

EEBus SPINE Technical Specification

Resource Specification

Version 1.1.1

Cologne, 2018-12-21

EEBus Initiative e.V.

Butzweilerhof Allee 4
50829 Cologne
GERMANY

Rue d'Arlon 25
1050 Brussels
BELGIUM

Phone: +49 221 / 47 44 12 - 20
Fax: +49 221 / 47 44 12 - 1822

info@eebus.org
www.eebus.org

District court: Cologne
VR 17275

Table of contents

Table of contents.....	2
List of figures	7
List of tables	12
1 Introduction.....	21
1.1 References.....	21
1.1.1 EEBUS SPINE documents	21
1.1.2 Other EEBUS documents	21
1.1.3 Other documents	21
1.1.4 Websites	21
1.1.5 Normative References.....	21
1.2 Terms and definitions.....	22
1.3 Document structure	24
1.4 How to read this document.....	25
1.4.1 Graphical representation	25
1.4.2 Namespaces	28
1.4.3 "datagram" vs. specific function – XMLs with different roots and scopes	28
1.4.4 Simplification of XML examples	29
1.4.5 Used requirement keywords.....	30
1.4.6 Presence indications in Feature Types	30
1.4.7 Presence indications in SmartEnergyManagementPs.....	30
2 Overall model hierarchy concept overview	31
2.1 Resource Type Definition Concept	31
2.1.1 Introduction.....	31
2.1.2 Device Model Facets	32
2.1.3 Vendor specific extensions.....	33
2.1.4 Feature Group	33
2.1.5 Specific usage	34
2.2 Class Concept	34
2.2.1 Standard Classes.....	34
2.2.2 Complex Classes	34
2.2.3 SPINE class hierarchy (Class / Sub-Class / Function-Group / Function / Element).....	34
3 Common technical details.....	36
3.1 Introduction.....	36
3.2 Time information (absolute / relative / recurring).....	36
3.3 List Data concept	36
3.4 Identifier concept	37
3.4.1 Introduction.....	37
3.4.2 Identifier rules	37
3.5 Multidimensional data representation with "flat" list-based data model.....	40
3.6 Restricted function exchange for list based functions	43
3.7 Relations between thematically connected SPINE classes	43
3.7.1 Relations between Alarm / ElectricalConnection / HVAC / LoadControl / Measurement / Setpoint / SupplyCondition / TariffInformation / Threshold / TimeTable.....	44
3.7.2 Relations between DirectControl / HVAC / LoadControl / OperatingConstraints / PowerSequences (SmartEnergyManagementPs) / TaskManagement	45

48	3.8	On the use of "label" and "description"	46
49	3.9	Empty elements as "tags"	47
50	3.10	Common data types	48
51	3.10.1	General	48
52	3.10.2	Time-related	60
53	3.10.3	Address-related	66
54	3.11	Result.....	68
55	4	Device model definitions.....	70
56	4.1	Device Types.....	70
57	4.1.1	ChargingStation	70
58	4.1.2	Dishwasher	70
59	4.1.3	Dryer	70
60	4.1.4	ElectricitySupplySystem	70
61	4.1.5	EnergyManagementSystem	70
62	4.1.6	EnvironmentSensor	70
63	4.1.7	Generic	71
64	4.1.8	HeatGenerationSystem	71
65	4.1.9	HeatSinkSystem	71
66	4.1.10	HeatStorageSystem	71
67	4.1.11	HVACController	71
68	4.1.12	Inverter	72
69	4.1.13	SubMeter.....	72
70	4.1.14	Washer	72
71	4.2	Entity Types	72
72	4.2.1	Battery.....	72
73	4.2.2	BatterySystem	72
74	4.2.3	CEM	72
75	4.2.4	ChargingOutlet	72
76	4.2.5	Compressor	73
77	4.2.6	DeviceInformation.....	73
78	4.2.7	DHWCircuit.....	73
79	4.2.8	DHWStorage	73
80	4.2.9	Dishwasher	73
81	4.2.10	Dryer	73
82	4.2.11	ElectricImmersionHeater.....	73
83	4.2.12	ElectricityGenerationSystem	74
84	4.2.13	ElectricityStorageSystem	74
85	4.2.14	EV.....	74
86	4.2.15	EVSE	74
87	4.2.16	Fan	74
88	4.2.17	GasHeatingAppliance	74
89	4.2.18	Generic	74
90	4.2.19	GridConnectionPointOfPremises	75
91	4.2.20	HeatingBufferStorage.....	75
92	4.2.21	HeatingCircuit.....	75
93	4.2.22	HeatingObject.....	75

94	4.2.23	HeatingZone	75
95	4.2.24	HeatPumpAppliance.....	75
96	4.2.25	HeatSinkCircuit	76
97	4.2.26	HeatSourceCircuit.....	76
98	4.2.27	HeatSourceUnit	76
99	4.2.28	Household	76
100	4.2.29	HVACController	76
101	4.2.30	HVACRoom	76
102	4.2.31	InstantDHWHeater	77
103	4.2.32	Inverter	77
104	4.2.33	OilHeatingAppliance.....	77
105	4.2.34	Pump	77
106	4.2.35	PVSystem.....	77
107	4.2.36	RefrigerantCircuit	77
108	4.2.37	SmartEnergyAppliance	77
109	4.2.38	SolarDHWStorage.....	78
110	4.2.39	SolarThermalCircuit.....	78
111	4.2.40	SubMeterElectricity	78
112	4.2.41	TemperatureSensor.....	78
113	4.2.42	Washer	78
114	4.3	Feature Types	78
115	4.3.1	ActuatorLevel	78
116	4.3.2	ActuatorSwitch	80
117	4.3.3	Alarm	82
118	4.3.4	Bill	83
119	4.3.5	DataTunneling	89
120	4.3.6	DeviceClassification	91
121	4.3.7	DeviceConfiguration.....	95
122	4.3.8	DeviceDiagnosis.....	99
123	4.3.9	DirectControl	103
124	4.3.10	ElectricalConnection.....	105
125	4.3.11	Generic	111
126	4.3.12	HVAC.....	111
127	4.3.13	Identification	119
128	4.3.14	IncentiveTable	120
129	4.3.15	LoadControl	132
130	4.3.16	Measurement	136
131	4.3.17	Messaging.....	143
132	4.3.18	NetworkManagement	144
133	4.3.19	NodeManagement	144
134	4.3.20	OperatingConstraints	144
135	4.3.21	PowerSequences	151
136	4.3.22	Sensing.....	151
137	4.3.23	Setpoint	155
138	4.3.24	SmartEnergyManagementPs.....	159
139	4.3.25	SupplyCondition	206

140	4.3.26	TariffInformation	209
141	4.3.27	TaskManagement	209
142	4.3.28	Threshold	213
143	4.3.29	TimeInformation	216
144	4.3.30	TimeSeries	219
145	4.3.31	TimeTable	225
146	5	Class descriptions	229
147	5.1	Introduction	229
148	5.2	Complex Classes	229
149	5.2.1	IncentiveTable	229
150	5.2.2	NodeManagement	238
151	5.2.3	SmartEnergyManagementPs	238
152	5.3	Standard Classes	259
153	5.3.1	ActuatorLevel	259
154	5.3.2	ActuatorSwitch	261
155	5.3.3	Alarm	264
156	5.3.4	Bill	266
157	5.3.5	BindingManagement	275
158	5.3.6	DataTunneling	280
159	5.3.7	DeviceClassification	283
160	5.3.8	DeviceConfiguration	287
161	5.3.9	DeviceDiagnosis	295
162	5.3.10	DirectControl	300
163	5.3.11	ElectricalConnection	304
164	5.3.12	HVAC	313
165	5.3.13	Identification	326
166	5.3.14	LoadControl	327
167	5.3.15	Measurement	340
168	5.3.16	Messaging	350
169	5.3.17	NetworkManagement	354
170	5.3.18	OperatingConstraints	379
171	5.3.19	PowerSequences	388
172	5.3.20	Sensing	415
173	5.3.21	Setpoint	420
174	5.3.22	SubscriptionManagement	427
175	5.3.23	SupplyCondition	432
176	5.3.24	TariffInformation	437
177	5.3.25	TaskManagement	462
178	5.3.26	Threshold	468
179	5.3.27	TimeInformation	473
180	5.3.28	TimeSeries	478
181	5.3.29	TimeTable	486
182	5.3.30	UseCaseInformation	493
183	5.3.31	Version	495
184	Annex A - SPINE XSDs		498
185	A.1	Complex Classes	498

186	A.1.1	IncentiveTable	498
187	A.1.2	NodeManagement	506
188	A.1.3	SmartEnergyManagementPs.....	518
189	A.2	Standard Classes.....	528
190	A.2.1	ActuatorLevel	528
191	A.2.2	ActuatorSwitch	529
192	A.2.3	Alarm	530
193	A.2.4	Bill	531
194	A.2.5	BindingManagement	537
195	A.2.6	DataTunneling	538
196	A.2.7	DeviceClassification	539
197	A.2.8	DeviceConfiguration	540
198	A.2.9	DeviceDiagnosis.....	543
199	A.2.10	DirectControl	545
200	A.2.11	ElectricalConnection.....	546
201	A.2.12	HVAC.....	550
202	A.2.13	Identification	555
203	A.2.14	LoadControl	556
204	A.2.15	Measurement	560
205	A.2.16	Messaging.....	564
206	A.2.17	NetworkManagement	565
207	A.2.18	OperatingConstraints	570
208	A.2.19	PowerSequences	573
209	A.2.20	Sensing.....	580
210	A.2.21	Setpoint	582
211	A.2.22	SubscriptionManagement	584
212	A.2.23	SupplyCondition	585
213	A.2.24	TariffInformation	588
214	A.2.25	TaskManagement	595
215	A.2.26	Threshold.....	598
216	A.2.27	TimeInformation.....	600
217	A.2.28	TimeSeries	601
218	A.2.29	TimeTable	604
219	A.2.30	UseCaseInformation	606
220	A.2.31	Version.....	607
221	A.3	Further XSDs	608
222	A.3.1	CommandCommonDefinitions	608
223	A.3.2	CommandFrame	614
224	A.3.3	CommonDataTypes	615
225	A.3.4	Datagram	628
226	A.3.5	Result.....	629
227	A.3.6	Overview example	629
228	Annex B - Resource Definitions Overview		631
229	B.1	Introduction.....	631
230	B.2	Device Types.....	631
231	B.3	Entity Types	631

232	B.4	Feature Types	634
233	B.5	Complex Classes	635
234	B.6	Standard Classes.....	635
235	B.7	Identifier list	637
236			

237 List of figures

238	Figure 1: XSD "simple type"	25
239	Figure 2: XSD "complex type" with no content.....	25
240	Figure 3: XSD "complex type" with content.....	25
241	Figure 4: XSD element with type	26
242	Figure 5: XSD sequence	26
243	Figure 6: XSD choice	26
244	Figure 7: XSD all.....	26
245	Figure 8: Optional vs. mandatory.....	27
246	Figure 9: Lists / multiplicity	27
247	Figure 10: XSD expanded view	27
248	Figure 11: Datagram structure (red line: "way" to the functions)	28
249	Figure 12: Payload structure (red line: "way" to the functions)	28
250	Figure 13: Dependencies of the standardised types and identifiers.....	31
251	Figure 14: SPINE device model example	32
252	Figure 15: SPINE class hierarchy.....	35
253	Figure 16: Absolute or relative time type	36
254	Figure 17: Non-list data function example.....	36
255	Figure 18: List data function example	37
256	Figure 19: Excerpt of the list element "measurementData" of the SPINE function measurementListData	41
258	Figure 20: SPINE identifiers can create a multidimensional space of data.	41
259	Figure 21: The flat data structure of Figure 19 can also be rearranged (virtually) to emphasize the priorities of the identifiers.	43
261	Figure 22: Relations between Alarm / ElectricalConnection / HVAC / LoadControl / Measurement / Setpoint / SupplyCondition / TariffInformation / Threshold / TimeTable	45
263	Figure 23: Relations between DirectControl / HVAC / LoadControl / OperatingConstraints / PowerSequences (SmartEnergyManagementPs) / TaskManagement	46
265	Figure 24: Empty elements example.....	47
266	Figure 25: ScaledNumberType	50
267	Figure 26: ScaledNumberSetType	50
268	Figure 27: ScaledNumberRangeType	51
269	Figure 28: PossibleOperationsType.....	60
270	Figure 29: FunctionPropertyType.....	60
271	Figure 30: TimePeriodType	60
272	Figure 31: TimestampIntervalType	61
273	Figure 32: DaysOfWeekType	63
274	Figure 33: AbsoluteOrRecurringTimeType	65
275	Figure 34: RecurrenceInformationType	66

276	Figure 35: DeviceAddressType	67
277	Figure 36: EntityAddressType.....	67
278	Figure 37: FeatureAddressType.....	68
279	Figure 38: resultData function overview.....	68
280	Figure 39: IncentiveTable example (single incentiveSlot) for different tiers	121
281	Figure 40: IncentiveTable function-group overview	229
282	Figure 41: incentiveTableData function overview	230
283	Figure 42: IncentiveTableIncentiveSlotType overview	231
284	Figure 43: incentiveTableDescriptionData function overview	233
285	Figure 44: IncentiveTableDescriptionTierType overview	234
286	Figure 45: incentiveTableConstraintsData overview	237
287	Figure 46: SmartEnergyManagementPs function-group overview.....	239
288	Figure 47: Concept of a power sequence.....	240
289	Figure 48: smartEnergyManagementPsData function overview, part 1.....	244
290	Figure 49: smartEnergyManagementPsData function overview, part 2.....	244
291	Figure 50: smartEnergyManagementPsData function overview, part 3.....	245
292	Figure 51: smartEnergyManagementPsData function overview, part 4.....	246
293	Figure 52: smartEnergyManagementPsConfigurationRequestCall function overview.....	255
294	Figure 53: smartEnergyManagementPsPriceData function overview	256
295	Figure 54: smartEnergyManagementPsPriceCalculationRequestCall function overview.....	258
296	Figure 55: ActuatorLevel function-group overview	259
297	Figure 56: actuatorLevelData function overview	259
298	Figure 57: actuatorLevelDescriptionData function overview	260
299	Figure 58: ActuatorSwitch function-group overview	262
300	Figure 59: actuatorSwitchData function overview.....	262
301	Figure 60: actuatorSwitchDescription function overview.....	263
302	Figure 61: Alarm function-group overview	264
303	Figure 62: alarmListData function overview	264
304	Figure 63: Bill function-group overview	266
305	Figure 64: billListData function overview.....	267
306	Figure 65: billListData function detail 1, "total"	267
307	Figure 66: billListData function detail 2, "position"	268
308	Figure 67: billConstraintsListData function overview	273
309	Figure 68: billDescriptionListData function overview	274
310	Figure 69: BindingManagement function-group overview (data)	275
311	Figure 70: BindingManagement function-group overview (call)	275
312	Figure 71: bindingManagementEntryListData function overview	276
313	Figure 72: bindingManagementRequestCall function overview.....	278
314	Figure 73: bindingManagementDeleteCall function overview	279
315	Figure 74: DataTunneling function-group overview (call).....	280
316	Figure 75: dataTunnelingCall function overview.....	282
317	Figure 76: DeviceClassification function-group overview	283
318	Figure 77: deviceClassificationManufacturerData function overview.....	284
319	Figure 78: deviceClassificationUserData function overview	287
320	Figure 79: DeviceConfiguration function-group overview	288
321	Figure 80: deviceConfigurationKeyValueListData function overview	288

322	Figure 81: deviceConfigurationKeyValueDescriptionListData function overview	290
323	Figure 82: deviceConfigurationKeyValueConstraintsListData function overview.....	292
324	Figure 83: deviceConfigurationKeyValueConstraintsListData function detail	293
325	Figure 84: DeviceDiagnosis function-group overview.....	295
326	Figure 85: deviceDiagnosisStateData function overview	296
327	Figure 86: deviceDiagnosisHeartbeatData function overview	298
328	Figure 87: deviceDiagnosisServiceData function overview.....	299
329	Figure 88: DirectControl function-group overview	300
330	Figure 89: directControlActivityListData function overview	301
331	Figure 90: directControlDescriptionData function overview	303
332	Figure 91: ElectricalConnection function-group overview.....	304
333	Figure 92: electricalConnectionParameterDescriptionListData function overview.....	305
334	Figure 93: electricalConnectionPermittedValueSetListData function overview.....	308
335	Figure 94: electricalConnectionStateListData function overview	310
336	Figure 95: electricalConnectionStateDescriptionData function overview	312
337	Figure 96: HVAC function-group overview.....	314
338	Figure 97: hvacSystemFunctionListData function overview	314
339	Figure 98: hvacSystemFunctionOperationModeRelationListData function overview.....	316
340	Figure 99: hvacSystemFunctionSetpointRelationListData function overview	317
341	Figure 100: hvacSystemFunctionPowerSequenceRelationListData function overview.....	318
342	Figure 101: hvacSystemFunctionDescriptionListData function overview.....	319
343	Figure 102: hvacOperationModeDescriptionListData function overview.....	321
344	Figure 103: hvacOverrunListData function overview.....	323
345	Figure 104: hvacOverrunDescriptionListData function overview	324
346	Figure 105: Identification function-group overview	326
347	Figure 106: identificationListData function overview	326
348	Figure 107: LoadControl function-group overview	328
349	Figure 108: loadControlNodeData function overview	328
350	Figure 109: loadControlEventListData function overview	330
351	Figure 110: loadControlStateListData function overview	332
352	Figure 111: loadControlLimitListData function overview	334
353	Figure 112: loadControlLimitConstraintsListData function overview	336
354	Figure 113: loadControlLimitDescriptionListData function overview	338
355	Figure 114: Measurement function-group overview.....	340
356	Figure 115: measurementListData function overview	341
357	Figure 116: measurementConstraintsListData function overview	345
358	Figure 117: measurementDescriptionListData function overview	346
359	Figure 118: measurementThresholdRelationListData function overview	349
360	Figure 119: Messaging function-group overview.....	350
361	Figure 120: messagingListData function overview.....	350
362	Figure 121: NetworkManagement function-group overview (call)	354
363	Figure 122: NetworkManagement function-group overview (data)	354
364	Figure 123: networkManagementAddNodeCall function overview	356
365	Figure 124: networkManagementRemoveNodeCall function overview	357
366	Figure 125: networkManagementModifyNodeCall function overview	359
367	Figure 126: networkManagementScanNetworkCall function overview.....	361

368	Figure 127: networkManagementDiscoverCall function overview.....	362
369	Figure 128: networkManagementAbortCall function overview	363
370	Figure 129: networkManagementProcessStateData function overview	364
371	Figure 130: networkManagementJoiningModeData function overview.....	365
372	Figure 131: networkManagementReportCandidateData function overview	366
373	Figure 132: networkManagementDeviceDescriptionListData function overview, part 1	368
374	Figure 133: networkManagementDeviceDescriptionListData function overview, part 2	369
375	Figure 134: networkManagementEntityDescriptionListData function overview, part 1.....	373
376	Figure 135: networkManagementEntityDescriptionListData function overview, part 2.....	373
377	Figure 136: networkManagementFeatureDescriptionListData function overview, part 1.....	375
378	Figure 137: networkManagementFeatureDescriptionListData function overview, part 2.....	376
379	Figure 138: networkManagementFeatureDescriptionListData function overview, part 3.....	377
380	Figure 139: OperatingConstraints function-group overview	380
381	Figure 140: operatingConstraintsInterruptListData function overview.....	381
382	Figure 141: operatingConstraintsDurationListData function overview	382
383	Figure 142: operatingConstraintsPowerDescriptionListData function overview.....	383
384	Figure 143: operatingConstraintsPowerRangeListData function overview	385
385	Figure 144: operatingConstraintsPowerLevelListData function overview.....	386
386	Figure 145: operatingConstraintsResumImplicationListData function overview	387
387	Figure 146: Concept overview of power sequences: Power P over time t; sequences are constituted	
388	of slots	389
389	Figure 147: Example of a device that first consumes and later on generates power. In this case	
390	“consumption” is assigned a positive value.	390
391	Figure 148: Example of a sequence with only one slot with some possible parameter boundaries..	390
392	Figure 149: PowerSequences function-group overview (data).....	392
393	Figure 150: PowerSequences function-group overview (call)	392
394	Figure 151: powerTimeSlotScheduleListData function overview	393
395	Figure 152: powerTimeSlotValueListData function overview.....	395
396	Figure 153: powerTimeSlotScheduleConstraintsListData function overview.....	398
397	Figure 154: powerSequenceAlternativesRelationListData function overview	400
398	Figure 155: powerSequenceDescriptionListData function overview.....	401
399	Figure 156: powerSequenceStateListData function overview	403
400	Figure 157: powerSequenceScheduleListData function overview.....	405
401	Figure 158: powerSequenceScheduleConstraintsListData function overview	406
402	Figure 159: powerSequencePriceListData function overview	408
403	Figure 160: powerSequenceSchedulePreferenceListData function overview	409
404	Figure 161: powerSequenceNodeScheduleInformationData function overview	411
405	Figure 162: powerSequenceScheduleConfigurationRequestCall function overview.....	413
406	Figure 163: powerSequencePriceCalculationRequestCall function overview	414
407	Figure 164: Sensing function-group overview.....	415
408	Figure 165: sensingListData function overview	415
409	Figure 166: sensingDescriptionData function overview	419
410	Figure 167: Setpoint function-group overview	421
411	Figure 168: setpointListData function overview	421
412	Figure 169: setpointConstraintsListData function overview.....	423
413	Figure 170: setpointDescriptionListData function overview.....	425

414	Figure 171: SubscriptionManagement function-group overview (data)	427
415	Figure 172: SubscriptionManagement function-group overview (call)	427
416	Figure 173: subscriptionManagementEntryListData function overview	428
417	Figure 174: subscriptionManagementRequestCall function overview	429
418	Figure 175: subscriptionManagementDeleteCall function overview.....	431
419	Figure 176: SupplyCondition function-group overview	432
420	Figure 177: supplyConditionListData function overview	432
421	Figure 178: supplyConditionDescriptionListData function overview.....	435
422	Figure 179: supplyConditionThresholdRelationListData function overview.....	436
423	Figure 180: TariffInformation relations, sub-class level.....	438
424	Figure 181: TariffInformation relations, function level.....	438
425	Figure 182: TariffInformation Sub-class overview	439
426	Figure 183: TariffInformation, sub-class Tariff, function-group overview	440
427	Figure 184: tariffOverallConstraintsData function overview	441
428	Figure 185: tariffListData function overview.....	442
429	Figure 186: tariffBoundaryRelationListData function overview	443
430	Figure 187: tariffTierRelationListData function overview	444
431	Figure 188: tariffDescriptionListData function overview	446
432	Figure 189: TariffInformation, sub-class TierBoundary, function-group overview.....	447
433	Figure 190: tierBoundaryListData function overview	448
434	Figure 191: tierBoundaryDescriptionListData function overview.....	450
435	Figure 192: TariffInformation, sub-class Commodity, function-group overview.....	452
436	Figure 193: commodityListData function overview	452
437	Figure 194: TariffInformation, sub-class Tier, function-group overview	453
438	Figure 195: tierListData function overview.....	454
439	Figure 196: tierIncentiveRelationListData function overview.....	455
440	Figure 197: tierDescriptionListData function overview	457
441	Figure 198: TariffInformation, sub-class Incentive, function-group overview.....	458
442	Figure 199: incentiveListData function overview.....	459
443	Figure 200: incentiveDescriptionListData function overview	461
444	Figure 201: TaskManagement function-group overview	462
445	Figure 202: taskManagementJobListData function overview.....	463
446	Figure 203: taskManagementJobRelationListData function overview	465
447	Figure 204: taskManagementJobDescriptionListData function overview	466
448	Figure 205: taskManagementOverviewData function overview	468
449	Figure 206: Threshold function-group overview.....	469
450	Figure 207: thresholdListData function overview	469
451	Figure 208: thresholdConstraintsListData function overview.....	470
452	Figure 209: thresholdDescriptionListData function overview	472
453	Figure 210: TimeInformation function-group overview (data).....	474
454	Figure 211: TimeInformation function-group overview (call).....	474
455	Figure 212: timeInformationData function overview	474
456	Figure 213: timeDistributorData function overview	475
457	Figure 214: timePrecisionData function overview	476
458	Figure 215: timeDistributorEnquiryCall function overview	477
459	Figure 216: TimeSeries function-group overview	478

460	Figure 217: timeSeriesListData function overview	479
461	Figure 218: timeSeriesDescriptionListData function overview	481
462	Figure 219: timeSeriesConstraintsListData function overview	484
463	Figure 220: TimeTable function-group overview	486
464	Figure 221: timeTableListData function overview	487
465	Figure 222: timeTableConstraintsListData function overview	490
466	Figure 223: timeTableDescriptionListData function overview	492
467	Figure 224: UseCaseInformation function-group overview	493
468	Figure 225: useCaseInformationListData function overview	494
469	Figure 226: Version function-group overview	496
470	Figure 227: specificationVersionListData function overview	496

471

472 List of tables

473	Table 1: ScaledNumberSetType	50
474	Table 2: ScaledNumberRangeType	51
475	Table 3: Enumeration CommodityTypeEnumType	51
476	Table 4: Enumeration EnergyDirectionEnumType value rules	52
477	Table 5: Enumeration EnergyModeEnumType	52
478	Table 6: Enumeration EnergyModeEnumType value rules	53
479	Table 7: Enumeration UnitOfMeasurementEnumType	55
480	Table 8: Enumeration CurrencyEnumType	56
481	Table 9: Enumeration ScopeTypeEnumType	57
482	Table 10: TimePeriodType	61
483	Table 11: TimestampIntervalType	61
484	Table 12: RecurringIntervalEnumType	62
485	Table 13: MonthType	62
486	Table 14: DayOfWeekType	63
487	Table 15: OccurrenceEnumType	64
488	Table 16: AbsoluteOrRecurringTimeType	65
489	Table 17: RecurrenceInformationType	66
490	Table 18: resultData function detailed description of elements	69
491	Table 19: Values specified for element errorNumber	69
492	Table 20: actuatorLevelData Element rules	79
493	Table 21: Element "function" value rules	79
494	Table 22: actuatorLevelDescriptionData Element rules	80
495	Table 23: actuatorSwitchData Element rules	81
496	Table 24: Element "function" value rules	81
497	Table 25: actuatorSwitchDescriptionData Element rules	81
498	Table 26: alarmListData Element rules	83
499	Table 27: Element "alarmType" value rules	83
500	Table 28: billListData Element rules	86
501	Table 29: Element "billType" value rules	86
502	Table 30: Element "costType" value rules	86
503	Table 31: Element "positionType" value rules	86

504	Table 32: billConstraintsListData Element rules.....	87
505	Table 33: billDescriptionListData Element rules	88
506	Table 34: dataTunnelingCall Element rules.....	91
507	Table 35: deviceClassificationManufacturerData Element rules	94
508	Table 36: Element "powerSource" value rules.....	94
509	Table 37: deviceClassificationUserData Element rules	95
510	Table 38: deviceConfigurationKeyValueListData Element rules	96
511	Table 39: Element "value" sub-element rules.....	97
512	Table 40: deviceConfigurationKeyValueDescriptionListData Element rules.....	97
513	Table 41: Element "keyName" value rules.....	98
514	Table 42: Element "valueType" value rules.....	98
515	Table 43: deviceConfigurationKeyValueListData Element rules	99
516	Table 44: deviceDiagnosisStateData Element rules	101
517	Table 45: Element "operatingState" value rules	101
518	Table 46: Element "powerSupplyCondition" value rules	102
519	Table 47: deviceDiagnosisHeartbeatData Element rules	102
520	Table 48: deviceDiagnosisServiceData Element rules.....	103
521	Table 49: directControlActivityListData Element rules	104
522	Table 50: Element "activityState" value rules	105
523	Table 51: directControlActivityListData Element rules	105
524	Table 52: electricalConnectionParameterDescriptionListData Element rules	107
525	Table 53: Element "voltageType" value rules	107
526	Table 54: Element "acMeasuredPhases" value rules.....	108
527	Table 55: Element "acMeasuredInReferenceTo" value rules	108
528	Table 56: Element "acMeasurementType" value rules.....	108
529	Table 57: Element "acMeasurementVariant" value rules.....	108
530	Table 58: electricalConnectionPermittedValueSetListData Element rules.....	109
531	Table 59: electricalConnectionStateListData Element rules	110
532	Table 60: electricalConnectionDescriptionListData Element rules	110
533	Table 61: hvacSystemFunctionListData Element rules	113
534	Table 62: hvacSystemFunctionOperationModeRelationListData Element rules	114
535	Table 63: hvacSystemFunctionSetpointRelationListData Element rules	114
536	Table 64: hvacSystemFunctionPowerSequenceRelationListData Element rules.....	115
537	Table 65: hvacSystemFunctionDescriptionListData Element rules.....	115
538	Table 66: Element "systemFunctionType" value rules.....	116
539	Table 67: hvacOperationModeDescriptionListData Element rules.....	116
540	Table 68: Element "operationModeType" value rules.....	117
541	Table 69: hvacOverrunListData Element rules.....	117
542	Table 70: Element "overrunStatus" value rules	118
543	Table 71: hvacOverrunDescriptionListData Element rules	118
544	Table 72: Element "overrunType" value rules	119
545	Table 73: Relation between "overrunStatus" and "isOverrunActive"	119
546	Table 74: identificationListData Element rules	120
547	Table 75: Element "identificationType" value rules.....	120
548	Table 76: incentiveTableData Element rules.....	126
549	Table 77: incentiveTableDescriptionData Element rules	129

550	Table 78: Element "tier. tierDescription. tierType" value rules	129
551	Table 79: Element "tier. boundaryDescription. boundaryType" value rules	130
552	Table 80: Element "tier. incentiveDescription. incentiveType" value rules.....	130
553	Table 81: incentiveTableConstraintsData Element rules	131
554	Table 82: loadControlNodeData Element rules	133
555	Table 83: loadControlLimitListData Element rules.....	134
556	Table 84: loadControlLimitConstraintsListData Element rules	135
557	Table 85: loadControlLimitDescriptionListData Element rules	136
558	Table 86: Element "limitType" value rules	136
559	Table 87: Element "limitCategory" value rules	136
560	Table 88: measurementListData Element rules	138
561	Table 89: Element "valueType" value rules.....	139
562	Table 90: Element "valueSource" value rules	139
563	Table 91: Element "valueTendency" value rules.....	139
564	Table 92: Element "valueState" value rules	139
565	Table 93: measurementConstraintsListData Element rules.....	140
566	Table 94: measurementConstraintsListData Element rules.....	141
567	Table 95: Element "measurementType" value rules.....	142
568	Table 96: measurementThresholdRelationListData Element rules.....	142
569	Table 97: messagingListData Element rules	144
570	Table 98: Element "type" value rules	144
571	Table 99: operatingConstraintsInterruptListData Element rules	146
572	Table 100: operatingConstraintsDurationListData Element rules	147
573	Table 101: operatingConstraintsPowerDescriptionListData Element rules.....	148
574	Table 102: operatingConstraintsPowerRangeListData Element rules	148
575	Table 103: operatingConstraintsPowerLevelListData Element rules	149
576	Table 104: operatingConstraintsResumeImplicationListData Element rules.....	150
577	Table 105: sensingListData Element rules.....	152
578	Table 106: Element "state" value rules	154
579	Table 107: sensingDescriptionData Element rules.....	154
580	Table 108: Element "sensingType" value rules	155
581	Table 109: setpointListData Element rules	157
582	Table 110: setpointConstraintsListData Element rules	157
583	Table 111: setpointDescriptionListData Element rules	158
584	Table 112: Element "setpointType" value rules	158
585	Table 113: Server messages overview. The "write (restricted)" denotes if the message is intended for	
586	partial modification.	160
587	Table 114: Description of complex class function "smartEnergyManagementPsData" for	
588	powerSequences information	191
589	Table 115: Dedicated "selectors" of "smartEnergyManagementPsData" to request a specific	
590	powerSequences information.	192
591	Table 116: Description of complex class function	
592	"smartEnergyManagementPsConfigurationRequestCall" for powerSequences configuration request	
593	call	193
594	Table 117: Description of "restricted write" operation for powerSequences configuration.	198
595	Table 118: Description of "restricted write" operation for powerSequences state change.	200

596	Table 119: Description of smartEnergyManagementPsPriceData for powerSequences price information.....	201
598	Table 120: Dedicated "selectors" of "smartEnergyManagementPsPriceData" to request a specific powerSequences price information.	202
600	Table 121: Description of "restricted write" operation for a power sequence price modification....	204
601	Table 122: Description of complex class function "smartEnergyManagementPsPriceCalculationRequestCall" for powerSequences price calculation request call	205
604	Table 123: supplyConditionListData Element rules	207
605	Table 124: Element "eventType" value rules	208
606	Table 125: Element "originator" value rules	208
607	Table 126: Element "gridCondition" value rules	208
608	Table 127: supplyConditionDescriptionListData Element rules	209
609	Table 128: supplyConditionThresholdRelationListData Element rules	209
610	Table 129: taskManagementJobListData Element rules	211
611	Table 130: taskManagementJobRelationListData Element rules	212
612	Table 131: taskManagementJobDescriptionListData Element rules	212
613	Table 132: Element "jobSource" value rules	212
614	Table 133: taskManagementOverviewData Element rules	213
615	Table 134: thresholdListData Element rules	214
616	Table 135: thresholdConstraintsListData Element rules	215
617	Table 136: thresholdDescriptionListData Element rules	215
618	Table 137: Element "thresholdType" value rules	216
619	Table 138: timeInformationData Element rules	217
620	Table 139: timeDistributorData Element rules	218
621	Table 140: timePrecisionData Element rules	218
622	Table 141: timeSeriesListData Element rules	221
623	Table 142: timeSeriesDescriptionListData Elements	223
624	Table 143: Element "timeSeriesType" value rules	223
625	Table 144: timeSeriesConstraintsListData Element rules	225
626	Table 145: timeTableListData Element rules	226
627	Table 146: timeTableConstraintsListData Element rules	227
628	Table 147: timeTableDescriptionListData Element rules	227
629	Table 148: Element "timeSlotTimeMode" value rules	228
630	Table 149: incentiveTable detailed description of elements	230
631	Table 150: IncentiveTableIncentiveSlotType detailed description of elements	232
632	Table 151: incentiveTableDescriptionData function detailed description of elements	234
633	Table 152: IncentiveTableDescriptionTierType detailed description of elements	236
634	Table 153: Enumeration TierTypeEnumType	236
635	Table 154: Enumeration TierBoundaryTypeEnumType	236
636	Table 155: Enumeration IncentiveTypeEnumType	237
637	Table 156: incentiveTableConstraints detailed description of elements	238
638	Table 157: Description of complex class function "smartEnergyManagementPsData"	254
639	Table 158: Dedicated "selectors" of "smartEnergyManagementPsData"	255
640	Table 159: Description of complex class function "smartEnergyManagementPsConfigurationRequestCall"	256

642	Table 160: Description of smartEnergyManagementPsPriceData.....	257
643	Table 161: Dedicated "selectors" of "smartEnergyManagementPsPriceData".....	257
644	Table 162: Description of complex class function	
645	"smartEnergyManagementPsPriceCalculationRequestCall".....	258
646	Table 163: actuatorLevelData function detailed description of elements.....	259
647	Table 164: Enumeration ActuatorLevelFctEnumType.....	260
648	Table 165: actuatorLevelDescriptionData function detailed description of elements	261
649	Table 166: actuatorSwitchData function detailed description of elements	262
650	Table 167: Enumeration ActuatorSwitchFctEnumType	262
651	Table 168: actuatorSwitchDescription function detailed description of elements	263
652	Table 169: alarmListData function detailed description of elements	265
653	Table 170: Enumeration AlarmTypeEnumType	265
654	Table 171: billListData function detailed description of elements	270
655	Table 172: Enumeration BillTypeEnumType	270
656	Table 173: Enumeration BillCostTypeEnumType	270
657	Table 174: Enumeration BillPositionTypeEnumType	271
658	Table 175: billConstraintsListData function detailed description of elements.....	273
659	Table 176: billDescriptionListData function detailed description of elements.....	275
660	Table 177: bindingManagementEntryListData function detailed description of elements.....	276
661	Table 178: bindingManagementRequestCall function detailed description of elements	278
662	Table 179: bindingManagementDeleteCall function detailed description of elements.....	279
663	Table 180: dataTunnelingData function detailed description of elements	282
664	Table 181: deviceClassificationManufacturerData function detailed description of elements.....	285
665	Table 182: Enumeration PowerSourceEnumType.....	286
666	Table 183: deviceClassificationUserData function detailed description of elements.....	287
667	Table 184: deviceConfigurationKeyValueListData function detailed description of elements.....	289
668	Table 185: Complex type DeviceConfigurationKeyValueValueType	289
669	Table 186: deviceConfigurationKeyValueDescriptionListData function detailed description of	
670	elements.....	291
671	Table 187: Enumeration DeviceConfigurationKeyNameEnumType	291
672	Table 188: Enumeration DeviceConfigurationKeyValueValueType	291
673	Table 189: deviceConfigurationKeyValueListData function detailed description of elements.....	294
674	Table 190: deviceDiagnosisStateData function detailed description of elements	297
675	Table 191: Enumeration DeviceDiagnosisOperatingStateEnumType	297
676	Table 192: Enumeration PowerSupplyConditionEnumType	297
677	Table 193: deviceDiagnosisHeartbeatData function detailed description of elements	298
678	Table 194: deviceDiagnosisServiceData function detailed description of elements	300
679	Table 195: directControlActivityListData function detailed description of elements.....	302
680	Table 196: Enumeration DirectControlActivityStateEnumType.....	302
681	Table 197: directControlDescriptionData function detailed description of elements.....	303
682	Table 198: electricalConnectionParameterDescriptionListData function detailed description of	
683	elements.....	306
684	Table 199: Enumeration ElectricalConnectionVoltageTypeEnumType.....	306
685	Table 200: Enumeration ElectricalConnectionPhaseNameEnumType.....	306
686	Table 201: Enumeration ElectricalConnectionAcMeasurementTypeEnumType	307
687	Table 202: Enumeration ElectricalConnectionMeasurandVariantEnumType.....	307

688	Table 203: electricalConnectionPermittedValueSetListData function detailed description of elements	
689	308
690	Table 204: electricalConnectionStateListData function detailed description of elements.....	310
691	Table 205: electricalConnectionDescriptionListData function detailed description of elements	313
692	Table 206: hvacSystemFunctionListData function detailed description of elements	315
693	Table 207: hvacSystemFunctionOperationModeRelationListData function detailed description of	
694	elements.....	316
695	Table 208: hvacSystemFunctionSetpointRelationListData function detailed description of elements	
696	317
697	Table 209: hvacSystemFunctionPowerSequenceRelationListData function detailed description of	
698	elements.....	318
699	Table 210: hvacSystemFunctionDescriptionListData function detailed description of elements	320
700	Table 211: Enumeration HvacSystemFunctionTypeEnumType	320
701	Table 212: hvacOperationModeDescriptionListData function detailed description of elements	321
702	Table 213: Enumeration HvacOperationModeTypeEnumType	322
703	Table 214: hvacOverrunListData function detailed description of elements	323
704	Table 215: Enumeration HvacOverrunStatusEnumType.....	323
705	Table 216: hvacOverrunDescriptionListData function detailed description of elements.....	325
706	Table 217: Enumeration HvacOverrunTypeEnumType	325
707	Table 218: identificationListData Element rules	327
708	Table 219: Enumeration IdentificationTypeEnumType.....	327
709	Table 220: loadControlNodeData function detailed description of elements.....	329
710	Table 221: loadControlEventListData function detailed description of elements.....	330
711	Table 222: Enumeration LoadControlEventActionEnumType.....	331
712	Table 223: loadControlStateListData function detailed description of elements.....	333
713	Table 224: Enumeration LoadControlEventStateEnumType.....	333
714	Table 225: loadControlLimitListData function detailed description of elements	334
715	Table 226: loadControlLimitConstraintsListData function detailed description of elements.....	336
716	Table 227: loadControlLimitDescriptionListData function detailed description of elements.....	339
717	Table 228: Enumeration LoadControlLimitTypeEnumType	339
718	Table 229: Enumeration LoadControlLimitCategoryEnumType.....	339
719	Table 230: measurementListData function detailed description of elements	342
720	Table 231: Enumeration MeasurementValueTypeEnumType	342
721	Table 232: Enumeration MeasurementValueSourceEnumType.....	343
722	Table 233: Enumeration MeasurementValueTendencyEnumType	343
723	Table 234: Enumeration MeasurementValueStateEnumType	343
724	Table 235: measurementConstraintsListData function detailed description of elements	345
725	Table 236: measurementDescriptionListData function detailed description of elements	347
726	Table 237: Enumeration MeasurementTypeEnumType	348
727	Table 238: measurementThresholdRelationListData function detailed description of elements.....	349
728	Table 239: messagingListData function detailed description of elements	351
729	Table 240: Enumeration MessagingTypeEnumType	351
730	Table 241: networkManagementAddNodeCall function element description.....	356
731	Table 242: networkManagementRemoveNodeCall function element description	358
732	Table 243: networkManagementModifyNodeCall function element description.....	360
733	Table 244: networkManagementScanNetworkCall function element description	361

734	Table 245: networkManagementDiscoverCall function element description	362
735	Table 246: networkManagementProcessStateData function element description.....	364
736	Table 247: Enumeration NetworkManagementProcessStateStateType	364
737	Table 248: networkManagementJoiningModeData function element description	366
738	Table 249: networkManagementReportCandidateData function element description.....	367
739	Table 250: networkManagementDeviceDescriptionData function element description	370
740	Table 251: Enumeration NetworkManagementFeatureSetType	370
741	Table 252: Enumeration NetworkManagementStateChangeType	371
742	Table 253: networkManagementEntityDescriptionData function element description	374
743	Table 254: networkManagementFeatureDescriptionListData function element description.....	378
744	Table 255: operatingConstraintsInterruptListData function element description	381
745	Table 256: operatingConstraintsDurationListData function element description.....	382
746	Table 257: operatingConstraintsPowerDescriptionListData function element description	384
747	Table 258: operatingConstraintsPowerRangeListData function element description.....	385
748	Table 259: operatingConstraintsPowerLevelListData element description.....	386
749	Table 260: operatingConstraintsResumImplicationListData function element description.....	388
750	Table 261: powerTimeSlotScheduleListData function element description.....	394
751	Table 262: powerTimeSlotValueListData function element description	396
752	Table 263: Enumeration PowerTimeSlotValueTypeEnumType	396
753	Table 264: powerTimeSlotScheduleConstraintsListData function element description	398
754	Table 265: powerSequenceAlternativesRelationListData function element description	400
755	Table 266: powerSequenceDescriptionListData function element description	402
756	Table 267: Enumeration PowerSequenceScopeEnumType	402
757	Table 268: powerSequenceStateListData function element description.....	404
758	Table 269: Enumeration PowerSequenceStateEnumType	405
759	Table 270: powerSequenceScheduleListData function element description	406
760	Table 271: powerSequenceScheduleConstraintsListData function element description	407
761	Table 272: powerSequencePriceListData function element description	409
762	Table 273: powerSequenceSchedulePreferenceListData function element description	410
763	Table 274: powerSequenceNodeScheduleInformationData function element description.....	412
764	Table 275: powerSequenceScheduleConfigurationRequestCall function element description	413
765	Table 276: powerSequencePriceCalculationRequestCall function element description.....	414
766	Table 277: sensingListData function detailed description of elements	416
767	Table 278: Enumeration SensingStateEnumType	417
768	Table 279: sensingDescriptionData function detailed description of elements	419
769	Table 280: Enumeration SensingTypeEnumType.....	420
770	Table 281: setpointListData function detailed description of elements.....	422
771	Table 282: setpointConstraintsListData function detailed description of elements	423
772	Table 283: setpointDescriptionListData function detailed description of elements	425
773	Table 284: Enumeration SetpointTypeEnumType	426
774	Table 285: subscriptionManagementEntryListData function detailed description of elements	428
775	Table 286: subscriptionManagementRequestCall function detailed description of elements.....	430
776	Table 287: subscriptionManagementDeleteCall function detailed description of elements	431
777	Table 288: supplyConditionListData function detailed description of elements.....	433
778	Table 289: Enumeration SupplyConditionEventTypeEnumType	433
779	Table 290: Enumeration SupplyConditionOriginatorEnumType.....	434

780	Table 291: Enumeration GridConditionEnumType	434
781	Table 292: supplyConditionDescriptionListData function detailed description of elements	435
782	Table 293: supplyConditionThresholdRelationListData function detailed description of elements ..	436
783	Table 294: tariffOverallConstraintsData function detailed description of elements.....	442
784	Table 295: tariffListData function detailed description of elements	443
785	Table 296: tariffBoundaryRelationListData function detailed description of elements	444
786	Table 297: tariffTierRelationListData function detailed description of elements.....	445
787	Table 298: tariffDescriptionListData function detailed description of elements.....	446
788	Table 299: tierBoundaryListData function detailed description of elements.....	448
789	Table 300: tierBoundaryDescriptionListData function detailed description of elements	451
790	Table 301: Enumeration TierBoundaryTypeEnumType	451
791	Table 302: commodityListData function detailed description of elements.....	453
792	Table 303: tierListData function detailed description of elements.....	454
793	Table 304: tierIncentiveRelationListData function detailed description of elements	456
794	Table 305: tierDescriptionListData function detailed description of elements	457
795	Table 306: Enumeration TierTypeEnumType	457
796	Table 307: incentiveListData function detailed description of elements	459
797	Table 308: Enumeration IncentiveValueTypeEnumType	460
798	Table 309: incentiveDescriptionListData function detailed description of elements	461
799	Table 310: Enumeration IncentiveTypeEnumType	462
800	Table 311: taskManagementJobListData function detailed description of elements	463
801	Table 312: taskManagementJobRelationListData function detailed description of elements.....	465
802	Table 313: taskManagementJobDescriptionListData function detailed description of elements.....	467
803	Table 314: Enumeration TaskManagementJobSourceEnumType	467
804	Table 315: taskManagementOverviewData function detailed description of elements	468
805	Table 316: thresholdListData function detailed description of elements.....	469
806	Table 317: thresholdConstraintsListData function detailed description of elements	471
807	Table 318: thresholdDescriptionListData function detailed description of elements	472
808	Table 319: Enumeration ThresholdTypeEnumType	473
809	Table 320: timeInformationData function detailed description of elements.....	475
810	Table 321: timeDistributorData function detailed description of elements.....	476
811	Table 322: timePrecisionData function detailed description of elements.....	477
812	Table 323: timeSeriesListData function detailed description of elements	479
813	Table 324: Complex type TimeSeriesSlotType	480
814	Table 325: timeSeriesDescriptionListData function detailed description of elements.....	482
815	Table 326: Enumeration TimeSeriesTypeEnumType	482
816	Table 327: timeSeriesConstraintsListData function detailed description of elements.....	485
817	Table 328: timeTableListData function detailed description of elements	487
818	Table 329: timeTableConstraintsListData function detailed description of elements	491
819	Table 330: timeTableDescriptionListData function detailed description of elements	492
820	Table 331: Enumeration TimeSlotTimeModeEnumType	492
821	Table 332: useCaseInformationListData function detailed description of elements.....	494
822	Table 333: specificationVersionListData function detailed description of elements	496
823	Table 334: Well known device types.....	631
824	Table 335: Well known entity types.....	634
825	Table 336: Well known feature types	635

826	Table 337: Well known complex classes	635
827	Table 338: Well known standard classes	637
828	Table 339: Identifier overview	638
829		
830		

1 Introduction

The SPINE data model is described within this technical specification (please keep in mind that some information may only be specified in the SPINE XSDs, see below). It contains the SPINE resource model, consisting of the device types, entity types and feature types as well as descriptions of the (standard and complex) classes as they are defined through the SPINE XSDs that are part of each release of the SPINE specifications. Additionally, some general rules, relations and data type descriptions are included.

The underlying SPINE data model of this document is identified by the XML namespace "http://docs.eebus.org/spine/xsd/v1".

1.1 References

1.1.1 EEBUS SPINE documents

[Introduction]	EEBus_SPINE_TR_Introduction.pdf
[ProtocolSpecification]	EEBus_SPINE_TS_ProtocolSpecification.pdf
[DataModelXSDs]	EEBus_SPINE_TS_ActuatorLevel.xsd, ..., EEBus_SPINE_TS_Version.xsd

1.1.2 Other EEBUS documents

[SHIPSpecification]	SHIP_Specification_V1.0.0.pdf
[SPINEUseCaseXSDs]	XSDs delivered within the EEBUS Use Case Package

1.1.3 Other documents

[SGReady]	2012-11-01_SG_Ready_Regularien_Version1.1.pdf
------------------	---

1.1.4 Websites

[EEBus]	http://www.eebus.org	Official EEBus Initiative e.V. website.
[W3C]	http://www.w3.org/	Official World Wide Web Consortium website.
[W3Schools]	http://www.w3schools.com/	Tutorials and examples for XML and others.

1.1.5 Normative References

[RFC2119]	IETF RFC 2119: 1997, Key words for use in RFCs to indicate requirement levels Please see section 1.4.5 for details.
------------------	--

1.2 Terms and definitions

Binding

Concept for connecting functionally matching features.

(Standard or Complex) Class

Set of SPINE functions used to model data for a specific functionality. A class can be considered as a topic where functions are defined for. For example the SPINE class "Measurement" is a collection of SPINE functions that are used to describe measurement values.

Classifier

Specifies whether a message serves to read, reply, write, etc.

Client

Role that specifies that a node uses data from a "server" or can change it.

Command

The functional part of a Message.

Complex Class

SPINE class that is build up by parts of SPINE standard classes and combines them in a new, ordered way.

(SPINE) Data model

Definition of the possible data that can be used for SPINE communications. Defined as XSD.

Device (specific node)

SPINE node that can include a set of entities. It has a "Device Type". With regards to the hierarchy of SPINE nodes a device is a root node for all functionalities offered by a device.

"device" (address information)

SPINE address part for the (physical) device.

Device Type

Specific type of physical device (e.g. "WashingMachine", "HeatPump", "FridgeFreezer", etc.).

Discovery

Process of finding appropriate partners for communication. Dependent on the context this can be either finding other devices or examination of a device's potential functionalities.

DSO

Distribution System Operator of an electricity grid.

Element

Item (or "attribute") of a SPINE function. Holds one information (e.g. "timestamp", "value", etc.) or contains further sub-elements.

Entity (specific node)

SPINE node that can include a set of (child) entities or features. It has an "Entity Type". With regards to the hierarchy of SPINE nodes an entity is a child element of a device or another entity.

- 900 **“entity”** (address information)
901 SPINE address part for the (logical) entity.
- 902 **Entity Type**
903 Specific type of logical device (e.g. "Freezer" is one logical part of a physical device "FridgeFreezer").
- 904 **EV**
905 Electric Vehicle
- 906 **EVSE**
907 Electric Vehicle Supply Equipment
- 908 **Feature** (specific node)
909 SPINE node that can include a set of functions (of a class). It has a "Feature Type". With regards to
910 the hierarchy of SPINE nodes a feature is a child element of an entity.
- 911 **“feature”** (address information)
912 SPINE address part for one feature.
- 913 **Feature Type**
914 Defines optional or mandatory rules and a general behaviour of the underlying Class (standard or
915 complex).
- 916 **(SPINE) Function**
917 A (SPINE) function is the smallest structure to model “actual data” (“functional data”). I.e. functions
918 usually consist of child elements that each hold an information (e.g. "timestamp", "value", etc.).
919 Information between communication partners is exchanged via the exchange of a function (as part of
920 a so-called “payload”).
- 921 **Header**
922 SPINE Header, including elements for addressing, unique identification of messages, timestamp, etc.
- 923 **Message**
924 One SPINE transfer from a sender to a receiver.
- 925 **(XML) Namespace**
926 XML namespaces provide a simple method for qualifying element and attribute names used in XML
927 documents by associating them with namespaces identified by URI references (source: www.w3.org).
- 928 **Node**
929 Common term for a SPINE instance that has a SPINE address. Dependent on the situation a node can
930 be either a device or an entity (of a specific device) or a feature (of a specific device-entity).
- 931 **Payload**
932 SPINE Payload, containing the functional SPINE data.
- 933 **Role**
934 Each Feature has a functional role, usually either “server” (data owner) or “client”. For some special
935 features (NodeManagement, e.g.) the role “special” is defined.

936 Scope (Type)

937 Some feature types define scope types for identifying specific functionalities unambiguously (e.g.
938 *outsideAirTemperature*).

939 Server

940 Role that specifies that a node offers own data to be read or written by a node with role client. A
941 server can notify its data to other nodes (with role client).

942 SPINE

943 **Smart Premises Interoperable Neutral-message Exchange**

944 Standard Class

945 All basic/standard functions are defined in standard classes. Functions of standard classes follow very
946 simple patterns and do not have deeply nested data structures.

947 Subscription

948 Enables the receiving of messages of interest from another device without polling it.

949 Use Case

950 Textual description of a re-usable functionality consisting of one or more messages of one or more
951 participating actors. May be visualized with a sequence diagram. E.g. "A CEM shifts the energy usage
952 of a washing machine."

953 User Story

954 Complete (but specific) business case described from the perspective of a user. Can be separated into
955 several use cases. E.g. "The user wants to get the laundry done by 8:00pm."

956 XML (Extensible Markup Language)

957 Human- and machine-readable markup language containing data. Used to model SPINE messages.

958 XSD (XML Schema Definition)

959 Definition format for XMLs, written in XML. Specifies how a well-formed XML (in regards to this XSD)
960 can be built. The SPINE data model is defined in XSD and supplementary documents (as not every
961 rule can be specified with XSD only). Other formats than XML can be derived from an XSD, too (e.g.
962 JSON).

963

964 1.3 Document structure

965 The main focus of this document is the description of the SPINE resources and the data model.
966 Before going into technical details, chapter 2 presents a coarse overview of the overall concept.
967 Chapter 3 collects a number of technical low-level details and can be left out at first reading. Chapter
968 4 defines types of different facets of a device (device types, entity types, feature types). These types
969 contain also rules which are relevant to establish an interoperable use of exchanged data. Finally,
970 chapter 5 describes SPINE classes with so-called "functions". These "functions" are the basic data
971 structures that are used within "messages" to represent or modify the content of a functionality or
972 data point. This chapter describes the potential purpose and use of functions, whereas chapter 4
973 (and esp. the feature types defined in section 4.3) impose rules on their use.

974

1.4 How to read this document

1.4.1 Graphical representation

1.4.1.1 Introduction

Documentation provided by the EEBus Initiative e.V. sometimes makes use of graphical representations of parts of the SPINE data models in order to ease the readability. All graphical representations are created using a specific commercial tool (oXygen). Please note that these graphical representations are not standardized. Furthermore, the explanations given below are tailored to the needs of the SPINE data models. In general, the explanations are not exhaustive.

1.4.1.2 Explanation of graphical XSD representation

The graphical XSD explanation uses a set of images, which are described to accurately understand the diagrams. This section gives an overview of these images.

1.4.1.2.1 XSD "simple type"

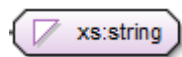


Figure 1: XSD "simple type"

This image represents an XSD "simple type" (in this case a string). Simple types are signed with a left upper corner triangle. A simple type can either be given by the W3C (starting with "xs:" in this document; see also 1.4.2) or can be self-defined. Self-defined simple types can only be restrictions of given W3C simple types, such as making a string an enumeration (only allowing certain words/literals) or restrict a number to a certain range. The kind of restriction is not graphically represented, but can only be read in the XSD file itself.

1.4.1.2.2 XSD "complex type" with no content

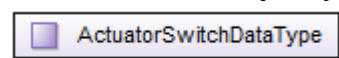


Figure 2: XSD "complex type" with no content

This image represents an XSD "complex type". Complex types are signed with a square. A complex type can consist of different other complex and/or simple types, which are embedded. A complex type might, however, also be completely empty, which is the case shown in this image. An empty "complex type" has no effect if used, other than to be a placeholder for easier extensibility.

1.4.1.2.3 XSD "complex type" with content

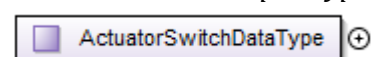


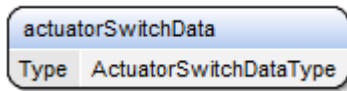
Figure 3: XSD "complex type" with content

This image shows an XSD "complex type" with a content (shown by the "+").

1010

1011

1.4.1.2.4 XSD element with type



1012

1013

Figure 4: XSD element with type

1014 This element represents an XML element “actuatorSwitchData”. The contents of this element are
1015 defined by the type “ActuatorSwitchDataType”. An element can be defined using a type, or can be
1016 directly defined. Using a type has the advantage of being easily extensible.

1017 Please note that figure above may vary. This depends whether the type is empty and whether it is
1018 referenced from a separate XSD.

1019

1020

1.4.1.2.5 XSD sequence



1021

1022

Figure 5: XSD sequence

1023 This image shows an XSD “sequence”. The element connected on the left (not shown in the image) is
1024 the one, which contains this sequence (the “parent”). The elements connected on the right are
1025 contained in that sequence (the “children”). A sequence is an ordered list of elements on the right,
1026 contained in an element connected on the left. (An unordered list is an “all”).

1027

1028

1.4.1.2.6 XSD choice



1029

1030

Figure 6: XSD choice

1031 This image shows an XSD “choice”. The element connected on the left shall contain exactly one of
1032 the elements connected on the right (none of these elements are shown here).

1033

1034

1.4.1.2.7 XSD all



1035

1036

Figure 7: XSD all

1037 This image shows an XSD “all”. The element connected on the left can contain any of the elements
1038 connected on the right (“children”; none of these elements are shown here). This is similar to an XSD
1039 sequence. The difference is that each “child” is implicitly optional (i.e. can be absent in an XML) and
1040 that the order of the children in an XML is arbitrary.

1041

1.4.1.2.8 Optional vs. mandatory

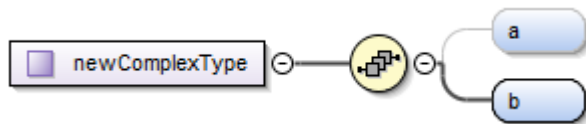


Figure 8: Optional vs. mandatory

There are different styles for the display of connections between building blocks. These styles are used to display whether a building block is optional or mandatory, among others. The image shows a complex type “newComplexType”, consisting of a sequence of an optional element “a” (light grey line) and a mandatory element “b” (dark line).

1.4.1.2.9 Lists / multiplicity

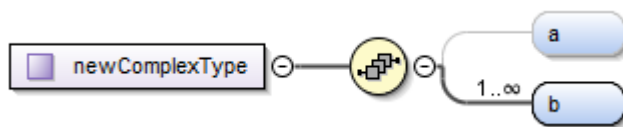


Figure 9: Lists / multiplicity

Elements might also be present multiple times. The shown image shows a complex type “newComplexType”, containing a sequence of an optional element “a”, as well as 1 up to infinite elements “b”.

1.4.1.2.10 Example: “Expanded view”

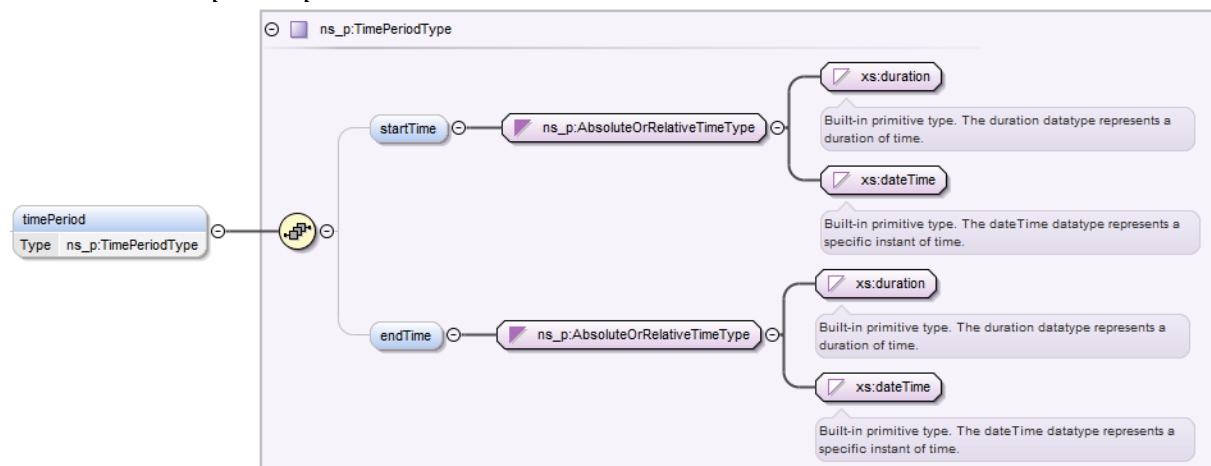


Figure 10: XSD expanded view

This image shows an element called “timePeriod”, which is based upon the Type “TimePeriodType”. Attached to it (on the right, embedded in a purple box) is the definition of the type “TimePeriodType”. The type is a sequence of two optional elements called “startTime” and “endTime”, which are both using the “AbsoluteOrRelativeTimeType” format. There are also informative comments shown in this image, which are embedded in little speech bubbles / balloons.

An XML part conforming to this XSD at least follows the structure of:

```

1066     <timePeriod>
1067         <startTime>2006-05-04T18:13:51.0Z</startTime>
1068         <endTime>2006-05-04T18:13:51.0Z</endTime>
1069     </timePeriod>

```

1070

1071 1.4.2 Namespaces

1072 The SPINE data models make use of many types defined by W3C. These types are referenced within
 1073 the SPINE data models under a so-called namespace of the W3C XML Schema. The SPINE data
 1074 models currently denote this namespace using the prefix "xs:". Thus, a W3C XML Schema type like
 1075 "dateTime" finally leads to the reference "xs:dateTime".

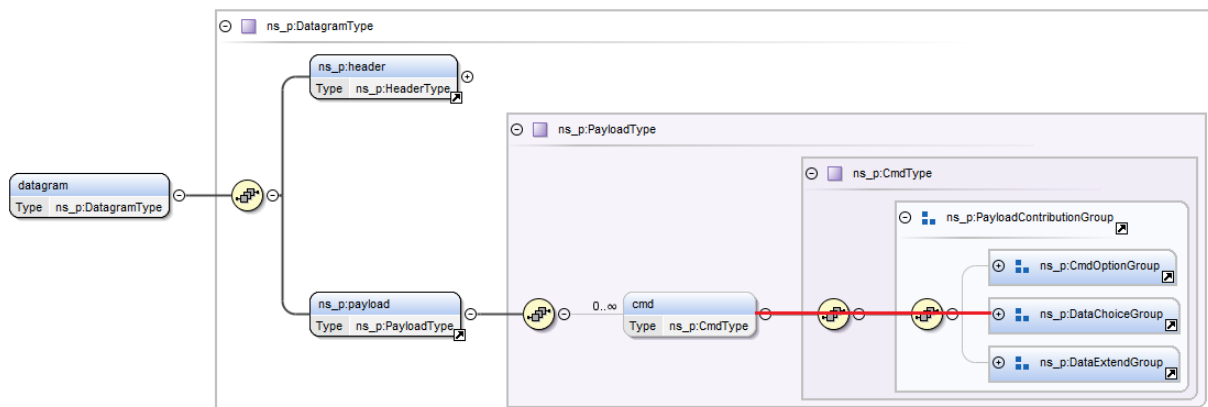
1076 It shall just be noted that the SPINE data models put the definition of their own types and elements
 1077 under namespaces of the EEBus Initiative e.V.

1078

1079 1.4.3 "datagram" vs. specific function – XMLs with different roots and scopes

1080 The SPINE data models provide definitions for different levels/layers. The subsequent figure shows a
 1081 (partly collapsed) graphical representation of the SPINE XML schema definition of "datagram", with a
 1082 specific branch highlighted in red for a subsequent example.

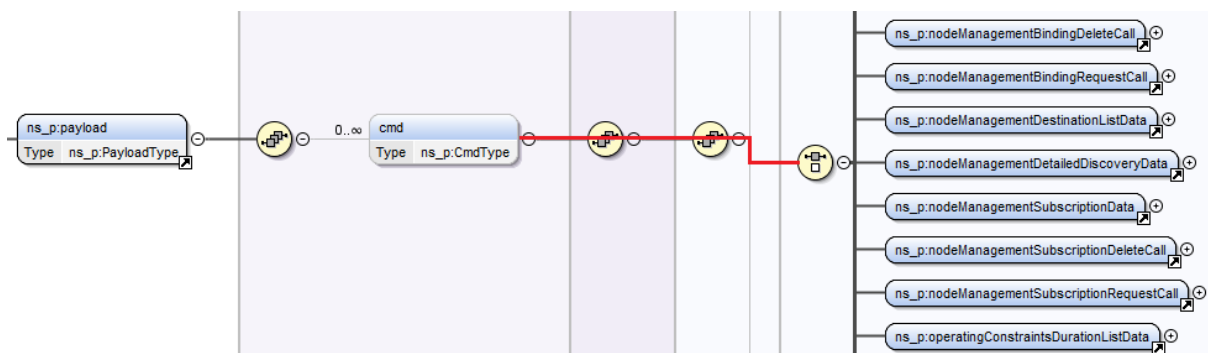
1083



1084

1085 Figure 11: Datagram structure (red line: "way" to the functions)

1086 The next figure shows a graphical excerpt of possible children of the highlighted branch:



1087

1088 Figure 12: Payload structure (red line: "way" to the functions)

1089 These children are also called "functions" within the SPINE specification.

1090 It is possible to create valid XMLs with either "datagram" or any of payload's child elements (like a
1091 concrete function as from the second picture) as root element. It depends on scope and "layer" (in
1092 terms of the ISO-OSI layer model, e.g.) which kind of XML is required.

1093 Throughout this document most explanations of classes will make use of XMLs with specific functions
1094 as root tag. However, it shall be noted that also XMLs with "datagram" as root tag will be discussed
1095 and that different kinds of "payload" contents will be explained.

1096

1097 **1.4.4 Simplification of XML examples**

1098 Supposed the SPINE data model of "datagram" is available for download from
1099 "http://docs.eebus.org/spine/xsd/v1/EEBus_SPINE_TS_Datagram.xsd". The beginning of a valid
1100 SPINE XML with a root tag ("datagram" in the subsequent example) could look like this:

```
1101 <?xml version="1.0" encoding="UTF-8"?>
1102 <datagram xmlns="http://docs.eebus.org/spine/xsd/v1"
1103   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1104   xsi:schemaLocation="http://docs.eebus.org/spine/xsd/v1
1105 EEBus_SPINE_TS_Datagram.xsd">
1106 ...
```

1107 The schemaLocation information of the SPINE data models is not finally decided yet, is not always
1108 mandatory within an XML (see W3C definitions for details) and is not required for the explanation of
1109 the SPINE data models. Hence, it is usually left out throughout this document:

```
1110 <?xml version="1.0" encoding="UTF-8"?>
1111 <datagram xmlns="http://docs.eebus.org/spine/xsd/v1">
1112 ...
```

1113 The W3C recommendation does not oblige the XML declaration which contains the XML version
1114 information, but it clearly recommends its use. However, this part is not of interest for the
1115 presentation of examples.

1116 The XML namespace is omitted within this document, too, as there is no need to mention it in all
1117 examples.

1118 Thus, the example is further simplified:

```
1119 <datagram>
1120 ...
```

1121 As most examples in this document are specific for a single function, the information of the header is
1122 not of interest and therefore not included in the examples. The <datagram> declaration as well as
1123 the subjacent element <payload> is not needed either. Most examples begin with the function itself,
1124 or with the <function> element, if "restricted function exchange" (with "<filter>") is used before the
1125 actual function.

1126 Please see [ProtocolSpecification] for more information about the structure of SPINE XMLs.

1127

1.4.5 Used requirement keywords

The following keywords are used:

- SHALL
- SHALL NOT
- SHOULD
- SHOULD NOT
- MAY

They apply only if written in capital letters!

For the meaning of the keywords, please refer to [RFC2119].

1.4.6 Presence indications in Feature Types

If no presence indication keyword is stated for an element, the keyword "MAY" SHALL be applied (in other words: if not stated otherwise, an element is optional).

1.4.7 Presence indications in SmartEnergyManagementPs

The following abbreviations on the presence of elements or the support of entities or features are used:

- M = mandatory
- R = recommended
- O = optional
- C = "choice", i.e. a presence depends also on the selection from multiple possibilities

meaning that a functionality SHALL be supported (mandatory), SHOULD be supported (recommended) or MAY be supported (optional). If "choice" is used, the element description needs to be considered to find out which alternative applies.

These abbreviations are used exactly the same way as stated in the section "Element presence indications" of the document [ProtocolSpecification]. The following information is a brief excerpt from this document:

In case of elements (of data model definitions), the presence indications "M" and "C" are always meant relative to the respective parent element. I.e. if a parent element is optional ("O") and a child is mandatory ("M") the child element can only be present if the parent element is present as well. The presence indications from the data model definitions describe general requirements on the presence of elements. These requirements can be strengthened (but not weakened) by process rules.

For more information and examples please consider the above mentioned document. In general, the design concept of section "Data model specialization: ..." of the document [ProtocolSpecification] applies here as well.

2 Overall model hierarchy concept overview

The following figure gives an overview on the miscellaneous levels that contribute to the definition and use of interoperable functionality.

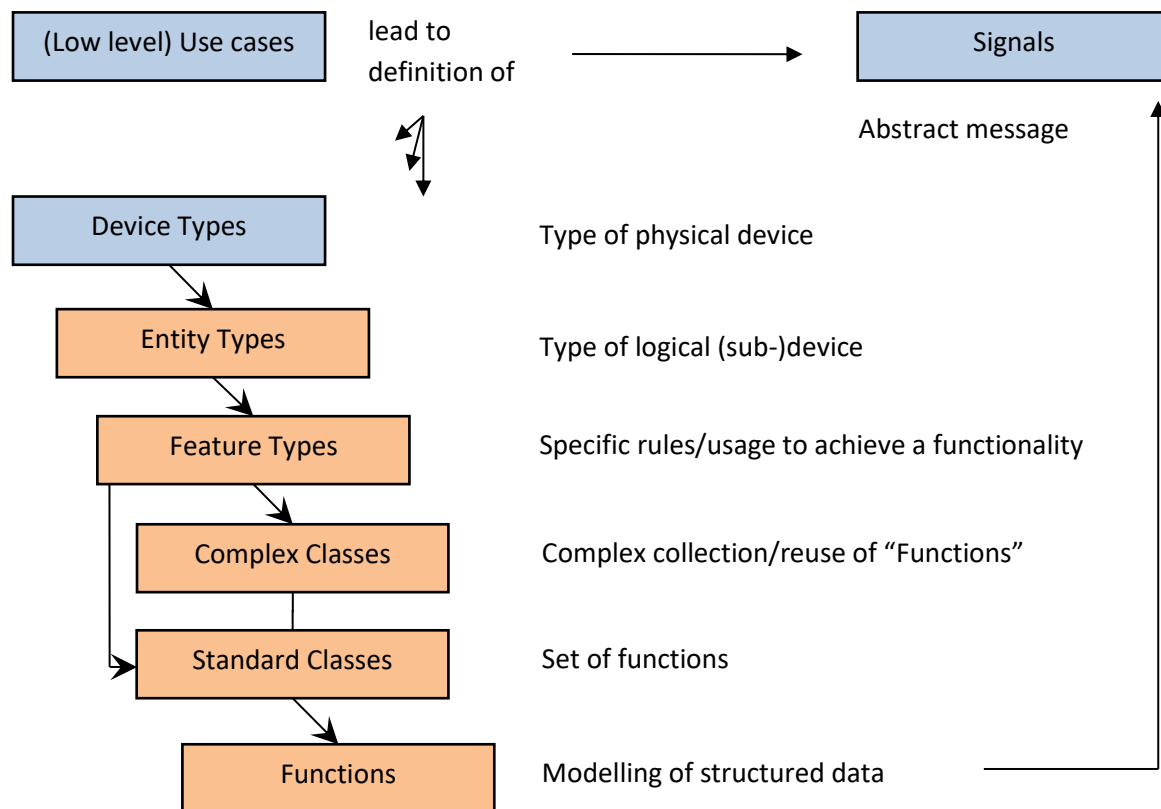


Figure 13: Dependencies of the standardised types and identifiers

Apart from user stories, the most abstract level deals with use cases and abstract signals exchanged between actors. Starting from the top of Figure 13, each level defines its own set of definitions for its own hierarchy of the problem. The device facets “device type”, “entity type”, and “feature type” define logical structures of a device and rules to achieve a required functionality. These facets are introduced in more detail in section 2.1.2. The required functionality makes use of well-defined data structures (so-called “functions”) which are organised in so-called classes. The class concept is explained in section 2.2. The data within the functions is expressed with so-called “elements”. The rules and the functions together permit the modelling of signals. This is used in document [ProtocolSpecification] to model a neutral message framework.

2.1 Resource Type Definition Concept

2.1.1 Introduction

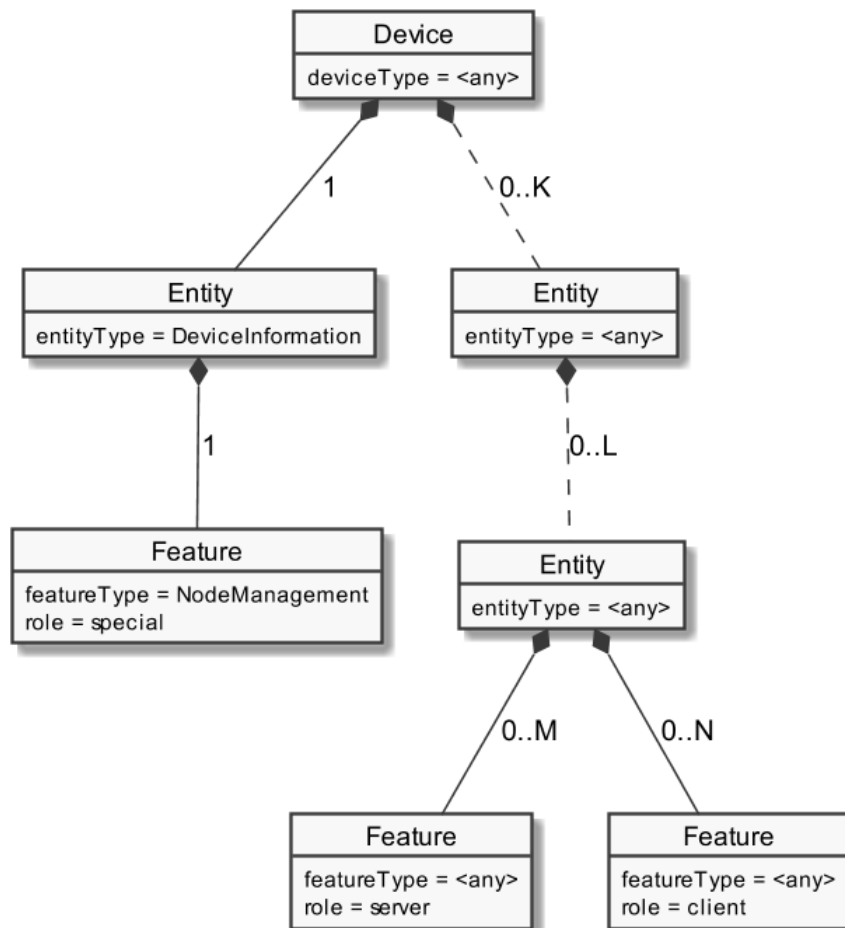
The SPINE device model consists of three layers

- Device
- Entity (with sub-Entities)
- Feature

1186 where the following applies:

- 1187 - 1 Device has 1..K Entities
- 1188 - 1 Entity has 0..L sub-Entities
- 1189 - 1 Entity has 1 parent (Device or Entity)
- 1190 - 1 Entity has 0..M Features
- 1191 - 1 Feature has 1 parent (Entity).

1192 These relations can be combined. The following figure gives an example for this:



1193

1194 *Figure 14: SPINE device model example*

1195 The dotted line in the figure indicates that further Entities MAY be present between the depicted
1196 instances.

1197 In the following sub-sections these components are described more in detail.

1198

1199 2.1.2 Device Model Facets

1200 2.1.2.1 Device Type

1201 The description of the (physical) device (that can be seen as kind of a root instance within the SPINE
1202 device model) includes a short description of the device. As the Device Type is only of informative
1203 character, no further information is needed.

Vendors may use official Device Types as defined in section 4.1 or create own Device Types that then have to be marked as vendor specific ones (see section 2.1.3). Any kind of Device Type may contain any combination of Entity Types (official or vendor specific ones).

1207

1208 **2.1.2.2 Entity Type**

In contrast to the only informative Device Type, the Entity Type is more important for the interoperable working of SPINE. Within the Use Case definitions, Entity Types are mapped to the Use Case Actor's functionality. Hence, a discovery of the correct Entity Type (and its path, see below) may be of relevance.

As already denoted in sub-section 2.1.1, Entities may be nested, meaning an Entity may have other Entities as child. This can lead to a tree structure of Entities. This hierarchy can be retrieved from the detailed discovery (see [ProtocolSpecification] for details).

1216

1217 **2.1.2.3 Feature Type**

Feature Types define optional or mandatory rules for the underlying Class (mostly with the same name, e.g. the Feature Type Measurement is based on the underlying Class Measurement), their functions and the elements and values within the functions. As most rules must be deriveable from the elements and values presented on the SPINE Features, the corresponding rules are provided for each element and value within the tables of the "Feature Types" section.

1223

1224 **2.1.3 Vendor specific extensions**

The Device-/Entity-/Feature-Type structure mentioned in the sections above relies on definitions provided by the EEBus Initiative e.V. for ensuring interoperability. If a vendor needs type definitions that are not already provided, a vendor specific extension allows the definition of new types that add further functionality. These extensions are completely vendor specific and will not be supported by the EEBus Initiative e.V. at all. A vendor may bring these definitions into the EEBUS working-groups and, if of interest, they potentially could become part of a future SPINE release.

These types defined by the vendors themselves SHALL be marked as vendor specific as long as they are not part of an official SPINE release. For details please see [ProtocolSpecification], section "Rules for vendor specific extensions".

1234

1235 **2.1.4 Feature Group**

Some features may need another feature to fulfil a functionality in a sensible way. E.g. the Setpoint Class should be linked to some Measurement instance, so the desired value (from Setpoint) can be compared to the actual one (Measurement). Therefore, feature groups can be used. They are always entity-wide defined. Two entities may define the same feature group for a different meaning. If two or more features shall be linked, the same feature group is assigned to them (e.g. "#1"). More than one feature group may be defined for one feature (e.g. if a Measurement instance is linked to a Setpoint instance as well as to a SmartEnergyManagementPs feature). If the features that reference

each other are only defined once on an entity, no feature group is needed (picking up the previous example: if there is only one Measurement and one Setpoint instance on an entity, no ambiguity is given). If, without the feature groups, there would be an ambiguity, feature groups SHALL be used to prevent the ambiguity and identify the associated features.

Examples:

- #1
- #1#2
- #2#5

2.1.5 Specific usage

This functionality is deprecated.

2.2 Class Concept

2.2.1 Standard Classes

The SPINE data model relies on the standard classes (like ActuatorSwitch, Measurement, etc.). They define the functions with all their elements, some examples and a general description what can be communicated with this functionality. No (or just very few) rules are given in the class sections as the definition of rules is the scope of the device model (see section 2.1.2).

Please note that some classes (or at least function type definitions) are also used in so-called complex classes (see section 2.2.2). In fact, for some classes there may even be no standard Feature Type defined if they are only used within a complex class. In this version of the specification this is the case for the classes

- BindingManagement
- SubscriptionManagement
- Version

as they are primarily re-used within the complex class NodeManagement.

2.2.2 Complex Classes

Complex classes combine one or more functionalities from standard classes in new functions that are treated the same way as functions from standard classes. They do not define new elements as leaves (in a tree structure), but only order the standard class functions in another way. Lists may be added and (leave) elements may be deleted.

2.2.3 SPINE class hierarchy (Class / Sub-Class / Function-Group / Function / Element)

To identify different levels of SPINE data-structures, the following naming conventions are defined:

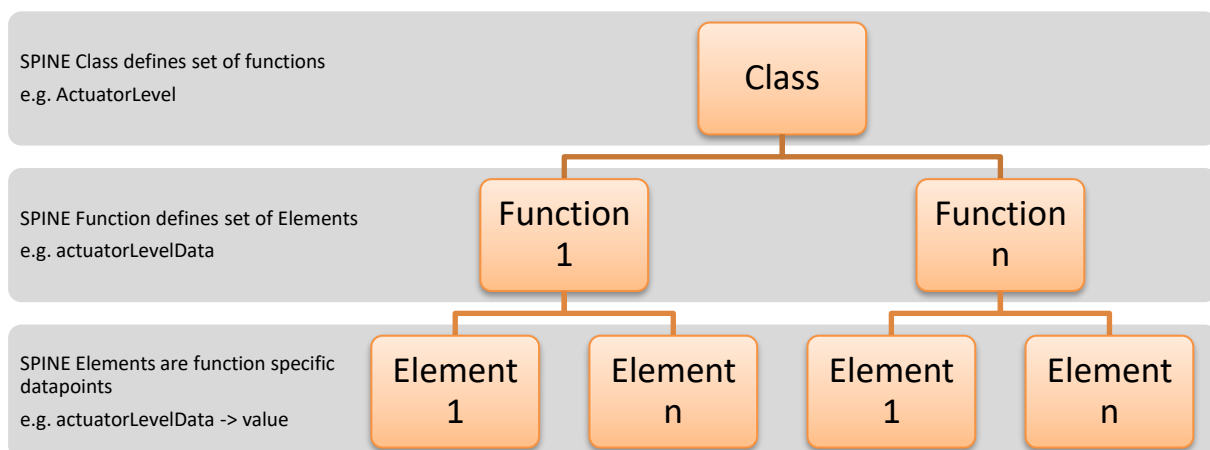
1278 **Class:** A SPINE class is the highest level of data-grouping. E.g. all functions relating to
 1279 *PowerSequences* are described within the *PowerSequences* class. The first letter of the SPINE class
 1280 name is always capitalized.

1281 **Sub-Class:** Classes may have sub-classes. They are introduced with a short sub-section in the
 1282 corresponding class description. E.g. the *PowerSequences* class contains the sub-classes
 1283 *PowerTimeSlot*, *PowerSequence* and *PowerSequenceNode*. These are not marked as own groups in
 1284 the XSD. The sub-classes are not used in any identifier and are not relevant for interoperability
 1285 considerations. The first letter of the SPINE sub-class name is always capitalized.

1286 **Function-Group:** A function-group consists of the function itself (xListData or xData) and (in case of a
 1287 ListData) its corresponding xListDataSelectors. The first letter of the SPINE functions-group name is
 1288 always lowercase.

1289 **Function:** The resources that are used for communication are called functions (e.g.
 1290 *powerTimeSlotValueListData*). The first letter of the SPINE function name is always lowercase.

1291 **Element:** Elements contain the actual data information (e.g. value, timestamp, etc.). The first letter
 1292 of the element name is always lowercase.



1293
 1294 *Figure 15: SPINE class hierarchy*

1295

3 Common technical details

3.1 Introduction

The subsequent sections define and discuss miscellaneous technical details that are required by classes.

3.2 Time information (absolute / relative / recurring)

Nearly all time information is of type: *AbsoluteOrRelativeTimeType* (see below).

Exceptions:

- *timeInformationData* -> utc

- *timePrecisionData* -> lastSyncAt

The simple type *AbsoluteOrRelativeTimeType* is a union of *xs:dateTime* and *xs:duration*. So it is possible to either model absolute times (*xs:dateTime* format, e.g. 2014-04-28T17:43:14.0Z) or relative time information (*xs:duration* format, e.g. P2Y8M15DT7H32M17S).

Unless specified otherwise, an *xs:duration* based time of *AbsoluteOrRelativeTimeType* always relates to "now".

Absolute times SHALL always be stated as UTC-Time ("Z" at the end of the time information).

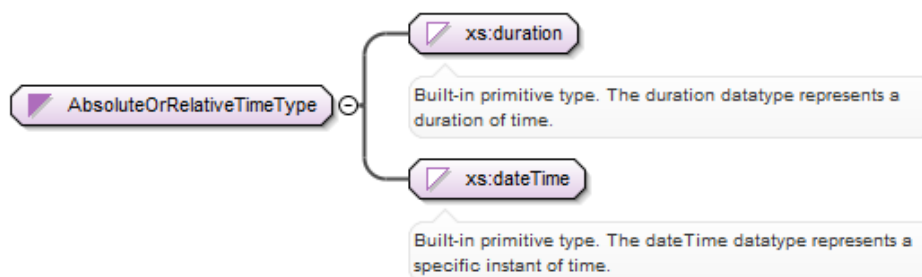


Figure 16: Absolute or relative time type

For recurring times, another type (*AbsoluteOrRecurringTimeType*, see section 3.10.2.13) is defined.

3.3 List Data concept

Some SPINE classes provide lists of data instances. These lists always consist of 0 (hence, may be empty) to unlimited count of single data instances (and nothing else). A function can either contain a (non-list) data as resource or the list data. If a list data is defined in a SPINE class, only this list data may be used as resource, not its non-list data (which is only used as reference in the list data).

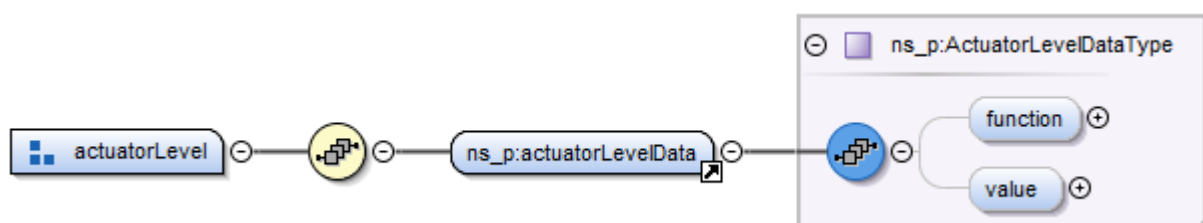


Figure 17: Non-list data function example



Figure 18: List data function example

3.4 Identifier concept

3.4.1 Introduction

Identifiers are used esp. in ListData-functions to identify single list entries unambiguously. They are also used for references within the class itself (to relate the static and non-static information) as well as for references from other classes (e.g. the Setpoint class (see section 5.3.21) references the Measurement class (see section 5.3.15), to announce for which actual measurand a specific setpoint is used for). References are only possible within the same entity, not between different entities or devices.

A special kind of identifiers are the address constituents “device”, “entity”, and “feature” (the types are defined in section 3.10.3). Among others, the element “entity” can occur multiple time (i.e. as list) within some address definitions. More details are explained in section 5.3.17.1.1.

Note: For some functions or use cases a specific data instance is only uniquely identified with the use of multiple identifiers (e.g. within the Power Sequences class a “slot” with number 2 of “sequence” 5 is different to “slot” 2 of “sequence” 7; this typically requires the use of sequenceId AND slotNumber in order to identify a specific slot). In some cases a data instance may carry all these identifiers "at the same place" (in the data model next or close to each other, e.g.). In other cases the identifiers may be organized rather hierarchically in a data structure (this finally depends on the chosen structure of a function; it is esp. true for so-called “complex classes”, if a “parent” identifier is placed closer to the root of a function’s data structure than a “child” identifier).

If not stated otherwise in the Feature Type, the order given in the XSD structure denotes the hierarchical order of the identifiers. An overview of all identifiers can be found in section B.7.

Feature Types define which kind of Identifier is associated to a particular element (see section 3.4.2.2).

3.4.2 Identifier rules

3.4.2.1 General

All primary identifiers SHALL be set if not stated otherwise in the feature type. Non-primary identifiers SHOULD be set, if available and if not stated otherwise in the feature type.

Primary identifiers are those that are used to reference the classes functionality within the Class itself as well as from other features (e.g. `measurementId` within the Measurement class, referenced from `ElectricalConnection`) and are unique within the feature.

3.4.2.2 Identifier keywords

In the Feature Types the following keywords are used for identifiers:

- PRIMARY IDENTIFIER: A primary identifier is unique for the whole Feature. Primary identifiers are used only as "primary" or "top-level" identifier of a Function.. MAY be combined with further SUB IDENTIFIERS if needed to identify a list entry uniquely (for lists with more than one dimension).
Note: A Feature Type may define two or more PRIMARY IDENTIFIERS. In this case the Feature Type describes a relation between the PRIMARY IDENTIFIERS.
- SUB IDENTIFIER: One or more SUB IDENTIFIERS MAY be used within a function to identify the further dimensions of list entries (if the list has more than one dimension (e.g. `parameterId` within `electricalConnectionParameterDescriptionListData`)).
- FOREIGN IDENTIFIER: Used to refer to other functionality on the same entity. It is not used to create further dimensions of list entries.
Usually, a FOREIGN IDENTIFIER contains a value that is also set as PRIMARY IDENTIFIER in another Function. This is the most common relation between list entries of different Features. However, there can also be Use Cases that make use of FOREIGN IDENTIFIERS only, i.e. without an explicit instance of a list entry where a given FOREIGN IDENTIFIER value is set as PRIMARY IDENTIFIER. Such list entries with the same FOREIGN IDENTIFIER values express a relationship between these list entries as well.

The according rules only apply if the identifier keyword is written in capital letters.

3.4.2.3 Order of identifiers

The denoted order of the identifiers in the function description within the Feature Types determines the order of the identifiers.

There might be a variance of that in some classes where a SUB IDENTIFIER appears before a PRIMARY IDENTIFIER in the data model. In this case, the order is

- PRIMATY IDENTIFIER
- SUB IDENTIFIER
- [eventually further SUB IDENTIFIERS]

It MAY be allowed to skip identifiers (e.g. if a Feature Type defines three identifiers, the one in the middle MAY be omitted in some cases). If so, it is defined within the Feature Type.

3.4.2.4 timestamp identifier rules

If timestamp is used as identifier, some special rules need to be considered:

- If used as part of a notification or reply, it may be of type `xs:duration` or `xs:dateTime`.

- If used as part of a write operation:
 - If the time in timestamp shall express "now", this can be modelled in two ways:
 - Using *xs:duration*: duration of 0 seconds (e.g. PT0S)
 - Using *xs:dateTime*: *the current time and date*
 - Due to the problems that may occur when transforming the different time representations, only relative time representation SHALL be used for write operations related to "now" where timestamp is an identifier!
 - For times others than "now", the client SHALL only use time representations (absolute or relative) supported by the server. The support currently may only be discovered by analyzing the messages sent by the server.
 - For history values that are writeable, only absolute time representation SHOULD be used!
- Normally a "binary comparison" of the identifier with the ones stored on the own device finds the appropriate data. But as timestamp can be modelled using absolute or relative time representation, a more complex comparison has to be chosen!
- The feature type may define further rules (i.e. restrictions) on the use of timestamp. This may affect its use as identifier as well as a restriction of its value range (e.g. if only *xs:duration* shall be used but not *xs:dateTime*).

3.4.2.5 Optional identifiers with default values

Some identifiers in some Feature Types (e.g. "billId" in the Feature Type "Bill") are defined with a default value and can be omitted in a message if the feature contains data of at maximum one identifier value of the respective list. In this case, default values for omitted identifiers apply. However, there are typical pitfalls implementers need to consider:

1. In general, identifiers SHALL be evaluated if they are present! Esp. extensions with new use cases/applications can easily lead to a situation where a formerly absent identifier element suddenly is set because there are now two or more identifier values used!
2. As soon as an Identifier is set within one list entry, all list entries SHALL set the according Identifier.
3. As mentioned above, default values are defined for some identifiers. This must be considered properly:
 - a. We assume there is a Feature "A" with a Function that omits a particular PRIMARY or SUB IDENTIFIER in its messages as the feature server applies internally the default value for the list entries. Now we consider an other Feature "B" with a Function that contains a FOREIGN IDENTIFIER to refer to respective list entries of Feature "A"'s Function with the omitted identifier: In this case the FOREIGN IDENTIFIER in "B" must be set to this default value.
 - b. The rule in item 2 above applies regardless of the value of the identifier. However, implementations should in particular consider cases briefly mentioned above, where suddenly two identifier values are used: In addition to the list elements with the default identifier value (usually "1") there are now also list elements with a new identifier value ("8", e.g.). In this case also for the formerly omitted Identifiers the corresponding list entries SHALL then contain the identifier explicitly.

1434 A typical implementation error is to ignore the possibility of multiple identifier values and the
1435 implementation of wrong default values. In this case, client features may not be able to analyze
1436 Function (data) changes properly. This can especially occur for the case described above in item 1.

1437 Generally, it is recommended to never omit identifiers if they are set to the default value, if possible,
1438 as it is more explicit and less prone to errors.

1439

1440 **3.4.2.6 Optional identifiers without default values**

1441 A Feature Type may also define an optional Identifier that has no default value. The following rules
1442 apply:

1443 1. If the Identifier is omitted:

1444 In this case the Identifier SHALL be omitted in all list entries. This means it is not possible then to
1445 submit or refer to only particular entries of this list.

1446 2. If the Identifier is present:

1447 In this case the Identifier SHALL be present in all list entries. As usual, the Identifier is then used
1448 to identify particular list entries.

1449 A typical implementation error is to ignore that a given list with such an Identifier may sometimes
1450 omit the Identifier and sometimes may have it set. Switching from one list type to the other always
1451 denotes a complete replacement of the former list by the new one.

1452

1453 **3.5 Multidimensional data representation with "flat" list-based data model**

1454 Many applications need to work with multidimensional data. The functions of SPINE standard classes,
1455 on the other hand, usually are modelled as lists (i.e. one-dimensional objects) of rather flat data
1456 structures. However, this list concept is comparable to common table-based databases (SQL, e.g.)
1457 that are also frequently used for the management of multidimensional data. As there are similar
1458 concepts, most of the SPINE standard classes can as well be used for the modelling of
1459 multidimensional data. This section gives a brief example.

1460 For this example we will use an excerpt of the function "measurementListData" of the SPINE
1461 Measurement class. It can take a list of "measurementData" elements, each comprising of the
1462 elements

- 1463 • measurementId
- 1464 • valueType
- 1465 • timestamp
- 1466 • value
- 1467 • evaluationPeriod
- 1468 • and some more elements

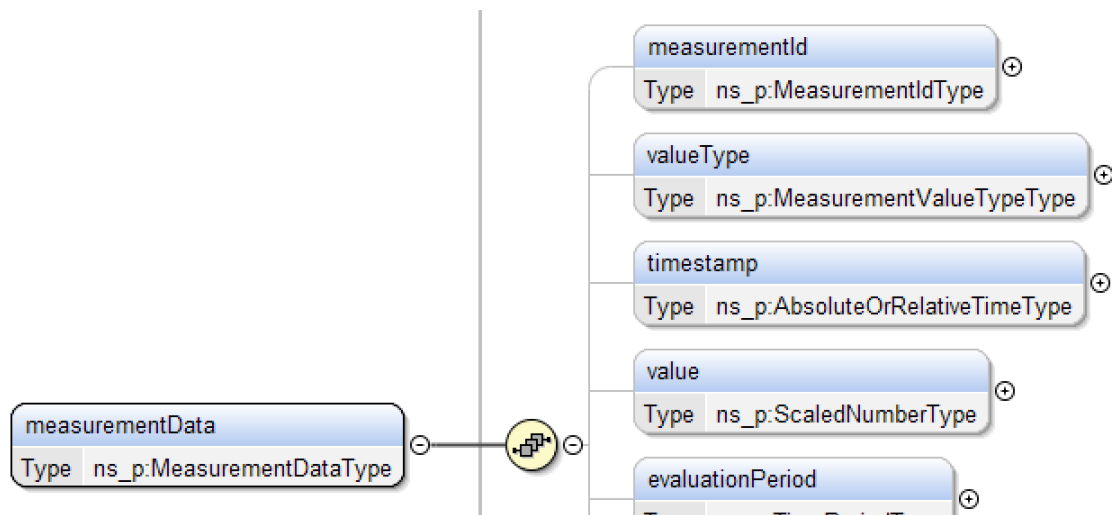


Figure 19: Excerpt of the list element "measurementData" of the SPINE function measurementListData

In the Measurement class definition, the first three elements "measurementId", "valueType", and "timestamp" are marked as so-called "identifiers". This means a particular measurementData instance is identified by these elements (just to give a comparison: In table-based databases such kinds of elements are frequently called "keys"). Mathematically, an identifier can be seen as the axe of a single dimension. The combination of distinct identifiers then creates a multidimensional space. At each position of this space the remaining elements ("value", "evaluationPeriod", etc.) can be placed with proper data:

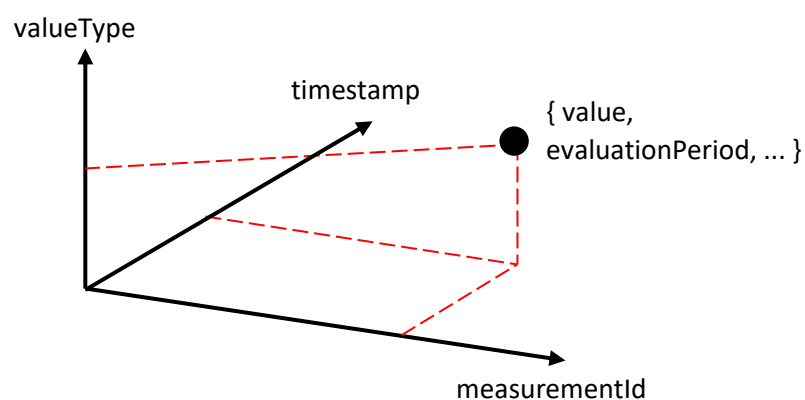


Figure 20: SPINE identifiers can create a multidimensional space of data.

As can be seen in Figure 19, the "position" in the multidimensional space is part of the measurementData structure. Therefore, a list of measurementData instances can build a complete multidimensional space with data. Among others, a particular value of measurementId can then contain data (value, evaluationPeriod, etc.) for multiple valueType values and multiple timestamp values. The following XML shows an example of a list of measurementData instances (for brevity the example just contains the element "value" and skips "evaluationPeriod" and further elements):

```
<measurementListData>
  <measurementData>
    <measurementId>1</measurementId>
    <valueType>value</valueType>
    <timestamp>2018-07-03T17:16:28.0Z</timestamp>
    <value>
```

```

1492         <number>15</number>
1493         <scale>-1</scale>
1494     </value>
1495 </measurementData>
1496 <measurementData>
1497     <measurementId>1</measurementId>
1498     <valueType>value</valueType>
1499     <timestamp>2018-07-03T17:16:33.0Z</timestamp>
1500     <value>
1501         <number>21</number>
1502         <scale>-1</scale>
1503     </value>
1504 </measurementData>
1505 <measurementData>
1506     <measurementId>1</measurementId>
1507     <valueType>minValue</valueType>
1508     <timestamp>2018-07-03T17:16:33.0Z</timestamp>
1509     <value>
1510         <number>1</number>
1511     </value>
1512 </measurementData>
1513 <measurementData>
1514     <measurementId>2</measurementId>
1515     <valueType>value</valueType>
1516     <timestamp>2018-07-03T17:16:33.0Z</timestamp>
1517     <value>
1518         <number>55</number>
1519     </value>
1520 </measurementData>
1521 </measurementListData>

```

Semantically, this example XML models data for two different measurementId values, where

1. measurementId 1 has data for two different valueTypes:
 - a. For valueType "value" there is data available for two different timestamps.
 - b. For valueType "minValue" there is data available for only one timestamp.
2. measurementId 2 just has data for a single valueType = "value" at a particular timestamp.

Please note that SPINE identifiers have a "priority" within a SPINE class (or Feature Type, resp.), as explained in section 3.4. This means that a list-based SPINE function cannot be created in a way that identifiers of lower priority are used without the mandatory higher priority identifiers. With regards to the example above this means it would be invalid to create a measurementData instance where "valueType" is set but "measurementId" is absent. The following figure compares the flat list-based structure from Figure 19 with a virtual arrangement that considers the priorities of the identifiers.

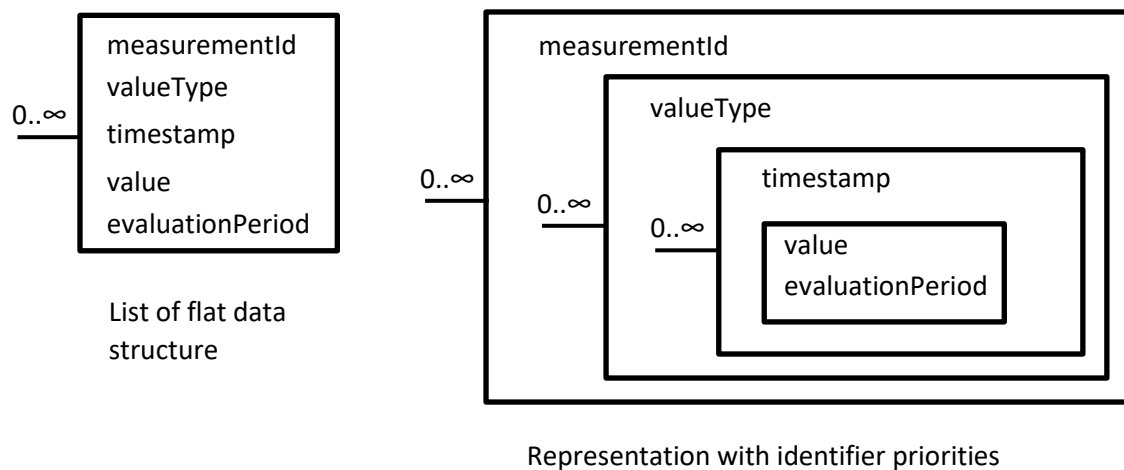


Figure 21: The flat data structure of Figure 19 can also be rearranged (virtually) to emphasize the priorities of the identifiers.

3.6 Restricted function exchange for list based functions

As defined in [ProtocolSpecification], section "Restricted function exchange with cmdOptions", in the communication between server and client not the complete server data has always to be considered but can be restricted considering some given rules as long as it is supported by the server (see [ProtocolSpecification], section "Minimum restricted function exchange support").

Further rules concerning the SPINE resources are defined in this section as default rules for all feature types that make use of SPINE standard class functions:

- If a server feature supports "write" operations with cmdControl "partial": The server feature SHALL accept received "xListData" containing at least one "xData". The server feature MAY accept xListData with more than one xData.
- If a server feature supports "write" operations with cmdControl "delete": The server feature SHALL accept the deletion request (using "<SELECTORS>") for at least a single "xData" of its "xListData". The server feature MAY support the deletion of more than one xData.
- If a server feature supports "read" operations with cmdControl "partial": The server feature SHALL accept the read request (using "<SELECTORS>") for at least a single "xData" of its "xListData". The server feature MAY return more than one xData.
- Complete function exchange SHALL be supported in any case (operations on the complete xListData).

Note: Feature types may still impose more restrictions than the above ones!

3.7 Relations between thematically connected SPINE classes

Some SPINE classes are thematically connected. The relations are indicated by the use of identifiers. To get a better understanding of the relations, the following diagrams visualize them. For details about the classes and the identifiers please consider the corresponding sections in this documentation. Relations are only possible within the same entity, not between different entities or devices.

1562

1563 **3.7.1 Relations between Alarm / ElectricalConnection / HVAC / LoadControl / Measurement /**
1564 **Setpoint / SupplyCondition / TariffInformation / Threshold / TimeTable**

1565 The **Measurement** class is the central functionality of these related classes. It can be used to
1566 measure a lot of different measurands, describe the measurand and model constraints, etc.

1567 If a setpoint shall be set, only the setpoint itself has to be stated in the **Setpoint** class. Via the
1568 *measurementId*, the type of measurand, that a setpoint is set for, is denoted. The period in time, this
1569 setpoint shall be applied can be specified in a time table from the **TimeTable** class. With the
1570 *timeTableId*, the reference is given.

1571 The **HVAC** class references the **Setpoint** class as well as the **TimeTable** class.

1572 If thresholds shall be set - either to activate an alarm if the threshold is exceeded or to do previously
1573 configured actions (implementation specific) - the **Threshold** class is used. The value that shall be
1574 measured to check if it is over- or underrun is the one from the Measurement class. The Threshold
1575 class does not reference the Measurement class, instead, in the Measurement class a relation
1576 function exists that lists all relevant thresholds.

1577 If a threshold is exceeded, the **Alarm** class notifies all subscribed clients. To identify which threshold
1578 was exceeded, the Alarm class references the *thresholdId*.

1579 The **SupplyCondition** class can reference to thresholds, too, to eventually generate events in case of
1580 exceeding some supply concerning limits.

1581 Specifically for electrical measurands, the **ElectricalConnection** class describes all parameters
1582 needed. The values themselves are modelled with the Measurement class.

1583 The **LoadControl** class enables a client to set limits on the server instance. To model further
1584 information about the measurand that is limited, the Measurement class is used.

1585 The **TariffInformation** class references the **Measurement** class as well as the **TimeTable** class to
1586 model simple or complex tariff related functionality.

1587

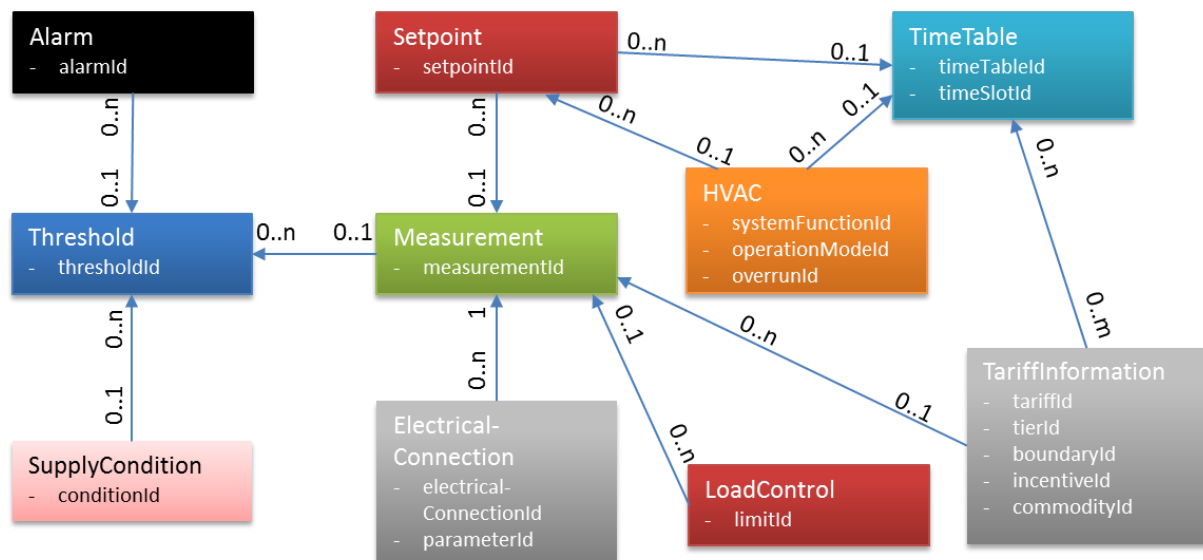


Figure 22: Relations between Alarm / ElectricalConnection / HVAC / LoadControl / Measurement / Setpoint / SupplyCondition / TariffInformation / Threshold / TimeTable

3.7.2 Relations between DirectControl / HVAC / LoadControl / OperatingConstraints / PowerSequences (SmartEnergyManagementPs) / TaskManagement

The **PowerSequences** (or the **SmartEnergyManagementPs** complex) class can be used to model sequences and slots, characterising a specific demand of energy that may be shifted in some range.

The **OperatingConstraints** class adds constraints to a power sequence to avoid misconfigurations of a device and models resume implications like additional energy or costs generated by pausing and then resuming some functionality. When using DirectControl (see below), the OperatingConstraints SHALL be considered, too, if linked with the same feature group (see section 2.1.4).

Pausing and reactivating devices can be done with the **DirectControl** class, which can relate to some power sequence that shall be controlled in a direct manner.

The **HVAC** class may reference power sequences that characterize specific functionality in more detail.

Another way the demand of a functionality may be affected is by some more or less external instruction (e.g. by the DSO) via the **LoadControl** class.

The **TaskManagement** class lists all the jobs from these classes (except for OperatingConstraints) and gives the needed relation in an extra function.

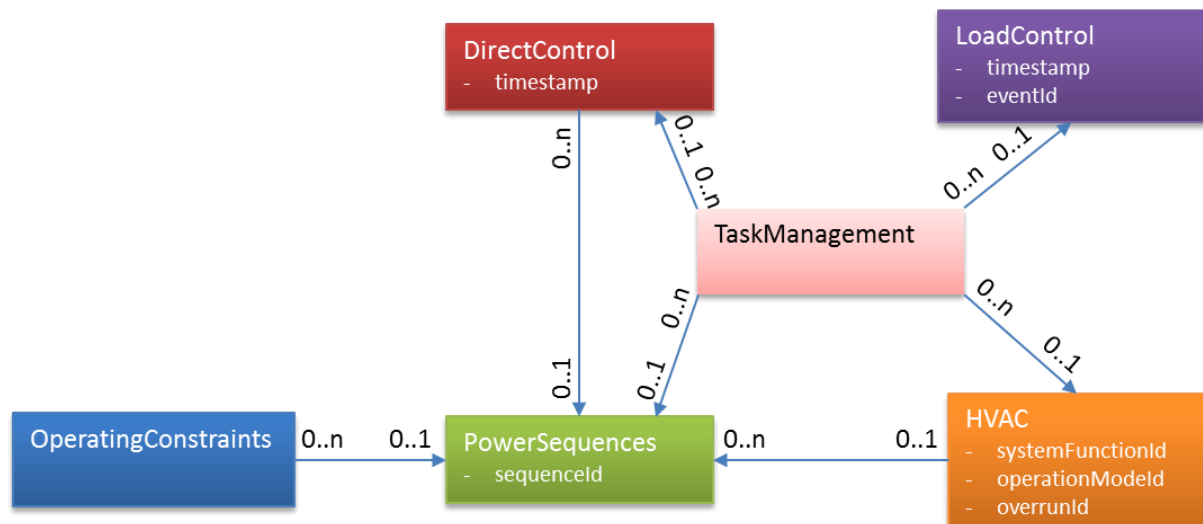


Figure 23: Relations between DirectControl / HVAC / LoadControl / OperatingConstraints / PowerSequences (SmartEnergyManagementPs) / TaskManagement

3.8 On the use of "label" and "description"

In most functions with rather static information, elements called "label" and "description" are defined. These two elements MAY contain human readable strings (they are always optional) that SHALL NOT be used for automatisms (there are other elements for this purpose).

In general, the label element should contain a very short string which could be displayed below an icon, e.g. The description may contain a longer string that is only shown if the user clicks on a corresponding button or similar.

The following examples show how the strings can be used:

label: ActuatorLevel TestDevice
description: Device for testing SPINE compliance.

label: SmartPlug_1
description: SmartPlug Test Installation 01.04.2012

label: mains connection
description: 1-phase mains connection of the device

label: battery connection
description: Backup battery connection of the device

label: TempSensor1
description: Temperature Sensor 1 (Bedroom)

3.9 Empty elements as "tags"

In some cases elements shall only be used as "tags" without any content as some kind of boolean information (element is present = true, element is not present = false). This can be achieved by assigning the element a certain complex type that has no content (see section 3.10.1.1).

An example is the *DaysOfWeekType* (see section 3.10.2.10) used (e.g.) for the element *daysOfWeek* within the *start-* and *endTime* of the *timeTableListData* function (see section 5.3.29.2):

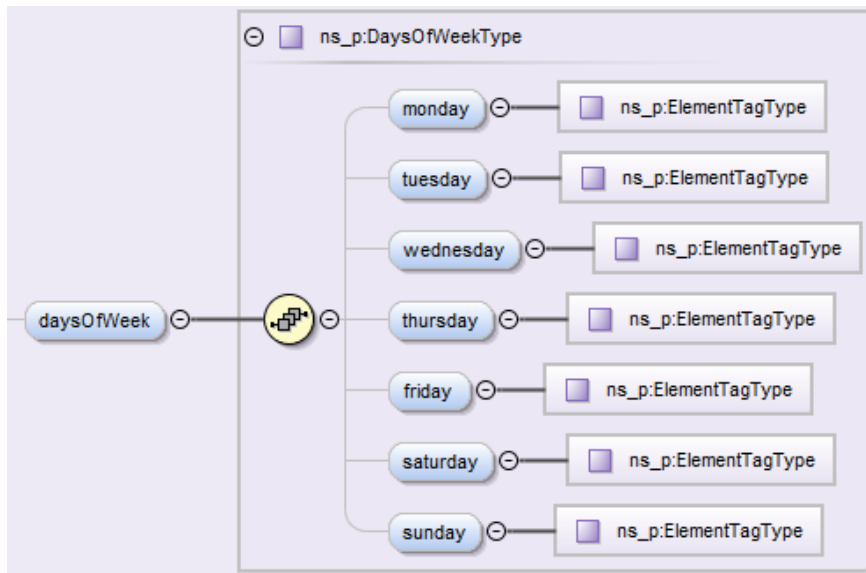


Figure 24: Empty elements example

An XML where (e.g.) the elements *monday* and *wednesday* are present would look like this:

```

...
<daysOfWeek>
  <monday/>
  <wednesday/>
</daysOfWeek>
...

```

In future versions of the SPINE specification, the type of an empty element may change and hence may include child elements, e.g. Please note that compatibility requirements and concepts always consider the content of a message (a concrete XML document, e.g.) and not the type names: The types are defined or used within XSD files, but type names are not explicitly mentioned in XML documents in this specification. Consequently, a subsequent version of the specification may well define the same (or properly extended) messages as a former version, but with partially renamed types as long as document/message compatibility is maintained. This concept permits also to extend formerly "empty" elements with child elements: If an element "foo" is defined as complex type with no further child element in version 1.0.0, then it is well permissive to assign "foo" a different complex type ("NewFooType", e.g.) with child elements in version 1.1.0.

3.10 Common data types

Some types are used by several classes. They are described in this section. For the exact naming and the complete list of all common data types please consider the underlying revision of the SPINE data model.

3.10.1 General

3.10.1.1 *ElementTagType*

Used for elements that have no content (empty tag).

A *complexType*, containing no elements.

3.10.1.1.1 *Default rules for ElementTagType*

If used as type for an element, the element SHALL have no further content in this SPINE version.

Future SPINE versions may add content that SHALL be ignored by devices implementing the SPINE version at hand.

3.10.1.2 *LabelType*

A default type used for all "label" elements.

A *simpleType* (restricted string).

3.10.1.2.1 *Default rules for LabelType*

The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.

3.10.1.3 *DescriptionType*

A default type used for all "description" elements.

A *simpleType* (restricted string).

3.10.1.3.1 *Default rules for DescriptionType*

The string-length SHOULD NOT be longer than 4096 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 4096 characters.

3.10.1.4 *SpecificationVersionType*

The SPINE specification version.

A *simpleType* (restricted string), with the pattern

[1-9][0-9]*\.[0-9]+\.[0-9]+

1692 allowing version strings like "1.0.0", "2.5.3", etc. The maximum string length is limited to 128
1693 characters.

1694

1695 **3.10.1.5 EnumExtendType**

1696 Used for extending enumeration with vendor specific ones.

1697 Some enumerations may be extended by the vendors with own values.

1698 A *simpleType* (restricted string), with the pattern

1699 $_ (i: [1-9] [0-9]^* | n: [a-zA-Z0-9-]^+) _ [^\p{Cc} \p{Cf} \p{Z}]^+$

1700 For details please see section 2.1.3. The maximum string length is limited to 256 characters.

1701

1702 **3.10.1.5.1 Default rules for EnumExtendType**

1703 The values SHALL be marked as vendor specific by adding the IANA PEN code of the vendor ("i:") or
1704 some vendor name ("n:") surrounded by "_".

1705 The IANA PEN SHOULD be used!

1706

1707 **3.10.1.6 NumberType**

1708 Used to represent a numeric (integer) value. See also section 3.10.1.8!

1709 A *simpleType* (xs:long).

1710

1711 **3.10.1.7 ScaleType**

1712 Used to scale a numeric value. See also section 3.10.1.8!

1713 A *simpleType* (xs:short).

1714

1715 **3.10.1.8 ScaledNumberType**

1716 Numeric values (e.g. numerical values, setpoints, thresholds, etc.) use this number representation.

1717 A *complexType*, containing the elements *number* (type: *NumberType*, see section 3.10.1.6) and *scale*
1718 (type: *ScaleType*, see section 3.10.1.7). See Figure 25.

1719

1720 **3.10.1.8.1 Default rules for ScaledNumberType**

1721 Throughout this specification, this type and its elements are used as follows unless stated otherwise:

- 1722 1. This type represents a real number (subsequently called "scaledNumber") that is calculated from
1723 the type's elements as follows:

$$1724 \text{ scaledNumber} = \text{number} * 10^{\text{scale}}$$

1725

- 1726 2. If the element "scale" is omitted a default value of "0" SHALL be applied for "scale". Then, item 1
1727 applies, or (equivalently)

$$1728 \text{ scaledNumber} = \text{number}$$

3. The element "number" SHALL be set.

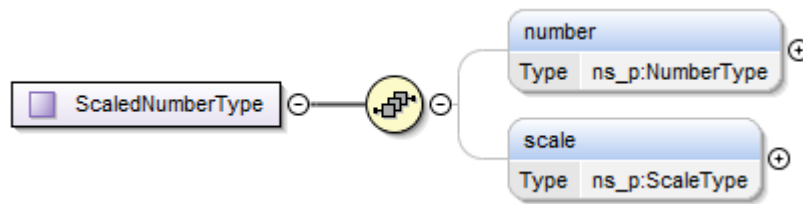


Figure 25: ScaledNumberType

3.10.1.9 ScaledNumberSetType

This type can be used to describe any sets of scaledNumbers. The set can have gaps compared to a simple value range and include many different values and value ranges.

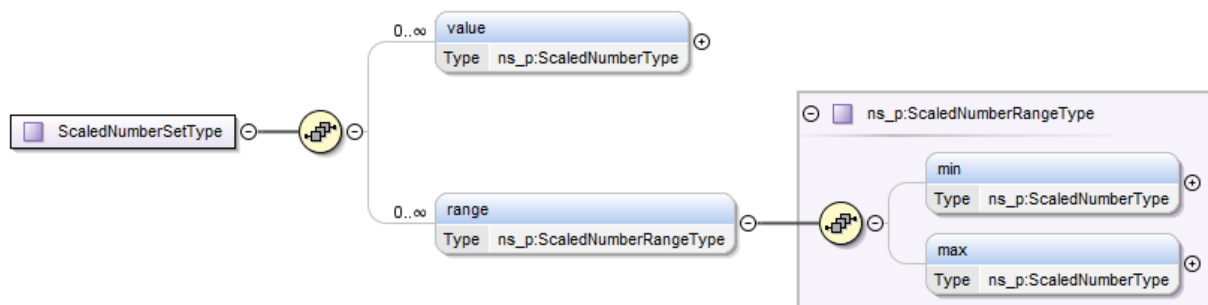


Figure 26: ScaledNumberSetType

Name	Type	Description
value (list)	Common data type "ScaledNumberType". See section 3.10.1.8.	Allows to describe numbers that are part of the set: Each "value" instance contributes its value as single number to the set.
range (list)	Common data type "ScaledNumberRangeType". See section 3.10.1.10.	Allows to describe number ranges that are part of the set. Each "range" instance contributes its value range to the set.

Table 1: ScaledNumberSetType

3.10.1.9.1 Default rules for ScaledNumberSetType

The set MAY have gaps.

3.10.1.10 ScaledNumberRangeType

This type can be used to describe a value range. All ScaledNumbers between min and max are supported in this case, at least if no further constraints apply, e.g. a value step size.

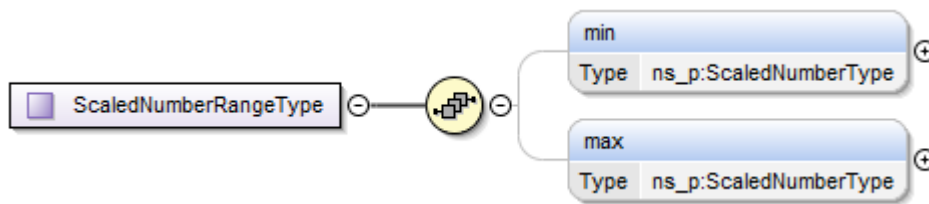


Figure 27: ScaledNumberRangeType

Name	Type	Description
min	Common data type "ScaledNumberType". See section 3.10.1.8.	The min element defines the lower bound of the ScaledNumberRange.
max	Common data type "ScaledNumberType". See section 3.10.1.8.	The max element defines the upper bound of the ScaledNumberRange.

Table 2: ScaledNumberRangeType

3.10.1.11 CommodityTypeType

Union of:

- CommodityTypeEnumType (see section 3.10.1.12)
- EnumExtendType (see section 3.10.1.5)

3.10.1.12 CommodityTypeEnumType

The commodity, used for measurements, pricing, etc.

A *simpleType* (restricted string), containing enumerations for the commodity.

Value	Description
electricity	Electricity (AC or DC).
gas	Gas, as used by a gas heater.
oil	Oil, as used by an oil heater.
water	(Drinking) water.
wasteWater	(Domestic) waste water.
domesticHotWater	Domestic hot (drinking, e.g.) water.
heatingWater	Water circulating through the heating system of a building.
steam	Steam, as used in (e.g.) a steam boiler or within a coal-fired power station.
heat	Heat (e.g. in the form of hot water, hot air, etc.).
coolingLoad	Cooling load used to cool down goods or a room / building.
air	Air, as commonly available outside (no special gas or chemical compound).

Table 3: Enumeration CommodityTypeEnumType

3.10.1.13 EnergyDirectionType

Union of:

- EnergyDirectionEnumType (see section 3.10.1.14)

- EnumExtendType (see section 3.10.1.5)

3.10.1.14 EnergyDirectionEnumType

The energy direction. Can be applied for other measurands than energy, too (such as power or current).

A *simpleType* (restricted string), containing the enumerations *consume* and *produce*.

3.10.1.14.1 Default rules for EnergyDirectionEnumType

If this type is used for an Element, the related data SHALL be interpreted in a way as defined by the rules below.

Value	Rule
consume	Positive values SHALL be interpreted as consumption; negative values SHALL be interpreted as production.
produce	Positive values SHALL be interpreted as production; negative values SHALL be interpreted as consumption.

Table 4: Enumeration EnergyDirectionEnumType value rules

3.10.1.15 EnergyModeType

Union of:

- EnergyModeEnumType (see section 3.10.1.16)

- EnumExtendType (see section 3.10.1.5)

3.10.1.16 EnergyModeEnumType

Different modes for energy demand.

A *simpleType* (restricted string), containing the enumerations for the energy mode.

Value	Description
consume	Used in cases where the functionality consumes energy.
produce	Used in cases where the functionality produces energy.
idle	Used in cases where the functionality neither consumes nor produces energy.
auto	Used in cases where the functionality may consume or produce energy (or none of both), decided on its internal business logic.

Table 5: Enumeration EnergyModeEnumType

3.10.1.16.1 Default rules for EnergyModeEnumType

Value	Rule
consume	The related functionality SHALL only consume energy.

produce	The related functionality SHALL only produce energy.
idle	The related functionality SHALL neither consume or produce energy.
auto	The related functionality MAY consume or produce energy or be in idle-mode based on its own decision.

Table 6: Enumeration EnergyModeEnumType value rules

3.10.1.17 UnitOfMeasurementType

Union of:

- UnitOfMeasurementEnumType (see section 3.10.1.18)

- EnumExtendType (see section 3.10.1.5)

3.10.1.18 UnitOfMeasurementEnumType

Unit, used for measurements, etc.

A *simpleType* (restricted *string*), containing several units.

In general the following format is used to model mathematical operations as strings:

 $x/y \equiv x$ divided by y (i.e. $x \div y$) $x^y \equiv x$ raised to the power of y (i.e. x^y) $x_y \equiv x$ multiplied by y (i.e. $x * y$)**Examples:** $m^3 \equiv m^3$ $m^3/h \equiv (m^3) \div h$ $m/s^2 \equiv m \div (s^2)$ $kg_m^2 \equiv kg * (m^2)$

Unit Name	Unit Description
unknown	Unknown unit
1	1 (unitless)
m	metre (SI unit, used for length)
kg	kilogram (SI unit, used for mass)
s	second (SI unit, used for time)
A	ampere (SI unit, used for electric current)
K	kelvin (SI unit, used for temperature)
mol	mole (SI unit, used for the amount of a substance)
cd	candela (SI unit, used for luminous intensity)
V	volt (used for voltage)
W	watt (used for power)
Wh	watt hour (used for energy)
VA	volt-ampere (used for apparent power)
VAh	volt-ampere hour (used for apparent energy)
var	volt-ampere reactive (used for reactive power)
varh	volt-ampere-reactive hours (used for reactive energy)
degC	degree Celsius (°C, used for temperature)
degF	degree Fahrenheit (°F, used for temperature)

Lm	lumen (used for total quantity of visible light emitted by a source)
lx	lux (used for illuminance and luminous emittance, measuring luminous flux per unit area)
Ohm	ohm (used for electrical resistance)
Hz	hertz (used for frequency)
dB	decibel (used to express the ratio of one value of a physical property to another on a logarithmic scale)
dBm	decibel with a reference value of one milliwatt
pct	percentage
ppm	parts-per-million
l	litre (used for volume)
l/s	litre per second (used for flow rate)
l/h	litre per hour (used for flow rate)
deg	degree of arc (used for planar angle; a full rotation is 360 degrees)
rad	radian (used for planar angle; a full rotation is 2π)
rad/s	radian per second (used for rotational speed (angular velocity))
sr	steradian or square radian (used for solid angles in three-dimensional geometry)
Gy	gray (used for ionizing radiation dose)
Bq	becquerel (used for radioactivity)
Bq/m ³	becquerel per cubic metre (used for specific radioactive activity in air and gases)
Sv	sievert (used for ionizing radiation dose)
Rd	rutherford (used for radioactive decay)
C	coulomb (used for electric charge)
F	farad (used for electrical capacitance)
H	henry (used for electrical inductance)
J	joule (used for energy)
N	newton (used for force)
N_m	newton metre (used for torque (also called <i>moment</i>))
N_s	newton second (used for impulse)
Wb	weber (used for magnetic flux)
T	tesla (used for magnetic flux density)
Pa	pascal (used for pressure)
bar	bar (used for pressure)
atm	standard atmosphere (used for pressure)
psi	pound-force per square inch (used for pressure)
mmHg	millimetre of mercury (used for pressure)
m ²	square metre (used for area)
m ³	cubic metre (used for volume)
m ³ /h	cubic metre per hour (used for (volumetric) flow rate)
m/s	metre per second (used for speed (scalar) and velocity (vector quantity which specifies both magnitude and a specific direction))
m/s ²	metre per second squared (used for acceleration)
m ³ /s	cubic metre per second (used for (volumetric) flow rate)
m/m ³	metre per cubic metre (used for fuel efficiency)
kg/m ³	kilogram per cubic metre (used for density)

kg_m	kilogram metre (used for payload-distance)
m^2/s	square metre per second (used for kinematic viscosity)
W/m_K	watts per meter-kelvin, W/(m·K) (used for thermal conductivity)
J/K	joule per kelvin (used for heat capacity (or thermal capacity) and (thermodynamic) entropy)
1/s	1 per second (used for frequency)
W/m^2	watt per square metre (used for heat flux density, irradiance)
J/m^2	joule per square metre (used for surface tension, stiffness)
S	siemens (used for electric conductance, electric susceptance and electric admittance)
S/m	siemens per metre (used for electrical conductivity)
K/s	kelvin per second (used for heating rate)
Pa/s	pascal per second (used for power density)
J/kg_K	joule per kilogram kelvin, J/(kg·K) (used for specific heat capacity)
Vs	volt-second (used for magnetic flux)
V/m	volt per metre (used for electric field strength)
V/Hz	volt per hertz (used for magnetic flux)
As	ampere second (used for electric charge)
A/m	ampere per metre (used for magnetization, magnetic field strength)
Hz/s	hertz per second (used for frequency drift)
kg/s	kilogram per second (used for mass flow rate)
kg_m^2	kilogram square metre (used for moment of inertia)
J/Wh	joule per watt-hour (used for heat rate)
W/s	watt per second (used for ramp rate)
ft^3	cubic foot (used for volume)
ft^3/h	cubic foot per hour (used for (volumetric) flow rate)
ccf	centum cubic feet = 100 ft ³ (used for volume)
ccf/h	centum cubic feet per hour = 100 ft ³ /h (used for (volumetric) flow rate)
US.liq.gal	US (liquid) gallon (used for (fluid) capacity, volume)
US.liq.gal/h	US (liquid) gallon per hour (used for flow rate)
Imp.gal	imperial (UK) gallon
Imp.gal/h	imperial (UK) gallon per hour (used for flow rate)
Btu	british thermal unit (used for heat)
Btu/h	british thermal unit per hour (used for power)
Ah	ampere hour (used for electric charge)
kg/Wh	kilogram per watt-hour (used for (e.g.) fuel economy or CO2 emission)

Table 7: Enumeration UnitOfMeasurementEnumType

3.10.1.19 CurrencyType

Union of:

- CurrencyEnumType (see section 3.10.1.20)

- EnumExtendType (see section 3.10.1.5)

1816 **3.10.1.20 CurrencyEnumType**

1817 Currency as used by price information, e.g.

1818 A *simpleType* (restricted *string*), containing several currencies (as defined in ISO 4217:2015).

AED	BRL	CZK	HNL	KYD	MXV	RUB	TND	XBA
AFN	BSD	DJF	HRK	KZT	MYR	RWF	TOP	XBB
ALL	BTN	DKK	HTG	LAK	MZN	SAR	TRY	XBC
AMD	BWP	DOP	HUF	LBP	NAD	SBD	TTD	XBD
ANG	BYR	DZD	IDR	LKR	NGN	SCR	TWD	XCD
AOA	BZD	EGP	ILS	LRD	NIO	SDG	TZS	XDR
ARS	CAD	ERN	INR	LSL	NOK	SEK	UAH	XOF
AUD	CDF	ETB	IQD	LYD	NPR	SGD	UGX	XPD
AWG	CHE	EUR	IRR	MAD	NZD	SHP	USD	XPF
AZN	CHF	FJD	ISK	MDL	OMR	SLL	USN	XPT
BAM	CHW	FKP	JMD	MGA	PAB	SOS	UYI	XSU
BBD	CLF	GBP	JOD	MKD	PEN	SRD	UYU	XTS
BDT	CLP	GEL	JPY	MMK	PGK	SSP	UZS	XUA
BGN	CNY	GHS	KES	MNT	PHP	STD	VEF	XXX
BHD	COP	GIP	KGS	MOP	PKR	SVC	VND	YER
BIF	COU	GMD	KHR	MRO	PLN	SYP	VUV	ZAR
BMD	CRC	GNF	KMF	MUR	PYG	SZL	WST	ZMW
BND	CUC	GTQ	KPW	MVR	QAR	THB	XAF	ZWL
BOB	CUP	GYD	KRW	MWK	RON	TJS	XAG	
BOV	CVE	HKD	KWD	MXN	RSD	TMT	XAU	

1819 Table 8: Enumeration CurrencyEnumType

1820

1821 **3.10.1.21 ScopeTypeType**

1822 Union of:

1823 - ScopeTypeEnumType (see section 3.10.1.22)

1824 - EnumExtendType (see section 3.10.1.5)

1825

1826 **3.10.1.22 ScopeTypeEnumType**

1827 Some feature types use scope types for specific functionalities. These are listed in the

1828 ScopeTypeEnumType.

1829 A *simpleType* (restricted *string*), containing enumerations for the scope type.

Scope Type	Description
ac	Alternating current.
acCosPhiGrid	Alternating current cosine phi of the electricity grid.
acCurrentA	Alternating current current on phase A.
acCurrentB	Alternating current current on phase B.
acCurrentC	Alternating current current on phase C.
acFrequencyGrid	Alternating current frequency of the electricity grid.
acPowerA	Alternating current power on phase A.
acPowerB	Alternating current power on phase B.
acPowerC	Alternating current power on phase C.
acPowerLimitPct	Alternating current power limit in percentage.

acPowerTotal	Alternating current power total (all phases).
acVoltageA	Alternating current voltage on phase A.
acVoltageB	Alternating current voltage on phase B.
acVoltageC	Alternating current voltage on phase C.
acYieldDay	Alternating current yield one (this) day.
acYieldTotal	Alternating current yield total (since last reset).
dcCurrent	Direct current.
dcPower	Direct current power.
dcString1	Direct current of string 1 (no info here whether voltage/current/etc.!).
dcString2	Direct current of string 2 (no info here whether voltage/current/etc.!).
dcString3	Direct current of string 3 (no info here whether voltage/current/etc.!).
dcString4	Direct current of string 4 (no info here whether voltage/current/etc.!).
dcString5	Direct current of string 5 (no info here whether voltage/current/etc.!).
dcString6	Direct current of string 6 (no info here whether voltage/current/etc.!).
dcTotal	Direct current of all strings (no info here whether voltage/current/etc.!).
dcVoltage	Direct current voltage.
dhwTemperature	Domestic hot water temperature (for example in a domestic hot water storage).
flowTemperature	Flow temperature (for example of a heating circuit).
outsideAirTemperature	Temperature of the outside air.
returnTemperature	Return (backflow) temperature (for example of a heating circuit).
roomAirTemperature	Air temperature inside a room.
charge	(Battery-)Charge-related values.
stateOfCharge	State-of-charge-related values of a (battery) appliance.
discharge	(Battery-)Discharge-related values.
gridConsumption	Values related to the consumption from the electricity grid.
gridFeedIn	Values related to the electricity grid feed in.
selfConsumption	Self consumption (optimization).
overloadProtection	Overload protection of the local grid, e.g.
acPower	AC power.
acEnergy	AC energy.
acCurrent	AC current.
acVoltage	AC voltage.
batteryControl	Battery control (e.g. charge, discharge, idle).
simpleIncentiveTable	Simple incentive table (limiting the possible variations of the Feature Type to an easy to implement solution).

Table 9: Enumeration ScopeTypeEnumType

3.10.1.23 RoleType

If a role has to be stated, this type is used.

A *simpleType* (restricted *string*), containing the enumeration values *client*, *server* or *special*.

3.10.1.24 FeatureGroupType

Use for connecting related features (see section 2.1.4).

A *simpleType* (restricted *string*), with the pattern

(# [1-9] [0-9] *) *

1840 allowing feature group strings like "#1", "#1#5", etc. The maximum string length is limited to 128
1841 characters.

1842

1843 **3.10.1.25 DeviceTypeType**

1844 Union of:

1845 - DeviceTypeEnumType (see section 3.10.1.26)

1846 - EnumExtendType (see section 3.10.1.5)

1847

1848 **3.10.1.26 DeviceTypeEnumType**

1849 The type of the device.

1850 A *simpleType* (restricted *string*), containing the device types defined by the EEBus Initiative e.V. See
1851 Table 334 for all enumerations.

1852

1853 **3.10.1.27 EntityTypeType**

1854 Union of:

1855 - EntityTypeEnumType (see section 3.10.1.28)

1856 - EnumExtendType (see section 3.10.1.5)

1857

1858 **3.10.1.28 EntityTypeEnumType**

1859 The type of the entity.

1860 A *simpleType* (restricted *string*), containing the entity types defined by the EEBus Initiative e.V. See
1861 Table 335 for all enumerations.

1862

1863 **3.10.1.29 FeatureTypeType**

1864 Union of:

1865 - FeatureTypeEnumType (see section 3.10.1.30)

1866 - EnumExtendType (see section 3.10.1.5)

1867

1868 **3.10.1.30 FeatureTypeEnumType**

1869 The type of the feature.

1870 A *simpleType* (restricted *string*), containing the feature types defined by the EEBus Initiative e.V. See
1871 Table 336 for all enumerations.

1872

1873 **3.10.1.31 FeatureSpecificUsageType**

1874 Deprecated.

1875

1876 **3.10.1.32 FeatureHvacSpecificUsageEnumType**

1877 Deprecated.

1878

1879 **3.10.1.33 FeatureMeasurementSpecificUsageEnumType**

1880 Deprecated.

1881

1882 **3.10.1.34 FeatureSetpointSpecificUsageEnumType**

1883 Deprecated.

1884

1885 **3.10.1.35 FunctionType**

1886 Union of:

1887 - FunctionEnumType (see section 3.10.1.36)

1888 - EnumExtendType (see section 3.10.1.5)

1889

1890 **3.10.1.36 FunctionEnumType**

1891 A list of all SPINE functions defined by the EEBus Initiative e.V.

1892 A *simpleType* (restricted *string*), containing enumerations of all SPINE functions. Please consider the
1893 class sections where all SPINE functions are specified.

1894

1895 **3.10.1.37 PossibleOperationsClassifierType**

1896 If the element *partial* is used, filtering the further content is permitted. Otherwise only full
1897 operations (no filtering) are allowed.

1898 A *complexType*, containing the element *partial* (type: *ElementTagType*, see section 3.10.1.1).

1899

1900 **3.10.1.38 PossibleOperationsReadType**

1901 If read operations are permitted, this element is set. Elements from the base type can be used (see
1902 the corresponding section).

1903 A *complexType* with the base type *PossibleOperationsClassifierType* (see section 3.10.1.37).

1904

1905 **3.10.1.39 PossibleOperationsWriteType**

1906 If write operations are permitted, this element is set. Elements from the base type can be used (see
1907 the corresponding section).

1908 A *complexType* with the base type *PossibleOperationsClassifierType* (see section 3.10.1.37).

1909

3.10.1.40 PossibleOperationsType

Defines which operations on a function are allowed.

A *complexType*, containing the elements *read* (type: *PossibleOperationsReadType*, see section 3.10.1.38) and *write* (type: *PossibleOperationsWriteType*, see section 3.10.1.39).

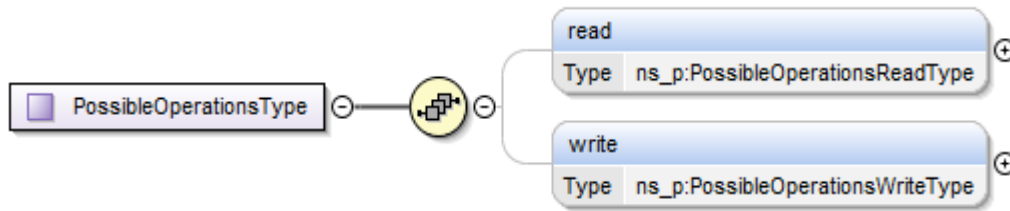


Figure 28: PossibleOperationsType

3.10.1.41 FunctionPropertyType

If a function together with its possible operations will be denoted, this complex type can be used.

A *complexType*, containing the two elements *function* (type: *FunctionType*, see section 3.10.1.35) and *possibleOperations* (type: *PossibleOperationsType*, see section 3.10.1.40).

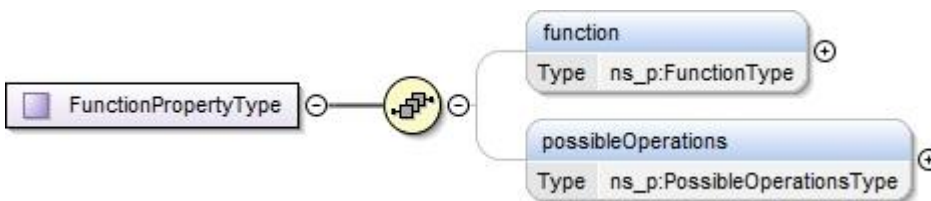


Figure 29: FunctionPropertyType

3.10.1.42 MaxResponseDelayType

Used to represent the maxResponseDelay.

A *simpleType* (xs:duration).

3.10.2 Time-related**3.10.2.1 TimePeriodType**

The typically used type for all time periods in the SPINE data model.

A *complexType* containing the two elements *startTime* and *endTime* (both of type *AbsoluteOrRelativeTimeType*, see section 3.10.2.3).

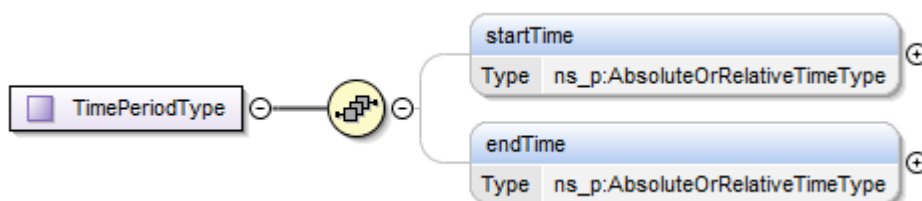


Figure 30: TimePeriodType

Name	Type	Description
startTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The startTime element defines the start time.
endTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The endTime element defines the end time.

Table 10: TimePeriodType

3.10.2.2 TimestampIntervalType

The typically used type for all timestamp intervals in selectors in the SPINE data model.

A *complexType*, containing the two elements *startTime* and *endTime* (both of type *AbsoluteOrRelativeTimeType*, see section 3.10.2.3).

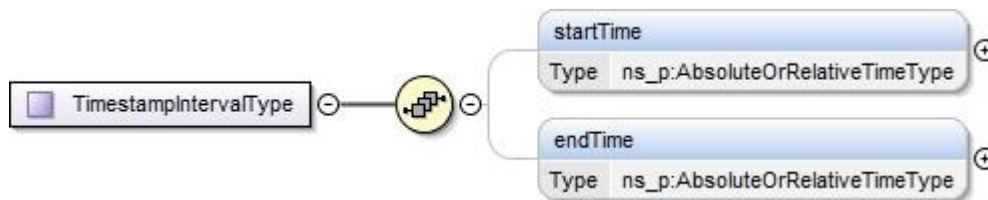


Figure 31: TimestampIntervalType

Name	Type	Description
startTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The startTime element defines the start time.
endTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The endTime element defines the end time.

Table 11: TimestampIntervalType

3.10.2.3 AbsoluteOrRelativeTimeType

The commonly used type for time information in the SPINE data model. Allows the modelling of absolute (*xs:dateTime*) or relative (*xs:duration*) times (see section 3.2).

A *union* with the two members (allowed types) *xs:duration* and *xs:dateTime*.

xs:dateTime is used to model absolute times.

xs:duration is used to model relative times. If not stated otherwise a duration of 0 seconds (e.g. "PT0S", or "PT0H", etc.) equals "now", while negative durations denote times in the past and positive values denote times in the future relative to "now".

1955 **3.10.2.4 *RecurringIntervalType***

1956 Union of:

1957 - *RecurringIntervalEnumType* (see section 3.10.2.5)

1958 - *EnumExtendType* (see section 3.10.1.5)

1959

1960 **3.10.2.5 *RecurringIntervalEnumType***

1961 Recurring interval length make use of this type.

1962 A *simpleType* (restricted *string*), containing enumerations for the recurring interval length.

yearly
monthly
weekly
daily
hourly
everyMinute
everySecond

1963 Table 12: *RecurringIntervalEnumType*

1964

1965 **3.10.2.6 *MonthType***

1966 All month names are included in this type.

1967 A *simpleType* (restricted *string*), containing enumerations for the months.

january	july
february	august
march	september
april	october
may	november
june	december

1968 Table 13: *MonthType*

1969

1970 **3.10.2.7 *DayOfMonthType***

1971 This type is used for stating the day of the month.

1972 A *simpleType* (restricted *xs:unsignedByte*), with the allowed value range from "1" to "31".

1973

1974 **3.10.2.8 *CalendarWeekType***

1975 This type is used for stating the calendar week.

1976 A *simpleType* (restricted *xs:unsignedByte*), with the allowed value range from "1" to "53".

1977

1978 **3.10.2.9 *DayOfWeekType***

1979 This type contains all weekdays as enumeration (only one selectable).

1980 A *simpleType* (restricted *xs:string*), containing enumerations for the weekdays.

monday
tuesday
wednesday
thursday
friday
saturday
sunday

Table 14: DayOfWeekType

3.10.2.10 DaysOfWeekType

This type contains all weekdays as flags (more than one selectable).

A *complexType*, containing the elements monday, tuesday, wednesday, thursday, friday, saturday, sunday.

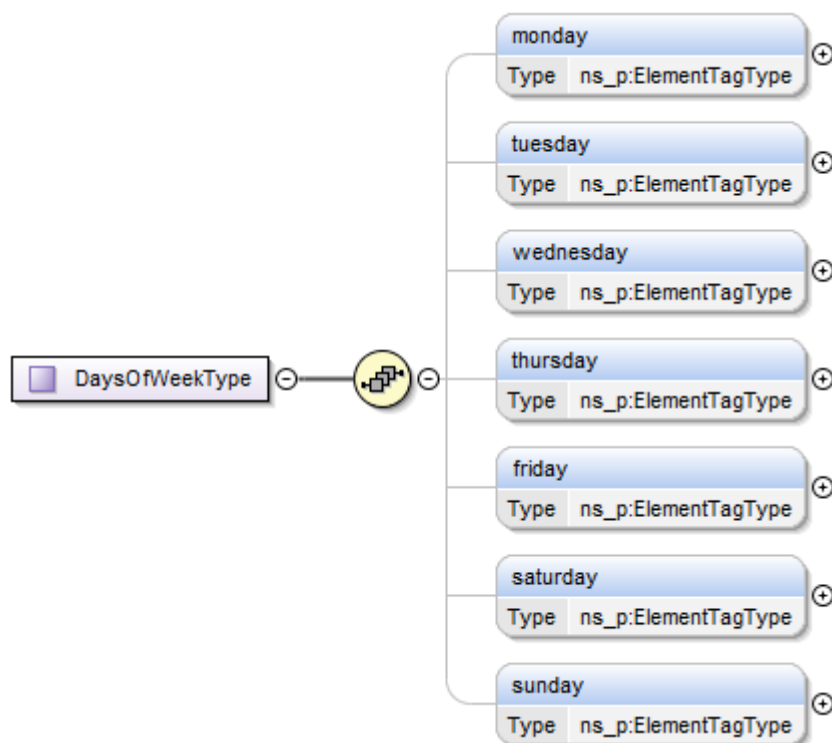


Figure 32: DaysOfWeekType

3.10.2.11 OccurrenceType

Union of:

- OccurrenceEnumType (see section 3.10.2.12)
- EnumExtendType (see section 3.10.1.5)

3.10.2.12 OccurrenceEnumType

If the occurrence of something will be modelled, this type can be used.

A *simpleType* (restricted *string*), containing enumerations for some occurrences.

Value	Description
first	First occurrence of the related event where this type is used.
second	Second occurrence of the related event where this type is used.
third	Third occurrence of the related event where this type is used.
fourth	Fourth occurrence of the related event where this type is used.
last	Last occurrence of the related event where this type is used.

Table 15: OccurrenceEnumType

1998

1999

2000 **3.10.2.13 AbsoluteOrRecurringTimeType**

2001 If absolute (specific point in time) or recurring (relative point in time that is repeated) times will be
 2002 modelled, this type is used.

2003 A complexType, containing the elements described in Table 16.

2004 The typical usage is that either *dateTime* is denoted or one of the following combinations:

- 2005 - month + dayOfMonth + time (specific day once per year)
- 2006 - dayOfMonth + time (specific day once per month)
- 2007 - calendarWeek + daysOfWeek + time (specific day of week once per year)
- 2008 - month + dayOfWeekOccurrence + daysOfWeek + time (first, second, ... occurrence of specific
 2009 day of week in a specific month once per year, e.g. every second Thursday of July)
- 2010 - daysOfWeek + time (specific day of week once per week)
- 2011 - time (every day at a specific time)

2012 The *dayOfWeekOccurrence* element only makes sense in combination with *daysOfWeek* where
 2013 *calendarWeek* is not set. It defines which week in a month was meant. For example, the *first* week, or
 2014 the *second*, etc.

2015 Instead of only one day, combinations of several days may be stated in the element *daysOfWeek*
 2016 (e.g. "monday" and "tuesday" and "friday" or "saturday" and "sunday").

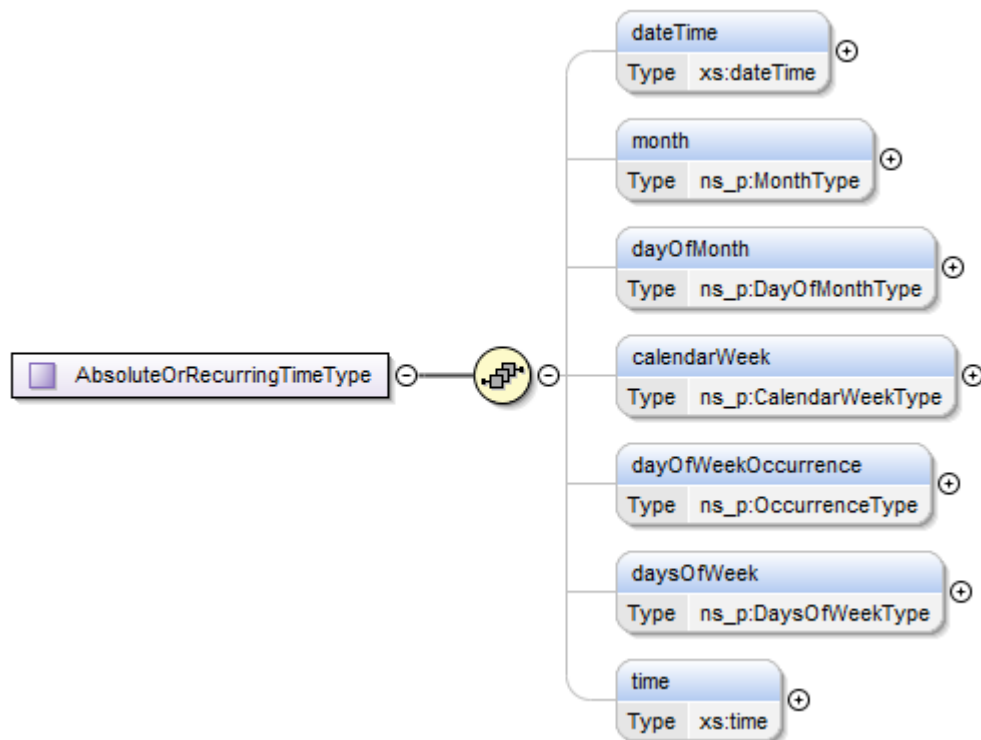


Figure 33: AbsoluteOrRecurringTimeType

Element	Type	Description
dateTime	Common data type "RecurringIntervalType". See section 3.10.2.4.	An absolute time.
month	Common data type "MonthType". See section 3.10.2.6.	The month information for the recurring time (e.g. "January").
dayOfMonth	Common data type "DayOfMonthType". See section 3.10.2.7.	The day of month information for the recurring time (e.g. "25").
calendarWeek	Common data type "CalenderWeekType". See section 3.10.2.8.	The calendar week information for the recurring time (e.g. "50").
dayOfWeekOccurrence	Common data type "OccurrenceType". See section 3.10.2.11.	The day of week occurrence information for the recurring time (e.g. "second", if the second occurrence of this weekday within a month is chosen).
daysOfWeek	Common data type "DaysOfWeekType". See section 3.10.2.10.	The days of week information for the recurring time (e.g. the elements "monday" and "wednesday" could be set).
time	xs:time (W3C standard type)	The time information for the recurring time (e.g. "18:15:31Z").
relative	xs:duration (W3C standard type)	The relative time information for the recurring time (e.g. "PT23H15M20S").

Table 16: AbsoluteOrRecurringTimeType

3.10.2.14 RecurrenceInformationType

If the recurrence of some time information will be modelled, this type is used.

A complexType, containing the elements described in Table 17.

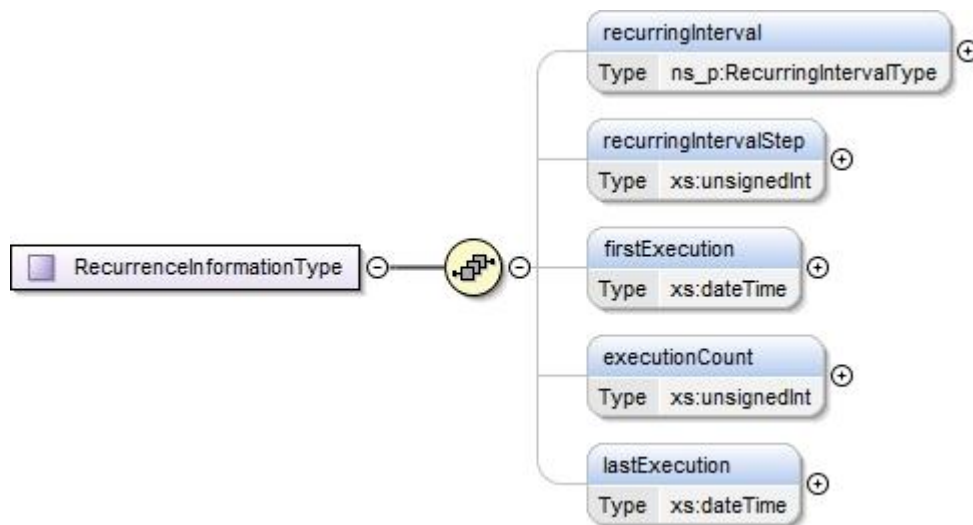


Figure 34: RecurrenceInformationType

Element	Type	Description
recurringInterval	Common data type "RecurringIntervalType". See section 3.10.2.4.	With the <i>recurringInterval</i> the frequency of the recurrence is defined (e.g. "weekly").
recurringIntervalStep	xs:unsignedInt (W3C standard type)	If not every occurrence will be active, the <i>recurringIntervalStep</i> is used (e.g. if only every second occurrence will be active, <i>recurringIntervalStep</i> would have a value of "2").
firstExecution	xs:dateTime (W3C standard type)	If the recurring interval will not start with the next occurrence, the <i>firstExecution</i> element is used to specify this first time.
executionCount	xs:unsignedInt (W3C standard type)	To end the recurring execution, the <i>executionCount</i> can be used (e.g. after "10" occurrences, the interval ends).
lastExecution	xs:dateTime (W3C standard type)	To end the recurring execution, the <i>lastExecution</i> can be used, stating a specific point in time after that the interval will be not active any more (the <i>lastExecution</i> point in time does not have to be an interval occurrence).

Table 17: RecurrenceInformationType

3.10.3 Address-related

3.10.3.1 AddressDeviceType

Used to address a (physical) device.

A *simpleType* (restricted *string*), containing the device address part, with the pattern

`d:_(i:[1-9][0-9]*|n:[a-zA-Z0-9-]+)_[^p{Cc}\p{Cf}\p{Z}]+`

For details please see [ProtocolSpecification], section "Rules for devices". The maximum string length is limited to 256 characters.

3.10.3.2 AddressEntityType

Used to address a logical device.

A *simpleType* (*unsignedInt*), containing an entity address part.

3.10.3.3 AddressFeatureType

Used to address a specific feature.

A *simpleType* (*unsignedInt*), containing the feature address part.

3.10.3.4 DeviceAddressType

Address type including only the device part.

A *complexType* with the element *device* (type: *AddressDeviceType*, see section 3.10.3.1).

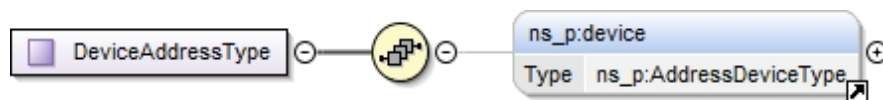


Figure 35: DeviceAddressType

3.10.3.5 EntityAddressType

Address type including the device part and a list of entities.

A *complexType* with the base type *DeviceAddressType* (see section 3.10.3.4) and a further list of element *entity* (type: *AddressEntityType*, see section 3.10.3.2).

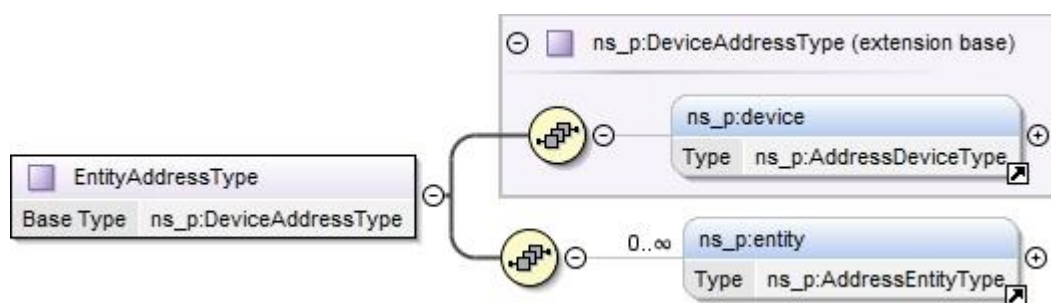


Figure 36: EntityAddressType

3.10.3.6 FeatureAddressType

Address type including the device part, a list of entities and the feature part.

A *complexType* with the base type *EntityAddressType* (see section 3.10.3.5) and a further element *feature* (type: *AddressFeatureType*, see section 3.10.3.3).

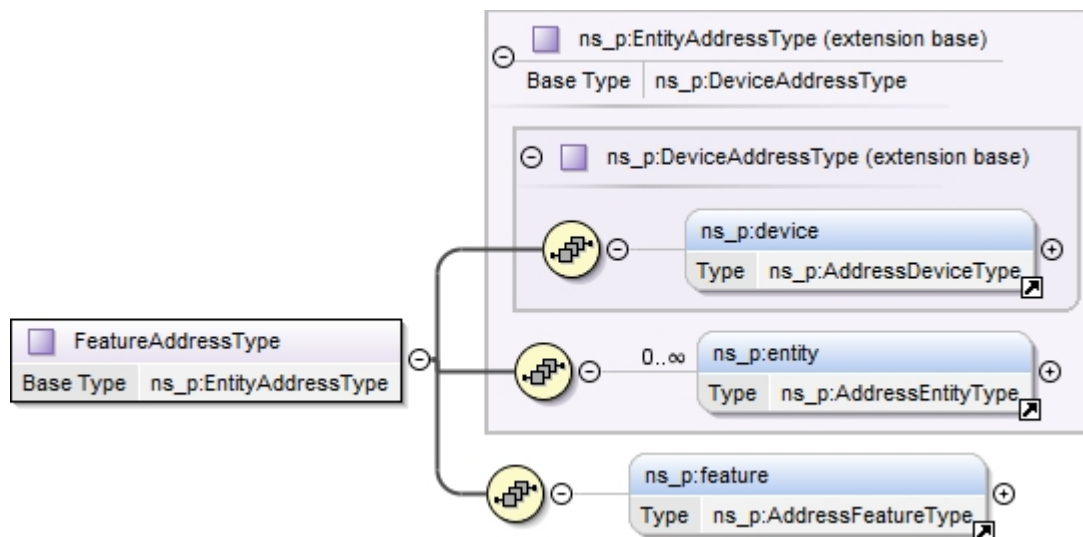


Figure 37: FeatureAddressType

3.11 Result

Note: This section discusses the SPINE function “resultData” and NOT the classifier “result”!

The result function concept is used for modelling acknowledges or errors for messages exchanged between communications partners. It is intended for these use cases:

- Notification of an error related to another function (e.g. as a reaction to a failed write operation).
- Substitution of a "regular reply" with an "error response". This applies if it is not possible to create a valid and proper reply that belongs to the function group of the corresponding read function.

Further information about the usage of result functions can be found in [ProtocolSpecification], section "Acknowledgement message".

The result function concept is modelled similar to (standard) classes. Its only function is called "resultData".

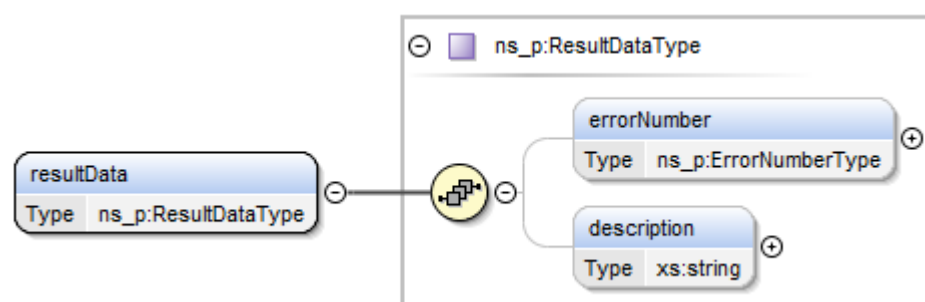


Figure 38: resultData function overview

The elements `errorNumber` and `description` are defined as follows:

Name	Type	Description
errorNumber	xs:unsignedInt (W3C standard type) See Table 19 for currently specified error numbers.	The error number specifies whether the result is "OK" (value "0") or an error (value ">0").
description	xs:string (W3C standard type)	Descriptive information on the result.

Table 18: resultData function detailed description of elements

Values specified for element **errorNumber**:

Value	Description
0	No Error.
1	General Error.
2	Timeout.
3	Overload.
4	Destination unknown.
5	Destination unreachable.
6	Command not supported.
7	Command rejected.
8	Restricted function exchange combination not supported.
9	Binding is necessary for this command.

Table 19: Values specified for element errorNumber

Further error numbers will be added in future releases.

Example of an error indication:

```
<resultData>
  <errorNumber>2</errorNumber>
  <description>Timeout</description>
</resultData>
```

4 Device model definitions

The subsequent sections define details on the use of "Device Type", "Entity Type" and "Feature Type". An overview of all types can be found in Annex B.

4.1 Device Types

4.1.1 ChargingStation

4.1.1.1 Descriptive information

Represents a Charging Station, typically used in the context of electrical vehicle charging.

4.1.2 Dishwasher

4.1.2.1 Descriptive information

Represents a Dishwasher

4.1.3 Dryer

4.1.3.1 Descriptive information

Represents a Dryer.

4.1.4 ElectricitySupplySystem

4.1.4.1 Descriptive information

Represents a set of sources or storages of electric power of a premises (for example a local PV system, the electricity grid connection point or a battery storage).

4.1.5 EnergyManagementSystem

4.1.5.1 Descriptive information

Represents an Energy Management System (EMS).

4.1.6 EnvironmentSensor

4.1.6.1 Descriptive information

Represents a Sensor typically used for outdoor measurements.

2124 **4.1.7 Generic**

2125 **4.1.7.1 Descriptive information**

2126 This is a generic device, which can either be used for client-only functionality or for development
2127 purposes where no specific type of device can be stated yet.

2128

2129 **4.1.8 HeatGenerationSystem**

2130 **4.1.8.1 Descriptive information**

2131 This Device Type is used to represent the subsystem of an HVAC system that generates heat. It can
2132 be used to represent single heat generators like gas heating boilers or heat pumps, but also be used
2133 to represent a heat generator cascade (like a cascade of several gas heating boilers or a combination
2134 of heat pump appliances and gas heating boilers). It can also be used to represent a solar thermal
2135 system.

2136 The Device Type “HeatGenerationSystem” serves as a container for specific Entity Types that
2137 represent the single heat generators.

2138

2139 **4.1.9 HeatSinkSystem**

2140 **4.1.9.1 Descriptive information**

2141 Represents the end user relevant heat sinks of the HVAC system (heating circuits, domestic hot water
2142 (DHW) circuits, air conditioning system, ...) and the end user relevant system functions (space
2143 heating, space cooling, DHW, air conditioning, ...)

2144

2145 **4.1.10 HeatStorageSystem**

2146 **4.1.10.1 Descriptive information**

2147 Represents a system of heat storages consisting of one or more heat storage appliances or
2148 appliances that are attached to a heat storage (for example the Entity Type InstantDHWHeater (see
2149 section 4.2.31) (fresh water station attached to a heating buffer storage)). All heat storages of the
2150 HVAC system should be represented by this Device Type.

2151

2152 **4.1.11 HVACController**

2153 **4.1.11.1 Descriptive information**

2154 Represents the central controller appliance of an HVAC System.

2155 A device of Device Type HVACController is used to deliver overall information about the HVAC
2156 system.

2157

2158 **4.1.12 Inverter**

2159 **4.1.12.1 Descriptive information**

2160 Represents a power Inverter.

2161

2162 **4.1.13 SubMeter**

2163 **4.1.13.1 Descriptive information**

2164 A SubMeter measures the demand or generation of a commodity (e.g. electricity).

2165

2166 **4.1.14 Washer**

2167 **4.1.14.1 Descriptive information**

2168 Represents a Washing Machine.

2169

2170 **4.2 Entity Types**

2171 **4.2.1 Battery**

2172 **4.2.1.1 Descriptive information and further definitions**

2173 Represents a single battery unit.

2174

2175 **4.2.2 BatterySystem**

2176 **4.2.2.1 Descriptive information**

2177 A system consisting of one or more battery storages and one or more battery inverters.

2178

2179 **4.2.3 CEM**

2180 **4.2.3.1 Descriptive information and further definitions**

2181 Abbreviation for Customer Energy Manager and typically used to represent an Energy Manager
2182 within SPINE.

2183

2184 **4.2.4 ChargingOutlet**

2185 **4.2.4.1 Descriptive information and further definitions**

2186 Represents a Charging Outlet of a Charging Station.

2187

2188 **4.2.5 Compressor**

2189 **4.2.5.1 Descriptive information and further definitions**

2190 Represents the compressor of (e.g.) an (electrical) heat pump appliance.

2191

2192 **4.2.6 DeviceInformation**

2193 **4.2.6.1 Descriptive information and further definitions**

2194 This entity has a special role and contains general information about the complete device.

2195

2196 **4.2.7 DHWCircuit**

2197 **4.2.7.1 Descriptive information and further definitions**

2198 Represents all elements for domestic hot water supply like cold water line, domestic hot water line,
2199 circulation line, tapping points.

2200 Should not include DHW storages, these should be represented as Entity Type "DHWStorage"

2201

2202 **4.2.8 DHWStorage**

2203 **4.2.8.1 Descriptive information and further definitions**

2204 Represents a heat storage appliance for storing domestic hot water.

2205

2206 **4.2.9 Dishwasher**

2207 **4.2.9.1 Descriptive information and further definitions**

2208 Represents a Dishwasher

2209

2210 **4.2.10 Dryer**

2211 **4.2.10.1 Descriptive information and further definitions**

2212 Represents a Dryer.

2213

2214 **4.2.11 ElectricalImmersionHeater**

2215 **4.2.11.1 Descriptive information and further definitions**

2216 Represents an electrical immersion heater (electrical resistance heater) for example as part of a heat
2217 pump appliance, a heat source unit, a DHW storage.

2218

2219 **4.2.12 ElectricityGenerationSystem**

2220 **4.2.12.1 Descriptive information and further definitions**

2221 A system that is capable of producing electric power through local generation (for example PV-
2222 systems, wind turbines or block heating and power plants).

2223

2224 **4.2.13 ElectricityStorageSystem**

2225 **4.2.13.1 Descriptive information and further definitions**

2226 A system that is capable of storing electric power and releasing the majority of it as electric power
2227 (for example a battery system or a capacitor).

2228

2229 **4.2.14 EV**

2230 **4.2.14.1 Descriptive information and further definitions**

2231 Represents an Electrical Vehicle.

2232

2233 **4.2.15 EVSE**

2234 **4.2.15.1 Descriptive information and further definitions**

2235 Abbreviation for Electrical Vehicle Supply Equipment, typically used to connect an Electrical Vehicle to
2236 a Charging Station.

2237

2238 **4.2.16 Fan**

2239 **4.2.16.1 Descriptive information and further definitions**

2240 Represents an air fan, for example of an air water heat pump unit or a ventilation unit.

2241

2242 **4.2.17 GasHeatingAppliance**

2243 **4.2.17.1 Descriptive information and further definitions**

2244 Represents a heating appliance that generates heat (for example for heating or domestic hot water)
2245 by burning natural gas.

2246

2247 **4.2.18 Generic**

2248 **4.2.18.1 Descriptive information and further definitions**

2249 This is a generic entity, which can either be used for client-only functionality, or for development
2250 purposes, where no specific entity can be stated yet.

2251

2252 4.2.19 GridConnectionPointOfPremises**2253 4.2.19.1 Descriptive information and further definitions**

2254 The electrical connection point between the utility electricity grid and the electric circuit of a
2255 premises.

2256

2257 4.2.20 HeatingBufferStorage**2258 4.2.20.1 Descriptive information and further definitions**

2259 Represents a heat storage appliance for storing heating water.

2260

2261 4.2.21 HeatingCircuit**2262 4.2.21.1 Descriptive information and further definitions**

2263 Represents a single heating circuit for space heating (or space cooling like active or passive cooling),
2264 including flow and return lines, optional mixing units (for mixed heating circuits) and the heat
2265 distribution:

2266 - directly to one or more heated rooms (see Entity Type “HVACRoom”, section 4.2.30)

2267 - to one or more heating zones (see Entity Type “HeatingZone”, section 4.2.23)

2268

2269 4.2.22 HeatingObject**2270 4.2.22.1 Descriptive information and further definitions**

2271 Represents a Heating Object, e.g. a radiator.

2272

2273 4.2.23 HeatingZone**2274 4.2.23.1 Descriptive information and further definitions**

2275 Represents a heating zone that provides heating energy (or cooling energy) to one or more heated
2276 rooms (see Entity Type HVACRoom).

2277

2278 4.2.24 HeatPumpAppliance**2279 4.2.24.1 Descriptive information and further definitions**

2280 Represents either a complete heat pump appliance (for example a brine water heat pump, indoor; an
2281 air water heat pump, completely indoor or completely outdoor) or the indoor part of a refrigerant
2282 split heat pump combination.

2283

2284 4.2.25 HeatSinkCircuit

2285 4.2.25.1 Descriptive information and further definitions

2286 Represents a (hydraulic) circuit of a heat generator that delivers heat energy from the heat generator
2287 to a heat sink. For example Entity Type "HeatSinkCircuit" could represent a (hydraulic) circuit of a gas
2288 heating appliance that delivers heat energy to a heating circuit.

2289

2290 4.2.26 HeatSourceCircuit

2291 4.2.26.1 Descriptive information and further definitions

2292 Represents a (hydraulic) circuit of a heat generator (especially of heat pumps) that delivers heat
2293 energy from an ambient energy source to the heat generator. For example Entity Type
2294 "HeatSourceCircuit" could represent a brine circuit of a heat pump appliance that delivers heat
2295 energy from an ambient heat source to the heat pump appliance.

2296

2297 4.2.27 HeatSourceUnit

2298 4.2.27.1 Descriptive information and further definitions

2299 Represents an additional unit that provides the heat from the environmental heat source to the heat
2300 pump appliance. This could for example be:

- 2301 - an outdoor ventilation unit that transfers the heat energy from the outside air to the heat source
- 2302 circuit of the (indoor) heat pump appliance (i.e. air water split heat pump)
- 2303 - an outdoor ventilation unit that transfers the heat energy from the outside air to the refrigerant
- 2304 circuit of the (indoor) heat pump appliance (refrigerant split heat pump)

2305

2306 4.2.28 Household

2307 4.2.28.1 Descriptive information and further definitions

2308 This is a rather virtual entity representing a complete household that aggregates all relevant
2309 information of the household (e.g. the complete energy demand, etc.).

2310

2311 4.2.29 HVACController

2312 4.2.29.1 Descriptive information and further definitions

2313 Represents the central controller appliance of the HVAC System (as a SPINE Entity).

2314

2315 4.2.30 HVACRoom

2316 4.2.30.1 Descriptive information and further definitions

2317 Represents one physical room in a building/house that can be heated / cooled, ventilated or air
2318 conditioned.

2319

2320 **4.2.31 InstantDHWHeater**2321 **4.2.31.1 Descriptive information and further definitions**

2322 Represents an appliance for instant generation of domestic hot water (for example by using heat
2323 from a heating buffer storage).

2324

2325 **4.2.32 Inverter**2326 **4.2.32.1 Descriptive information and further definitions**

2327 Represents a power Inverter.

2328

2329 **4.2.33 OilHeatingAppliance**2330 **4.2.33.1 Descriptive information and further definitions**

2331 Represents a heating appliance that generates heat (for example for heating or domestic hot water)
2332 by burning oil.

2333

2334 **4.2.34 Pump**2335 **4.2.34.1 Descriptive information and further definitions**

2336 Represents a pump to generate a flow rate (for example in a heating circuit).

2337

2338 **4.2.35 PVSystem**2339 **4.2.35.1 Descriptive information and further definitions**

2340 A system consisting of one or more PV inverters with one or more PV modules. The PVSystem
2341 provides aggregated PV related system parts of the house or premises.

2342

2343 **4.2.36 RefrigerantCircuit**2344 **4.2.36.1 Descriptive information and further definitions**

2345 Represents the refrigerant circuit of a heat pump appliance.

2346

2347 **4.2.37 SmartEnergyAppliance**2348 **4.2.37.1 Descriptive information and further definitions**

2349 An appliance that allows a client to shift the load announced by the server can be modelled with this
2350 Entity Type. Non-shiftable loads (so-called "fixed forecasts") can be provided by this Entity, too.

2351

2352 **4.2.38 SolarDHWStorage**2353 **4.2.38.1 Descriptive information and further definitions**

2354 Represents a heat storage appliance for storing domestic hot water. The heat energy can be provided
2355 especially by solar thermal energy (compare to Entity Type "DHWStorage").

2356

2357 **4.2.39 SolarThermalCircuit**2358 **4.2.39.1 Descriptive information and further definitions**

2359 Represents a single circuit of a solar thermal system including a solar thermal collector field and flow
2360 and return pipes to/from the collector field and a solar circuit pump.

2361

2362 **4.2.40 SubMeterElectricity**2363 **4.2.40.1 Descriptive information and further definitions**

2364 Represents an electrical Sub-Meter.

2365

2366 **4.2.41 TemperatureSensor**2367 **4.2.41.1 Descriptive information and further definitions**

2368 Represents a sensor that can provide a temperature measurement value.

2369

2370 **4.2.42 Washer**2371 **4.2.42.1 Descriptive information and further definitions**

2372 Represents a laundry washer that washes clothes (laundry).

2373

2374 **4.3 Feature Types**2375 **4.3.1 ActuatorLevel**2376 **4.3.1.1 Dependencies to other Feature Types**

2377 None.

2378

2379 **4.3.1.2 Introduction**

2380 Independent from the *ActuatorSwitch* class, the *ActuatorLevel* class enables a user or application to
2381 model *LEVEL* commands (*start*, *up*, *percentageAbsolute*, *relative*, ...). This can be used to dim a light,
2382 set the speed of an electric motor, etc. Its simple functions can easily be mapped to other
2383 technologies.

2384

2385 **4.3.1.3 Rules**

2386 None.

2387

2388 **4.3.1.4 actuatorLevelData**2389 **4.3.1.4.1 Description**

2390 This function is used for setting the desired operation mode of the node or getting the actual mode
 2391 respectively.

2392

2393 **4.3.1.4.2 Rules**

2394 None.

2395

2396 **4.3.1.4.3 Element rules**

Element	Rule	Description
function	SHALL always be set. Value rules of Table 21 SHALL be applied.	The element "function" contains the mode.
value	None.	The value is only needed for some functions. E.g. setting <i>function</i> to <i>percentageAbsolute</i> just makes sense together with a proper value.

2397 Table 20: actuatorLevelData Element rules

2398 Element "**function**" value rules:

Value	Rule	Description
start	None.	Actuator starts.
up	None.	Actuator moves up.
down	None.	Actuator moves down.
stop	None.	Actuator stops.
percentageAbsolute	None.	The percentage position or value of the actuator (e.g. 25 %).
percentageRelative	None.	Changes the current value by a percentage value (e.g. move plus 10 percent).
absolute	None.	The absolute position or value of the actuator (e.g. 0.91 m).
relative	None.	Changes the current value by some value (e.g. move plus 0.23 m).

2399 Table 21: Element "function" value rules

2400

2401 **4.3.1.5 actuatorLevelDescriptionData**2402 **4.3.1.5.1 Description**

2403 The rather static descriptive data of the *actuatorLevel* class can contain a default unit, among others.
 2404 This is useful for some *absolute* commands.

2405

2406 **4.3.1.5.2 Rules**
 2407 None.

2408

2409 **4.3.1.5.3 Element rules**

Element	Rule	Description
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the actuator.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the actuator.
levelDefaultUnit	None.	If an actuator can specify a unit for its level-controlling, it can be denoted in this element.

2410 *Table 22: actuatorLevelDescriptionData Element rules*

2411

2412 **4.3.1.6 Descriptive information and further definitions**
 2413 None.

2414

2415 **4.3.2 ActuatorSwitch**

2416 **4.3.2.1 Dependencies to other Feature Types**
 2417 None.

2418

2419 **4.3.2.2 Introduction**

2420 Basic on/off operations on a simple actuator can be modelled with the *ActuatorSwitch* class.
 2421 Whether the function turns a device itself *on* or *off*, or whether it switches a specific functionality,
 2422 depends on the implementation. E.g. one could model the super freeze program of a freezer using
 2423 *ActuatorSwitch* class. An *on* command would then activate the super freeze program and an *off*
 2424 command would deactivate it. This example shall just give an idea how *ActuatorSwitch* can be used
 2425 for more purposes than only turning devices on and off.

2426 Please note a device's different features may be modelled with separate implementations of this
 2427 class, each associated to a different node address. Continuing with the example above, a "defrost"
 2428 function may be modelled on another node address with *ActuatorSwitch*.

2429

2430 **4.3.2.3 Rules**
 2431 None.

2432

2433 **4.3.2.4 actuatorSwitchData**2434 **4.3.2.4.1 Description**2435 The *actuatorSwitchData* function models the functionality described in section 5.3.2.1.

2436

2437 **4.3.2.4.2 Rules**

2438 None.

2439

2440 **4.3.2.4.3 Element rules**

Element	Rule	Description
function	None. Value rules of Table 24 SHALL be applied.	The only element in this function to model an <i>on</i> , <i>off</i> or <i>toggle</i> command.

2441 *Table 23: actuatorSwitchData Element rules*2442 Element "**function**" value rules:

Value	Rule
on	If used in a write, the actuator SHALL be activated, if the write is accepted. If used in a notify or reply, the actuator SHALL be active.
off	If used in a write, the actuator SHALL be deactivated, if the write is accepted. If used in a notify or reply, the actuator SHALL NOT be active.
toggle	SHALL only be used in a write. If used in a write, an inactive actuator SHALL be activated and an active actuator SHALL be deactivated, if the write is accepted.

2443 *Table 24: Element "function" value rules*

2444

2445 **4.3.2.5 actuatorSwitchDescriptionData**2446 **4.3.2.5.1 Description**

2447 The (typically) non-changing information on the actuator is modelled with this function.

2448

2449 **4.3.2.5.2 Rules**

2450 None.

2451

2452 **4.3.2.5.3 Element rules**

Element	Rule	Description
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the actuator.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the actuator.

2453 *Table 25: actuatorSwitchDescriptionData Element rules*

2454

2455 **4.3.2.6 Descriptive information and further definitions**

2456 None.

2457

2458 **4.3.3 Alarm**

2459 **4.3.3.1 Dependencies to other Feature Types**

2460 - Threshold

2461

2462 **4.3.3.2 Introduction**

2463 There are several reasons for generating an alarm. This might be due to exceeding (in a positive or
2464 negative way) a threshold (modelled with the Threshold class, see section 5.3.26), or because of
2465 some other occurrence that provokes the generation of an alarm. The Alarm class provides the
2466 needed data model.

2467

2468 **4.3.3.3 Rules**

2469 None.

2470

2471 **4.3.3.4 alarmListData**

2472 **4.3.3.4.1 Description**

2473 This function models data for an alarm. Depending on the type of alarm, different elements are used.
2474 It can be used to notify an alarm or as a reply to a read command.

2475

2476 **4.3.3.4.2 Rules**

2477 None.

2478

2479 **4.3.3.4.3 Element rules**

Element	Rule	Description
alarmId	SHALL be set as PRIMARY IDENTIFIER.	Identifier of the alarm.
thresholdId	If the alarm is related to a threshold, the value SHALL be set as FOREIGN IDENTIFIER. Otherwise it SHALL be omitted.	If the alarm relates to a threshold, the corresponding ID can be stated here.
timestamp	SHOULD be set.	The time of creation of the alarm.
alarmType	SHOULD be set. Value rules of Table 27 SHALL be applied.	The type of alarm is given here.
measuredValue	None.	The value that was measured which exceeded the threshold is denoted in this element.

evaluationPeriod	None.	If a time period can be stated in which
scopeType	None.	Specifies a more detailed meaning of the alarm (e.g. exceeding of a threshold).
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the alarm.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the alarm.

Table 26: alarmListData Element rules

Element "**alarmType**" value rules:

Value	Rule
alarmCancelled	SHALL be used, if a previously generated alarm is cancelled.
underThreshold	SHALL be used, if the value is under a related threshold (<i>thresholdId</i> SHALL be set).
overThreshold	SHALL be used, if the value is over a related threshold (<i>thresholdId</i> SHALL be set).

Table 27: Element "alarmType" value rules

4.3.3.5 Descriptive information and further definitions

None.

4.3.4 Bill

4.3.4.1 Dependencies to other Feature Types

None.

4.3.4.2 Introduction

The Bill Feature can be used to issue bills, containing positions and their costs.

4.3.4.3 Rules

None.

4.3.4.4 billListData

4.3.4.4.1 Description

billData holds the bill data, including descriptive elements like currency, unit and billType.

A billData has a "total" element, including total values and corresponding costs in its sub-values.

Additionally, positions of the total data can be provided, to describe the composition of the total

data. To link total costs and values in a position the valuelId and costId are used within a billData.

2503

2504 *4.3.4.4.2 Rules*

2505 The identifier positionId MAY be omitted, if only one position is included in the bill. Otherwise it
 2506 SHALL be set.

2507 The identifier valueId MAY be omitted, if only one value is included in the bill. Otherwise it SHALL be
 2508 set.

2509 The identifier costId MAY be omitted, if only one cost is included in the bill. Otherwise it SHALL be
 2510 set.

2511

2512 *4.3.4.4.3 Element rules*

Element	Rule	Description
billId	SHALL be set as PRIMARY IDENTIFIER, at least if more than one bill exists.	Allows the identification of a bill.
billType	Value rules of Table 29 SHALL be applied.	The type of the bill.
scopeType	None.	Specifies the scope of the bill.
total	None.	Total information of the bill (in contrast to the positions, see below).
total. timePeriod	None.	Allows to define a time, or time period for a bill.
total. value (list)	None.	Different values can be listed for a bill. E.g. energy values or just a number of items.
total. value. valueId	This Element SHALL be interpreted as SUB IDENTIFIER. If there is only one "value" entry within "total", the "valueId" MAY be omitted. Otherwise it SHALL be stated. If omitted, "valueId" SHALL equal "1".	Allows the identification of a value within the bill.
total. value. unit	If the value has a unit, it SHOULD be stated here.	Allows definition of a unit for the value.
total. value. value	Sales from the perspective of the Feature Server SHALL be expressed with negative values and purchases with positive values. In "total" all position sales and purchases SHALL be accumulated.	Allows to express the total amount that was bought or sold of a certain commodity during the time or time period given in total.timePeriod.
total. cost (list)	None.	Different types of costs can be listed for a value. E.g. monetary costs like a price as well as environmental costs as CO2 emission.
total. cost. costId	This Element SHALL be interpreted as SUB IDENTIFIER. If there is only one "cost" entry within "total", the "costId" MAY be omitted. Otherwise it SHALL be stated. If omitted, "costId" SHALL equal "1".	Allows identification of costs within bill.

total. cost. costType	Value rules of Table 30 SHALL be applied.	Allows to describe the type of costs.
total. cost. valueId	If the cost is linked to a value, the corresponding valueId SHALL be set.	Allows to link cost to a value.
total. cost. unit	None.	Costs that are not monetary may need to describe a unit (e.g. m ³).
total. cost. currency	None.	Monetary costs may need to describe a currency.
total. cost. cost	None.	Total costs generated during the time or time period given in total.timePeriod.
total. label	Default rules of section 3.10.1.2.1 SHALL be applied.	Allows to express a user friendly label for the bill.
total. description	Default rules of section 3.10.1.3.1 SHALL be applied.	Allows to express a user friendly description for the bill.
position (list)	None.	For each total data different positions can be assigned that describe the composition of the total data.
position. positionId	This Element SHALL be interpreted as SUB IDENTIFIER. If there is only one "position" entry the "positionId" MAY be omitted. Otherwise it SHALL be stated. If omitted, "positionId" SHALL equal "1".	Identifier of a position within a specific bill.
position. positionType	Value rules of Table 31 SHALL be applied.	Allows to describe the type of the a position E.g. different types of energy consumption like self produced or electricity grid energy.
position. timePeriod	None.	Allow to define a time, or time period for a position if the time or time period is different from the overall time period described in total.
position. value (list)	None.	Different values can be listed for each position.
position. value. valueId	This Element SHALL be interpreted as SUB IDENTIFIER. If there is only one "value" entry for the "position", the "valueId" MAY be omitted. Otherwise it SHALL be stated. If omitted, "valueId" SHALL equal "1".	Identifier of a value within a specific bill (see also "total. value. valueId").
position. value. value	Allows to describe a value for each position. This can be helpful if some positions have negative while other positions have positive values, as this would can not be express with valuePercentage.	Allows to describe a value for each position. This can be helpful if some positions have negative while other positions have positive values, as this would can not be express with valuePercentage.
position. value. valuePercentage	Percentage of the total value (see "total").	Percentage of the total value (see "total").
position. cost (list)	Different costs can be listed for each position	Different costs can be listed for each position

position. cost. costId	This Element SHALL be interpreted as SUB IDENTIFIER. If there is only one "cost" entry for the "position", the "costId" MAY be omitted. Otherwise it SHALL be stated. If omitted, "costId" SHALL equal "1".	Identifier of a cost within a specific bill (see also "total. cost. costId").
position. cost. cost	None.	The actual costs for this position. This can be helpful if some positions have negative while other positions have positive costs, as this can not be expressed with costPercentage.
position. cost. costPercentage	None.	Percentage of the total costs (see "total").
position. label	Default rules of section 3.10.1.2.1 SHALL be applied.	Allows to provide a user friendly label for each bill position.
position. description	Default rules of section 3.10.1.3.1 SHALL be applied.	Allows to provide user friendly information for each bill position.

Table 28: billListData Element rules

Element "**billType**" value rules:

Value	Rule	Description
chargingSummary	This summary SHOULD NOT be used for actual billing as it may also contain approximated values.	This bill type allows to summarize a recharging cycle, e.g. containing total energy charged, total costs as well as billing positions for different types of energy.

Table 29: Element "billType" value rules

Element "**costType**" value rules:

Value	Rule	Description
absolutePrice	None.	Used for absolute prices.
relativePrice	None.	Used for relative prices, e.g. a price that shows how much cheaper the costs are related to the average tariff in place.
co2Emission	None.	In this case the cost value is used to express how much CO2 emission was emitted.
renewableEnergy	None.	In this case the cost value is used to express how much renewable energy was used.
radioactiveWaste	None.	In this case the cost value is used to express how much radioactive waste was generated.

Table 30: Element "costType" value rules

Element "**positionType**" value rules:

Value	Rule	Description
gridElectricEnergy	None.	This position type specifically describes the electricity grid energy portion of the total data.
selfProducedElectricEnergy	None.	This position type specifically describes the self produced energy portion of the total data.

Table 31: Element "positionType" value rules

4.3.4.5 billConstraintsListData**4.3.4.5.1 Description**

The billConstraintsListData function allows a server to define certain constraints for each linked billListData list entry. This is especially important if a particular billListData list entry is writeable and allows the server to communicate the corresponding constraints before a client performs a write command. The client can match the server constraints with own constraints and perform a corresponding write command.

4.3.4.5.2 Rules

None.

4.3.4.5.3 Element rules

Element	Rule	Description
billId	SHALL be set as PRIMARY IDENTIFIER.	Allows the identification of a bill. Allows linking of the different functions to the same bill. Allows linking of the bill within other features that are placed in the same entity.
positionCountMin	If a related billData exists, it SHALL NOT contain less positions.	Minimum amount of possible positions within the specified bill.
positionCountMax	If a related billData exists, it SHALL NOT contain more positions.	Maximum amount of possible positions within the specified bill.

Table 32: billConstraintsListData Element rules

4.3.4.6 billDescriptionListData**4.3.4.6.1 Description**

The billDescriptionListData function is especially important if a linked billListData list entry is writeable and allows a server to define if a bill is writeable and which billTypes are supported for the corresponding billId, so that a client can perform corresponding writes. Additionally, the server can request updates from a client with the updateRequired flag.

4.3.4.6.2 Rules

None.

4.3.4.6.3 Element rules

Element	Rule	Description
billId	SHALL be set as PRIMARY IDENTIFIER.	Allows the identification of a bill. Allows linking of the different functions to the same bill. Allows

		linking of the bill within other features that are placed in the same entity.
billWriteable	If set to "true", a client MAY write the related billData entry. If there is no billData entry with a corresponding billId, the client MAY still write the complete billData entry and the server SHALL accept it, if the write command matches the rules given in the related billDescriptionData and billConstraintsData entries for the corresponding billId.	Indicates whether the related billData entry is writeable.
updateRequired	With updateRequired the server can request an update of writeable or changeable data related to the same PRIMARY IDENTIFIER from a client. The server SHALL ensure that only one responsible client is permitted to update the related data. To request an update the server SHALL set updateRequired to "true". Note: In this case, the server expects the responsible client to update the writeable or changeable data related to the same PRIMARY IDENTIFIER. However, also if updateRequired is set to "false" a server SHOULD in general allow updates of the data from the responsible client. The server SHALL set the updateRequired back to "false", as soon as "billDescriptionData" was updated successfully (if writeable or changeable) OR the update of the other writeable or changeable data related to the same PRIMARY IDENTIFIER was successful. The server MAY choose to withdraw the update request at any time by setting updateRequired back to "false". Note: The client does not need to stop an ongoing update process (e.g. if multiple functions are written), when updateRequired is set back to "false".	Indicates whether the server requests an update for the bill.
supportedBillType (list)	None.	List of all supported types for this billId.

Table 33: billDescriptionListData Element rules

4.3.4.7 Descriptive information and further definitions

None.

2550

2551 **4.3.5 DataTunneling**2552 **4.3.5.1 Dependencies to other Feature Types**

2553 None.

2554

2555 **4.3.5.2 Introduction**

2556 DataTunneling provides a mechanism to submit data of proprietary protocols using the SPINE data
2557 models (esp. addressing other nodes). However, the nature of proprietary definitions imposes
2558 restrictions:

2559 Unless stated otherwise, the term “payload” refers to the sub-element of "dataTunnelingCall" within
2560 this section (in contrast to the definition of “payload” as sub-element of “datagram” as defined in
2561 document [ProtocolSpecification]).

2562 Please note that the use of the above mentioned elements will always be proprietary. The SPINE
2563 specification will never define any specific protocol for dataTunneling.

2564 To put it in other words, the SPINE data tunnelling model is not intended for any kind of flexible
2565 extension of the SPINE classes. This is mainly because a so-called technology specific mapping could
2566 hardly be defined in a standardized way. Another reason is that it shall always be transparent that
2567 data tunnelling contains proprietary data. This should not be considered a limitation of SPINE as the
2568 EEBus Initiative e.V. committees may define specific classes with functions (like messagingListData,
2569 timeInformationData, e.g.) including mappings for specific protocols, if need be.

2570

2571 **4.3.5.3 Rules**

2572 None.

2573

2574 **4.3.5.4 dataTunnelingCall**2575 **4.3.5.4.1 Description**

2576 In general, several conditions must be fulfilled in order to make use of SPINE data tunnelling. This is
2577 explained with an example:

2578 A manufacturer "X" of a device "A" created a proprietary extension of a communications protocol
2579 (e.g. KNX, ZigBee, LON, ... – we call it "CP" here) in order to configure device "A" with special
2580 parameters. We assume there is no specific SPINE function defined so far for these parameters.
2581 There is also no mapping defined for it.

2582 Another manufacturer "Y" creates a device "B" that runs "SPINE applications" (i.e. applications
2583 communicating with an internal interface using XMLs of the SPINE data models). The device "B" can
2584 as well communicate via "CP".

2585 Now, manufacturer "Y" wants to make use of the proprietary configuration of device "A". Therefore,
2586 manufacturer "Y" needs to implement the proprietary extension of manufacturer "X"'s
2587 communications protocol as well. I.e. the "CP" implementation of device "B" must be extended
2588 properly. Furthermore, manufacturer "Y" needs to define how the proprietary protocol shall be
2589 represented as SPINE data tunnelling towards the "SPINE applications" running on device "B" (i.e.
2590 define the proprietary mapping) and of course needs to implement this mapping. Finally, the
2591 applications need to implement the recognition and use of the corresponding SPINE data tunnelling
2592 XMLs.

2593 The scenario above may look "unattractive" or even raise concerns. However, it shall be mentioned
2594 that proprietary extensions of standardized protocols are not unusual. This practice shall not be
2595 judged here.

2596 In the aforementioned scenario it is manufacturer "B" who implements a number of extensions in
2597 order to comply with the extensions of device "A". But it could also be possible that manufacturer
2598 "A" already provided some definitions and implementation for other manufacturers. Then,
2599 manufacturer "B" could integrate this solution and potentially implements less by its own.

2600 Some may find a comparison with Virtual Private Networks helpful (though VPNs have a different
2601 scope): Two computers "A", "B" shall each run a compatible (proprietary) program (i.e.
2602 mapping/implementation) that know how to communicate with each other (i.e. set up the
2603 communication and provide a specific authentication and encryption). There are manufacturers of
2604 VPN gateways that provide client software for different operating systems. This way a proprietary
2605 implementation is made available for others.

2606 The combination { `purposeld`, `channelld`, `sequenceld` } should be considered unique for a
2607 communication from "A" to "B". Within a system (e.g. house) several communications may be used
2608 in parallel. The combinations used in these communications are in general independent of each
2609 other.

2610 Please note that an instance of a `dataTunnelingCall` implementation permits bidirectional
2611 communication in general. I.e. if a `dataTunnelingCall` functionality of "B" submits a message to a
2612 `dataTunnelingCall` functionality of "A", it is also permissive that the `dataTunnelingCall` functionality of
2613 "A" submits `dataTunnelingCall` messages to the `dataTunnelingCall` functionality of "B". However,
2614 similar to a VPN, it may well be that the establishment for such message exchanges is
2615 "unidirectional" to some extent. I.e. it may be that the `dataTunnelingCall` functionality of "A"
2616 determines which kind of messages (or permissive values for "`purposeld`") are accepted on this
2617 functionality and that the functionality first of all acts as some kind of "listening port". This means
2618 the functionality of "A" may wait for an initial incoming message before it also sends
2619 `dataTunnelingCall` messages back.

2620 Please note that the example above does not prevent device "B" from implementing another
2621 `dataTunnelingCall` functionality as "listening port" with own permitted values for "`purposeld`" as well.
2622 In that case device "A" could send a proper initial message to this "listening functionality" of "B".

2623 SPINE data tunnelling is intended to be used for call-operations only. No read-reply or write
2624 procedures are supported.

2625

2626 4.3.5.4.2 *Rules*
 2627 None.

2628

2629 4.3.5.4.3 *Element rules*

Element	Rule	Description
header	SHALL be set.	A data tunneling specific header.
header. purposeld	SHALL be set as PRIMARY IDENTIFIER. The string-length SHOULD NOT be longer than 128 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 128 characters.	A string that identifies the purpose / kind of tunnelled data / protocol.
header. channelld	SHALL be set as SUB IDENTIFIER.	Between two nodes there might be several channels used in parallel using the same purposeld. channelld helps to separate these channels. It can also be used to differentiate subsequent (rather than parallel) sessions.
header. sequenceld	SHALL be set as SUB IDENTIFIER.	This helps to give the payload for each combination of { purposeld, channelld } a chronological order. Note: This “sequenceld” is data tunneling specific and should not be confused with “sequenceld” elements of other classes!
payload	SHALL be set.	The actual proprietary data.

2630 Table 34: dataTunnelingCall Element rules

2631

2632 4.3.5.5 *Descriptive information and further definitions*

2633 The DataTunneling Feature Type permits exchange of arbitrary data.

2634 Only "call" operations are permitted with this Feature Type.

2635 Unlike “conventional” features (i.e. features with functions where read/reply or write operations are
 2636 supported) the DataTunneling feature does not apply the “data owner” concept in a usual way at
 2637 first glance. However, the role concept still applies as follows:

2638 An instance of Feature Type DataTunneling and role “server” defines which kind of data is
 2639 exchanged. Specifically, the permitted values for the element “header.purposeld” (if used) are
 2640 determined by the server feature only.

2641

2642 4.3.6 *DeviceClassification*

2643 4.3.6.1 *Dependencies to other Feature Types*

2644 None.

2645

2646 **4.3.6.2 Introduction**

2647 Each device can have associated information useful for its identification and which kind of power it
 2648 requires. This information (mostly plaintext) is bundled in the *DeviceClassification* class. It is intended
 2649 to describe the physical device, but may be used for logical devices (i.e. *entities*), too.

2650

2651 **4.3.6.3 Rules**

2652 None.

2653

2654 **4.3.6.4 deviceClassificationManufacturerData**2655 **4.3.6.4.1 Description**

2656 Information which is usually NOT writable (hard coded in the device or preconfigured information,
 2657 e.g.), is modelled with *deviceClassificationManufacturerData*.

2658

2659 **4.3.6.4.2 Rules**

2660 No specific element is mandatory, but at least one SHALL be set.

2661 For visualisation purposes, the content of the element *vendorName* SHOULD be used as name of the
 2662 vendor only if the element *brandName* is not set. Otherwise *brandName* SHOULD be used.

2663

2664 **4.3.6.4.3 Element rules**

Element	Rule	Description
deviceName	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	The name of the (physical or logical) device as defined by the manufacturer.
deviceCode	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	A device code for the (physical or logical) device as defined by the manufacturer.

serialNumber	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	The serial number of the (physical or logical) device as defined by the manufacturer. Usually the same as printed on the case.
softwareRevision	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	The software revision of the (physical or logical) device as defined by the manufacturer.
hardwareRevision	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	The hardware revision of the (physical or logical) device as defined by the manufacturer.
vendorName	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	The name of the vendor of the (physical or logical) device as defined by the manufacturer.
vendorCode	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	A code for the vendor of the (physical or logical) device as defined by the manufacturer.

brandName	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	The name of the brand. Useful where the name of the brand and the vendor differs.
powerSource	Value rules of Table 36 SHALL be applied.	This element describes for which kind of primary power source the device is designed for. It is only used for informative purposes. It is NOT used to describe how or to which kind of power source the device is finally connected (this means it is not considered here whether the device's single mains is fed by a house's mains supply or a backup system, e.g.).
manufacturerNodeIdentification	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	A node identification for the (physical or logical) device as defined by the manufacturer. This could be used for the identification of a (physical or logical) device, even if it was removed from the network and rejoined later with changed node address.
manufacturerLabel	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the (physical or logical) device as defined by the manufacturer.
manufacturerDescription	Default rules of section 3.10.1.3.1 SHALL be applied.	A description for the (physical or logical) device as defined by the manufacturer.

2665 Table 35: deviceClassificationManufacturerData Element rules

2666 Element "**powerSource**" value rules:

Value	Rule
unknown	None.
mainsSinglePhase	None.
mains3Phase	None.
battery	None.
dc	None.

2667 Table 36: Element "powerSource" value rules

2668

4.3.6.5 deviceClassificationUserData**4.3.6.5.1 Description**

Information which is usually writable (changeable by a user / application) is modelled with *deviceClassificationUserData*.

4.3.6.5.2 Rules

No specific element is mandatory, but at least one SHALL be set.

4.3.6.5.3 Element rules

Element	Rule	Description
userNodeIdentification	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	Similar to <i>manufacturerNodeIdentification</i> but defined by a user or application.
userLabel	Default rules of section 3.10.1.2.1 SHALL be applied.	Similar to <i>manufacturerLabel</i> but defined by a user or application.
userDescription	Default rules of section 3.10.1.3.1 SHALL be applied.	Similar to <i>manufacturerDescription</i> but defined by a user or application.

Table 37: *deviceClassificationUserData* Element rules

4.3.6.6 Descriptive information and further definitions

None.

4.3.7 DeviceConfiguration**4.3.7.1 Dependencies to other Feature Types**

None.

4.3.7.2 Introduction

Some information cannot be represented by the "normal" SPINE classes, but are important for the interoperable functionality of a device, hence should not be modelled with the *DataTunneling* class (see section 5.3.6) or other proprietary ways. For this purpose, the *DeviceConfiguration* class should be used. It provides a key-value based modelling of information. Some data, important for interoperability, is already defined by the EEBus Initiative e.V. and listed in Table 187.

The information is split into a rather static part (see section 4.3.7.5) and the actual value (see section 4.3.7.4).

2696 **4.3.7.3 Rules**

2697 None.

2698

2699 **4.3.7.4 deviceConfigurationKeyValueListData**2700 **4.3.7.4.1 Description**

2701 The actual values belonging to a specific key are modelled within this function.

2702

2703 **4.3.7.4.2 Rules**

2704 None.

2705

2706 **4.3.7.4.3 Element rules**

Element	Rule	Description
keyId	SHALL be set as PRIMARY IDENTIFIER.	Enables the identification of different keys on one SPINE feature.
value	Exactly one of the child elements SHALL be set. This SHALL match with the content of element <i>valueType</i> within the key value description part (see section 5.3.8.3.2). Value rules of Table 39 SHALL be applied.	The actual value belonging to the <i>keyId</i> .
isValueChangeable	If set to "true" the server SHALL accept changes of the element "value" by appropriate clients (e.g. the bound client). Otherwise, the server SHALL decline change requests.	States whether the value belonging to the <i>keyId</i> is changeable by a client or not.

2707 Table 38: deviceConfigurationKeyValueListData Element rules

2708 Element "**value**" sub-element rules:

Element	Rule	Description
boolean	If the Element "boolean" is set, the "valueType" Element within the related deviceConfigurationDescriptionData entry SHALL be set to "boolean".	Used, if the <i>value</i> is of type <i>boolean</i> .
date	If the Element "date" is set, the "valueType" Element within the related deviceConfigurationDescriptionData entry SHALL be set to "date".	Used, if the <i>value</i> is of type <i>date</i> .
dateTime	If the Element "dateTime" is set, the "valueType" Element within the related deviceConfigurationDescriptionData entry SHALL be set to "dateTime".	Used, if the <i>value</i> is of type <i>dateTime</i> .
duration	If the Element "duration" is set, the "valueType" Element within the related deviceConfigurationDescriptionData entry SHALL be set to "duration".	Used, if the <i>value</i> is of type <i>duration</i> .
string	If the Element "string" is set, the "valueType" Element within the related deviceConfigurationDescriptionData entry SHALL be set to "string".	Used, if the <i>value</i> is of type <i>string</i> .

	The string-length of the Element SHOULD NOT be longer than 512 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 512 characters.	
time	If the Element "time" is set, the "valueType" Element within the related deviceConfigurationDescriptionData entry SHALL be set to "time".	Used, if the <i>value</i> is of type <i>time</i> .
scaledNumber	If the Element "scaledNumber" is set, the "valueType" Element within the related deviceConfigurationDescriptionData entry SHALL be set to "scaledNumber".	Used, if the <i>value</i> is of type <i>scaledNumber</i> .

Table 39: Element "value" sub-element rules

4.3.7.5 deviceConfigurationKeyValueDescriptionListData

4.3.7.5.1 Description

The rather static part of the device configuration keys is modelled with this function, including an identifier for the key, a name that defines the meaning of the key and further additional elements.

4.3.7.5.2 Rules

None.

4.3.7.5.3 Element rules

Element	Rule	Description
keyId	SHALL be set as PRIMARY IDENTIFIER.	The key identifier.
keyName	Value rules of Table 31 SHALL be applied.	A certain key name. If one of the keys defined by the EEBus Initiative e.V. is used, the meaning is defined in Table 187. Vendors may define own key names (but then have to mark them as vendor specific, see [ProtocolSpecification], section "Rules for vendor specific extensions").
valueType	Most <i>keyName</i> definitions specify which <i>valueType</i> shall be used. Value rules of Table 31 SHALL be applied.	The type of the "value" Element within the corresponding "deviceConfigurationKeyValueData" entry is denoted here.
unit	If set, the unit SHALL be applied to the value of the key.	The unit in which the value of the key is given. Not all keys need the element "unit".
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A label for this key.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	A description for this key.

Table 40: deviceConfigurationKeyValueDescriptionListData Element rules

2721 Element "**keyName**" value rules:

Value	Rule	Description
pvCurtailementLimitFactor	This value SHALL be in range of 0 to 100. If used, the content of the element "valueType" SHALL be "scaledNumber" and the element "unit" SHALL be set to "pct".	The percentage of the peak power of the pv system that is permitted to be fed into the electricity grid through the electricity grid connection point.
peakPowerOfPvSystem	This value SHALL be positive or 0. If used, the content of the element "valueType" SHALL be "scaledNumber" and the element "unit" SHALL be set to "W".	The nominal peak power of the production of the installed PV System.
asymmetricChargingSupported	If used, the content of the element "valueType" SHALL be "boolean" and the element "unit" SHALL be omitted.	This configuration parameter is used to indicate if it is possible to charge with different current per phase.
communicationsStandard	If used, the content of the element "valueType" SHALL be "string" and the element "unit" SHALL be omitted. As content of the element "value.string" within the linked deviceConfigurationKeyValueData entry, only one of the following strings SHALL be used: <ul style="list-style-type: none"> - iso15118-2ed1 - iso15118-2ed2 - iec61851 	Defines the used communications standard for the entity where the DeviceConfiguration feature is located.

2722 Table 41: Element "**keyName**" value rules2723 Element "**valueType**" value rules:

Value	Rule	Description
boolean	SHALL be used, if the corresponding key value is of type xs:boolean.	The corresponding key value is of type xs:boolean.
date	SHALL be used, if the corresponding key value is of type xs:date.	The corresponding key value is of type xs:date.
dateTime	SHALL be used, if the corresponding key value is of type xs:dateTime.	The corresponding key value is of type xs:dateTime.
duration	SHALL be used, if the corresponding key value is of type xs:duration.	The corresponding key value is of type xs:duration.
string	SHALL be used, if the corresponding key value is of type xs:string.	The corresponding key value is of type xs:string.
time	SHALL be used, if the corresponding key value is of type xs:time.	The corresponding key value is of type xs:time.
scaledNumber	SHALL be used, if the corresponding key value is of type " <i>ScaledNumberType</i> " (see section 3.10.1.8).	The corresponding key value is of type " <i>ScaledNumberType</i> " (see section 3.10.1.8).

2724 Table 42: Element "**valueType**" value rules

2725

2726 **4.3.7.6 deviceConfigurationKeyValueConstraintsListData**2727 **4.3.7.6.1 Description**

2728 In cases where a configuration value has some range, it could make sense to specify the upper and
 2729 lower bounds that are possible for this value as well as a step size. This can be achieved with the
 2730 deviceConfigurationKeyValueConstraintsListData function.

2731

2732 **4.3.7.6.2 Rules**

2733 None.

2734

2735 **4.3.7.6.3 Element rules**

Element	Rule	Description
keyId	SHALL be set as PRIMARY IDENTIFIER.	Enables the identification of different keys on one SPINE feature.
valueRangeMin	If set, the child element SHALL match with the content of the Element <i>valueType</i> within the related deviceConfigurationDescriptionData entry. For rules on the sub-elements see Table 39.	The lower bound that is possible for the related value.
valueRangeMax	If set, the child element SHALL match with the content of the Element <i>valueType</i> within the related deviceConfigurationDescriptionData entry. For rules on the sub-elements see Table 39.	The upper bound that is possible for the related value.
valueStepSize	If set, the child element SHALL match with the content of the Element <i>valueType</i> within the related deviceConfigurationDescriptionData entry. For rules on the sub-elements see Table 39.	The step size that is possible for the related value.

2736 Table 43: deviceConfigurationKeyValueListData Element rules

2737

2738 **4.3.8 DeviceDiagnosis**2739 **4.3.8.1 Dependencies to other Feature Types**

2740 None.

2741

2742 **4.3.8.2 Introduction**

2743 This class contains information about the state of a device, heartbeat counting and service data.

2744

2745 **4.3.8.3 Rules**

2746 None.

2747

2748 **4.3.8.4 deviceDiagnosisStateData**2749 **4.3.8.4.1 Description**

2750 The *deviceDiagnosisStateData* function provides information about dynamically changing state
 2751 information.

2752

2753 **4.3.8.4.2 Rules**

2754 No specific element is mandatory, but at least one SHALL be set.

2755

2756 **4.3.8.4.3 Element rules**

Element	Rule	Description
timestamp	None.	The time of creation of the state.
operatingState	Value rules of Table 45 SHALL be applied.	The current operating state of the device.
vendorStateCode	The string-length SHOULD NOT be longer than 128 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 128 characters.	A vendor specific state. May contain any state the vendor uses for its devices.
lastErrorCode	The string-length SHOULD NOT be longer than 128 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 128 characters. Even if the device's "operationState" has a value of <i>normalOperation</i> again, the error code SHOULD remain in the element lastErrorCode.	The last error code that occurred.
upTime	If used, the following rules apply: - If located on entity "DeviceInformation" (see section 4.2.6), the up time of the complete device SHALL be stated here. - If located on some other entity, only the up time of the specific functionality SHALL be stated. In case of nested entities, the up	The duration the functionality is running since the last start.

	time of all "child"-entities SHALL be considered, too.	
totalUpTime	If used the following rules apply: - If located on entity "DeviceInformation" (see section 4.2.6), the total up time of the complete device SHALL stated here. - If located on some other entity, only the total up time of the specific functionality SHALL be stated. In case of nested entities, the total up time of all "child"-entities SHALL be considered, too.	The total duration the functionality is running since installation (see <i>installationTime</i> element in <i>deviceDiagnosisServiceData</i> function).
powerSupplyCondition	Value rules of Table 46 SHALL be applied.	The condition of the power supply can be stated here.

2757 Table 44: *deviceDiagnosisStateData* Element rules2758 Element "**operatingState**" value rules:

Value	Rule	Description
normalOperation	None.	The device does its normal operation without any errors or limitations.
standby	None.	The device is in standby mode and does nothing but waits for user interaction and listens on its communications protocol for external commands.
failure	None.	A failure occurred and the device cannot run as desired. Its normal functionality is likely not given.
serviceNeeded	None.	The device needs some kind of service. Possibly the normal functionality is not given.
overrideDetected	None.	The device detected an override. Normal operation may be given or the device is in some safe mode.
inAlarm	None.	An alarm (or emergency) occurred and user interaction is needed. Normal operation may be given or the device is in some safe mode.
notReachable	None.	The device is not reachable via its non-SPINE communications protocol. This operating state can only be set by some technology gateway device that handles the normal communication to the device.
finished	None.	The device has finished its temporary operation. As long as it remains in state <i>finished</i> , the normal operation is not given.

2759 Table 45: Element "*operatingState*" value rules2760 Element "**powerSupplyCondition**" value rules:

Value	Rule	Description
good	None.	The power supply is in a good condition.
low	None.	The power supply has low capacity. It would make sense to check it (e.g. change battery).
critical	None.	The power supply is in critical state. If it is supplied by a battery, a change is advised.
unknown	None.	The power supply condition is unknown. A notification to a user could make sense.

error	None.	The power supply is in error condition.
-------	-------	---

Table 46: Element "powerSupplyCondition" value rules

4.3.8.5 deviceDiagnosisHeartbeatData

4.3.8.5.1 Description

The *deviceDiagnosisHeartbeatData* function enables sending a specific signal, which indicates a device is still running.

4.3.8.5.2 Rules

No specific element is mandatory, but at least one SHALL be set.

4.3.8.5.3 Element rules

Name	Rule	Description
timestamp	None.	The time of creation of the data.
heartbeatCounter	The value of the heartbeatCounter element SHALL be increased after every <i>heartbeatTimeout</i> (NOT with every sending of this function). The <i>deviceDiagnosisHeartbeatData</i> function can not only be sent initially by the device itself, but can be requested by another device, too. In this case, the element <i>heartbeatCounter</i> SHALL NOT be incremented and the <i>heartbeatTimeout</i> has (as always) its fixed value (i.e. not the remaining time to the next (automatic) notification by the device).	An incrementing counter of the heartbeat.
heartbeatTimeout	A notification of the <i>deviceDiagnosisHeartbeatData</i> function SHALL be sent to all subscribed clients after each heartbeatTimeout period.	The heartbeatTimeout element contains the period, in which the <i>deviceDiagnosisHeartbeatData</i> function is sent by the device (NOT the remaining time till the next sending).

Table 47: deviceDiagnosisHeartbeatData Element rules

4.3.8.6 deviceDiagnosisServiceData

4.3.8.6.1 Description

The *deviceDiagnosisServiceData* function informs about service information related to a device's installation or maintenance.

2779 **4.3.8.6.2 Rules**

2780 No specific element is mandatory, but at least one SHALL be set.

2781

2782 **4.3.8.6.3 Element rules**

Name	Rule	Description
timestamp	None.	The time of creation of the data.
installationTime	None.	The point in time the device was installed (commissioned).
bootCounter	None.	The count of boot processes the device completed since <i>installationTime</i> .
nextService	None.	The point in time, the next service should be done, is stated by this element. Alternatively, the decrementing duration till the next service can be indicated.

2783 *Table 48: deviceDiagnosisServiceData Element rules*

2784

2785 **4.3.8.7 Descriptive information and further definitions**

2786 None.

2787

2788 **4.3.9 DirectControl**2789 **4.3.9.1 Dependencies to other Feature Types**

- 2790 - OperatingConstraints
- 2791 - PowerSequences
- 2792 - SmartEnergyManagementPs

2793

2794 **4.3.9.2 Introduction**

2795 Some kind of electricity consuming or producing devices require (in contrast to the PowerSequences
 2796 Class, see section 5.3.19) a rather instantaneous energy consumption or generation control. This can
 2797 be modelled with two primary information sets:

- 2798 1. Notification of instantaneous consumption/generation as well as instantaneous control. This
 2799 is covered by DirectControl and discussed in this section.
- 2800 2. Information on the power/energy constraints and implications. This is covered by
 2801 OperatingConstraints and discussed in section 4.3.20.

2802 Information on and control of energy/power can be modelled using the directControlActivityListData
 2803 Function. The directControlDescriptionData Function is used to model some general rather constant
 2804 data.

2805

2806 **4.3.9.3 Rules**

2807 None.

2808

2809 4.3.9.4 *directControlActivityListData*

2810 4.3.9.4.1 *Description*

2811 The function *directControlActivityListData* combines some information on the activity of a given time.
 2812 Apart from information on state, power and energy it can contain information whether power,
 2813 energy, etc. can be modified by a client with a proper (partial) write command.

2814

2815 4.3.9.4.2 *Rules*

2816 The current values SHALL be represented by a list entry with timestamp = "zero seconds" (e.g. "PT0S"
 2817 or "PT0H"). The server SHALL only accept write commands on data instances with timestamp = "zero
 2818 seconds" (e.g. "PT0S" or "PT0H") to adjust current values of the Function.

2819

2820 4.3.9.4.3 *Element rules*

Name	Rules	Description
timestamp	SHALL be set as PRIMARY IDENTIFIER. The most actual <i>directControlActivityData</i> SHALL have a duration (i.e. relative time to "now") of zero seconds.	The timestamp of this <i>directControlActivityData</i> .
activityState	Value rules of Table 50 SHALL be applied.	The state of the activity.
isActivityStateChangeable	If this element is set to "true" the device's <i>activityState</i> MAY be modified by a client.	Indicates whether the "activityState" can be modified by a client or not.
energyMode	For further value rules see section 3.10.1.16.1.	Energy mode of the device.
isEnergyModeChangeable	If this element is set to <i>true</i> the device's "energyMode" MAY be modified by a client.	Indicates whether the "energyMode" can be modified by a client or not.
power	Target value that SHOULD be followed as good as possible.	The overall electrical power value belonging to this activity information. Please note: Phase-specific information is not given here.
isPowerChangeable	If this element is set to <i>true</i> the device's "power" MAY be modified by a client.	Indicates whether the "power" can be modified by a client or not.
energy	Target value that SHOULD be followed as good as possible.	The electrical energy value belonging to this activity information.
isEnergyChangeable	If this element is set to <i>true</i> the device's "energy" MAY be modified by a client.	Indicates whether the "energy" can be modified by a client or not.
sequenceId	If the activity is related to a power sequence, the according <i>sequenceId</i> SHOULD be stated here as FOREIGN IDENTIFIER.	The related power sequence identifier (if available).

2821 Table 49: *directControlActivityListData* Element rules

2822 Element "**activityState**" value rules:

Value	Rule	Description
running	If <i>isActivityStateChangeable</i> is set to <i>true</i> , it MAY be paused or deactivated by a client (if <i>sequenceId</i> is set, further information from the related sequence SHALL be considered).	The activity is running.
paused	The activity is paused. If <i>isActivityStateChangeable</i> is set to <i>true</i> , it MAY be activated or deactivated by a client (if <i>sequenceId</i> is set, further information from the related sequence SHALL be considered).	The activity is paused.
inactive	The activity is inactive. If <i>isActivityStateChangeable</i> is set to <i>true</i> , it MAY be activated by a client (if <i>sequenceId</i> is set, further information from the related sequence SHALL be considered).	The activity is inactive.

2823 Table 50: Element "activityState" value rules

2824

2825 **4.3.9.5 directControlDescriptionData**

2826 *4.3.9.5.1 Description*

2827 This function contains the rather static information available in this Feature Type.

2828

2829 *4.3.9.5.2 Rules*

2830 None.

2831

2832 *4.3.9.5.3 Element rules*

Element	Rule	Description
positiveEnergyDirection	SHOULD be set. For further value rules see section 3.10.1.14.1.	The <i>positiveEnergyDirection</i> states whether energy consumption or production will be counted as positive value.
powerUnit	SHOULD be set, if a power is stated in the function <i>directControlActivityListData</i> .	This is the unit for all power values related to consumption or production.
energyUnit	SHOULD be set, if an energy is stated in the function <i>directControlActivityListData</i> .	This is the unit for all energy values related to consumption or production.

2833 Table 51: *directControlActivityListData* Element rules

2834

2835 **4.3.9.6 Descriptive information and further definitions**

2836 None.

2837

2838 **4.3.10 ElectricalConnection**

2839 **4.3.10.1 Dependencies to other Feature Types**

2840 - Measurement

2841

2842 **4.3.10.2 Introduction**

2843 Electrical connections have some more complex parameters (e.g. which AC phase is measured,
 2844 real/reactive/apparent measurement, amplitude/rms, number of measured harmonic, etc.), which
 2845 are modelled with this class. Some electrical connection specific state information (e.g. AC or DC,
 2846 number of connected phases (only AC), positive energy direction, label, etc.) is modelled, too.

2847 Measuring electrical values is still done with the Measurement class (see section 5.3.15). But the
 2848 above-mentioned additional parameters are only described within the ElectricalConnection class.
 2849 The *measurementId* from the Measurement class is stated in the
 2850 *electricalConnectionParameterDescriptionListData* function, to have a proper reference.

2851

2852 **4.3.10.3 Rules**

2853 None.

2854

2855 **4.3.10.4 electricalConnectionParameterDescriptionListData**2856 **4.3.10.4.1 Description**

2857 The aforementioned parameters for electrical connection measurement are described in this
 2858 function. The value itself is modelled in the Measurement class (section 5.3.15). Only a reference is
 2859 given in this function.

2860

2861 **4.3.10.4.2 Rules**

2862 None.

2863

2864 **4.3.10.4.3 Element rules**

Element	Rule	Description
electricalConnectionId	SHALL be set as PRIMARY IDENTIFIER.	Reference to the electrical connection, described in the <i>electricalConnectionDescriptionListData</i> function (see section 5.3.11.5).
parameterId	SHALL be set as SUB IDENTIFIER.	One electrical connection can have multiple parameter combinations. The <i>parameterId</i> helps to distinguish them.
measurementId	The FOREIGN IDENTIFIER MAY be set. If set, the related electrical connection data SHALL be linked to a measurand or data of another Feature that uses the same measurementId.	This is a foreign identifier.
voltageType	If <i>powerSupplyType</i> in <i>electricalConnectionDescriptionData</i> has the value "dc", this element MAY be omitted. If stated it SHALL be set	Specifies which kind of electricity is measured ("ac" or "dc").

	to "dc". If <i>powerSupplyType</i> is of value "ac", the <i>voltageType</i> can either have the value "ac" (<i>voltageType</i> then MAY be omitted), or "dc". In the latter case, the DC-offset of the ac connection is measured and this element SHALL be set accordingly. Value rules of Table 53 SHALL be applied	
acMeasuredPhases	In case of <i>powerSupplyType</i> = "dc", this element SHALL be omitted. Value rules of Table 54 SHALL be applied.	In case of <i>powerSupplyType</i> = "ac", this element states the phase, which is measured. Combinations of phases (e.g. "ab") are possible.
acMeasuredInReferenceTo	In case of <i>powerSupplyType</i> = "dc", this element SHALL be omitted. Value rules of Table 55 SHALL be applied.	In case of <i>powerSupplyType</i> = "ac", this element states the phase, which <i>acMeasuredPhases</i> is measured against (e.g. "neutral").
acMeasurementType	In case of <i>powerSupplyType</i> = "dc", this element SHALL be omitted. Value rules of Table 56 SHALL be applied.	In case of <i>powerSupplyType</i> = "ac", this element states the kind of ac measurement is done (e.g. "real").
acMeasurementVariant	In case of <i>powerSupplyType</i> = "dc", this element SHALL be omitted. Value rules of Table 57 SHALL be applied.	In case of <i>powerSupplyType</i> = "ac", this element states the variation of the ac measurement (e.g. "amplitude").
acMeasuredHarmonic	In case of <i>powerSupplyType</i> = "dc", this element SHALL be omitted.	In case of <i>powerSupplyType</i> = "ac", this element states the harmonic, which is measured. A value of "1" indicates that the normal value is measured.
scopeType	None.	A certain meaning of the parameter combination.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A label for this electrical connection parameter.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	A description for this electrical connection parameter.

2865 Table 52: *electricalConnectionParameterDescriptionListData* Element rules2866 Element "**voltageType**" value rules:

Value	Rule	Description
ac	SHALL NOT be used if <i>powerSupplyType</i> in <i>electricalConnectionDescriptionData</i> has the value "dc".	Alternating current.
dc	None.	Direct current.

2867 Table 53: Element "*voltageType*" value rules2868 Element "**acMeasuredPhases**" value rules:

Value	rule	Description
a	None.	Phase a (or L1).
b	None.	Phase b (or L2).
c	None.	Phase c (or L3).
ab	None.	Phase a and b (or L1 and L2).

bc	None.	Phase b and c (or L2 and L3).
ac	None.	Phase a and c (or L1 and L3).
abc	None.	Phase a, b and c (or L1, L2 and L3).
neutral	None.	The neutral conductor of a connection.
ground	None.	The ground conductor of a connection.
none	None.	No phase specified or available.

2869 Table 54: Element "acMeasuredPhases" value rules

2870 Element "acMeasuredInReferenceTo" value rules:

Value	Description	Description
a	None.	Phase a (or L1).
b	None.	Phase b (or L2).
c	None.	Phase c (or L3).
ab	None.	Phase a and b (or L1 and L2).
bc	None.	Phase b and c (or L2 and L3).
ac	None.	Phase a and c (or L1 and L3).
abc	None.	Phase a, b and c (or L1, L2 and L3).
neutral	None.	The neutral conductor of a connection.
ground	None.	The ground conductor of a connection.
none	None.	No phase specified or available.

2871 Table 55: Element "acMeasuredInReferenceTo" value rules

2872 Element "acMeasurementType" value rules:

Value	Description	Description
real	None.	AC measurement of the real part.
reactive	None.	AC measurement of the reactive part.
apparent	None.	AC measurement of the apparent part.
phase	None.	AC measurement of the phase.

2873 Table 56: Element "acMeasurementType" value rules

2874 Element "acMeasurementVariant" value rules:

Value	Description	Description
amplitude	None.	Measurement of the amplitude.
rms	None.	Measurement of the RMS (Root Mean Square).
instantaneous	None.	Measurement of the instantaneous value.
angle	None.	Measurement of the angle.
cosPhi	None.	Measurement of the cosine phi.

2875 Table 57: Element "acMeasurementVariant" value rules

2876

2877 **4.3.10.5 electricalConnectionPermittedValueSetListData**2878 **4.3.10.5.1 Description**

2879 If an electrical connection parameter has some defined value(s) or value range(s) that are allowed or
 2880 possible, they can be modelled with this function. It enables the server to state single values, value
 2881 ranges or combinations of both.

2882

2883 *4.3.10.5.2 Rules*

2884 None.

2885

2886 *4.3.10.5.3 Element rules*

Element	Rule	Description
electricalConnectionId	SHALL be set as PRIMARY IDENTIFIER.	Identifier of the electrical connection, this rather static information belongs to.
parameterId	SHALL be set as SUB IDENTIFIER.	Relation to the according parameter.
permittedValuesSet (list)	At least one set of permitted values SHALL be stated.	The permittedValueSet allows to define an arbitrary set of permitted data. Compared to a simple value range the permittedValueSet also may have gaps. Please refer to the ScaledNumberSetType for more details.

2887 *Table 58: electricalConnectionPermittedValueSetListData Element rules*

2888

2889 **4.3.10.6 electricalConnectionStateListData**2890 *4.3.10.6.1 Description*

2891 Some electrical connection specific state information is modelled in this function.

2892

2893 *4.3.10.6.2 Rules*

2894 None.

2895

2896 *4.3.10.6.3 Element rules*

Element	Rule	Description
electricalConnectionId	SHALL be set as PRIMARY IDENTIFIER.	Electrical connection, this state information belongs to.
timestamp	None.	Point in time, the state information was generated.
currentEnergyMode	For further value rules see section 3.10.1.16.1.	Whether a device is consuming, producing or idle is stated by this element.
consumptionTime	SHOULD be set to the duration the device is in energy mode "consume" since the last reboot.	The duration the device is in energy mode "consume" since the last reboot (see section 5.3.9.2).
productionTime	SHOULD be set to the duration the device is in energy mode "produce" since the last reboot.	The duration the device is in energy mode "produce" since the last reboot (see section 5.3.9.2).

totalConsumptionTime	SHOULD be set to the duration, the device is in energy mode "consume" since its <i>installationTime</i> .	The duration the device is in energy mode "consume" since its <i>installationTime</i> (see section 5.3.9.4).
totalProductionTime	SHOULD be set to the duration, the device is in energy mode "producing" since its <i>installationTime</i> .	The duration the device is in energy mode "producing" since its <i>installationTime</i> (see section 5.3.9.4).

Table 59: electricalConnectionStateListData Element rules

4.3.10.7 electricalConnectionDescriptionListData

4.3.10.7.1 Description

Some electrical connection specific information, that are rather static, are modelled within this function.

4.3.10.7.2 Rules

None.

4.3.10.7.3 Element rules

Element	Rule	Description
electricalConnectionId	SHALL be set as PRIMARY IDENTIFIER.	Electrical connection, this information belongs to.
powerSupplyType	SHOULD be set.	States whether the electrical connection is of type "ac" or "dc".
acConnectedPhases	If <i>powerSupplyType</i> = "ac", this element SHOULD be set.	If <i>powerSupplyType</i> ="ac", this element contains the number of phases, this electrical connection has (typically 1, 2 or 3).
acRmsPeriodDuration	None.	If the "rms" value of a parameter combination (see element <i>acMeasurementVariant</i> in section 5.3.11.2.2) is measured, this element states the duration, the rms value is determined in.
positiveEnergyDirection	SHOULD be set. If it is omitted, it SHALL be interpreted as <i>positiveEnergyDirection</i> = "consume".	This element states whether energy consumption or production will be counted as positive value.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A label for this electrical connection.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	A description for this electrical connection.

Table 60: electricalConnectionDescriptionListData Element rules

2910 **4.3.10.8 Descriptive information and further definitions**

2911 None.

2912

2913 **4.3.11 Generic**

2914 **4.3.11.1 Dependencies to other Feature Types**

2915 None.

2916

2917 **4.3.11.2 Introduction**

2918 None.

2919

2920 **4.3.11.3 Rules**

2921 None.

2922

2923 **4.3.11.4 Function and Identifier rules**

2924 None.

2925

2926 **4.3.11.5 Element rules**

2927 None.

2928

2929 **4.3.11.6 Descriptive information and further definitions**

2930 The Generic Feature Type does not contain any information. It SHALL only be used as a client feature!

2931 Such a client MAY send any read, write or call messages to servers from this feature. This can be very
2932 useful if a client wants to communicate with different Feature Types from the same address.

2933 Additionally, unnecessary duplications of server Feature Types on client side are avoided (e.g. it is
2934 not needed to have a Measurement and a Setpoint client Feature; instead, only this Generic Feature
2935 is used by the client). Therefore, it is preferable that a client uses the Generic Feature Type.

2936

2937 **4.3.12 HVAC**

2938 **4.3.12.1 Dependencies to other Feature Types**

- 2939 - Measurement
- 2940 - PowerSequences
- 2941 - Setpoint
- 2942 - SmartEnergyManagementPs
- 2943 - TimeTable

2944

4.3.12.2 Introduction

Heating, Ventilation and Air Conditioning (HVAC) devices have some special functionality which is covered by this class.

All system functions (like heating, cooling, domestic hot water preparation (dhw), ventilation, etc.) have their specific HVAC operation modes (like on, off, auto, etc.). Relations between these are modelled as well as relations to (optional) power sequences, which represent a concrete run of a device's program.

HVAC overruns (like an one time domestic-hot-water (DHW) preparation), which do have an impact on the normal operation mode of a HVAC system function, are modelled in further SPINE functions of the HVAC Class.

4.3.12.3 Rules

None.

4.3.12.4 hvacSystemFunctionListData**4.3.12.4.1 Description**

The hvacSystemFunctionListData function announces the current status of the HVAC system functions, including the current HVAC operation mode, a flag that indicates, if the HVAC operation mode can be changed by some external communications partner (client) and a flag that indicates whether an HVAC overrun is currently active or not.

4.3.12.4.2 Rules

None.

4.3.12.4.3 Element rules

Element	Rule	Description
systemFunctionId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for a specific HVAC system function.
currentOperationModeId	SHALL be set as FOREIGN IDENTIFIER to the according operationModeId of the currently selected operation mode. If the function hvacSystemFunctionOperationModeRelationListData is used, only a operationModeId value related there to the according systemFunctionId SHALL be used here.	States which HVAC operation mode is currently active for this specific HVAC system function
isOperationModeIdChangeable	If set to "true", a client MAY change the currentOperationModeId. If absent or set to	Denotes whether the

	"false" currentOperationModelId SHALL NOT be writeable.	current HVAC operation mode identifier is changeable (by some external communication partner) or not.
currentSetpointId	The setpointId SHALL be set here as FOREIGN IDENTIFIER, if a setpoint is linked to the system function. If the function hvacSystemFunctionSetpointRelationListData is used, only a setpointId value related there to the according systemFunctionId SHALL be used here.	States which setpoint is active for this specific HVAC system function
isSetpointIdChangeable	If no currentSetpointId is set, this element SHALL NOT be set. If set to "true", a client MAY change the currentSetpointId. If absent or set to "false" currentSetpointId SHALL NOT be writeable.	Denotes whether the current setpoint identifier is changeable (by some external communication partner) or not.
isOverrunActive	If an overrun is active that has an effect on this system function, this element SHOULD be set to "true".	If an HVAC overrun is currently active, this element is set to <i>true</i> . The HVAC overrun itself is modelled with other functions (see section 5.3.12.8 and 5.3.12.9) and may be located on a different feature or even a different entity or device.

Table 61: hvacSystemFunctionListData Element rules

4.3.12.5 hvacSystemFunctionOperationModeRelationListData**4.3.12.5.1 Description**

Each HVAC system function has its own relation to all for this HVAC system function available HVAC operation modes. E.g. there may be three HVAC operation modes available on the feature: *on*, *off*, *auto* (see section 5.3.12.7). HVAC System function 1 only supports HVAC operation mode 1 (*on*) and 2 (*off*). HVAC System function 2 supports all three HVAC operation modes. The enumeration of the available HVAC system functions is described in section 5.3.12.6.2.

4.3.12.5.2 Rules

None.

4.3.12.5.3 Element rules

Element	Rule	Description
systemFunctionId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for a specific HVAC system function.
operationModelId (list)	SHALL be set as FOREIGN IDENTIFIER (at least one) to the according operationModelId of the referenced operation mode.	Lists the HVAC operation modes that are available (selectable) for a specific HVAC system function.

Table 62: hvacSystemFunctionOperationModeRelationListData Element rules

4.3.12.6 hvacSystemFunctionSetpointRelationListData**4.3.12.6.1 Description**

The setpoint class (see section 5.3.21) may be used to model some HVAC relevant setpoints like desired room temperature. An HVAC system function may relate to some of these setpoints. The relation is modelled with this SPINE function.

4.3.12.6.2 Rules

None.

4.3.12.6.3 Element rules

Element	Rule	Description
systemFunctionId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for a specific HVAC system function.
operationModelId	SHALL be set as SUB IDENTIFIER.	Identifier for a specific HVAC operation mode.
setpointId (list)	MAY be set as FOREIGN IDENTIFIER to the according setpointId of the referenced setpoint.	List of setpoints that are related with a specific HVAC system function.

Table 63: hvacSystemFunctionSetpointRelationListData Element rules

2997

2998 **4.3.12.7 hvacSystemFunctionPowerSequenceRelationListData**2999 **4.3.12.7.1 Description**

3000 Power sequences (see section 5.3.19) describe tasks of a functionality in detail. An HVAC system
 3001 function may relate to some of these power sequences. The relation is modelled with this function.

3002

3003 **4.3.12.7.2 Rules**

3004 None.

3005

3006 **4.3.12.7.3 Element rules**

Element	Rule	Description
systemFunctionId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for a specific HVAC system function.
sequenceId (list)	MAY be set as FOREIGN IDENTIFIER to the according sequenceId of the referenced power sequence.	List of power sequences that are related with a specific HVAC system function.

3007 *Table 64: hvacSystemFunctionPowerSequenceRelationListData Element rules*

3008

3009 **4.3.12.8 hvacSystemFunctionDescriptionListData**3010 **4.3.12.8.1 Description**

3011 The rather static information about the HVAC system functions is modelled within this SPINE
 3012 function.

3013

3014 **4.3.12.8.2 Rules**

3015 None.

3016

3017 **4.3.12.8.3 Element rules**

Element	Rule	Description
systemFunctionId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for a specific HVAC system function.
systemFunctionType	SHALL be set. Value rules of Table 66 SHALL be applied.	Denotes the type of the HVAC system function.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the HVAC system function.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the HVAC system function.

3018 *Table 65: hvacSystemFunctionDescriptionListData Element rules*3019 Element "**systemFunctionType**" value rules:

Value	Rule	Description
heating	None.	HVAC system function for space heating.
cooling	None.	HVAC system function for space cooling.
ventilation	None.	HVAC system function for ventilation.
dhw	None.	HVAC system function for domestic hot water preparation.

Table 66: Element "systemFunctionType" value rules

4.3.12.9 hvacOperationModeDescriptionListData

4.3.12.9.1 Description

HVAC operation modes are used to describe which behaviour the HVAC system functions have. Each HVAC operation mode may be used by several HVAC system functions.

4.3.12.9.2 Rules

None.

4.3.12.9.3 Element rules

Element	Rule	Description
operationModelId	SHALL be set as PRIMARY IDENTIFIER.	Identifier of the HVAC operation mode.
operationModeType	SHALL be set. Value rules of Table 68 SHALL be applied.	Denotes the type of the HVAC operation mode.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the HVAC operation mode.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the HVAC operation mode.

Table 67: hvacOperationModeDescriptionListData Element rules

Element "**operationModeType**" value rules:

Value	Rule	Description	
auto	None.	HVAC System Function	Meaning
		heating / cooling / dhw / ventilation	HVAC System function is controlled depending on a time table (i.e. the current setpoint for the HVAC system function is changed depending on a time table)
on	None.	HVAC System Function	Meaning
		heating / cooling / dhw / ventilation	HVAC System function is controlled (e.g. Setpoint) independant from time tables (i.e. the current setpoint for the HVAC system function is not depending on any time table).
off	None.	HVAC System Function	Meaning
		heating / dhw	Protection mode against freezing (no physically turning off). The (temperature)

			setpoint for freezing protection may be stated with a setpointId.
		cooling	Space cooling function not active.
		ventilation	Ventilation function not active (protection functions like humidity protection may be active).
eco	None.	HVAC System Function	Meaning
		Heating / cooling / dhw / ventilation	HVAC System function is controlled (e.g. current Setpoint) independant from time tables (i.e. the current setpoint for the HVAC system function is not depending on any time table). HVAC System function is controlled by a different setpoint as for "on"

Table 68: Element "operationModeType" value rules

4.3.12.10 hvacOverrunListData

4.3.12.10.1 Description

The normal HVAC operation may be overridden by some HVAC specific functionality. There are some HVAC overruns defined by the EEBus Initiative e.V., but vendors may add further vendor-specific HVAC overruns, too (which are then out of interoperability scope).

When the HVAC overrun is active, some (not all! See section 5.3.12.9) HVAC system functions have a specific behaviour, other than the normal operation would be. After the HVAC overrun is inactive again, the HVAC system returns to its normal operation.

4.3.12.10.2 Rules

None.

4.3.12.10.3 Element rules

Element	Rule	Description
overrunId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for the HVAC overrun.
overrunStatus	SHALL be set. Value rules of Table 70 SHALL be applied.	The current status of the HVAC overrun.
timeTableId	None.	Reference to a time table. Needed, if an overrun has some start and end point(s).
isOverrunStatusChangeable	If absent, the default value of "false" will be assumed.	Whether the overrun status is changeable (element set to "true") by a client or not (element set to "false") is denoted in this element.

Table 69: hvacOverrunListData Element rules

3049 Element "**overrunStatus**" value rules:

Value	Rule	Description
active	None.	HVAC overrun triggered, but not started yet.
running	None.	HVAC overrun running.
finished	MAY only be used as notification directly after HVAC overrun was finished. "overrunStatus" SHALL be changed to "inactive" after that. SHOULD NOT be used as part of a reply message!	HVAC overrun finished.
inactive	MAY be used as reply or as notification. The latter only, if HVAC overrun was not finished in a usual way (deactivated by user, e.g.).	HVAC overrun inactive.

3050 *Table 70: Element "overrunStatus" value rules*

3051

3052 **4.3.12.11 hvacOverrunDescriptionListData**

3053 *4.3.12.11.1 Description*

3054 Each HVAC overrun is defined by its description part. Here, the HVAC overrun type is specified as well
 3055 as the list of HVAC system functions that are affected by the HVAC overrun (if this information is
 3056 available).

3057

3058 *4.3.12.11.2 Rules*

3059 None.

3060

3061 *4.3.12.11.3 Element rules*

Element	Rule	Description
overrunId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for the HVAC overrun.
overrunType	SHALL be set. Value rules of Table 72 SHALL be applied.	The type of the overrun.
affectedSystemFunctionId (list)	If system functions are affected by this overrun, their according systemFunctionId SHOULD be stated here.	Identifiers of all HVAC system functions that are affected by this HVAC overrun.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the HVAC overrun.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the HVAC overrun.

3062 *Table 71: hvacOverrunDescriptionListData Element rules*

3063 Element "**overrunType**" value rules:

Value	Rule	Description
-------	------	-------------

oneTimeDhw	None.	A one-time domestic hot water loading (independend from current HVAC operation mode or time table).
party	None.	During a party, the HVAC system will stay in the normal day-operation mode and not turn off at the usual time. Example: HVAC system function "heating" is controlled by the day setpoint, independend from operation mode and time table, until HVAC overrun "party" is finished.
sgReadyCondition1	None.	The SGReady condition 1 will be applied. Please consider [SGReady].
sgReadyCondition3	None.	The SGReady condition 3 will be applied. Please consider [SGReady].
sgReadyCondition4	None.	The SGReady condition 4 will be applied. Please consider [SGReady].
oneDayAway	None.	Overrides the normal daily routine to a out-of-home configuration (e.g. eco-mode the whole day).
oneDayAtHome	None.	Overrides the normal daily routine to an at-home configuration (e.g. on-mode the whole day).
oneTimeVentilation	None.	Activates the ventilation once, independent from the configured routine.
hvacSystemOff	None.	Deactivates the HVAC system (frost protection is still active and the system is in a network-standby mode, in which it is reachable for re-configuration).

Table 72: Element "overrunType" value rules

4.3.12.12 Descriptive information and further definitions

Relation between "overrunStatus" element in function "hvacOverrunListData" and "isOverrunActive" element in function "hvacSystemFunctionListData":

overrunStatus	isOverrunActive
active	false
running	true
finished	false
inactive	false

Table 73: Relation between "overrunStatus" and "isOverrunActive"

The content of the according elements SHALL match the relation stated above.

4.3.13 Identification

4.3.13.1 Dependencies to other Feature Types

None.

4.3.13.2 Introduction

The Identification Class can be used to express certain identification values of the parent Entity.

4.3.13.3 Rules

None.

3081

3082 **4.3.13.4 identificationListData**3083 **4.3.13.4.1 Description**

3084 The identificationListData function allows to express certain identification values of the
 3085 corresponding Entity (the parent Entity of the Identification Feature).

3086

3087 **4.3.13.4.2 Rules**

3088 None.

3089

3090 **4.3.13.4.3 Element rules**

Element	Rule	Description
identificationId	SHALL be set as PRIMARY IDENTIFIER.	The identificationId is used to identify the different list entries within IdentificationListData.
identificationType	If set, the value rules of the IdentificationTypeEnumType SHALL apply.	The identificationType allows to describe a certain type of identification, e.g. a MAC address or an RFID tag.
identificationValue	The string-length SHOULD NOT be longer than 256 characters. If it is longer, the sender SHALL consider the possibility that the receiver will shorten the string to 256 characters.	This holds the value of the identification and therefore is the essential element of IdentificationData.

3091 Table 74: identificationListData Element rules

3092 Element "**identificationType**" value rules:

Value	Rule	Description
eui48	The identificationValue SHALL be interpreted as eui48 MAC address. The following pattern SHALL be used: (([A-F] [0-9]) {2} \-) {5} ([A-Z] [0-9]) {2}	eui48 MAC address
eui64	The identificationValue SHALL be interpreted as eui64 MAC address. The following pattern SHALL be used: (([A-F] [0-9]) {2} \-) {7} ([A-Z] [0-9]) {2}	eui64 MAC address

3093 Table 75: Element "identificationType" value rules

3094

3095 **4.3.14 IncentiveTable**3096 **4.3.14.1 Dependencies to other Feature Types**

3097 None.

3098

4.3.14.2 Introduction

The IncentiveTable data model allows to define tiers with different incentives and boundaries. In the case the incentive is price based, it also could be called a priceTable. Within the boundaries of a tier the incentives of the tier apply. This allows to distribute e.g. different types of power to different power consumers. E.g. a photovoltaic system produces more power than can be consumed and feed into the electricity grid (surplus power), in this case the power is for free and can be distributed by the CEM to consumers that have the flexibility to increase their consumption. When all surplus power is consumed the next tier would be the electricity grid feed-in power, which in this example would be cheaper than the electricity grid power. In the given example we would have three tiers, 1. surplus power on top of that the 2. electricity grid feed in power and beyond that the 3. electricity grid power. However, the electricity grid power can also get cheaper, e.g. in case the electricity grid has too much power available an incentive to flexible consumers could be given.

Although in this example power was used, the IncentiveTable may also be used for other things.

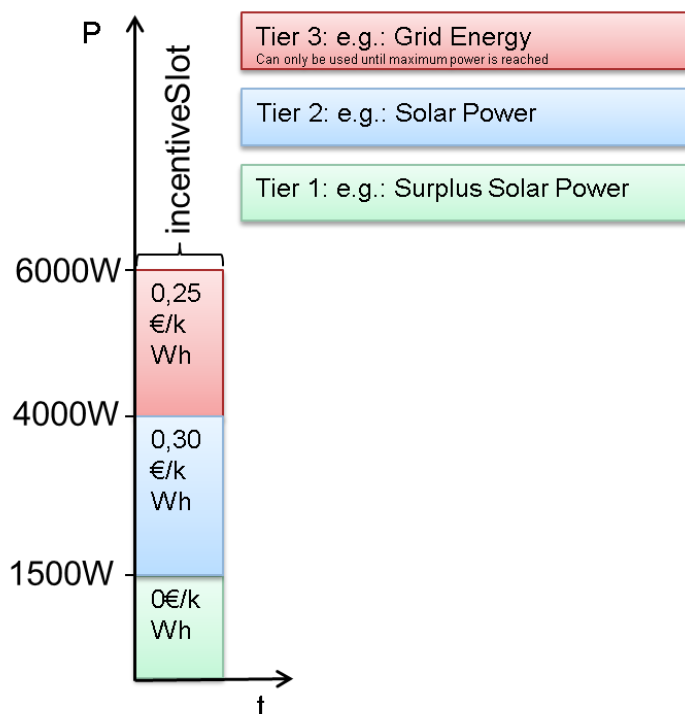


Figure 39: IncentiveTable example (single incentiveSlot) for different tiers

3114 Example 1: If the actual power value is 5000W:

- 3115 - 1500W are for free
- 3116 - 2500W cost 0,30€/kWh
- 3117 - and the last 1000W cost 0,25€/kWh.

3118 Example 2: If the actual power value is 1000W:

- 3119 - Only the lowest tier is used
- 3120 - the 1000W are for free.

3121 Example 3: If the actual power value is 2000W:

- 3122 - 1500W are for free
- 3123 - and the last 500W cost 0,30€/kWh.

3124

3125 The IncentiveTable has three different functions:

- 3126 1. incentiveTableData: holds the actual values of the boundaries and incentives
- 3127 2. incentiveTableDescriptionData: holds data that describes the values of incentiveTable
- 3128 3. incentiveTableConstraintsData: allows a server to define certain constraints regarding the
- 3129 incentiveTableData, this is especially important if the IncentiveTable is writeable

3130

3131 **4.3.14.3 Rules**

3132 Each tier is uniquely defined by the value pair of "tariffId" and "tierId".

3133

3134 **4.3.14.4 incentiveTableData**

3135 *4.3.14.4.1 Description*

3136 The incentiveTable within the incentiveTableData function holds the values of the IncentiveTable.

3137 This includes the boundary values as well as the incentive values for each tier.

3138 The Function "incentiveTableData" models rather time dependent information of each tier within the
3139 respective element "tier". However, the time dependency is not modelled within the "tier" elements
3140 directly. Instead, each "tariff" is split into time slots which are expressed by the element
3141 "incentiveSlot". This means each "incentiveSlot" entry finally contains time dependent data of one or
3142 more "tier" entries.

3143

3144 *4.3.14.4.2 Rules*

3145 If the same tier is used within different "incentiveSlot" entries the same "tierId" value SHALL be used
3146 in these "incentiveSlot" entries.

3147 Each tier defines its supply (difference between upperBoundaryValue and lowerBoundaryValue) that
3148 can be consumed or produced for the incentive(s) (e.g. price) given by the tier. If the boundaries of a

3149 tier are negative the supply can be produced. If the boundaries of a tier are positive the supply can
 3150 be consumed. The supply of each tier can only be consumed or produced, if the supply of the tiers
 3151 between this tier and zero are already consumed.

3152 If (e.g.) the lowerBoundaryValue of tier x is set to 2 units and the upperBoundaryValue is 4 units, tier
 3153 x can provide a supply of $4 - 2 = 2$ units. These 2 units can be consumed for the incentive(s) given by
 3154 tier x. If the actual value of a device surpasses the upperBoundaryValue, the device consumes the
 3155 whole 2 units of tier x for the corresponding incentive(s). If the actual value of a device is below the
 3156 upperBoundaryValue but above the lowerBoundaryValue (e.g. 3 units), the device consumes the
 3157 difference between the actual value and the lowerBoundaryValue of tier x (e.g. $3 - 2 = 1$) for the
 3158 incentive(s) given by the tier. As negative values are used for production, the meaning of the
 3159 boundaries switch. For production the lowerBoundaryValue of a tier will have the higher distance to
 3160 zero then the upperBoundaryValue. Therefore, the upperBoundaryValue for production has the
 3161 same meaning as the lowerBoundaryValue for consumption and the lowerBoundaryValue for
 3162 production has the same meaning as the upperBoundaryValue for consumption.

3163 For more examples, please refer to Figure 39.

3164

3165 4.3.14.4.3 Element rules

Element	Rule	Description
incentiveTable (list)		
incentiveTable. tariff		
incentiveTable. tariff. tariffId	SHALL be set as PRIMARY IDENTIFIER.	Allows the identification of a incentiveTable. Allows linking of the different functions within the IncentiveTable Feature. Allows linking of the incentiveTable within other Features that are placed in the same entity.
incentiveTable. incentiveSlot (list)		
incentiveTable. incentiveSlot. timeInterval	SHALL be set if an incentive has a time restriction or the incentiveTable contains more than one incentiveSlot entry. The timeInterval of different incentiveSlots within an incentiveTable SHALL NOT overlap in time.	Time period of the incentiveSlot.
incentiveTable. incentiveSlot. timeInterval. timeSlotId	MAY be used as SUB IDENTIFIER of tariffId to allow partial modification of the incentiveSlot list if slotIdSupport is set to "true" for this incentiveTable. SHALL NOT be used if the Element "slotIdSupport" within the Function "incentiveTableDescriptionData"	Allows identification of an incentiveSlot within an incentiveTable.

	is not set to "true". If used in a incentiveSlot it SHALL be used in all incentiveSlots of an incentiveTable. Please refer to section 3.4.2.6 for details.	
incentiveTable. incentiveSlot. timeInterval. startTime	SHALL be set.	
incentiveTable. incentiveSlot. timeInterval. startTime. dateTime		Absolute start time.
incentiveTable. incentiveSlot. timeInterval. startTime. relative		Relative start time containing a value relative to now. Positive values are in the future, negative ones are in the past and zero seconds represents "now".
incentiveTable. incentiveSlot. timeInterval. endTime	SHALL be set for the last slot. SHALL be set if a time gap follows after this slot. MAY be set for all other slots.	
incentiveTable. incentiveSlot. timeInterval. endTime. dateTime		Absolute end time.
incentiveTable. incentiveSlot. timeInterval. endTime. relative		Relative end time containing a value relative to now. Positive values are in the future, negative ones are in the past and zero seconds represents "now".
incentiveTable. incentiveSlot. tier (list)		
incentiveTable. incentiveSlot. tier. tier		
incentiveTable. incentiveSlot. tier. tier. tierId	SUB IDENTIFIER of tariffId to identify a tier of a tariff. SHALL be set if the incentiveTable contains more than one tier. If the tierId is omitted, a default value of "1" SHALL be applied.	Allows identification of a tier within a incentiveTable.
incentiveTable. incentiveSlot. tier. boundary (list)	The boundary list defines the boundaries of a tier. The boundary range of different tiers within an incentiveSlot defined by lowerBoundaryValue and upperBoundaryValue SHALL NOT overlap for boundaries with the same boundaryId. The boundary range of a tier includes all values that are equal or greater than lowerBoundaryValue and lower than upperBoundaryValue.	

incentiveTable. incentiveSlot. tier. boundary. boundaryId	SHALL be used as SUB IDENTIFIER of tierId. SHALL be set if the tier contains more than one boundary entry. If the boundaryId is omitted, a default value of "1" SHALL be applied.	Allows identification of a boundary within a tier.
incentiveTable. incentiveSlot. tier. boundary. lowerBoundaryValue	<p>Within a boundary lowerBoundaryValue SHALL be smaller than upperBoundaryValue.</p> <p>Lower and upper boundaries of adjacent tiers: If lowerBoundaryValue is omitted and upperBoundaryValue is set, the lowerBoundaryValue SHALL be interpreted as if it is equal (without a gap) to the upperBoundaryValue of the next lower tier within the incentiveSlot. Gaps between boundaries of adjacent "tier" instances can only be modelled by setting the upperBoundaryValue of the lower "tier" AND the lowerBoundaryValue of the higher "tier" instance explicitly.</p> <p>Minus infinity lowerBoundary rule: If lowerBoundaryValue is omitted and there exists no lower tier with a lowerBoundaryValue or upperBoundaryValue within the incentiveSlot, the omitted lowerBoundaryValue SHALL be interpreted as minus infinity.</p>	This represents the lower end value range boundary.
incentiveTable. incentiveSlot. tier. boundary. upperBoundaryValue	<p>Within a boundary lowerBoundaryValue SHALL be smaller than upperBoundaryValue.</p> <p>Lower and upper boundaries of adjacent tiers: If upperBoundaryValue is omitted and lowerBoundaryValue is set, the upperBoundaryValue SHALL be interpreted as equal (without any gap) to the lowerBoundaryValue of the next higher tier within the incentiveSlot. Gaps between boundaries of adjacent "tier"</p>	This represents the upper end value range boundary.

	instances can only be modelled by setting the upperBoundaryValue of the lower "tier" AND the lowerBoundaryValue of the higher "tier" instance explicitly. Infinity upperboundary rule: If upperBoundaryValue is omitted and there exists no higher tier with a lowerBoundaryValue or upperBoundaryValue within the incentiveSlot, the omitted upperBoundaryValue SHALL be interpreted as infinity.	
incentiveTable. incentiveSlot. tier. incentive (list)		
incentiveTable. incentiveSlot. tier. incentive. incentiveld	SHALL be used as SUB IDENTIFIER of tierId. SHALL be set if the tier contains more than one incentive entry. If the incentiveld is omitted, a default value of "1" SHALL be applied.	Allows identification of an incentive within a tier.
incentiveTable. incentiveSlot. tier. incentive. value		The value of the incentive.

Table 76: incentiveTableData Element rules

4.3.14.5 incentiveTableDescriptionData

4.3.14.5.1 Description

The incentiveTableDescription within the incentiveTableDescriptionData function holds the rather static description of the IncentiveTable. This includes the tier, boundary and incentive type as well as further elements.

4.3.14.5.2 Rules

None.

4.3.14.5.3 Element rules

Element	Rule	Description
incentiveTableDescription (list)		
incentiveTableDescription. tariffDescription		
incentiveTableDescription. tariffDescription. tariffId	SHALL be set as PRIMARY IDENTIFIER of an incentiveTable, if more than one incentiveTable is defined within IncentiveTable.	Allows the identification of a incentiveTable. Allows linking of the data in the different functions of IncentiveTable. Allows linking of the incentiveTables within

		other features that are placed in the same entity.
incentiveTableDescription. tariffDescription. measurementId	SHALL be set as FOREIGN IDENTIFIER, if a measurand or other feature is linked with the measurementId	If a measurand is linked to the incentive table, its identifier is denoted here.
incentiveTableDescription. tariffDescription. tariffWriteable	SHALL be set to "true" if the incentive table is writeable by a client. Otherwise it may be omitted.	Whether the incentive table is writeable by a client or not can be denoted in this element.
incentiveTableDescription. tariffDescription. updateRequired	With updateRequired the server can request an update of writeable or changeable data related to the same PRIMARY IDENTIFIER from a client. The server SHALL ensure that only one responsible client is able to update the related data. To request an update the server SHALL set updateRequired to "true". Note: In this case, the server expects the responsible client to update the writeable or changeable data related to the same PRIMARY IDENTIFIER. However, also if updateRequired is set to "false" a server SHOULD in general allow updates of the data from the responsible client. The server SHALL set the updateRequired back to "false", as soon as "incentiveTableDescriptionListData" was updated successfully (if writeable or changeable) OR the update of the other writeable or changeable data related to the same PRIMARY IDENTIFIER was successful. Note: The client does not need to stop an ongoing update process (e.g. if multiple functions are written), when updateRequired is set back to "false". The server MAY choose to withdraw the update request at any time by setting updateRequired back to "false".	Allows to request an update from the bound client.
incentiveTableDescription. tariffDescription. scopeType		A scopeType for the incentive table can be stated here.
incentiveTableDescription. tariffDescription. label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label on the incentive table.
incentiveTableDescription. tariffDescription. description	Default rules of section 3.10.1.3.1 SHALL be applied.	A description for the incentive table.

incentiveTableDescription. tariffDescription. slotIdSupport	If omitted, a default of “false” SHALL be applied. If set to “false” no incentiveSlot SHALL contain a timeSlotId. If set to “true” timeSlotId MAY be used within the incentiveSlot as described for the element timeSlotId within the function incentiveTableData.	Indicates support of timeSlotId within the corresponding tariff in the Function incentiveTableData.
incentiveTableDescription. tier (list)		Allows to describe tiers and the incentives and boundaries that are used by the tier. Please refer to the detailed description of <i>IncentiveTableDescriptionTierType elements</i> . See Table 152.
incentiveTableDescription. tier. tierDescription		
incentiveTableDescription. tier. tierDescription. tierId	SHALL be set as SUB IDENTIFIER of tariffId, if the incentiveTable contains more than one tier. If the tierId is omitted, a default value of "1" SHALL be applied.	Allows identification of a tier within a incentiveTable.
incentiveTableDescription. tier. tierDescription. tierType	TierTypeEnumType value rules SHALL apply, described below this table.	Allows to define the type of the tier.
incentiveTableDescription. tier. tierDescription. label	Default rules of section 3.10.1.2.1 SHALL be applied.	Allows to define user friendly name for the tier.
incentiveTableDescription. tier. tierDescription. description	Default rules of section 3.10.1.3.1 SHALL be applied.	Allows to define a user friendly description for the tier.
incentiveTableDescription. tier. boundaryDescription (list)		
incentiveTableDescription. tier. boundaryDescription. boundaryId	SHALL be set as SUB IDENTIFIER of tierId, if the tier contains more than one boundary entry. If the boundaryId is omitted, a default value of "1" SHALL be applied.	Allows identification of a boundary within a incentiveTable.
incentiveTableDescription. tier. boundaryDescription. boundaryType	TierBoundaryTypeEnumType value rules SHALL apply, described below this table. There SHALL NOT be multiple boundaries with the same boundaryType for the same tier.	Allows to define the type of the boundary.
incentiveTableDescription. tier. boundaryDescription. switchToTierIdWhenLower	FOREIGN IDENTIFIER used to refer to the tier that is going to be active as soon as the lowerBoundaryValue is underrun.	Link to the tier that will become active if the lowerBoundaryValue is underrun.
incentiveTableDescription. tier. boundaryDescription. switchToTierIdWhenHigher	FOREIGN IDENTIFIER used to refer to the tier that is going to be active as soon as the upperBoundaryValue is exceeded.	Link to the tier that will become active if the upperBoundaryValue is exceeded.

incentiveTableDescription. tier. boundaryDescription. boundaryUnit	If set, the unit SHALL be applied to the lowerBoundaryValue and upperBoundaryValue.	Allows to define the unit of the boundary values.
incentiveTableDescription. tier. incentiveDescription (list)		
incentiveTableDescription. tier. incentiveDescription. incentiveId	SHALL be set as SUB IDENTIFIER of tierId, if the tier contains more than one incentive entry. If the incentiveId is omitted, a default value of "1" SHALL be applied.	Allows identification of an incentive within a incentiveTable.
incentiveTableDescription. tier. incentiveDescription. incentiveType	IncentiveTypeEnumType value rules SHALL apply, described below this table.	Allows to define the type of the incentive.
incentiveTableDescription. tier. incentiveDescription. incentivePriority	Values SHALL be greater than zero. A value of "1" SHALL be interpreted as highest priority. The priority decreases with increasing value of incentivePriority.	Allows to define the priority of the incentive. E.g. the priority that the customer has assigned for this incentive.
incentiveTableDescription. tier. incentiveDescription. currency	If set, the currency SHALL be applied to the value of the incentive	Monetary incentives may need to describe a currency.
incentiveTableDescription. tier. incentiveDescription. unit	If set, the unit SHALL be applied to the value of the incentive	Allows definition of a unit, if the incentive has a unit.
incentiveTableDescription. tier. incentiveDescription. label	Default rules of section 3.10.1.2.1 SHALL be applied.	Allows to express a user friendly label for the incentive.
incentiveTableDescription. tier. incentiveDescription. description	Default rules of section 3.10.1.3.1 SHALL be applied.	Allows to express a user friendly description for the incentive.

3178 Table 77: incentiveTableDescriptionData Element rules

3179 Element "**incentiveTableDescription. tier. tierDescription. tierType**" value rules:

Value	Rule
fixedCost	The tier has a cost incentive that SHALL NOT vary over time
dynamicCost	The tier has a cost incentive that MAY vary over time

3180 Table 78: Element "tier. tierDescription. tierType" value rules

3181 Element "**incentiveTableDescription. tier. boundaryDescription. boundaryType**" value rules:

Value	Rule
powerBoundary	The boundary SHALL be an electrical power. E.g. if the surplus power is consumed, more expensive power has to be used. Positive powerBoundary values describe which power can be consumed with certain incentives, while negative powerBoundary values describe which power can be produced with certain incentives.
energyBoundary	The boundary SHALL be an electrical energy. E.g. if the stored self produced energy in a battery is consumed, more expensive energy has to be used.

	Positive energyBoundary values describe which energy can be consumed with certain incentives, while negative energyBoundary values describe which energy can be produced with certain incentives.
countBoundary	The boundary SHALL be a count. E.g. after a certain count the price of a good may be reduced.

Table 79: Element "tier. boundaryDescription. boundaryType" value rules

Element "incentiveTableDescription. tier. incentiveDescription. incentiveType" value rules:

Value	Rule	Description
absoluteCost	Each related value SHALL be an absolute cost with a specific currency. Positive values SHALL be interpreted as costs. Negative values SHALL be interpreted as profits. If also a unit is set, the cost SHALL contain the cost per unit. E.g. EUR/Wh.	Absolute costs with a currency and a unit. The total costs can be calculated.
relativeCost	Positive values SHALL be interpreted as costs. Negative values SHALL be interpreted as profits. The values SHALL have no currency and unit SHALL be set to percentage (pct). E.g. the price is relative to the fixed price in the tariff contract. The value SHALL be in the range from 0% to 100%.	Relative costs that cannot be interpreted as some monetary costs.
renewableEnergyPercentage	The unit of the incentive SHALL be set to percentage (pct) in this case. The value SHALL be in the range from 0% to 100%.	Percentage of renewable energy of the total obtained energy. This is an ecological incentive that allows optimizing consumption of renewable energy instead of costs.
co2Emission	The unit of the incentive SHALL be set to "kg/Wh". Only positive values SHALL be used.	CO2 emission related to the obtained energy. This is an ecological incentive that allows optimizing the CO2 emission instead of costs.

Table 80: Element "tier. incentiveDescription. incentiveType" value rules

4.3.14.6 incentiveTableConstraintsData

4.3.14.6.1 Description

The incentiveTableConstraints within the incentiveTableConstraintsData function holds additional constraints of the IncentiveTable, regarding how much tiers, boundaries, incentives and slots are allowed.

3192 4.3.14.6.2 Rules

3193 None.

3194

3195 4.3.14.6.3 Element rules

Element	Rule	Description
incentiveTableConstraints (list)		
incentiveTableConstraints. tariff		
incentiveTableConstraints. tariff. tariffId	SHALL be set as PRIMARY IDENTIFIER of a incentiveTable, if more than one incentiveTable is defined within IncentiveTable.	Allows the identification of a incentiveTable. Allows linking of the different functions within the IncentiveTable Feature. Allows linking of the incentiveTable within other Features that are placed in the same entity.
incentiveTableConstraints. tariffConstraints		
incentiveTableConstraints. tariffConstraints. maxTiersPerTariff	If set the incentiveTable SHALL NOT include more tiers than given in maxTiersPerTariff.	Allows the server to define a maximum tier count for this IncentiveTable. This is especially important if a client wants to write an IncentiveTable.
incentiveTableConstraints. tariffConstraints. maxBoundariesPerTier	If set the tier within the incentiveTable SHALL NOT include more boundaries in a tier than given in maxBoundareisPerTier.	Allows the server to define a maximum boundarie count for tiers. This is especially important if a client wants to write an IncentiveTable.
incentiveTableConstraints. tariffConstraints. maxIncentivesPerTier	If set the tier within the incentiveTable SHALL NOT include more incentives than given in maxIncentivesPerTier.	Allows the server to define a maximum incentive count for tiers. This is especially important if a client wants to write an IncentiveTable.
incentiveTableConstraints. incentiveSlotConstraints		
incentiveTableConstraints. incentiveSlotConstraints. slotCountMax	If set the incentiveTable SHALL NOT include more slots than given in slotCountMax.	Allows the server to define a maximum slot count for this IncentiveTable. This is especially important if a client wants to write an IncentiveTable.

3196 Table 81: incentiveTableConstraintsData Element rules

3197

3198 **4.3.14.7 Descriptive information and further definitions**

3199 None.

3200

3201 **4.3.15 LoadControl**

3202 **4.3.15.1 Dependencies to other Feature Types**

3203 *4.3.15.1.1 measurementId*

3204 LoadControl can use the FOREIGN IDENTIFIER measurementId from the Measurement Feature Type
3205 to define a relation to functions of an ElectricalConnection or Measurement Feature.

3206 If a measurementId is given within a LoadControlLimitDescriptionData entry:

- 3207 - If there is an electricalConnectionPermittedValueSetData entry that relates to the same
3208 measurementId, then the value Element of the according loadControlLimitData entry
3209 SHOULD only contain values that are defined within the permittedValueSet of the according
3210 electricalConnectionPermittedValueSetData entry.
- 3211 - If there is an electricalConnectionParameterDescriptionData entry that relates to the same
3212 measurementId, then the elements of the related
3213 electricalConnectionParameterDescriptionData entry SHOULD be considered for the
3214 according loadControlLimitData entry.

3215 If there is a measurementDescriptionData entry that relates to the same measurementId, then the
3216 unit Element of this measurementDescriptionData entry SHALL be ignored for the according
3217 loadControlLimitData entry.

3218

3219 **4.3.15.2 Introduction**

3220 The LoadControl class has two different approaches to control the load of an appliance:

- 3221 - Event-based load control with "loadControlEvent" (see section 5.3.14.3) and
3222 "loadControlState" (see section 5.3.14.4)
- 3223 - Limit-based load control with "loadControlLimit" (see section 5.3.14.5) and
3224 "loadControlLimitDescription" (see section 5.3.14.7)

3225 The event-based load control is reserved for future use.

3226 The limit-based load control is used where limits (changeable by a client) shall affect the behaviour of
3227 a node automatically. E.g. the power demand of an appliance could be limited to a certain value to
3228 protect the household of an overload.

3229 Additionally, with the information given in loadControlNode (see section 5.3.14.2), a node can define
3230 whether it is (at the moment) remote controllable or not.

3231

3232 **4.3.15.3 Rules**

3233 None.

3234

3235 **4.3.15.4 loadControlNodeData**3236 *4.3.15.4.1 Description*

3237 The loadControlNodeData includes node-wide information related to load control (e.g. whether the
3238 node is remote controllable).

3239

3240 *4.3.15.4.2 Rules*

3241 If the function is absent, a default of "true" SHALL be applied for the element
3242 *isNodeRemoteControllable*.

3243

3244 *4.3.15.4.3 Element rules*

Element	Rule	Description
isNodeRemoteControllable	SHOULD be set. If set to "false", write operations on this feature SHALL be declined. If absent, a default value of "true" SHALL be applied.	Defines whether the node is (at the moment) remote controllable by a client, or not.

3245 *Table 82: loadControlNodeData Element rules*

3246

3247 **4.3.15.5 loadControlEventsListData**

3248 Reserved for future use.

3249

3250 **4.3.15.6 loadControlStateListData**

3251 Reserved for future use.

3252

3253 **4.3.15.7 loadControlLimitListData**3254 *4.3.15.7.1 Description*

3255 The loadControlLimitListData function provides the value of the limit (and an optional time period for
3256 that limit) and further information whether the limit is active or not and if it is changeable by a client
3257 or not.

3258

3259 *4.3.15.7.2 Rules*

3260 None.

3261

3262 *4.3.15.7.3 Element rules*

Element	Rule	Description
---------	------	-------------

limitId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for the limit.
isLimitChangeable	SHOULD be set. If set to "false", the timePeriod, value and isLimitActive element SHALL NOT be writeable by a client. If omitted or set to "true", the timePeriod, value and isLimitActive element SHALL be writeable by a client.	States whether the limit may be changed by a client or not.
isLimitActive	SHOULD be set. If set to "false", the limit and its timePeriod and value element SHALL be ignored. If set to "true" or omitted, the timePeriod and value element SHALL be applied, at least if timePeriod or value are set.	Indicates whether the limit is currently active or not.
timePeriod	MAY be used to define a timePeriod for the value. In this case the value SHALL only be applied within the given timePeriod.	The period where the limit shall be active.
value	If <i>isLimitActive</i> is set to "true", the <i>value</i> SHALL be set. Otherwise the element MAY be omitted. If <i>isLimitActive</i> is set to "false", but <i>value</i> is set, the content of <i>value</i> SHALL be ignored. If in the related <i>loadControlLimitDescriptionData</i> entry (with the same PRIMARY IDENTIFIER <i>limitId</i>) <i>measurementId</i> is set, the rules of section 4.3.15.1 SHALL be applied.	The actual limit.

Table 83: *loadControlLimitListData* Element rules

4.3.15.8 *loadControlLimitConstraintsListData*

4.3.15.8.1 *Description*

Constraints that must be kept when trying to change a limit, are modelled with the *loadControlLimitConstraintsListData* function, including a minimum value, a maximum value and the stepsize of the values between.

4.3.15.8.2 *Rules*

None.

4.3.15.8.3 *Element rules*

Element	Rule	Description
limitId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for the limit.
valueRangeMin	SHOULD be set, if not already defined by other functionality (e.g. <i>electricalConnectionPermittedValueSetListData</i>). If set, the content of element <i>value</i> within <i>loadControlLimitListData</i> SHALL NOT contain a value below <i>valueRangeMin</i> .	This element allows to define a minimum limit for the element value within <i>loadControlLimitListData</i> .
valueRangeMax	SHOULD be set, if not already defined by other functionality (e.g. <i>electricalConnectionPermittedValueSetListData</i>).	This element allows to define a maximum limit for the element value within <i>loadControlLimitListData</i> .

	If set, the content of element <i>value</i> within <i>loadControlLimitListData</i> SHALL NOT contain a value above <i>valueRangeMax</i> .	
valueStepSize	SHOULD be set, if there is a step size limitation.	The minimum step size between two different limit values.

Table 84: loadControlLimitConstraintsListData Element rules

4.3.15.9 loadControlLimitDescriptionListData

4.3.15.9.1 Description

The rather static information about the available limit(s) are modelled within the loadControlLimitDescriptionListData function.

4.3.15.9.2 Rules

None.

4.3.15.9.3 Element rules

Name	Rule	Description
limitId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for the limit.
limitType	SHALL be set. Value rules of Table 86 SHALL be applied.	Whether the limit shall be used as a minimum or maximum is denoted within this element.
limitCategory	SHOULD be set. Value rules of Table 87 SHALL be applied.	There are different levels of how important the limit is (see enumeration description).
limitDirection	SHOULD be set. If omitted, the limitDirection SHALL be interpreted as "consume". Default rules of section 3.10.1.14.1 SHALL be applied.	If a device supports both, consumption and production, the direction which will be limited is stated in this element (e.g. if only the consumption of a battery (i.e. it is recharging) will be limited, the enumeration "consume" will be set; if both, consumption and production will be limited, two limits have to be set).
measurementId	SHALL be set as FOREIGN IDENTIFIER, if a measurand or other feature is linked with the measurementId. If set, the rules of section 4.3.15.1 SHALL be applied.	This is a foreign identifier.
unit	SHOULD be set, if there could be ambiguity how to interpret the value without the unit.	The unit, which is used for the limit value, is denoted with this element.
scopeType	None.	A certain meaning of the limit.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A label for the limit.

description	Default rules of section 3.10.1.3.1 SHALL be applied.	A description for the limit.
-------------	---	------------------------------

3286 *Table 85: loadControlLimitDescriptionListData Element rules*3287 Element "**limitType**" value rules:

Value	Rule	Description
minValueLimit	The limit value SHALL be interpreted as a minimum limit. The corresponding value SHALL stay over or equal to the given value of a minValueLimit, if isLimitActive = true and limitType = obligation.	The limit set in the "value" Element of the according "loadControlLimitData" entry shouldn't be underrun (value >= minValueLimit).
maxValueLimit	The limit value SHALL be interpreted as a maximum limit. The corresponding value SHALL stay under or equal to the given value of a maxValueLimit, if isLimitActive = true and limitType = obligation.	The limit set in the "value" Element of the according "loadControlLimitData" entry shouldn't be overrun (value <= maxValueLimit).

3288 *Table 86: Element "limitType" value rules*3289 Element "**limitCategory**" value rules:

Value	Rule	Description
obligation	If this category is used, the limit SHALL be kept under any circumstances. A loss of comfort MAY be risked for as long as needed.	Used for limits that are absolutely important to be hold. A loss of comfort could happen and will be accepted. The normal operation will be interrupted for keeping the limit (if needed).
recommendation	If this category is used, the limit SHOULD be kept wherever possible. A loss of comfort MAY be taken into account for a short period of time.	Used for limits that are quite important for the energy management. A loss of comfort could happen and will be accepted for a short period of time. The normal operation should not be interrupted.
optimization	If this category is used, the limit MAY be considered as good as possible. A loss of comfort SHOULD NOT be accepted.	Used for limits that just optimize the energy management. No loss of comfort will be accepted. The normal operation shall not be interrupted.

3290 *Table 87: Element "limitCategory" value rules*

3291

3292 **4.3.15.10 Descriptive information and further definitions**

3293 None.

3294

3295 **4.3.16 Measurement**3296 **4.3.16.1 Dependencies to other Feature Types**

3297 - ElectricalConnection

3298 - Threshold

3299

3300 **4.3.16.2 Introduction**

3301 The SPINE Measurement class provides functionality for transmitting measured, calculated or
 3302 empirically obtained values from sensors. Mainly these are sensors that provide some kind of
 3303 information about the environment or physical parameters. Clients can use the information for
 3304 further actions or just visualize them for the information of a user.

3305 Due to the nature of measurement, the measured values are only readable. Some control systems
 3306 permit a user to select a preferred value (e.g. a desired room temperature). Such “desired” values
 3307 obviously require writable fields. This can be achieved with the "Setpoint" class (see section 5.3.21).

3308

3309 **4.3.16.3 Rules**

3310 None.

3311

3312 **4.3.16.4 measurementListData**3313 **4.3.16.4.1 Description**

3314 The *measurementListData* command is used to communicate measurement values. Typically, a
 3315 sensor sends this command as notification as soon as it changes more than a specific threshold. Of
 3316 course, it may be readable, too.

3317

3318 **4.3.16.4.2 Rules**

3319 None.

3320

3321 **4.3.16.4.3 Element rules**

Element	Rule	Description
measurementId	SHALL be set as PRIMARY IDENTIFIER.	Enables the identification of different sensors on one SPINE feature.
valueType	SHOULD be set as SUB IDENTIFIER. Value rules of Table 89 SHALL be applied.	It is possible to model different types of measurement values. For example “value”, “averageValue”, “minValue”, etc. (see the SPINE data model for all possible value types). There can be multiple measurementData entries with different valueTypes for the same measurementId. This allows to express different facets of the measurand.
timestamp	SHOULD be set as SUB IDENTIFIER. If set, the timestamp of the creation of the message SHOULD be used.	The time of creation of a measurement value element (i.e. the time it was measured, calculated, ...). Some sensors and communications technologies may already provide this information natively. In other cases the technology

		interface may set timestamp to the time of creation of a <i>measurementData</i> instance. There can be multiple <i>measurementData</i> entries for the same <i>measurementId</i> , but with different timestamps. This allows to express historic values or a measurement timeline.
value	SHALL be set, if a value is available. Otherwise the whole list entry SHALL be omitted or the element <i>valueState</i> SHALL be set to "error". If <i>valueState</i> is set to "error", but <i>value</i> is set, the content of <i>value</i> SHALL be ignored by the client. If <i>valueState</i> is set to "outOfRange" and <i>measurementConstraintsListData.valueRangeMax</i> is set and <i>value</i> is equal to or greater than <i>valueRangeMax</i> , <i>value</i> SHALL be interpreted as greater than <i>valueRangeMax</i> . If <i>valueState</i> is set to "outOfRange" and <i>measurementConstraintsListData.valueRangeMin</i> is set and <i>value</i> is equal to or smaller than <i>valueRangeMin</i> , <i>value</i> SHALL be interpreted as lower than <i>valueRangeMin</i> . If "value" is set, <i>measurementDescriptionListData.measurementType</i> SHALL be set, too.	The measurement value itself, i.e. the magnitude according to "valueType".
evaluationPeriod	None.	Some <i>valueType</i> values can be detailed or even require information about the time period. For example, an <i>averageValue</i> is usually computed from measured values of a finite time period. For smart devices, the start and end time of the <i>evaluationPeriod</i> can be filled in directly. For bare ("simple") sensors, such "derived"/"accompanying" values (average, min, max, ...) could be calculated (if desired) by the technology interface component together with the corresponding <i>evaluationPeriod</i> .
valueSource	Value rules of Table 90 SHALL be applied.	Whether element "value" is measured, calculated or empirically obtained can be denoted with this element.
valueTendency	Value rules of Table 91 SHALL be applied.	This elements states whether the tendency is <i>rising</i> , <i>stable</i> or <i>falling</i> .
valueState	Value rules of Table 92 SHALL be applied.	A measurand may have a state.

3322 Table 88: *measurementListData* Element rules

3323 Element **"valueType"** value rules:

Value	Rule	Description
value	None.	The normal measurement value.
averageValue	None.	The average measurement value over the last time, stated in the element evaluationPeriod.
minValue	None.	The minimum measurement value of the last time, stated in the element evaluationPeriod.
maxValue	None.	The maximum measurement value of the last time, stated in the element evaluationPeriod.
standardDeviation	None.	Used to quantify the amount of variation or dispersion of a set of values.

3324 Table 89: Element **"valueType"** value rules

3325 Element **"valueSource"** value rules:

Value	Rule	Description
measuredValue	None.	The measurement value was measured with a dedicated sensor. The value is rather exact.
calculatedValue	None.	The measurement value was calculated with the values from some other parameters (e.g. water temperature, outside temperature, etc.) that have a calculable impact on the measurand. This value is not as exact as a measured value, but it differs not too much from the real value.
empiricalValue	None.	The measurement value is only an empirical one that can differ a lot from the real value. It can rely on some other parameters that have a known value, but there is no exactly calculable relation between them.

3326 Table 90: Element **"valueSource"** value rules

3327 Element **"valueTendency"** value rules:

Value	Rule	Description
rising	None.	The average measurement value is currently rising.
stable	None.	The average measurement value is currently stable.
falling	None.	The average measurement value is currently falling.

3328 Table 91: Element **"valueTendency"** value rules

3329 Element **"valueState"** value rules:

Value	Rule	Description
normal	None.	The measurement value could be measured / calculated / empirically determined as expected.
outOfRange	SHALL be used if the value is out of range	The measurement value is currently out of range of the sensor (as defined by "valueRangeMin" and "valueRangeMax", see section 5.3.15.3.2).
error	SHALL be used if the value has some error.	An error occurred while determining the measurement value.

3330 Table 92: Element **"valueState"** value rules

3331

3332 **4.3.16.5 measurementConstraintsListData**

3333 4.3.16.5.1 Description

3334 Some constraints concerning the possible values of a measurand are modelled within this function.

3335

3336 4.3.16.5.2 Rules

3337 None.

3338

3339 4.3.16.5.3 Element rules

Element	Rule	Description
measurementId	SHALL be set as PRIMARY IDENTIFIER.	Enables the identification of different sensors on one SPINE feature.
valueRangeMin	MAY be used to express the minimal value that can be measured.	The minimum possible measurement value measurable by the feature.
valueRangeMax	MAY be used to express the maximum value that can be measured.	The maximum possible measurement value measurable by the feature.
valueStepSize	MAY be used if values can only be delivered in a certain stepsize.	The minimum step size between two different measurement values.

3340 Table 93: measurementConstraintsListData Element rules

3341

3342 **4.3.16.6 measurementDescriptionListData**

3343 4.3.16.6.1 Description

3344 Descriptive information on a specific measurement type is modelled with a
 3345 *measurementDescriptionListData* function. This information is rather constant in comparison to
 3346 measurement values. Typically, it is requested once at the beginning, when a node (an external
 3347 display, e.g.) is configured to present or use a specific measurement value. As usual, for such
 3348 scenarios it is recommended the the server notifies changes to all relevant communication partners,
 3349 because a server cannot expect clients to periodically request this information.

3350

3351 4.3.16.6.2 Rules

3352 None.

3353

3354 4.3.16.6.3 Element rules

Element	Rule	Description
measurementId	SHALL be set as PRIMARY IDENTIFIER.	Enables the identification of different measurands on one SPINE feature.
measurementType	SHALL be set, if a measurement value is available in <i>measurementListData</i> .	To identify which type of measurand is measured (for example "temperature").

commodityType	SHOULD be set if a commodity is measured.	If a measurand of a commodity is measured, the type of commodity is stated here.
unit	SHALL be set, if there could be ambiguity how to interpret the value without the unit.	The unit, which is used for the value. It is always related to the normal value. Other <i>valueType</i> values (e.g. <i>averageDerivativeValue</i>) can have different units (derived from <i>unit</i>).
calibrationValue	None.	Some measurement sensors need a calibration value (e.g. set during production phase). It is automatically added to the internal measured value. I.e. the value stated in the <i>value</i> element in the <i>measurementListData</i> function is the already corrected value.
scopeType	None.	Specifies a more detailed meaning of the measurand (e.g. <i>outsideTemperature</i>).
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A (short) label can be useful to identify or group specific sensors.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	A (longer) description of a measurement device can be deposited here.

3355 Table 94: *measurementConstraintsListData* Element rules

3356 Element "**measurementType**" value rules:

Value	Rule	Description
acceleration	None.	The acceleration is measured.
angle	None.	An angle is measured.
angularVelocity	None.	The angular velocity is measured.
area	None.	An area is measured.
atmosphericPressure	None.	The atmospheric pressure is measured.
capacity	None.	The capacity is measured.
concentration	None.	A concentration of a substance is measured.
count	None.	A count is measured.
current	None.	The current is measured.
density	None.	The density is measured.
distance	None.	A distance is measured.
electricField	None.	The electric field is measured.
energy	None.	The energy is measured.
force	None.	The force is measured.
frequency	None.	The frequency is measured.
harmonicDistortion	None.	The harmonic distortion is measured.
heat	None.	The heat is measured.
heatFlux	None.	The heat flux is measured.
illuminance	None.	The illuminance is measured.
impulse	None.	An impulse is measured.
level	None.	A level is measured.
magneticField	None.	The magnetic field is measured.
mass	None.	A mass is measured.
massFlow	None.	The mass flow is measured.
particles	None.	The quantity of particles is measured.
percentage	None.	The percentage is measured.

power	None.	The power is measured.
powerFactor	None.	The power factor is measured.
pressure	None.	The pressure is measured.
radonActivity	None.	The radon activity is measured.
relativeHumidity	None.	The relative humidity is measured.
resistance	None.	The resistance is measured.
solarRadiation	None.	The solar radiation is measured.
speed	None.	The speed is measured.
temperature	None.	The temperature is measured.
time	None.	The time is measured.
torque	None.	The torque is measured.
unknown	None.	An unknown measurand is measured.
velocity	None.	The velocity is measured.
voltage	None.	The voltage is measured.
volume	None.	The volume is measured.
volumetricFlow	None.	The volumetric flow is measured.

Table 95: Element "measurementType" value rules

4.3.16.7 measurementThresholdRelationListData

4.3.16.7.1 Description

The relation between a measurand and dedicated thresholds (see section 5.3.26) is modelled with this function.

4.3.16.7.2 Rules

None.

4.3.16.7.3 Element rules

Element	Rule	Description
measurementId	SHALL be set as PRIMARY IDENTIFIER.	Enables the identification of different sensors on one SPINE address.
thresholdId (list)	SHALL be set as FOREIGN IDENTIFIER, if a threshold or other feature is linked with the thresholdId.	Related threshold identifier(s).

Table 96: measurementThresholdRelationListData Element rules

4.3.16.8 Descriptive information and further definitions

None.

3373 **4.3.17 Messaging**3374 **4.3.17.1 Dependencies to other Feature Types**

3375 None.

3376

3377 **4.3.17.2 Introduction**

3378 Messaging is basically intended for simple textual information transfer for displays and users. The
 3379 textual information is not intended to be used for any automatisms. It should be used for textual
 3380 information that are for example event based or otherwise will change over time.

3381 Some devices may keep a list of messages rather than (or in addition to) a single message.

3382

3383 **4.3.17.3 Rules**

3384 None.

3385

3386 **4.3.17.4 messagingListData**3387 **4.3.17.4.1 Description**

3388 The messagingListData function contains the necessary elements to model a simple text message.

3389

3390 **4.3.17.4.2 Rules**

3391 None.

3392

3393 **4.3.17.4.3 Element rules**

Element	Rule	Description
timestamp	SHOULD be set as SUB IDENTIFIER to the current timestamp at the creation of the message.	The time when the message was generated.
messagingNumber	SHALL be set as PRIMARY IDENTIFIER.	An identifier for one specific message. If a message is marked as obsolete (see below) this number can be used to identify the original message.
type	SHALL be set. Value rules of Table 98 SHALL be applied.	The type of the message.
text	If "type" has the value "obsolete", this element SHOULD be omitted. Otherwise it SHALL be set. The string-length SHOULD NOT be longer than 4096 characters. If it is longer, the sender SHALL consider	The message itself.

	the possibility that the receiver will shorten the string to 4096 characters.	
--	---	--

3394 *Table 97: messagingListData Element rules*3395 Element "**type**" value rules:

Value	Rule	Description
logging	Used for messages that should be stored in a log file.	Used for messages that are stored in a log file.
information	Messages that should be presented to the customer on a display; lower priority.	Messages that are presented to the customer on a display; lower priority.
warning	Messages that should be presented to the customer on a display; medium priority.	Messages that are presented to the customer on a display; medium priority.
alarm	Messages that should be presented to the customer on a display; high priority.	Messages that are presented to the customer on a display; high priority.
emergency	Messages that should be presented to the customer on a display; very high priority.	Messages that are presented to the customer on a display; very high priority.
obsolete	Previously sent messages that shall be marked as obsolete and eventually not be displayed anymore.	Used to mark previously sent messages as obsolete (so that they are not displayed anymore).

3396 *Table 98: Element "type" value rules*

3397

3398 **4.3.17.5 Descriptive information and further definitions**

3399 None.

3400

3401 **4.3.18 NetworkManagement**

3402 Reserved for future use.

3403

3404 **4.3.19 NodeManagement**

3405 All rules and procedures are described in the [ProtocolSpecification].

3406

3407 **4.3.20 OperatingConstraints**3408 **4.3.20.1 Dependencies to other Feature Types**

3409 None.

3410

3411 **4.3.20.2 Introduction**

3412 This class is designed to model some constraints of an appliance (or functionality) with regards to its
 3413 energy consumption or production. I.e. this class is NOT intended to control a device. Rather, it
 3414 enables an appliance to express what needs to be considered especially for the case it permits being

3415 controlled by another device. An example is a central energy management system that uses the class
 3416 DirectControl to control the appliance; in this case the energy management system needs to be
 3417 aware of the appliance's operating constraints.

3418

3419 **4.3.20.3 Rules**

3420 None.

3421

3422 **4.3.20.4 operatingConstraintsInterruptListData**

3423 *4.3.20.4.1 Description*

3424 This function models basic constraints for an interrupt of a device's current task.

3425

3426 *4.3.20.4.2 Rules*

3427 If used, at least one constraint SHALL be set.

3428

3429 *4.3.20.4.3 Element rules*

Element	Rule	Description
sequenceId	SHALL be set as FOREIGN IDENTIFIER, if the interrupt constraints are related to another feature defining sequenceId (see section 4.3.20.10 for details). Otherwise it SHALL be omitted.	If interrupt constraints are related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
isPausable	If set to "true", a client MAY pause the related sequence, functionality or task of a server. Otherwise the element SHALL be set to "false" or omitted.	States whether a functionality or task is pausable (true) or not (false).
isStoppable	If set to "true", a client MAY stop the related sequence, functionality or task of a server. Otherwise the element SHALL be set to "false" or omitted.	States whether a functionality or task is stoppable (true) or not (false).
notInterruptibleAtHighPower	If set to "true", a client SHALL NOT	States whether a functionality or task is NOT interruptible while it is consuming

	pause or stop the related sequence, functionality or task of a server, if the power consumption or production is high. Otherwise the element SHALL be set to "false" or omitted.	(or producing) energy (true) or whether it can be interrupted nevertheless (false).
maxCyclesPerDay	If set, the related sequence, functionality or task SHALL NOT restart more often within a day than stated in this element.	If a functionality or task may be started only a maximum number of times per day, this value is denoted here.

Table 99: operatingConstraintsInterruptListData Element rules

4.3.20.5 operatingConstraintsDurationListData

4.3.20.5.1 Description

This function models constraints on the minimum/maximum duration of active ("on") and pause ("off") phases. Example: A compressor can be damaged if it is started and then switched off too early.

4.3.20.5.2 Rules

If used, at least one constraint SHALL be set.

4.3.20.5.3 Element rules

Element	Rule	Description
sequenceId	SHALL be set as FOREIGN IDENTIFIER, if the interrupt constraints are related to another feature defining sequenceId (see section 4.3.20.10 for details). Otherwise it SHALL be omitted.	If duration constraints are related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
activeDurationMin	If set, the related sequence, functionality or task SHALL be active without interruption for at least the time stated in this element.	This is the minimum duration a device has to run without interruption.
activeDurationMax	If set, the related sequence, functionality or task SHALL at maximum be active without interruption for as long as the time stated in this element.	This is the maximum duration a device can run without interruption.

pauseDurationMin	If set, a pause of the related sequence, functionality or task SHALL be at least as long as the time stated in this element.	This is the minimum duration a device has to pause after the end of an activity.
pauseDurationMax	If set, a pause of the related sequence, functionality or task SHALL be at maximum as long as the time stated in this element.	This is the maximum duration a device can pause after the end of an activity.
activeDurationSumMin	If set, the related sequence, functionality or task SHALL be in sum active for at least the time stated in this element.	This is the minimum duration a device has to run in total for a given task (summation of all active times).
activeDurationSumMax	If set, the related sequence, functionality or task SHALL be in sum active for maximum the time stated in this element.	This is the maximum duration a device can run in total for a given task (summation of all active times).

Table 100: *operatingConstraintsDurationListData* Element rules**4.3.20.6 operatingConstraintsPowerDescriptionListData****4.3.20.6.1 Description**

This function models basic information relevant for *operatingConstraintsPowerRangeListData* and *operatingConstraintsPowerLevelListData*.

4.3.20.6.2 Rules

None.

4.3.20.6.3 Element rules

Element	Rule	Description
sequenceId	SHALL be set as FOREIGN IDENTIFIER, if the interrupt constraints are related to another feature defining sequenceId (see section 4.3.20.10 for details). Otherwise it SHALL be omitted.	If function elements are related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
positiveEnergyDirection	SHALL be set, if there could be ambiguity how to interpret the value of the related sequence, functionality or task. For further value rules see section 3.10.1.14.1.	The <i>positiveEnergyDirection</i> states whether energy consumption or production is counted as positive value.
powerUnit	SHALL be set, if there could be ambiguity how to	The unit for all power values in <i>operatingConstraintsPowerRangeListDa</i>

	interpret the power value of the related sequence, functionality or task without the unit.	ta and operatingConstraintsPowerLevelListData.
energyUnit	SHALL be set, if there could be ambiguity how to interpret the energy value of the related sequence, functionality or task without the unit.	The unit for all energy values in operatingConstraintsPowerRangeListData and operatingConstraintsPowerLevelListData.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information.

Table 101: operatingConstraintsPowerDescriptionListData Element rules

4.3.20.7 operatingConstraintsPowerRangeListData

4.3.20.7.1 Description

This function can be used to model a power/energy range(!) a device can operate. An alternative model for power/energy constraints is operatingConstraintsPowerLevelListData.

4.3.20.7.2 Rules

If used, at least one range SHALL be set.

4.3.20.7.3 Element rules

Element	Rule	Description
sequenceId	SHALL be set as FOREIGN IDENTIFIER, if the interrupt constraints are related to another feature defining sequenceId (see section 4.3.20.10 for details). Otherwise it SHALL be omitted.	If the power range is related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
powerMin	If set, the related sequence, functionality or task SHALL at least consume or produce the given value of powerMin.	The minimum power a device can consume/produce.
powerMax	If set, the related sequence, functionality or task SHALL at maximum consume or produce the given value of powerMax.	The maximum power a device can consume/produce.
energyMin	If set, the related sequence, functionality or task SHALL at least consume or produce the given value of energyMin.	The minimum energy a device can consume/produce.
energyMax	If set, the related sequence, functionality or task SHALL at maximum consume or produce the given value of energyMax.	The maximum energy a device can consume/produce.

Table 102: operatingConstraintsPowerRangeListData Element rules

4.3.20.8 operatingConstraintsPowerLevelListData**4.3.20.8.1 Description**

This function can be used to model a list of possible power values a device can operate. An alternative model for power constraints is operatingConstraintsPowerRangeListData, where a full range (not only discrete levels) can be modelled.

4.3.20.8.2 Rules

If used, at least one power value SHALL be set.

4.3.20.8.3 Element rules

Element	Rule	Description
sequenceId	SHALL be set as FOREIGN IDENTIFIER, if the interrupt constraints are related to another feature defining sequenceId (see section 4.3.20.10 for details). Otherwise it SHALL be omitted.	If the power levels are related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
power (list)	The values stated within the list SHALL define the allowed value set that MAY be used within the value of a related sequence, functionality or task.	Each “power” item contains a possible power level.

Table 103: operatingConstraintsPowerLevelListData Element rules

4.3.20.9 operatingConstraintsResumeImplicationListData**4.3.20.9.1 Description**

Pausing a device might be permitted according to above mentioned (duration) constraints. Nevertheless, it can be expected that it has implications if a device's task is paused temporarily and then resumed to continue the task. One possible implication is an additional mechanical stress which reduces the device's lifetime and could be interpreted as additional cost. Such additional costs or additional energy consumption can be modelled with this function in order to be considered for energy management. However, these are typically empirical (thus rather constant) data, therefore should not be considered to be precise.

4.3.20.9.2 Rules

None.

3489

3490 **4.3.20.9.3 Element rules**

Element	Rule	Description
sequenceId	SHALL be set as FOREIGN IDENTIFIER, if the interrupt constraints are related to another feature defining sequenceId (see section 4.3.20.10 for details). Otherwise it SHALL be omitted.	If the resume implications are related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
resumeEnergyEstimated	None.	Additional energy a device consumes before resuming to its normal operation (after a pause).
energyUnit	SHOULD be set, if there could be ambiguity how to interpret the value without the unit.	Unit for the energy stated in the element resumeEnergyEstimated.
resumeCostEstimated	None.	Additional costs for the resumption of a device to its normal operation.
currency	SHOULD be set, if "resumeCostEstimate" is set, too.	Currency for the price stated in the element resumeCostEstimated.

3491 *Table 104: operatingConstraintsResumeImplicationListData Element rules*

3492

3493 **4.3.20.10 Descriptive information and further definitions**

3494 If operatingConstraintsPowerRangeListData or operatingConstraintsPowerLevelListData is supported,
 3495 operatingConstraintsPowerDescriptionListData SHALL be supported, too.

3496 Section 5.3.18.1 mentions some recommendations to avoid ambiguities. Therefore, some rules are
 3497 defined:

- 3498 1. In every moment for each supported function "xListData" (with "x" as "prefix" placeholder for
 3499 functions mentioned in section 4.3.20) exactly one of the following conditions SHALL be valid:
 3500 1.1. "no sequenceId": xListData contains exactly one xData AND in xData the element
 3501 "sequenceId" is NOT present.
 3502 1.2. "all sequenceId": xListData contains one or more xData AND in every xData the element
 3503 "sequenceId" is present. No two xData items have the same value in "sequenceId".
 3504 1.3. "currently no sequence": xListData has no xData. This shall be treated as special case of item
 3505 1.2: In this case there is currently no related sequenceId.

3506 Please note that these rules DO NOT permit to have an xListData where at least one xData has
 3507 "sequenceId" set AND at least one xData has no "sequenceId" set.

- 3508 2. Further rules on the use of elements apply:
 3509 2.1. If an xData item has "sequenceId" set the other elements of xData are related to the
 3510 corresponding power sequence except for the following case:

3511 2.2. The use of element “maxCyclesPerDay” of function “operatingConstraintsInterruptData” is
3512 optional. If the element is used it is NOT related to any power sequence even if the element
3513 “sequenceId” is present.

3514 Remark: This means this specific constraint always applies to the feature and its related
3515 functionality in general.

3516 3. Use of parallel instances of this Feature Type in the same entity for the same functionality: It is
3517 valid to have in the same entity one Feature Type instance that follows the “no sequenceId” rule
3518 (item 1.1 above) AND one Feature Type instance that follows the “all sequenceId” / “currently no
3519 sequence” rules (items 1.2 and 1.3 above) for the same functionality. However, in this case the
3520 xData child elements of these two instances SHALL be disjoint (i.e. they SHALL NOT contradict
3521 each other).

3522 4. Remark: Esp. in case of the “no sequenceId” rule it is recommended to use the feature group
3523 concept (see section 2.1.4) in order to clearly relate the operating constraints to the
3524 actual/essential functionality.

3525

3526 **4.3.21 PowerSequences**

3527 Reserved for future use.

3528

3529 **4.3.22 Sensing**

3530 **4.3.22.1 Dependencies to other Feature Types**

3531 None.

3532

3533 **4.3.22.2 Introduction**

3534 The *Sensing* class is used for modelling data obtained by sensors in the meaning of more state-based
3535 sensors like contact sensors, switches and buttons, etc., in contrast to value-based sensors like
3536 temperature or humidity sensors that are treated with the *Measurement* class.

3537

3538 **4.3.22.3 Rules**

3539 None.

3540

3541 **4.3.22.4 sensingListData**

3542 **4.3.22.4.1 Description**

3543 The list function of sensing can be used to model sensor information over time. This may be of
3544 interest for a motion detector (when someone was at the backdoor), a contact sensor (when was the
3545 window open in the last 24 hours), etc.

3546 Some sensors may only store the latest value and therefore only support one entry in the list. To
3547 enable users and applications to query past changes, a communications technology interface may

3548 cache states/values of simple sensors that only support the single data function. That is no
 3549 mandatory feature, however.

3550

3551 4.3.22.4.2 Rules

3552 If used, at least one constraint SHALL be set.

3553

3554 4.3.22.4.3 Element rules

Element	Rule	Description
timestamp	SHOULD be set as PRIMARY IDENTIFIER to the timestamp at the creation of the message.	The creation time of the <i>sensingData</i> instance.
state	SHALL be set. Value rules of Table 106 SHALL be applied.	The state of the sensor is denoted in this element.
value	Whether this element SHALL be set or SHOULD NOT be set depends on the state (see Table 106 for details). If no information is given, as a default, "value" SHOULD NOT be set.	Some sensors deliver a value (e.g. a <i>levelSwitch</i>).

3555 Table 105: *sensingListData* Element rules

3556 Element "state" value rules:

Value	Rule	Description
on	"value" SHOULD NOT be set.	"on" was detected, meaning that the according functionality was switched on.
off	"value" SHOULD NOT be set.	"off" was detected, meaning that the according functionality was switched off.
toggle	"value" SHOULD NOT be set.	"toggle" was detected, meaning that the according functionality was toggled (e.g. from "on" to "off" or vice versa).
level	"value" SHALL be set.	"level" was detected, meaning that the according functionality changed its level (e.g. from 2.4 to 8.2) that can be found in the Element "value".
levelUp	"value" SHALL be set.	"levelUp" was detected, meaning that the according functionality changed its level upwards.
levelDown	"value" SHALL be set.	"levelDown" was detected, meaning that the according functionality changed its level downwards.
levelStart	"value" SHOULD NOT be set.	"levelStart" was detected, meaning that the according functionality started to change its level.
levelStop	"value" SHOULD NOT be set.	"levelStop" was detected, meaning that the according functionality stopped to change its level.
levelAbsolute	"value" SHALL be set.	"levelAbsolute" was detected, meaning that the according functionality changed its level to an absolute value stated in the Element "value" (e.g. 1190).
levelRelative	"value" SHALL be set.	"levelRelative" was detected, meaning that the according functionality moved its level in comparison to the old one. The relative change can be found in the Element "value" (e.g. a change

		from 22 to 33 would result in a content of the Element "value" of 11).
levelPercentageAbsolute	"value" SHALL be set.	"levelPercentageAbsolute" was detected, meaning that the according functionality changed its level to an absolute percentage value stated in the Element "value" (e.g. 90%).
levelPercentageRelative	"value" SHALL be set.	"levelPercentageRelative" was detected, meaning that the according functionality changed its percentage level relative to the old one. The relative changed can be found in the Element "value" (e.g. a change from 40% to 50% would result in a content of the Element "value" of 10).
pressed	"value" SHOULD NOT be set.	"pressed" was detected, meaning that the according functionality was pressed (e.g. a button, that was pressed).
longPressed	"value" SHOULD NOT be set.	"longPressed" was detected, meaning that the according functionality was pressed longer (e.g. a button, that was pressed longer). The timing difference between "pressed" and "longPressed" is up to the vendor.
released	"value" SHOULD NOT be set.	"released" was detected, meaning that the according functionality was released (e.g. a button, that was released).
changed	"value" SHOULD NOT be set.	"changed" was detected, meaning that the according functionality changed.
started	"value" SHOULD NOT be set.	"started" was detected, meaning that the according functionality started.
stopped	"value" SHOULD NOT be set.	"stopped" was detected, meaning that the according functionality stopped.
paused	"value" SHOULD NOT be set.	"paused" was detected, meaning that the according functionality paused.
middle	"value" SHOULD NOT be set.	"middle" was detected, meaning that the according functionality (e.g. a scroll bar or a throttle) is in position "middle".
up	"value" SHOULD NOT be set.	"up" was detected, meaning that the according functionality moves up.
down	"value" SHOULD NOT be set.	"down" was detected, meaning that the according functionality moves down.
forward	"value" SHOULD NOT be set.	"forward" was detected, meaning that the according functionality moves forward.
backwards	"value" SHOULD NOT be set.	"backwards" was detected, meaning that the according functionality moves backwards.
open	"value" SHOULD NOT be set.	"open" was detected, meaning that the according functionality is open (e.g. a door).
closed	"value" SHOULD NOT be set.	"closed" was detected, meaning that the according functionality is closed (e.g. a door).
opening	"value" SHOULD NOT be set.	"opening" was detected, meaning that the according functionality is opening (e.g. a door, that is changing from "closed" to "open").
closing	"value" SHOULD NOT be set.	"closing" was detected, meaning that the according functionality is closing (e.g. a door, that is changing from "open" to "closed").

high	"value" SHOULD NOT be set.	"high" was detected, meaning that the according functionality is at a high position or value.
low	"value" SHOULD NOT be set.	"low" was detected, meaning that the according functionality is at a low position or value.
day	"value" SHOULD NOT be set.	"day" was detected, meaning that the according functionality is in day-mode.
night	"value" SHOULD NOT be set.	"night" was detected, meaning that the according functionality is in night-mode.
detected	"value" SHOULD NOT be set.	"detected" was detected, meaning that the according functionality was detected (e.g. a motion sensor detected a motion).
notDetected	"value" SHOULD NOT be set.	"notDetected" was detected, meaning that the according functionality was detected (e.g. a motion sensor reports that no motion is detected).
alarmed	"value" SHOULD NOT be set.	"alarmed" was detected, meaning that the according functionality is in alarm mode (e.g. a smoke detector).
notAlarmed	"value" SHOULD NOT be set.	"notAlarmed" was detected, meaning that the according functionality is not in alarm mode (e.g. a smoke detector).

Table 106: Element "state" value rules

4.3.22.5 sensingDescriptionData

4.3.22.5.1 Description

The rather static (non-changing) elements of the *Sensing* class are combined in the *sensingDescriptionData* function. Typically, it is only requested once by a user or an application.

4.3.22.5.2 Rules

None.

4.3.22.5.3 Element rules

Element	Rule	Description
sensingType	SHALL be set. Value rules of Table 108 SHALL be applied.	Which type of sensor is modelled is denoted with this element.
unit	SHALL be set, if there could be ambiguity how to interpret the sensing value without the unit.	The unit in which the values of the sensor are given.
scopeType	None.	Specifies a more detailed meaning of the sensor (e.g. shutter).
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the sensor.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the sensor.

Table 107: sensingDescriptionData Element rules

3569 Element "**sensingType**" value rules:

Value	Rule	Description
switch	None.	A switch that has usually two states "on" and "off".
button	None.	A button that can be "pressed" (and eventually supports "longPressed" and "released")
level	None.	Some level functionality supporting one or more of the level-related sensing states.
levelSwitch	None.	Some level functionality supporting one or more of the level-related sensing states and a switch functionality ("on" / "off").
windowHandle	None.	A window handle supporting different positions.
contactSensor	None.	Sensor that can detect whether something is "open" or "closed".
occupancySensor	None.	Sensor that can detect whether someone is present in a room ("detected") or not ("notDetected").
motionDetector	None.	Sensor that can detect motion ("detected") and eventually supports the "notDetected" state, too.
fireDetector	None.	A sensor that detects fire ("detected") and eventually the absence of fire ("notDetected").
smokeDetector	None.	A sensor that detects smoke ("detected") and eventually the absence of smoke ("notDetected").
heatDetector	None.	A sensor that detects heat ("detected") and eventually the absence of heat ("notDetected").
waterDetector	None.	A sensor that detects water ("detected") and eventually the absence of water ("notDetected").
gasDetector	None.	A sensor that detects gas ("detected") and eventually the absence of gas ("notDetected").
alarmSensor	None.	A sensor providing alarm information ("alarmed" / "notAlarmed").
powerAlarmSensor	None.	A sensor reporting power alarms ("alarmed" / "notAlarmed").
dayNightIndicator	None.	A sensor detecting whether it is "day" or "night".

3570 *Table 108: Element "sensingType" value rules*

3571

3572 **4.3.22.6 Descriptive information and further definitions**

3573 None.

3574

3575 **4.3.23 Setpoint**

3576 **4.3.23.1 Dependencies to other Feature Types**

3577 - Measurement

3578 - TimeTable

3579

3580 **4.3.23.2 Introduction**

3581 Setting setpoint values is an often chosen way to control some functionality of a device. Especially in
 3582 the home automation and HVAC domain, several use cases depend on setpoints. This class handles

3583 the setpoint itself. Relations to Measurement and TimeTable class enables the control of a
 3584 measurand for specific time frames.

3585

3586 **4.3.23.3 Rules**

3587 None.

3588

3589 **4.3.23.4 setpointListData**

3590 *4.3.23.4.1 Description*

3591 The setpoint values and (if needed) the identifier of related measurands or time tables are modelled
 3592 within this function. It is possible to either define an exact value, that should be reached or a range
 3593 which shall not be fallen below or exceeded. Absolute or relative tolerances can be defined, too.

3594

3595 *4.3.23.4.2 Rules*

3596 One of the following SHALL be supported, at least (more of the combinations MAY be set):

- 3597 - value
- 3598 - valueMin and valueMax

3599

3600 *4.3.23.4.3 Element rules*

Element	Rule	Description
setpointId	SHALL be set as PRIMARY IDENTIFIER.	Identifier of the setpoint.
value	None.	The setpoint value itself.
valueMin	If set, the valueMin SHALL be interpreted as acceptable minimum setpoint value.	Instead of a fixed value (see above) a value range can be set. This would be the lower limit.
valueMax	If set, the valueMax SHALL be interpreted as acceptable maximum setpoint value.	Instead of a fixed value (see above) a value range can be set. This would be the upper limit.
valueToleranceAbsolute	If set, the valueToleranceAbsolute SHALL be added to the value to calculate the tolerated maximum and subtracted from the value to calculate the tolerated minimum. The valueToleranceAbsolute SHALL be positive.	If the value may vary some absolute value around the desired setpoint, the maximum variation allowed (in both directions) can be stated here.
valueTolerancePercentage	If set, the valueTolerancePercentage	If the value may vary some percentage around the desired setpoint, the

	SHALL be multiplied with the value and then added to the value to calculate the tolerated maximum and the valueTolerancePercentage SHALL be multiplied with the value and subtracted from the value to calculate the tolerated minimum. The valueTolerancePercentage SHALL be positive or zero.	maximum variation allowed (in both directions) can be stated here.
--	---	--

Table 109: setpointListData Element rules

4.3.23.5 setpointConstraintsListData**4.3.23.5.1 Description**

The constraints regarding the possible setpoints allowed are modelled within this function.

4.3.23.5.2 Rules

If used, at least one constraint SHALL be set.

4.3.23.5.3 Element rules

Element	Rule	Description
setpointId	SHALL be set as PRIMARY IDENTIFIER.	Identifier of the setpoint, these constraints belong to.
setpointRangeMin	The setpoint value SHALL NOT be smaller than setpointRangeMin.	The minimum value, the setpoint can be set to.
setpointRangeMax	The setpoint value SHALL NOT be larger than setpointRangeMax.	The maximum value, the setpoint can be set to.
setpointStepSize	SHOULD be used if values are only supported in a certain stepsize. Values that do not match the step size SHALL be rounded by the server.	The minimum step size between two different values.

Table 110: setpointConstraintsListData Element rules

4.3.23.6 setpointDescriptionListData**4.3.23.6.1 Description**

The rather static (non-changing) elements of the *Setpoint* class are combined in the *setpointDescriptionListData* function. Typically, it is only requested once by a user or an application.

3618 4.3.23.6.2 Rules
3619 None.

3620

3621 4.3.23.6.3 Element rules

Element	Rule	Description
setpointId	SHALL be set as PRIMARY IDENTIFIER.	Identifier of the setpoint.
measurementId	SHALL be set as FOREIGN IDENTIFIER, if the setpoint is linked to a measurement or another Feature that uses the same measurementId as FOREIGN IDENTIFIER. Otherwise it SHALL be omitted.	If the setpoint relates to a measurand, the corresponding ID can be stated here.
timeTableId	SHALL be set as FOREIGN IDENTIFIER to the according timeTableId, if the setpoint is linked to a timeTable or another Feature that uses the same timeTableId as FOREIGN IDENTIFIER. Otherwise it SHALL be omitted.	If the activation of the setpoint is time dependent, the ID of a corresponding time table can denoted in this element.
setpointType	SHOULD be set. If omitted, the setpoint SHALL be interpreted as <i>setpointType</i> "valueAbsolute". Value rules of Table 112 SHALL be applied.	This is the type of setpoint used for this specific setpoint.
unit	SHALL be set, if there could be ambiguity how to interpret the value without the unit.	The unit used for this setpoint (see SPINE data model for all possible values).
scopeType	None.	Specifies a more detailed meaning of the setpoint.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the setpoint.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the setpoint.

3622 Table 111: setpointDescriptionListData Element rules

3623 Element "**setpointType**" value rules:

Value	Rule	Description
valueAbsolute	None.	An absolute value, e.g. 28°C. A unit is needed in most cases.
valueRelative	None.	A relative value in percentage. The unit is "pct" in most cases.

3624 Table 112: Element "setpointType" value rules

3625

3626 4.3.23.7 Descriptive information and further definitions

3627 None.

3628

3629 **4.3.24 SmartEnergyManagementPs**3630 **4.3.24.1 Functions and message purposes**

3631 The following table shows which messages purposes are required or optional (see column “M/O”) for
 3632 a server. It also shows which function is used for the given purpose and which direction and classifier
 3633 applies:

Message purpose	Used complex function	M / O	Direction	Used classifiers	Explanation
powerSequences information	smartEnergyManagementPs Data	M	Out	reply, notify	Informs or updates a client about a forecast (fixed or with flexibilities).
powerSequences information request	smartEnergyManagementPs Data	M	In	read	Requests current information from a server.
powerSequences configuration request call	smartEnergyManagementPs ConfigurationRequestCall	O	Out	call	Informs the client that the server's power sequences need to be (re-)scheduled.
powerSequences configuration	smartEnergyManagementPs Data	M	In	write (restricted)	Is used to submit a new configuration to a server.
powerSequences state change	smartEnergyManagementPs Data	O	In	write (restricted)	Enables a power sequences client to change the current state of a power sequences server's sequence (if permitted by the server).
powerSequences price calculation request call	smartEnergyManagementPs PriceCalculationRequestCall	O	Out	call	Requests a price calculation from a client.
powerSequences price information	smartEnergyManagementPs PriceData	O	Out	reply, notify	Informs or updates a client about a price.
powerSequences price information request	smartEnergyManagementPs PriceData	O	In	read	Requests price information stored on the server.
powerSequences price change	smartEnergyManagementPs PriceData	O	In	write (restricted)	Enables a power sequences client to change the price

					information of a power sequences server's sequence (if permitted by the server).
--	--	--	--	--	--

Table 113: Server messages overview. The "write (restricted)" denotes if the message is intended for partial modification.

Details on the messages are defined in section 4.3.24.4.

4.3.24.2 Introduction

4.3.24.2.1 Scope of this Feature Type

The underlying SmartEnergyManagementPs complex class enables a client to shift the demand of a server within some given constraints. The corresponding rules for interoperable implementations are denoted within this feature type.

4.3.24.2.2 Theory of operation (informative)

Each power sequences server requires the establishment of a binding with a power sequences client (see 4.3.24.3.8) as only a bound power sequences client is permitted to configure a power sequences server. One power sequences client may be bound to many power sequences servers (see 4.3.24.3.8), while one power sequences server is only bound to exactly one power sequences client (see 4.3.24.3.9). This completes the basic setup for a power sequences server and client.

Please note: Binding does not imply subscription mechanisms! If a client is interested in data changes on the server (which is recommended), a subscription has to be established, too.

If the power sequences contain flexibilities and/or alternatives, the power sequences server issues a power sequences configuration request call. Once the power sequences client receives this message, it evaluates whether it will configure the according power sequences server or ignore the call (temporarily or permanent). The power sequences client configures the server by writing a power sequences configuration to the power sequence server. If the power sequences client does not configure the power sequences server, the power sequences server follows the configuration it has initially exposed, i.e. its own preference. If the power sequences server receives a configuration it should take into account, it first checks whether the configuration is valid. If the received configuration is valid, i.e. meets the constraints the power sequences server has laid out, it takes over the configuration.

During runtime, the power sequences client and server may re-negotiate the configuration multiple times, even if the according power sequence is already running (if the power sequences server supports it). However, every time the data on either the power sequence information, configuration or state changes, the power sequences server updates the subscribed nodes.

Please note: Devices that can't freely choose between different modes of operations (e.g. because they cannot interrupt an already running process) would not publish the power sequences which they cannot switch to "currently" until the situation has been resolved.

Dependent on the use case, a power sequence needs to contain different kind of data. This means there are different types of sequences possible. The description of the supported sequence types and how they are recognised (evaluated) is given in detail in section 4.3.24.3.5.5.

In the subsequent sections terms and rules are defined for the use of the complex class within this Feature Type. These rules impose restrictions on the use of the class as follows:

- A sequence contains a minimum of one slot, but may contain as many slots as needed.
- Slots which appear in a certain order within a power sequence cannot be interchanged, i.e. a slot can never be shifted so that it is executed before a formerly subsequent slot. The slot order within a power sequence cannot be changed. Thus, each time shift to a slot moves the subsequent slots accordingly.
- Within a sequence, each pause or gap is treated as regular slot of a power sequence and thus holds the same data a regular slot holds (but with an energy value of "0").
- For fixed forecasts, no constraints are specified, because no shifting is permitted (see 4.3.24.3.5.5.2).
- etc.

4.3.24.3 Common specification details

4.3.24.3.1 Guidance to the concept and description

The concept permits numerous possibilities for the representation of a device's energy management related data. In order to achieve interoperability, a formal definition of terms and rules is given subsequently. First of all, definitions and rules related to "slots" are given, followed by definitions and rules on "sequences".

However, one key to achieve interoperability is to focus on those "variants" of device's energy management information that are supported by this specification. These are called "permitted sequence types" and explained in section 4.3.24.3.5.5. It is recommended to start with this section for a first reading at least. These sequence types definitions make use of the terms that are explained in the previous sections.

Finally, rules on the evaluation of power sequences information and some necessary conventions are given.

4.3.24.3.2 Slots

4.3.24.3.2.1 Overview

Section 4.3.24.3.2 contains a number of basic terms and rules on the modelling of power sequence slots. In order to get a better overview and understanding of this section and the concepts it may be preferable to continue first with the following sections and then return to the other sections on demand:

1. Section 4.3.24.3.2.9 discusses some general duration facets of slots.
2. Section 4.3.24.3.2.7 discusses slots with configurable schedule.
3. Section 4.3.24.3.2.5 describes some requirements on power/energy values.

3707 4. Section 4.3.24.3.2.8 describes an impact of power/energy consumption/generation through
3708 the modification of a slot's duration.

3709

3710 *4.3.24.3.2.2 Determined vs. undetermined slot*

3711 *4.3.24.3.2.2.1 Introduction (informative)*

3712 A "determined slot" can be used to build detailed power/energy curves. This provides a basis for
3713 time-dependent forecasts, process configuration (in advance) and autonomous process execution.

3714 An "undetermined slot" on the other hand can be used for processes where only few information is
3715 available in advance.

3716

3717 *4.3.24.3.2.2.2 Definitions*

3718 A slot is called "determined" if all of the following information is present:

- 3719 1. "alternatives. powerSequence. powerTimeSlot. schedule. slotNumber"
- 3720 2. "alternatives. powerSequence. powerTimeSlot. schedule. defaultDuration"
- 3721 3. The slot is (explicitly or implicitly) activated or deactivated (see 4.3.24.3.2.4).
- 3722 4. The slot value forecast is determined (see 4.3.24.3.2.5).

3723 The value of "alternatives. powerSequence. powerTimeSlot. schedule. defaultDuration" SHALL be
3724 greater than 0.

3725 Remark: The element "alternatives. powerSequence. powerTimeSlot. schedule. durationUncertainty"
3726 may as well be present in a determined slot, like other elements as well. However, this is relevant for
3727 the duration estimation only and not for the classification as determined slot.

3728 A slot is called "undetermined" if it is NOT "determined" AND all of the following conditions are
3729 fulfilled:

- 3730 1. "alternatives. powerSequence. powerTimeSlot. schedule. slotNumber" is present.
- 3731 2. "alternatives. powerSequence. powerTimeSlot. schedule. defaultDuration" is NOT present.
- 3732 3. The slot value forecast is determined (see 4.3.24.3.2.5).

3733 Undetermined slots have no activation or deactivation information themselves. Rather, this kind of
3734 slot is designed for a specific purpose where the final execution is determined by its power
3735 sequence's state.

3736 A slot that is neither determined nor undetermined is malformed and not allowed in this
3737 specification.

3738 The element "alternatives. powerSequence. powerTimeSlot. schedule. timePeriod" SHALL NOT be
3739 used to make a slot schedule "determined". Rather, "timePeriod" can be used for supplementary
3740 purposes (e.g. display) only. If used, it SHALL NOT contradict regular schedule information (i.e.
3741 sequence start time, sum of preceding (and explicitly/implicitly activated) slot durations).

3742

4.3.24.3.2.2.3 *Process rules*

1. On numbering slot numbers in a power sequence:
Each slot of a power sequence SHALL be assigned a slot number within the element "slotNumber". The slot numbers within the sequence SHALL be ascending according to the chronological order of the slots. The combination of sequence ID and slot number and (optionally) the current repetition number (see sections 4.3.24.3.2.10 and 4.3.24.3.5.3.2) SHALL remain unique at any given point in time.
2. Some more process rules on the use of defaultDuration and its dependencies to some other values are described in section 4.3.24.3.2.7.3.

4.3.24.3.2.3 *Duration uncertainty*

This section (4.3.24.3.2.3 and subsections) applies for determined slots only!

4.3.24.3.2.3.1 *Introduction (informative)*

Slots exposed by a power sequences server typically have fixed durations. In some applications, the exact duration of a slot cannot be determined. Thus, a slot duration might be rather an average value, supplemented with an uncertainty of the duration.

Note: A duration uncertainty does NOT model a configuration range of a duration. A duration uncertainty also does NOT express any potential interrupt of the slot. A duration uncertainty only models that a power sequences server cannot estimate a duration more precisely.

4.3.24.3.2.3.2 *Definitions*

The duration of a determined slot SHALL be rather "exact" if NO duration uncertainty is given.

A duration uncertainty is given if the following information is present:

1. The slot schedule is determined.
2. "alternatives. powerSequence. powerTimeSlot. schedule. durationUncertainty" is present.

If "alternatives. powerSequence. powerTimeSlot. schedule. durationUncertainty" is present the value SHALL be positive.

If a duration uncertainty is given, the slot's final duration (i.e. the measured duration when the slot was executed completely) SHOULD be in the range from "lower duration bound" to (defaultDuration + durationUncertainty). The value of "lower duration bound" is determined by (defaultDuration - durationUncertainty) provided that (defaultDuration - durationUncertainty) is not negative. Otherwise, "lower duration bound" is 0.

If a slot provides information about a duration uncertainty and has a configurable duration (see 4.3.24.3.2.7), the duration uncertainty is considered to be constant for all configurable durations of the slot.

4.3.24.3.2.4 Activated vs. deactivated slot

This section (4.3.24.3.2.4) applies for determined slots only.

Usually, a slot is explicitly or implicitly activated, hence contributes with its data to the schedule and execution of its power sequence. However, in case a slot is “optional” it can instead be marked as “deactivated”. A deactivated slot does not contribute to the schedule and execution of a power sequence. I.e. a power sequence is scheduled and executed as if all deactivated slots have been removed from the sequence.

4.3.24.3.2.5 Determined vs. undetermined slot value forecast

A slot can be assigned power or energy values to model a potential or expected forecast of a power sequences server’s power or energy consumption or production.

A slot value forecast is called “determined” if all of the following conditions are fulfilled:

1. At least one instance of “alternatives. powerSequence. powerTimeSlot. valueList. value” is present for the given slot AND the child element “valueType” is present AND the child element “value” is present.
2. The list of the slot’s “alternatives. powerSequence. powerTimeSlot. valueList. value” instances fulfils the value types requirement (see subsequent definition).

Otherwise the slot value forecast is called “undetermined”.

The value types requirement is defined as follows: Within each slot of a power sequence, different pairs of the elements “value” and “valueType” may be provided. The availability of these values and more important the value types depends on the process(es) the power sequences server exposes. However, a power sequences server SHALL at least provide one of the values and value types or tuples of values and value types contained in the following list:

- power
- powerMin
- powerMax
- energy
- energyMin
- energyMax
- powerExpectedValue
- powerExpectedValue + powerStandardDeviation
- powerExpectedValue + powerStandardDeviation + powerSkewness
- energyExpectedValue
- energyExpectedValue + energyStandardDeviation
- energyExpectedValue + energyStandardDeviation + energySkewness

As an example, a power sequences server might expose all slots with a valueType of “power”. A power sequences server might as well expose all slots with the valueTypes “powerExpectedValue”, “powerStandardDeviation” and “powerSkewness”. However, it is invalid to only provide the valueType combination of “powerExpectedValue” and “powerSkewness” without “powerStandardDeviation” for one or more slots.

3820 Please note that “expectedValue” refers to the term “expected value” (also “expectation”) of
3821 probability theory. It may be used to express the estimation of an average value.

3822 Note: See section 4.3.24.3.2.8 on a dependency between power/energy forecast and configurable
3823 slot durations!

3824

3825 *4.3.24.3.2.6 Constrained vs. implicit slot schedule*

3826 This section (4.3.24.3.2.6 and subsections) applies for determined slots only!

3827

3828 *4.3.24.3.2.6.1 Introduction (informative)*

3829 This section is closely related to the question whether a slot schedule can be configured or not (see
3830 4.3.24.3.2.7). In case of configurable slot schedules there is a need to model the constraints of such a
3831 configuration. Non-configurable slots, on the other hand, do not require such constraints, hence
3832 their schedule is given implicitly.

3833

3834 *4.3.24.3.2.6.2 Definitions*

3835 A slot schedule is called “constrained” if all of the following conditions are fulfilled:

- 3836 1. The slot is determined.
3837 2. “alternatives. powerSequence. powerTimeSlot. scheduleConstraints” is present with at least
3838 one child element.

3839 A slot schedule is called “implicit” if all of the following conditions are fulfilled:

- 3840 1. The slot is determined.
3841 2. “alternatives. powerSequence. powerTimeSlot. scheduleConstraints” is absent or has no
3842 child element.

3843 An implicit slot schedule does NOT permit any direct variation of the slot’s schedule itself. I.e. it is not
3844 possible to modify the slot’s duration or start time or end time directly. This means the duration of
3845 the slot is not modifiable by a value assignment, and the slot’s start time is determined implicitly (by
3846 the end of the preceding slot or by the sequence’s start time if there is no preceding slot). It also
3847 means the slot is not optional, i.e. it cannot be deactivated individually (see 4.3.24.3.2.4).

3848 A constrained slot schedule denotes further rules that need to be considered for the configuration of
3849 the slot’s schedule. Among others, this may include the possibility for the complete deactivation of
3850 the slot (see 4.3.24.3.2.4).

3851

3852 *4.3.24.3.2.7 Configurable vs. non-configurable schedule of a determined slot*

3853 This section (4.3.24.3.2.7 and subsections) applies for determined slots only!

3854

3855 *4.3.24.3.2.7.1 Introduction (informative)*

3856 Some devices can permit the configuration of some slot durations or even the disabling of slots of a
3857 given power sequence. Such slot schedules are called “configurable”, whereas other slot schedules
3858 are called “non-configurable”. The following definition explains the proper conditions. Furthermore,
3859 process rules that affect both kinds of slot schedules are discussed.

3860

3861 *4.3.24.3.2.7.2 Definitions*

3862 The schedule information of a determined slot is configurable if all of the following conditions are
3863 fulfilled:

- 3864 1. The slot schedule is constrained (see 4.3.24.3.2.6).
- 3865 2. The slot’s sequence permits configuration.

3866 In this case the slot’s “alternatives. powerSequence. powerTimeSlot. schedule” can be modified with
3867 the proper modification according to section 4.3.24.3.2.7.3. However, this requires the consideration
3868 of the whole sequence’s schedule constraints (including all slot schedule constraints).

3869 The duration of a slot is configurable if all of the following conditions are fulfilled:

- 3870 1. The slot schedule is configurable.
- 3871 2. In “alternatives. powerSequence. powerTimeSlot. scheduleConstraints” the child elements
3872 minDuration AND maxDuration are present.

3873 If a slot’s duration is configurable the following rules apply:

- 3874 1. minDuration and maxDuration SHALL be non-negative.
- 3875 2. minDuration SHALL be smaller than maxDuration.
- 3876 3. defaultDuration SHALL be in the range [minDuration, maxDuration].

3877 Note: In case a duration uncertainty is given (see 4.3.24.3.2.3), the final duration of a slot (i.e. the
3878 measured duration once the slot was completed) may violate a boundary given by minDuration or
3879 maxDuration. This is because minDuration and maxDuration denote only configuration boundaries.
3880 This is independent from the capability of forecasting (i.e. estimating) a final duration from a given
3881 configuration.

3882 The element “alternatives. powerSequence. powerTimeSlot. scheduleConstraints” is malformed if
3883 only one of the child elements minDuration and maxDuration is present.

3884 A slot can be activated or deactivated if all of the following conditions are fulfilled:

- 3885 1. The slot schedule is configurable.
- 3886 2. In “alternatives. powerSequence. powerTimeSlot. scheduleConstraints” the child element
3887 optionalSlot is present and set to “true”.

3888 See also section 4.3.24.3.2.7.3 on process related restrictions to schedule configurability!

3889 The schedule information of a determined slot with implicit schedule (see 4.3.24.3.2.6) is not
3890 configurable.

3891

3892 *4.3.24.3.2.7.3 Process rules*

3893 This section discusses rules that affect determined slots with configurable as well as non-configurable
3894 schedule.

3895 The schedule configuration of a power sequence's slots is modelled within the element "alternatives.
3896 powerSequence" of a "powerSequences configuration" message (see section 4.3.24.4.4), where each
3897 instance of the child element "powerTimeSlot. schedule" contains the schedule configuration of a
3898 single slot.

3899 1. Uniqueness and order:

3900 The slot schedule configuration list SHALL be unique and ordered: Within "alternatives.
3901 powerSequence" each slotNumber SHALL occur not more than once. The slots in the list
3902 SHALL be ordered by the slotNumber according to section 4.3.24.3.2.2.3.

3903 2. Slots with non-configurable schedule:

3904 Slots with non-configurable schedule do not need to be listed in "alternatives.
3905 powerSequence" unless stated otherwise (some of the following rules REQUIRE the presence
3906 for specific cases; however, the presence is not required in general). If listed anyhow, the
3907 slot's schedule configuration MAY be empty (except for slotNumber) and SHOULD NOT
3908 contradict the slot's schedule information. If evaluated, the schedule configuration of slots
3909 with non-configurable schedule SHALL be ignored.

3910 3. Completeness of a slot's schedule configuration:

3911 Within a message to configure one or more slots, each instance of "alternatives.
3912 powerSequence. powerTimeSlot. schedule" of a configurable slot SHALL be complete with
3913 regards to the configurable options. I.e. if a schedule configuration is given, it SHALL
3914 configure ALL schedule information the slot permits for configuration.

3915 Example: If a slot can be deactivated and the duration can be modified it is not sufficient to
3916 configure only the activation or only the duration.

3917 4. Contributions to the schedule configuration list:

3918 Within "alternatives. powerSequence" there can be "alternatives. powerSequence.
3919 powerTimeSlot. schedule" instances for "past slots", the "current slot", and "future slots". A
3920 "past slot" is a slot that was already executed. The "current slot" is the slot that is currently
3921 executed. A "future slot" is a slot that is not executed yet.

3922 5. On "past slots":

3923 A "past slot" SHALL be treated as non-configurable slot, even if it was configurable before its
3924 execution. Its "alternatives. powerSequence. powerTimeSlot. schedule" instance MAY be
3925 part of "alternatives. powerSequence". However, such a slot's instance of "alternatives.
3926 powerSequence. powerTimeSlot. schedule" has no further functional relevance within this
3927 specification.

3928 6. On the "current slot":

3929 a. It SHALL NOT be possible to deactivate a "current slot". This means even if the slot's
3930 child element "optionalSlot" (see 4.3.24.3.2.7.2) is available and set to "true" it
3931 SHALL be interpreted as if the element is set to "false".

3932 Remark: This rule is only about the slot's specific flag "optionalSlot". It is
3933 independent from any possibility to modify a whole power sequence's state.

- b. If a configurable slot becomes the current slot it MAY remain configurable or it MAY become non-configurable (this is finally implementation specific to the power sequences server).
 - c. The current slot's instance of "alternatives. powerSequence. powerTimeSlot. schedule" SHALL be listed in "alternatives. powerSequence" (regardless whether it is configurable or not; see rules above).
 - d. If the power sequences server provides a "alternatives. powerSequence. state" instance of the power sequence with the current slot, the following rules apply:
 - i. The child element "remainingSlotTime" SHALL be set to the duration the current slot still needs to be active. I.e. in case the power sequence itself is paused for a while (hence the current slot can also not be active) the value of remainingSlotTime SHALL NOT decrease.
 - ii. The child element "elapsedSlotTime" MAY be present. If it is present it SHALL be set to the duration the current slot has been active. I.e. in case the power sequence itself is paused for a while (hence the current slot can also not be active) the value of elapsedSlotTime SHALL NOT increase.
 - iii. The sum of remainingSlotTime's value and the duration the current slot has been active SHALL be at least approximately equal to the slot's defaultDuration. At least a deviation according to durationUncertainty is permitted.

Note: This also means the current slot's defaultDuration SHALL always cover the time it began (i.e. the past) up to the time it will end (i.e. the future). I.e. this value SHALL NOT be reduced just because a part of the slot's overall activity was already completed. It also means that phases where the slot (sequence) was paused do not require an update of defaultDuration.
7. On "future slots":
- All "future slots" instances of "alternatives. powerSequence. powerTimeSlot. schedule" SHALL be listed in "alternatives. powerSequence" (regardless whether they are configurable or not; see rules above).

Please note: Some of the rules describe situations where the configurability of a slot changes. Of course, this requires a proper information update from the power sequences server to the power sequences client in general. However, this update process is not subject of this section.

4.3.24.3.2.8 Dependency between power/energy and slot duration configuration

This section (4.3.24.3.2.8) applies for slots with configurable duration according to section 4.3.24.3.2.7 only!

According to section 4.3.24.3.2.5 miscellaneous power OR energy values can be used as forecast of a slot's consumption or production. It shall be noted that this requires special consideration in conjunction with slots of configurable duration. In general a slot with a configurable duration can be modelled with an energy forecast OR a power forecast, resulting in two fundamentally different consequences:

1. A slot with a configurable duration and an exposed "valueType" element stating an energy-related information, is expected to (nearly) have the same energy for all possible

3977 configurations of durations it offers. I.e. the power is reduced when extending the duration
3978 and increased when shortening the duration.

3979 2. A slot with a configurable duration and an exposed “valueType” element stating a power-
3980 related information is expected to (nearly) have the same power for all possible
3981 configurations of durations it offers. I.e. the energy is reduced when shortening the duration
3982 and increased when extending the duration.

3983 Note: A slot that provides a configurable duration SHALL NOT provide “valueType”s related to energy
3984 and power for the same slot!

3985

3986 *4.3.24.3.2.9 Duration facets of determined slots, overview*

3987 This section (4.3.24.3.2.9) applies for determined slots only! It provides a brief overview on several
3988 aspects of a slot duration.

3989 The following aspects can be modelled:

- 3990 1. A slot’s duration is “exact” or just given with an uncertainty: See 4.3.24.3.2.3.
- 3991 2. A slot’s duration is configurable (i.e. modifiable) or not: See 4.3.24.3.2.7.
- 3992 3. A slot is optional (i.e. can be deactivated) or not: See 4.3.24.3.2.7.

3993 Thus, for optional as well as non-optional slots the following combinations are possible:

- 3994 1. A slot with an “exact” duration (i.e. without any duration uncertainty) and no configurable
3995 duration.
- 3996 2. A slot with a (average) duration and a duration uncertainty and no configurable duration.
- 3997 3. A slot with an “exact” duration (i.e. without any duration uncertainty) and a configurable
3998 duration.
- 3999 4. A slot with a (average) duration and a duration uncertainty and a configurable duration.

4000

4001 *4.3.24.3.2.10 Repeated slots vs. one-time slots*

4002 Within a given sequence, each phase is represented by a slot. Each slot can be assigned a unique
4003 number. In this situation, each slot is processed only one time.

4004 However, some implementations favour a “repetition concept” where a sequence of phases repeats
4005 a number of times. This permits to model only the first occurrence of a phase. E.g. if a sequence has
4006 three phases “A”, “B”, “C” and needs to process the phase sequence { A, B, C } two times, i.e. { A, B,
4007 C, A, B, C }, it is possible to model this with three slots only (e.g. slotNumber values 1, 2, 3) and an
4008 additional repetition counter. This leads to the case where the slots are processed in the order { 1, 2,
4009 3, 1, 2, 3 }, i.e. each slot is repeated. However, this concept does not contradict any rule of section
4010 4.3.24.3.2. As this concepts requires the consideration of a whole sequence, more details are
4011 discussed in section 4.3.24.3.5.3.

4012

4013 4.3.24.3.3 Sequences

4014 4.3.24.3.3.1 Chronological classification of single sequence

4015 This section defines some terms used to classify power sequences chronologically. It also imposes
4016 restrictions on the use of the power sequence's "state" information element ("alternatives.
4017 powerSequence. state").

4018 1. "Past sequence":

4019 A "past sequence" is a sequence that was relevant in the past. This means the sequence was
4020 processed or was at least offered. The sequence is neither "now" nor in the future offered or
4021 considered by the power sequences server. The power sequence's "state" information
4022 element SHALL be either "completed" or "invalid".

4023 2. "Pre-announced sequence":

4024 A "pre-announced sequence" is a sequence that can be relevant for a time not earlier than
4025 "now". However, it is not a "current sequence". The power sequence's "state" information
4026 element SHALL be either "scheduled" or "scheduledPaused" or "inactive".

4027 3. "Current sequence":

4028 A "current sequence" is a sequence that is currently processed by a power sequences server.
4029 The power sequence's "state" information element SHALL be either "running" or "paused".

4030 Note: If a "pre-announced sequence" becomes a "current sequence" the natural "state"
4031 transition is as follows: "scheduled" changes to "running" and "scheduledPaused" changes to
4032 "paused".

4033 Note: The above mentioned "relevance" does not only apply to sequences that were or will be
4034 executed. It also applies to sequences just offered as choice (to the bound power sequences client).

4035 Note: The term "past sequence" does not say anything about the time frame the sequence was
4036 relevant for. A "past sequence" may have been a "pre-announced sequence" recently and notified
4037 with other "pre-announced sequences" within an alternatives group (see 4.3.24.3.3.2) by the power
4038 sequences server. Suddenly a sequence became "invalid" for any reason, hence is a "past sequence",
4039 now. This example shows a "past sequence" can belong to the same time frame as a "pre-announced
4040 sequence" (e.g.). Nevertheless, it is not offered any more by the power sequences server for a choice
4041 by the bound power sequences client.

4042

4043 4.3.24.3.3.2 Chronological classification of alternatives group

4044 Sequences belonging to the same "alternatives" element constitute an alternatives group (see
4045 section 4.3.24.2.2). This section defines some terms used to classify such a group chronologically. It
4046 also imposes rules on the creation of such groups.

4047 1. "Past group":

4048 A "past group" is a group that contains only "past sequences".

4049 2. "Pre-announced group":

4050 A "pre-announced group" is a group that contains only "pre-announced sequences".

4051 3. "Current group":

4052 A "current group" is a group that SHALL contain exactly one "current sequence".

4053 Furthermore, it MAY contain any number of “past sequences”. It SHOULD NOT contain any
4054 “pre-announced sequence”.

4055 Note: A power sequences server may face a condition where a previously notified “pre-announced
4056 sequence” is not valid anymore and becomes a “past sequence”. According to the rules above this
4057 sequence SHALL NOT be part of any update of the alternatives group it formerly did belong to.
4058 However, it is still possible to notify the state update of the sequence only.

4059

4060 4.3.24.3.3.3 *Sequence schedule information*

4061 A sequence has “sequence schedule information” if all the following conditions are fulfilled:

- 4062 1. The information element “alternatives. powerSequence. schedule” is present and at least its
4063 child element “startTime” is present as well.

4064 Furthermore, the following rules apply:

- 4065 1. A “pre-announced sequence” SHALL NOT have a startTime referring to the past.
4066 2. If the child element “endTime” is available its value SHALL point to a time later than
4067 “startTime”.

4068 Remark: For “sequence schedule modification” see 4.3.24.3.3.7.

4069

4070 4.3.24.3.3.4 *Sequence selection types*

4071 This section applies to “pre-announced groups” and “current groups” only.

4072 Esp. for a sensible control of a power sequences server it is important to identify if or which
4073 sequence of an alternatives group is “currently preferred or chosen” and if or which sequences are
4074 available as alternative (i.e. offered as choice to the bound power sequences client). The following
4075 selection types and rules are defined:

- 4076 1. “Chosen sequence”:
- 4077 a. A sequence is called a “chosen sequence” if it has sequence schedule information. Its
4078 state SHALL be “scheduled” or “scheduledPaused” if it is a “pre-announced
4079 sequence”. It SHALL be “running” or “paused” if it is a “current sequence”.
- 4080 b. If it is a “pre-announced sequence” the power sequences server SHALL execute the
4081 sequence autonomously at the sequence’s startTime (i.e. it becomes a “current
4082 sequence”) provided that the sequence is still valid and no other sequence of the
4083 alternatives group has been selected (by the bound power sequences client (through
4084 configuration) or by the power sequences server (for internal reasons or by user’s
4085 choice, e.g.)) as “chosen sequence”.
- 4086 Note: The autonomous start may well result in an immediate “paused” state if the
4087 sequence’s state was “scheduledPaused” before (see section 4.3.24.3.3.1).
- 4088 c. Each “current sequence” SHALL be a “chosen sequence”.
- 4089 d. Within an alternatives group at maximum one sequence can be a “chosen
4090 sequence”.
- 4091 2. “Optional sequence”:

- 4092 a. A sequence that has no sequence schedule information and its state information
- 4093 element set to “inactive” is called an “optional sequence”.
- 4094 b. An “optional sequence” SHALL NOT be executed by the power sequences server
- 4095 autonomously. If, for some reason, there is a request or need to execute this
- 4096 sequence it has to be modified to become a “chosen sequence”.
- 4097 c. Remark: Within a “pre-announced group” it is permissive to have only “optional
- 4098 sequences” and no “chosen sequence”.
- 4099 3. “Unavailable sequence”:
- 4100 a. A sequence that is neither a “chosen sequence” nor an “optional sequence” as called
- 4101 an “unavailable sequence”. It SHALL NOT have a sequence schedule information and
- 4102 it SHALL be a “past sequence”.
- 4103 Please note: This section is NOT about “past groups”.
- 4104 b. An “unavailable sequence” SHALL NOT be executed by the power sequences server.

4105

4106 4.3.24.3.3.5 Sequence modifiability

4107 4.3.24.3.3.5.1 Introduction (informative)

4108 Modifiability is required if a bound power sequences client wants to change any of the following data
4109 of the power sequences server’s sequence:

- 4110 1. The sequence overall schedule (information elements “startTime”, “endTime”).
- 4111 2. The sequence state (interruptibility, i.e. pausing/resuming or aborting a current sequence).
- 4112 3. The sequence’s slot data (slot durations, activation/deactivation, energy/power).

4113 Dependent on the data further specific conditions may be required.

4114 Furthermore, modifiability is also required if a power sequences client wants to make any of the
4115 offered “optional sequences” a “chosen sequence”.

4116

4117 4.3.24.3.3.5.2 Definition

4118 A sequence is called “modifiable” if the sequence’s information element
4119 “sequenceRemoteControllable” is available and set to “true”. Otherwise it is not modifiable.

4120

4121 4.3.24.3.3.6 Sequence interruptibility

4122 4.3.24.3.3.6.1 Definition

4123 A sequence is called “interruptible” if all of the following conditions are fulfilled:

- 4124 1. The sequence is modifiable.
- 4125 2. The information element “alternatives. powerSequence. operatingConstraintsInterrupt” is
- 4126 available AND at least one of the child elements “isPausable” and “isStoppable” are set to
- 4127 “true”.

4128 Otherwise a sequence is not interruptible. This also means the absence of the information element
4129 “alternatives. powerSequence. operatingConstraintsInterrupt” assigns “isPausable” and
4130 “isStoppable” the value “false” implicitly.

4131 The process of interrupting/resuming a sequence may be further constrained by additional data as
4132 explained in 4.3.24.3.3.6.2.

4133

4134 4.3.24.3.3.6.2 Process rules

4135 Only “current sequences” and partly “pre-announced sequences” can be interrupted and resumed by
4136 the bound power sequences client. However, further conditions apply:

- 4137 1. A “current sequence” can be paused, resumed or aborted dependent on the interruptibility
4138 data (see 4.3.24.3.3.6.1) and the state of the sequence (see 4.3.24.4.5.1.2).
- 4139 2. A “chosen sequence” of a “pre-announced group” (i.e. a “chosen pre-announced sequence”)
4140 can be changed as follows:
 - 4141 a. Its state can be changed from “scheduled” to “scheduledPaused” if the sequence is
4142 interruptible and “isPausable” is set to true.
 - 4143 b. Its state can be changed from “scheduledPaused” to “scheduled”.
 - 4144 c. It can be changed to become an “unavailable sequence”, provided that the sequence
4145 is interruptible in general and “isStoppable” is set to “true”. This requires the
4146 modification of the sequence’s information element “state” to “invalid” (see
4147 4.3.24.4.5.1.2).

4148 Remark: This item just discusses “direct” state changes. It does not discuss “indirect” state
4149 changes (e.g. from the configuration of another sequence).

- 4150 3. There can be further constraints which need to be considered as explained below.

4151 The process of interrupting/resuming a power sequences server REQUIRES that the bound power
4152 sequences client advises the power sequences server to change its state. Consequently, a bound
4153 power sequences server accepts such commands only if it permits the proper interruptibility as
4154 stated above. Please note that this process is in general independent from a sequence configuration
4155 through the bound power sequences client.

4156 If a sequence is interruptible and further constraints are specified in “alternatives. powerSequence.
4157 operatingConstraintsInterrupt” or “alternatives. powerSequence. operatingConstraintsDuration”,
4158 these constraints SHALL be considered by the bound power sequences client (esp. in order to
4159 prevent damage of the power sequences server).

4160 The information element “alternatives. powerSequence. operatingConstraintsResumeImplication”
4161 SHALL only be available if “alternatives. powerSequence. operatingConstraintsInterrupt” is available
4162 AND the child element “isPausable” is available. It SHALL NOT be considered if “isPausable” is set to
4163 “false”.

4164

4165 *4.3.24.3.3.7 Sequence schedule modification*

4166 *4.3.24.3.3.7.1 Introduction (informative)*

4167 Section 4.3.24.3.3 describes “sequence schedule information” itself. This section (4.3.24.3.3.7)
4168 describes some rules to set or modify “sequence schedule information”.

4169

4170 *4.3.24.3.3.7.2 Definition*

4171 A sequence is called to have a “modifiable sequence schedule” if all of the following conditions are
4172 fulfilled:

- 4173 1. The sequence is either a “chosen sequence” or an “optional sequence”.
4174 2. The sequence is modifiable (see 4.3.24.3.3.5).
4175 3. The information element “alternatives. powerSequence. scheduleConstraints” is available.

4176 Otherwise a sequence’s schedule information is not modifiable.

4177

4178 *4.3.24.3.3.7.3 Process rules*

4179 If a “chosen sequence” has a “modifiable sequence schedule” a bound power sequences client MAY
4180 modify the sequence’s “sequence schedule information”.

4181 If an “optional sequence” has a “modifiable sequence schedule” a bound power sequences client
4182 MAY assign the sequence a “sequence schedule information”.

4183

4184 *4.3.24.3.3.8 Sequences configuration*

4185 *4.3.24.3.3.8.1 Introduction (informative)*

4186 This section discusses the relation and dependency of modifiability aspects as described in section
4187 4.3.24.3.3.5.1. Furthermore it describes how the configuration of one sequence may affect another
4188 sequence of the same alternatives group.

4189

4190 *4.3.24.3.3.8.2 Definition*

4191 A sequence is called “configurable” if all of the following conditions are fulfilled:

- 4192 1. The sequence has a “modifiable sequence schedule” according to section 4.3.24.3.3.7.
4193 2. The sequence belongs to a “current group” or “pre-announced group”.

4194 Otherwise a sequence is called “not configurable”.

4195

4196 *4.3.24.3.3.8.3 Process rules*

4197 Throughout this section only “current groups” and “pre-announced groups” are considered.

4198 Furthermore, only “chosen sequences” and “optional sequences” of these groups are considered and
4199 are called “applicable sequences” within this section. The following rules apply:

- 4200 1. Applies to “pre-announced groups” only: Each “optional sequence” SHALL be configurable.
- 4201 2. Applies to “pre-announced groups” only: If a bound power sequences client assigns a
- 4202 configurable “optional sequence” a “sequence schedule information” (see 4.3.24.3.3.7.3) the
- 4203 sequence becomes then a “chosen sequence”. If there was another sequence of the
- 4204 alternatives group a “chosen sequence”, this other sequence becomes an “optional
- 4205 sequence” or (implementation specific) a “past sequence” then.
- 4206 3. A “current sequence” SHALL NOT become an “optional sequence” through the configuration
- 4207 of another sequence that is an “optional sequence” of the alternatives group. I.e. once a
- 4208 sequence becomes a “current sequence” it remains the “current sequence” until it
- 4209 completes successfully or is aborted finally (if applicable).
- 4210 4. Each “applicable sequence” that is configurable MAY be interruptible according to section
- 4211 4.3.24.3.3.6.
- 4212 5. Each “applicable sequence” that is configurable MAY have configurable slots according to
- 4213 section 4.3.24.3.2.7. If a sequence needs to have configurable slots the sequence itself SHALL
- 4214 be configurable.
- 4215 6. A configurable sequence MAY become a non-configurable sequence or vice versa at any
- 4216 time. Whether this happens is up to the power sequences server and implementation
- 4217 specific. However, specific process rules may impose restrictions on such changes.

4218 Note: If a “chosen sequence” becomes an “optional sequence” (or even “unavailable sequence”) its
 4219 information changes. Among others, this can “remove” a sequence’s schedule information (see
 4220 section 4.3.24.3.3.4).

4221 Please note that the configuration of a sequence through the bound power sequences client is in
 4222 general independent from a sequence’s interruptibility. However, dependent on the implementation
 4223 or further process rules it may well happen that the process of sequence configuration leads to a
 4224 change of the sequence’s interruptibility.

4225

4226 4.3.24.3.4 Overall energy management information

4227 This section describes briefly some “global information” and rules on its use.

4228 The information element “nodeScheduleInformation” contains information on the device’s overall
 4229 energy management state, i.e. regardless of a specific sequence or slot. Some of its child elements
 4230 are rather static (constant) while others typically vary over time. However, the following rule applies
 4231 in every case: The content of “nodeScheduleInformation” SHALL NOT contradict the sequence- or
 4232 slot-specific information. This also means that the content does not “supersede” sequence- or slot-
 4233 specific information. This is relevant esp. for the following information:

- 4234 1. Remote controllability
- 4235 A power sequences server is “remote controllable” exactly if it has at least one sequence that
- 4236 is configurable (see 4.3.24.3.3.8.2).
- 4237 2. Scheduling of single slots vs. multiple slots
- 4238 Even if a power sequence permits the configuration of slots it might be that the power
- 4239 sequences server requires individual configuration messages for each slot. I.e. it may not
- 4240 support the configuration of several slots within a single message. This is denoted by the
- 4241 element “supportsSingleSlotSchedulingOnly”.

3. One-time sequence selection

A power sequences server may offer to select a sequence (i.e. an “optional sequence” becomes a “chosen sequence”). As long as the sequence did not start it might be possible to alter the selection, i.e. make another sequence a “chosen sequence”. However, some device may not permit such multiple re-selections and permit just a one-time selection. The device’s possibility for re-selection is denoted by the element “supportsReselection”.

The information element “nodeScheduleInformation” can contain more than the items mentioned above. However, these are explained in the definition of the element and are not relevant for process considerations.

4.3.24.3.5 Rules for creation of information

This section primarily focuses on rules to be considered by the power sequences server. However, the rules are relevant for a power sequences client as well.

4.3.24.3.5.1 On the use of “alternativesId”

As described in section 4.3.24.3.3.2 each “alternatives” element of the server’s power sequences information contains the power sequences of the same alternatives group. Each of these groups can be identified with the element “alternatives. relation. alternativesId”. Therefore, each alternatives group exposed by a power sequences server SHALL always have a unique value in “alternativesId”, i.e. there SHALL never be concurrently valid, duplicate values of “alternativesId”. Furthermore, a power sequences server SHALL NOT reuse values of “alternativesId” immediately after an alternatives group is not present anymore on the power sequences server.

As power sequence IDs are unique across all alternatives groups (see section 4.3.24.3.5.2) the use and evaluation of alternativesId is required just in case of “restricted function exchange” (see [ProtocolSpecification]). This is in particular the case if a new power sequence is notified for an already existing alternatives group.

4.3.24.3.5.2 On the use of (power) “sequenceId”

Each power sequence exposed by a power sequences server SHALL always have a unique value in its power sequence ID (element “sequenceId”), i.e. there SHALL never be concurrently valid, duplicate power sequences IDs. Furthermore, a power sequences server SHALL NOT reuse power sequences IDs immediately after sequences with the according IDs have been completed. As an example: A washing machine should not expose its cycles with a fixed sequence ID of “1” all the time. Instead, it should increase the sequence ID with every new sequence. In general, a power sequence ID is just a simple identifier (to select a specific sequence from a list) and SHALL NOT be interpreted to carry any additional information, such as information about which sequence runs after another judging from ascending or descending IDs.

4.3.24.3.5.3 On the embedding of slots in sequences

Sequences are used together with slots within this specification.

4282

4283 *4.3.24.3.5.3.1 Common rules*

4284 The following common rules apply:

- 4285 4. Each sequence SHALL have at least one slot.
- 4286 5. Each sequence SHALL EITHER have only determined slots OR exactly one undetermined slot
4287 (see 4.3.24.3.2.2).
- 4288 6. Applies only to sequences with at least one determined slot: For at least one slot it SHALL
4289 NOT be possible to deactivate it and it SHALL always have a duration greater than 0 seconds.
4290 Remark: This requirement has no impact on the potential possibility to “deactivate” the
4291 sequence itself.

4292

4293 *4.3.24.3.5.3.2 Embedding slots in case of “repetition concept”*

4294 Some additional considerations are required in case of the “repetition concept” as briefly explained
4295 in section 4.3.24.3.2.10. Again we consider the example where a sequence consists of three different
4296 phases “A”, “B”, “C” that are processed two times, resulting in an overall phase cycle { A, B, C, A, B,
4297 C }. We assume this is represented with three slots 1, 2, 3 and accordingly the number of executions
4298 is set in the element “alternatives. powerSequence. description. repetitionsTotal” to two (please
4299 note that “repetition” here means “number of execution”). Then, the number of the current
4300 repetition has to be set in the element “alternatives. powerSequence. state.
4301 activeRepetitionNumber”.

4302 As just three explicit slot information entities are used in this example (instead of the effectively six
4303 phases/slots) this means that rules have to be defined and considered whether a slot’s information
4304 belongs to the first repetition (i.e. first execution) only or to every repetition or is updated with each
4305 repetition. These rules are defined in the element “alternatives.powerSequence.slotInformation” of
4306 message “smartEnergyManagementPsData”. As a rule of thumb, slot information of this message
4307 usually belongs to the current repetition unless stated otherwise. This means information for slot
4308 number 1 still belongs to the current repetition (in contrast to the next repetition) even if it was
4309 already processed and slot number 2 is currently processed. Constraints, on the other hand, usually
4310 (but not always) apply to each repetition. In general a configuration of slots shall be considered to
4311 apply to every repetition.

4312

4313 *4.3.24.3.5.4 Prevent group overlap in time*

4314 Pre-announced alternatives groups SHALL NOT overlap in time. No pre-announced alternatives group
4315 SHALL overlap in time with a current alternatives group.

4316 This means no schedule or schedule constraints of any sequence of alternatives group “A” shall
4317 overlap with any schedule or schedule constraints of any sequence of alternatives group “B”.

4318 Remark: Of course, sequences of the same alternatives group may well overlap in time.

4319 As a consequence, some restrictions SHALL be applied if

- 4320 1. EITHER one current alternatives group and at least one pre-announced group,

4321 2. OR at least two pre-announced groups

4322 are created:

- 4323 1. All pre-announced sequences of the alternatives group that is later in time SHALL have a
4324 schedule with startTime AND endTime or they SHALL have sequence schedule constraints
4325 with earliestStartTime AND latestEndTime.

4326

4327 4.3.24.3.5.5 Permitted sequence types

4328 4.3.24.3.5.5.1 Introduction

4329 The data models, rules (and messages described subsequently) are designed to permit a limited
4330 number of different sequence types or processes. These types are briefly described in the following
4331 sections. The description primarily makes use of terms defined in previous sections and is extended
4332 with specific restrictions or rules where required. This approach shall also help to compare the
4333 sequence types.

4334

4335 4.3.24.3.5.5.2 Sequence type "fixed forecast"

4336 This type is designed to notify a sequence where a power sequences client will not be given any kind
4337 of control. I.e. this sequence denotes a completely autonomous schedule of the power sequences
4338 server.

4339 A power sequence with this sequence type SHALL be initially announced as follows:

- 4340 1. It SHALL be notified by the power sequences server before its execution as "pre-announced
4341 sequence" unless the sequence was created suddenly and begins immediately. If it was
4342 created suddenly and begins immediately it SHALL be notified as "current sequence". In
4343 either case the sequence SHALL be the only sequence in the alternatives group (which is
4344 accordingly a "pre-announced group" or "current group", resp.).
- 4345 2. It SHALL be notified as "chosen sequence". Additionally, its "alternatives. powerSequence.
4346 schedule" child element "endTime" SHALL be set.
- 4347 Remark: Of course, slot duration uncertainties can have an impact on the precision of the
4348 whole sequence's end time as well.
- 4349 3. The sequence SHALL NOT be interruptible.
- 4350 4. The sequence SHALL NOT be configurable.
- 4351 5. The sequence SHALL have at least one slot.
- 4352 6. All slots of the sequence SHALL be determined.
- 4353 7. All slots of the sequence SHALL have an implicit slot schedule (hence are also not
4354 configurable).

4355 If the sequence becomes a "current sequence" it SHALL remain non-interruptible and non-
4356 configurable.

4357

4.3.24.3.5.5.3 Sequence type “flexible chosen forecast”

This type is designed to notify a sequence where a power sequences client usually will be given some control. I.e. this sequence denotes the preference or current configuration a power sequences server will execute autonomously unless the power sequences client takes its possibilities to modify the schedule or state.

A power sequence with this sequence type SHALL be initially announced as follows:

1. It SHALL be notified by the power sequences server before its execution as “pre-announced sequence”. It SHALL only be notified within a “pre-announced group”.
2. It SHALL be notified as “chosen sequence”. Additionally, its “alternatives. powerSequence. schedule” child element “endTime” MAY be set.
3. The sequence MAY be interruptible.
4. The sequence SHALL be configurable.
5. The sequence SHALL have at least one slot.
6. All slots of the sequence SHALL be determined.
7. Each slot SHALL have EITHER a constrained schedule OR an implicit schedule. The general rules on configurable and non-configurable slots of section 4.3.24.3.2.7 apply.

If the sequence becomes a “current sequence” its interruptibility and configurability MAY be changed by the power sequences server.

4.3.24.3.5.5.4 Sequence type “flexible optional forecast”

This type is similar to the type of section 4.3.24.3.5.5.3 but denotes just an option. I.e. a power sequences server will NOT execute this sequence autonomously. However, a power sequences client can take its possibilities to modify the schedule or state in order to make this sequence a “chosen sequence” (see below also).

A power sequence with this sequence type SHALL be initially announced as follows:

1. It SHALL be notified by the power sequences server before its execution as “pre-announced sequence”. It SHALL only be notified within a “pre-announced group”.
2. It SHALL be notified as “optional sequence”.
3. The sequence MAY be interruptible.
4. The sequence SHALL be configurable.
5. The sequence SHALL have at least one slot.
6. All slots of the sequence SHALL be determined.
7. Each slot SHALL have EITHER a constrained schedule OR an implicit schedule. The general rules on configurable and non-configurable slots of section 4.3.24.3.2.7 apply.

See section 4.3.24.3.3.8 in order to modify the kind of the sequence or its state.

In case this sequence becomes a “chosen sequence” please see section 4.3.24.3.5.5.3, as this change corresponds to the change of the sequence type.

4396 *4.3.24.3.5.5.5 Sequence type “optional sequence-based immediate control”*

4397 This type is designed for devices that cannot forecast their consumption or production with
4398 “sufficient precision” but that can at least announce an optional task with upper or lower bounds
4399 (e.g. schedule constraints or min/max values). For this sequence type the client is given at least some
4400 kind of control in order to pause and resume this task (considering constraints imposed by the power
4401 sequences server). This sequence type is initially optional and requires a proper configuration from
4402 the power sequences client in order to start at the configured time. Afterwards the power sequences
4403 client can use its options on the interrupt/resumption of the sequence.

4404 A power sequence with this sequence type SHALL be initially announced as follows:

- 4405 1. It SHALL be notified by the power sequences server before its execution as “pre-announced
4406 sequence”. It SHALL only be notified within a “pre-announced group”.
- 4407 2. It SHALL be notified as “optional sequence”.
- 4408 3. The sequence SHALL be interruptible.
- 4409 4. The sequence SHALL be configurable.
- 4410 5. The sequence SHALL have exactly one slot.
- 4411 6. The slot of the sequence SHALL be undetermined.
- 4412 Remark: This item and item 5 are in accordance with the rules in section 4.3.24.3.5.3.
- 4413 7. Remark: Due to item 6 the slot itself is not configurable.
- 4414 8. The child element “endTime” of “alternatives. powerSequence. schedule” SHALL NOT be set
4415 unless the sequence becomes a “past sequence”. I.e. neither can the power sequences
4416 server announce “endTime” in advance nor after the configuration of the schedule nor shall
4417 the power sequences client try to configure “endTime”.
- 4418 9. Remark: The absence of any duration prevents a price forecast (see 4.3.24.4.9.1).

4419 See section 4.3.24.3.3.8 in order to modify the kind of the sequence or its state.

4420 If the sequence becomes a “current sequence” it SHALL NOT be configurable any more. Its
4421 interruptibility MAY be changed by the power sequences server (temporarily).

4422 Please note: This sequence type typically requires at the power sequences client a strong
4423 consideration of constraints imposed by the power sequences server. Furthermore, this sequence
4424 type is typically more likely to be aborted at some point of time by the power sequences server
4425 autonomously.

4426

4427 *4.3.24.3.6 Rules for evaluation*

4428 This section primarily focuses on rules to be considered by the power sequences client. It is based on
4429 the sequence grouping concept and section 4.3.24.3.5.

4430 A power sequences client should analyse received information on sequences as follows:

- 4431 1. Order alternatives groups chronologically:
4432 Received alternatives groups SHALL be classified chronologically according to section
4433 4.3.24.3.3.2. Afterwards the chronological order of current or pre-announced groups shall be
4434 derived according to the rules defined in section 4.3.24.3.5.4 to prevent overlapping groups.
- 4435 2. Within each alternatives group consider rules of section 4.3.24.3.5.

4436

4437 *4.3.24.3.7 Conventions*

4438 This section describes some further conventions that SHALL apply in every case.

4439

4440 *4.3.24.3.7.1 Production vs. generation, sign convention:*

4441 The basic PowerSequences class defines the element “positiveEnergyDirection” in order to model
4442 whether positive or negative values (of power/energy) denote the production of energy. This
4443 element is not used explicitly within the complex class of this specification and is replaced by the
4444 following convention instead:

- 4445 1. Power and energy values less than zero (negative value sign) denote the production of
4446 energy.
- 4447 2. Power and energy values greater than zero (positive value sign) denote the consumption of
4448 energy.

4449 This also means an implicit assignment of “positiveEnergyDirection = consume”.

4450

4451 *4.3.24.3.7.2 Update of time information elements*

4452 All time information that identifies a point in time SHALL be updated continuously with every
4453 message communicated (e.g. the startTime of a sequence). Time information that specifies a
4454 duration of some slot or sequence that do not change over time SHALL keep the same value (e.g. the
4455 defaultDuration of a slot).

4456

4457 *4.3.24.3.7.3 Contiguous sequence data*

4458 Within a message instance, information on a given power sequence SHALL be contiguous. I.e.
4459 information on the power sequence with ID "1" is not interrupted with information on the power
4460 sequence with ID "2". The contiguity also means each power sequence SHALL be split into as few
4461 data groups as possible (unless stated otherwise by a specific message definition).

4462

4463 *4.3.24.3.8 Role concept*

4464 If a device is a power sequences server and a power sequences client at the same time, the instances
4465 of the power sequences server and the power sequences client SHALL NOT be present on the same
4466 entity. There SHALL NOT be more than one instance of a power sequences server on the same entity
4467 as well. However, there MAY be multiple instances of power sequences clients on one entity.

4468

4469 *4.3.24.3.9 Binding considerations*

4470 Power sequences servers SHALL only accept one binding to a power sequences client for each
4471 instance of a power sequences server entity. Once the power sequence instance is bound to a client,
4472 the power sequences server SHALL reject additional binding requests for this power sequence
4473 instance.

4474 If a power sequences server detects that a bound power sequence client is not accessible any more
4475 (e.g. since it has been sold to another household), it is a vendor specific implementation to either
4476 delete the binding or continue waiting for the bound device. Thus, it might only be possible for
4477 device owners to re-bind the power sequences server with a factory reset of the device.

4478 Devices which are interested in data of a power sequences server, but don't configure the power
4479 sequences server flexibilities, SHALL only use data subscriptions or explicit read commands without
4480 bindings to receive updated information and SHALL never try to create bindings to a power
4481 sequences server.

4482 For more details about binding, please consider [ProtocolSpecification] .

4483

4484 *4.3.24.3.10 General error handling*

4485 *4.3.24.3.10.1 Acknowledgement concept*

4486 The acknowledgement concept as specified in [ProtocolSpecification] applies. A bound power
4487 sequences server and bound power sequences client SHOULD explicitly request acknowledgement
4488 messages for message exchanged among each others (provided the message type permits
4489 acknowledgement request).

4490

4491 *4.3.24.3.10.2 Valid vs. invalid classifiers*

4492 Subsequent message specific sections contain sub-sections with the headline "When to send". These
4493 sub-sections determine explicitly under which condition a given message shall be sent and also which
4494 classifier(s) shall be used. These classifiers are subsequently considered valid for the respective
4495 messages and conditions. Other classifiers are considered invalid for the respective messages.

4496 A brief overview of valid classifiers is given in Table 113. Furthermore, sub-sections with the headline
4497 "When received" usually mention valid classifiers only.

4498

4499 **4.3.24.4 Message purpose specification details**

4500 This section defines details of the message purposes that were briefly announced in Table 113 of
4501 section 4.3.24.1.

4502

4503 *4.3.24.4.1 powerSequences information*

4504 *4.3.24.4.1.1 General data model description and rules*

4505 *4.3.24.4.1.1.1 General aspects*

4506 With a power sequences information message a power sequences server exposes all power
4507 sequences (potentially) relevant for the power sequences client, which may be a combination of:

4508 - Alternative processes to choose from:

4509 This has already been introduced in section 4.3.24.2.2 as choice of power sequences: A

4510 device may have different operation modes or (more generally speaking) mutually exclusive
4511 processes, which it offers to a power sequences client. The client can only choose one of
4512 these mutually exclusive processes. These processes SHALL be exposed within one
4513 “alternatives” element.

4514 - Single or multiple, sequenced processes:
4515 A device may have different processes which are executed in a serialized manner, i.e. cannot
4516 be parallelized or run concurrently. These processes SHALL be exposed within different
4517 “alternatives” elements.

4518 A power sequences server can also expose a combination of these options. As an example, we
4519 consider the case of an HVAC system, which knows that it has to run in the morning and in the
4520 evening. These processes cannot be run concurrently. However, it has 2 modes of heating: One with
4521 high power consumption, but shorter heating cycle and one with a lower power consumption and
4522 longer heating cycle. In this case, the HVAC system would expose 2 “alternatives” elements, each
4523 containing 2 power sequences. One of these pairs of power sequences would reflect the different
4524 power consumption modes during the morning whereas the second pair does the same for the
4525 evening.

4526

4527 *4.3.24.4.1.1.2 Cancelling (deleting/aborting) offered power sequences*

4528 There might be situations in which a power sequences server offers one or more power sequences,
4529 but encounters a situation which renders one or more of the offered power sequence(s) invalid. In
4530 such situations the affected power sequences SHALL be cancelled/deleted. As an example, a washing
4531 machine might offer a power sequence representing a washing cycle, but a few seconds later the
4532 user of the washing machine cancels the washing cycle, because he has chosen the wrong washing
4533 program. If a power sequences server has offered one or more power sequences which are now
4534 invalid, it SHALL send a “powerSequences information” with a notify classifier and set the according
4535 power sequence to an “invalid” state. Afterwards it SHALL remove the according power sequence ID
4536 and respect the rules on reusing sequence IDs (see section 4.3.24.3.5.2). The deletion of the
4537 sequence MAY result in a delete-notification (see section "Restricted function exchange with
4538 cmdOptions" in [ProtocolSpec]), but it is not recommended to do so.

4539 In general, a power sequences server SHALL cancel all power sequences that have been announced
4540 and possibly also configured, but are no longer valid.

4541 Errors might occur if a power sequences client does not receive the notification(s) about cancelled
4542 power sequence(s), e.g. because the power sequences client is currently offline. However, in these
4543 situations the power sequences client SHOULD be aware of potentially lost packets and SHOULD
4544 apply appropriate mechanisms, such as explicitly requesting the power sequences available.

4545

4546 *4.3.24.4.1.1.3 Processes that might be run in parallel*

4547 Please note a single instance of a power sequences server SHALL NOT expose processes that
4548 (potentially or partly) run at the same time! To give an example: Consider a fridge/freezer
4549 combination with independent temperature control. I.e. for both the fridge temperature control as
4550 well as for the freezer temperature control a process can be exposed. However, as these processes

are independent from each other (hence can also run in parallel) each process has to be exposed by an individual power sequences server instance. This means there has to be a power sequences server for cooling processes and another power sequences server for freezing processes. As stated in section 4.3.24.3.8, these power sequences server instances SHALL be located on different entities.

4.3.24.4.1.1.4 Power sequences with and without flexibilities

Please note that power sequences might or might not contain flexibilities, i.e. offer time-shifting of a power sequence or slot or offer slots which have durations which can be reduced or extended, i.e. have a configurable duration. A power sequence with schedule flexibilities SHALL contain schedule constraints. A power sequence not containing flexibilities, i.e. a fixed forecast, SHALL NOT contain schedule constraints.

4.3.24.4.1.1.5 Consistency of a power sequence

In case of a sequence with flexible schedule (and determined slots), a sequence's "earliest start time" and "latest end time" SHALL always frame the slots that are part of the sequence. I.e. the sum of durations of the slots in a sequence SHALL always fit into the frame given by these elements. Care has to be taken if slots can be extended regarding their duration, i.e. which have a configurable duration: Extending all durations to their maximum may exceed the time frame permitted by a sequence's "earliest start time" and "latest end time". I.e. such configurations are invalid and shall be rejected by the power sequences server. However, there SHALL at least be one possible configuration of the duration of all slots in a sequence that fits into the frame given by "earliest start time" and "latest end time" of the sequence.

As an example, consider the slots 1, 2 and 3 of a sequence. All of these slots may have a typical duration of 5 minutes, but all of them can be extended so they have a duration of 15 minutes each. This results in a minimum length of 15 minutes and a maximum length of 45 minutes for the whole sequence. However, the power sequences server might be limited to an "earliest start time" and "latest end time", which only provide a duration of 35 minutes for the whole sequence. In this example, the power sequences client would be free to either extend the duration of all slots equally so they fit into the frame, but it might also only extend slot 1 and 2 to their maximum duration. The power sequences client might, however, also choose to not extend the duration of any of the slots at all.

4.3.24.4.1.1.6 Possible state values

The states delivered in a power sequences state can have several, pre-defined values. This Feature Type uses as well the definition of PowerSequenceStateType (see Table 269, section 5.3.19.9.2), but restricts the permitted values. The following values are allowed within the scope of this specification:

- "running": The according sequence is currently executed.
- "paused": The according sequence is paused.
- "scheduled": The according sequence will be executed by the power sequences server at the scheduled time. It will then usually be in the state "running".

- 4591 • “scheduledPaused”: The according sequence will be executed by the power sequences server
- 4592 at the scheduled time – but it will then usually be in the state “paused”.
- 4593 • “inactive”: The according sequence has been offered as optional choice, but has not been
- 4594 configured.
- 4595 • "completed": The according sequence has been completed.
- 4596 • "invalid": The according sequence has been cancelled. Please consider section 4.3.24.4.1.1.2.

4597

4598 *4.3.24.4.1.1.7 Data model description and rules*

4599 The complex function “smartEnergyManagementPsData” of the complex class

4600 “SmartEnergyManagementPs” is used with the following restrictions and rules:

Element name	M/O/C (1.4.7)	Explanation
nodeScheduleInformation	M	
nodeScheduleInformation. nodeRemoteControllable	M	If nodeRemoteControllable is equal to "false" no power sequence is remote controllable, independent from the value of the element "sequenceRemoteControllable". If nodeRemoteControllable is set to "true", it depends on the value of the element "sequenceRemoteControllable" if a specific sequence is remote controllable, or not.
nodeScheduleInformation. supportsSingleSlotSchedulingOnly	M	If set to “true” the power sequences server applies a restriction on received “powerSequences configuration” messages: The element “alternatives. powerSequence” SHALL NOT contain more than one child element “powerTimeSlot”. This restriction does not apply if supportsSingleSlotSchedulingOnly is set to “false”.
nodeScheduleInformation. alternativesCount	M	
nodeScheduleInformation. totalSequencesCountMax	M	
nodeScheduleInformation. supportsReselection	M	
alternatives	O	SHALL only be left out if no power sequences are present within its group. Otherwise, it SHALL be present.
alternatives. relation	M	
alternatives. relation. alternativesId	M	See section 4.3.24.3.5.1.
alternatives. powerSequence	M	
alternatives. powerSequence. description	M	

alternatives. powerSequence. description. sequenceId	M	
alternatives. powerSequence. description. description	O	Default rules of section 3.10.1.3.1 SHALL be applied.
alternatives. powerSequence. description. powerUnit	O	Element SHOULD be set. If omitted, "W" SHALL be assumed as unit for the value implicitly.
alternatives. powerSequence. description. energyUnit	O	Element SHOULD be set. If omitted, "Wh" SHALL be assumed as unit for the value implicitly.
alternatives. powerSequence. description. valueSource	O	If absent, the source of the forecasted values is undefined.
alternatives. powerSequence. description. taskIdentifier	O	
alternatives. powerSequence. description. repetitionsTotal	O	MAY be present. Absence of the element is equal to a presence with a value of 1 (one). SHALL be absent if the value is 1. If a power sequence executes its sequence of slots more than one time, the element SHALL be present and contain the total number of executions.
alternatives. powerSequence. state	M	
alternatives. powerSequence. state. state	M	Consider esp. section 4.3.24.3.3.1 on restrictions of permitted values from the enumeration!
alternatives. powerSequence. state. activeSlotNumber	O	If state is set to "running" or "paused" this element SHALL contain the currently active slot number. Otherwise it SHALL be omitted.
alternatives. powerSequence. state. elapsedSlotTime	O	If state is set to "running" or "paused" AND the slot is determined (see 4.3.24.3.2.2) this element MAY contain the time the slot has already been in "running" state (this also means the value remains constant during a "paused" state). Otherwise it SHALL be omitted.
alternatives. powerSequence. state. remainingSlotTime	O	If state is set to "running" or "paused" AND the slot is determined (see 4.3.24.3.2.2) this element SHALL contain the time the slot still needs to be in "running" state (this also means the value remains constant during a "paused" state). Otherwise it SHALL be omitted.

alternatives. powerSequence. state. sequenceRemoteControllable	M	Denotes whether the sequence is modifiable (value “true”) or not (value “false”), see 4.3.24.3.3.5. Modifiability is required to configure sequences and slots (see section 4.3.24.4.4). It is also required to change a power sequence state (if supported; see 4.3.24.4.5).
alternatives. powerSequence. state. activeRepetitionNumber	O	SHALL be present if alternatives. powerSequence. description. repetitionsTotal is present and has a value greater than 1. Otherwise, it SHALL be absent.
alternatives. powerSequence. state. remainingPauseTime	O	This element SHALL ONLY be present if the power sequence is interruptible. Otherwise, it SHALL be omitted. In case of the sequence’s interruptibility the following rules apply: If the element is absent this means there is no explicit pause duration restriction for the current slot; a value of 0s denoted the slot does not permit being paused.
alternatives. powerSequence. schedule	O	If the sequence has a schedule, this element with its sub-elements SHALL be present. Otherwise it SHALL be omitted.
alternatives. powerSequence. schedule. startTime	M	
alternatives. powerSequence. schedule. endTime	O	If the power sequence has an endTime, it SHALL be put here. Otherwise the element SHALL be omitted.
alternatives. powerSequence. scheduleConstraints	O	If there are constraints available, this element and its sub-elements SHALL be present. Otherwise it SHALL be omitted.
alternatives. powerSequence. scheduleConstraints. earliestStartTime	M	Only “xs:duration” value types SHALL be used to denote a relative time which relates to “now” as time 0.
alternatives. powerSequence. scheduleConstraints. latestEndTime	M	Only “xs:duration” value types SHALL be used to denote a relative time which relates to “now” as time 0.
alternatives. powerSequence. schedulePreference	O	MAY be present. A power sequences client should analyse this element when it configures the power sequences server. Note: There may be several options available which cannot be fulfilled altogether. This means the options can in general only be considered “as good as it can get”.
alternatives. powerSequence. schedulePreference. greenest	O	MAY be present. Absence of this element is equal to the presence with value “false”.
alternatives. powerSequence. schedulePreference. cheapest	O	MAY be present. Absence of this element is equal to the presence with value “false”.

alternatives. powerSequence. operatingConstraintsInterrupt	O	If the sequence may be paused or stopped by the power sequences client, this element and the required sub-elements SHALL be present. Otherwise it SHALL be omitted.
alternatives. powerSequence. operatingConstraintsInterrupt. isPausable	O	If the sequence is pausable by the bound power sequences client, this element SHALL be present and set to true. Otherwise it SHALL be omitted.
alternatives. powerSequence. operatingConstraintsInterrupt. isStoppable	O	If the sequence is stoppable by the bound power sequences client, this element SHALL be present and set to true. Otherwise it SHALL be omitted.
alternatives. powerSequence. operatingConstraintsInterrupt. maxCyclesPerDay	O	
alternatives. powerSequence. operatingConstraintsDuration	O	The element SHALL ONLY be present if at least one of its child elements is present. Otherwise it SHALL be omitted.
alternatives. powerSequence. operatingConstraintsDuration. activeDurationMin	O	
alternatives. powerSequence. operatingConstraintsDuration. activeDurationMax	O	
alternatives. powerSequence. operatingConstraintsDuration. pauseDurationMin	O	
alternatives. powerSequence. operatingConstraintsDuration. pauseDurationMax	O	
alternatives. powerSequence. operatingConstraintsDuration. activeDurationSumMin	O	
alternatives. powerSequence. operatingConstraintsDuration. activeDurationSumMax	O	
alternatives. powerSequence. operatingConstraintsResumel mplication	O	The element SHALL ONLY be present if at least one of its child elements is present. Otherwise it SHALL be omitted.

alternatives. powerSequence. operatingConstraintsResumeI mplication. resumeEnergyEstimated	O	
alternatives. powerSequence. operatingConstraintsResumeI mplication. energyUnit	O	For electric energy, this element SHALL be omitted and Wh SHALL be assumed as unit for the value implicitly. For any other energy type, the unit SHALL be stated explicitly.
alternatives. powerSequence. operatingConstraintsResumeI mplication. resumeCostEstimated	O	
alternatives. powerSequence. operatingConstraintsResumeI mplication. currency	O	
alternatives. powerSequence. powerTimeSlot	M	
alternatives. powerSequence. powerTimeSlot. schedule	M	
alternatives. powerSequence. powerTimeSlot. schedule. slotNumber	M	
alternatives. powerSequence. powerTimeSlot. schedule. timePeriod	O	SHALL ONLY be present if at least one of its child elements is present.
alternatives. powerSequence. powerTimeSlot. schedule. timePeriod. startTime	O	
alternatives. powerSequence. powerTimeSlot. schedule. timePeriod. endTime	O	The following equation SHALL apply: $endTime - startTime = defaultDuration$.
alternatives. powerSequence. powerTimeSlot. schedule. defaultDuration	O	SHALL be present in case of “determined slot” and state the duration of the slot. If the slot has a configurable length, this element SHALL reflect the currently configured length.
alternatives. powerSequence. powerTimeSlot. schedule. durationUncertainty	O	May only be used if alternatives. powerSequence. powerTimeSlot. schedule. defaultDuration is set. Otherwise it SHALL be omitted.

alternatives. powerSequence. powerTimeSlot. schedule. slotActivated	O	If the slot is optional this element SHALL be present and reflect the current status of the slot. If the slot is not optional, this element SHALL be absent.
alternatives. powerSequence. powerTimeSlot. schedule. description	O	Default rules of section 3.10.1.3.1 SHALL be applied.
alternatives. powerSequence. powerTimeSlot. valueList	M	
alternatives. powerSequence. powerTimeSlot. valueList. value	M (1..6)	<p>The following restriction to the list size applies: 1..6 occurrences of “value” are permitted.</p> <p>If more than just one “value” in “valueList” is present, each “value” in “valueList” SHALL state a different valueType than each other “value” in “valueList” of this slot.</p> <p>See section 4.3.24.3.2.5 for details about allowed values.</p>
alternatives. powerSequence. powerTimeSlot. valueList. value. valueType	M	
alternatives. powerSequence. powerTimeSlot. valueList. value. value	M	
alternatives. powerSequence. powerTimeSlot. scheduleConstraints	O	<p>If a slot has information about constraints for a time slot, this element and the required sub-elements SHALL be present. Otherwise AND for fixed forecasts (i.e. sequences without any flexibility), this element SHALL be omitted.</p> <p>Note: In case of a power sequence with repetition of its sequence of slots, this element often applies to the first execution only! Details are explained in the description of each child element.</p>
alternatives. powerSequence. powerTimeSlot. scheduleConstraints. earliestStartTime	O	<p>Only “xs:duration” value types SHALL be used to denote a relative time which relates to “now” as time 0.</p> <p>Note: This element applies to the first execution of the slot number only!</p>
alternatives. powerSequence. powerTimeSlot. scheduleConstraints. latestEndTime	O	<p>Only “xs:duration” value types SHALL be used to denote a relative time which relates to “now” as time 0.</p>

		Note: This element applies to the first execution of the slot number only!
alternatives. powerSequence. powerTimeSlot. scheduleConstraints. minDuration	O	If the slot has a configurable duration, this element SHALL be present and denote the minimum supported configuration. Please also see section 4.3.24.3.2.7. Note: This element applies to every execution (repetition) of the slot number.
alternatives. powerSequence. powerTimeSlot. scheduleConstraints. maxDuration	O	If the slot has a configurable duration, this element SHALL be present and denote the maximum supported configuration. Please also see section 4.3.24.3.2.7. Note: This element applies to every execution (repetition) of the slot number.
alternatives. powerSequence. powerTimeSlot. scheduleConstraints. optionalSlot	O	SHALL be present and set to “true” if the slot can be omitted. Otherwise the element SHALL be omitted or set to “false”. I.e. the value “false” SHALL be assumed implicitly if the element is omitted. Note: This element applies to every execution (repetition) of the slot number.

Table 114: Description of complex class function “smartEnergyManagementPsData” for powerSequences information

4.3.24.4.1.2 When received

If a power sequences server receives this message, it SHALL consider this as erroneous receipt.

4.3.24.4.1.3 When to send

A power sequences server SHALL produce this message with a “notify” classifier and send it to the subscribed nodes if the data for one of its power sequences on this server instance changes.

A power sequences server SHALL produce a proper message with a “reply” classifier upon a received “read” request according to the rules described in section 4.3.24.4.2. This reply may then be a complete or restricted function exchange.

4.3.24.4.1.4 Error considerations

See section 4.3.24.4.1.2.

4.3.24.4.2 powerSequences information request

4.3.24.4.2.1 General data model description and rules

A power sequences information request is a request for the power sequences information of a power sequences server.

- 4620 A power sequences server SHALL answer to power sequences information requests, even if most of
4621 the power sequences communication relies on event based (unsolicited) transmissions.
- 4622 The request to get power sequences information is sent with “read” classifier and an “empty”
4623 complex class function “smartEnergyManagementPsData”.
- 4624 Optionally, the request MAY be modelled for “restricted function exchange” using a dedicated
4625 selectors. This version of the specification RECOMMENDS to support the request for information of a
4626 specific power sequence as shown in Table 115.

Element name	M/O/C (1.4.7)	Explanation
smartEnergyManagementPsDataSelectors	O	SHALL ONLY be present if a child element is required. Otherwise it SHALL be absent.
smartEnergyManagementPsDataSelectors . alternativesRelation	-	Not considered in this recommendation.
smartEnergyManagementPsDataSelectors . alternativesRelation. alternativesId	-	Not considered in this recommendation.
smartEnergyManagementPsDataSelectors . powerSequenceDescription	M	SHALL ONLY be present if the following child element is required. Otherwise it SHALL be absent.
smartEnergyManagementPsDataSelectors . powerSequenceDescription. sequenceId	M	SHALL be present and set to the sequenceId of the power sequence whose information is requested.
smartEnergyManagementPsDataSelectors . powerTimeSlotSchedule	-	Not considered in this recommendation.
smartEnergyManagementPsDataSelectors . powerTimeSlotSchedule. slotNumber	-	Not considered in this recommendation.

4627 Table 115: Dedicated “selectors” of “smartEnergyManagementPsData” to request a specific powerSequences information.

4628

4629 4.3.24.4.2.2 When received

- 4630 If a power sequences server receives a power sequences information request with a “read” classifier,
4631 it SHALL answer with a proper “power sequence information”.

- 4632 If the request is modelled for “restricted function exchange” as shown in Table 115, a reply with
4633 “smartEnergyManagementPsData” (as described in section 4.3.24.4.1) is modified as follows: The
4634 response SHALL only contain information of the power sequence with the same sequenceId value of
4635 the “selectors” content.

4636

4637 4.3.24.4.2.3 When to send

- 4638 A power sequences server SHALL never produce this message.

4639

4.3.24.4.2.4 Error considerations

If the power sequences server gets a power sequence request containing an invalid sequenceId (a sequenceId which does not exist at the time of the request on this instance of the power sequences server), the power sequences server SHALL consider this as erroneous receipt.

4.3.24.4.3 powerSequences configuration request call

4.3.24.4.3.1 General data model description and rules

This function can be sent from a power sequences server to its bound power sequences client if it wants one or “all” of its power sequences to be configured. It is intended as “trigger” for the bound power sequences client that may then initiate a power sequences configuration to configure the power sequences server’s power sequences (see 4.3.24.4.4).

The server’s message uses the complex function

“smartEnergyManagementPsConfigurationRequestCall” of the complex class

“SmartEnergyManagementPs” with the following restrictions and rules:

Element name	M/O/C (1.4.7)	Explanation
scheduleConfigurationRequest	M	
scheduleConfigurationRequest . sequenceId	O	If present, denotes the sequence ID of the power sequences server which SHOULD be configured. More precisely, it denotes the current power sequence preference of a single alternatives group. If “sequenceId” is not present, “all” power sequences of the power sequences server SHOULD be configured (more precisely: from each alternatives group a configuration of a proper power sequence should be done, if feasible).

Table 116: Description of complex class function “smartEnergyManagementPsConfigurationRequestCall” for powerSequences configuration request call

4.3.24.4.3.2 When received

If a power sequences server receives this message, it SHALL consider this as erroneous receipt.

4.3.24.4.3.3 When to send

A power sequences server SHALL only send this message with a “call” classifier to its bound power sequences client. It SHALL never send this message to clients that are only subscribed.

Once a power sequences server has a bound power sequences client and at least one of the power sequences contains flexibilities, it MAY send a power sequences configuration request call. If none of

4665 the power sequences contains flexibilities, it SHALL NOT send a power sequences configuration
4666 request call.

4667 A power sequences server MAY send this message containing a “sequenceld” to explicitly trigger the
4668 configuration of a specific power sequence or any of the (permissive) sequences of the same
4669 alternatives group. However, this SHALL NOT be done for fixed forecasts.

4670 A power sequences server MAY also omit the “sequenceld” in the message. This denotes the request
4671 to get a configuration for each alternatives group except for those with fixed forecast.

4672

4673 4.3.24.4.3.4 *Error considerations*

4674 If a power sequences server encounters a failure with its transmission of the power sequences
4675 configuration request call (e.g. explicitly from a received error message, or from any internal
4676 transmission API), the power sequences server SHOULD consider that the bound power sequences
4677 client may currently be not reachable or able to process the information, hence the client will not
4678 send a configuration. The absence of a power sequences configuration from the client SHALL NOT be
4679 considered as a problem, because the power sequences client may need more time for the
4680 calculation or needs to collect some further information beforehand.

4681 If the bound power sequences client has not sent a power sequences configuration within 15
4682 minutes, the power sequences server MAY consider to re-announce its power sequences and do a
4683 power sequences configuration request call or run as it would do in stand-alone mode, if it lacks time
4684 to wait for a configuration.

4685 However, in which way a power sequences server handles errors is up to the manufacturer. For a
4686 good user experience, a power sequences server SHOULD, however, not behave in a different way
4687 than it would do in stand-alone mode. A washing machine SHOULD consider starting a cycle with a
4688 configuration defined on its own, while a battery storage MAY consider to idle or execute a
4689 maintenance cycle, e.g.

4690

4691 4.3.24.4.4 *powerSequences configuration*

4692 4.3.24.4.4.1 *General data model description and rules*

4693 The power sequences configuration is for use with a “write” classifier together with proper
4694 restriction information from the power sequences client to a bound power sequences server only.

4695 A power sequences server may hold more than one power sequence with configurability at a time.
4696 However, a power sequences client SHALL only configure one power sequence within one command,
4697 i.e. configuring one power sequence after the other.

4698 If the power sequences server offered multiple power sequences which were mutual alternatives, a
4699 configuration of one of these power sequences SHALL be interpreted as a choice for this specific
4700 power sequence. A consecutive configuration of another of the alternative power sequences SHALL
4701 be interpreted as a superseding choice between the mutual alternative power sequences.

4702 There are 3 possibilities for configuring a power sequence:

- 4703 - Configuration option A: By configuring the start time of a whole sequence
 - 4704 ○ If a power sequences server uses this method, it SHALL start the sequence at the
 - 4705 given time and follow its own preference of the slot durations.
 - 4706 ○ MAY only be used by the power sequences client to configure the bound power
 - 4707 sequences server if the durations of all slots are already set (by the server as its own
 - 4708 preference). Otherwise the client SHALL NOT use this configuration method.
- 4709 - Configuration option B: By configuring the end time of a whole sequence
 - 4710 ○ If a power sequences server uses this method, it SHALL calculate the start time of the
 - 4711 sequence by subtracting the expected durations of all slots (including all repetitions,
 - 4712 if any) from the configured end time. Then, the server SHALL start the sequence at
 - 4713 the calculated time and follow its own preference of the slot durations.
 - 4714 Please note that the sum of expected durations SHOULD consider slot duration
 - 4715 uncertainties in order to finish not later than the configured endTime.
 - 4716 ○ MAY only be used by the power sequences client to configure the bound power
 - 4717 sequences server if the durations of all slots are already set (by the server as its own
 - 4718 preference). Otherwise the client SHALL NOT use this configuration method.
- 4719 - Configuration option C: By configuring all configurable slots of a sequence
 - 4720 ○ If a power sequences server uses this method, it SHALL set the durations of all slots
 - 4721 according to the given configuration, if all constraints are meeting the server's
 - 4722 requirements.
 - 4723 ○ If a power sequences client wants to use this method in its configuration message, it
 - 4724 SHALL retain the order of the slot numbers it received from the power sequences
 - 4725 server message with power sequences information. Additionally, at least the start
 - 4726 time or the end time of the sequence SHALL be configured (since only durations are
 - 4727 included within the slots).
 - 4728 ○ Please also note that this is the only method to re-configure a power-sequence that
 - 4729 has already begun (i.e. at least the first slot has been started), provided that this
 - 4730 power sequence still offers flexibilities.
 - 4731 ○ Configurable slots are:
 - 4732 ▪ Slots whose duration can be changed
 - 4733 ▪ Slots that are marked as optional (and hence can be set active or inactive)
 - 4734 ▪ Slots with a combination of the above stated configurabilities

4735 Note: This section is just about the configuration of a sequence, not about a "state change": Apart
 4736 from a configuration a power sequences server's active power sequence may also be modified by a
 4737 state change (provided that this functionality is enabled). This is described in 4.3.24.4.5.

4738 The configuration message is modelled with a specific "write" operation to the power sequences
 4739 server's instance of "smartEnergyManagementPsData" (i.e. the server's powerSequences
 4740 information as described in section 4.3.24.4.1). The message SHALL be composed as follows:

- 4741 1. The classifier SHALL be set to "write".
- 4742 2. The destination of the message SHALL be the power sequences server's instance of the
- 4743 complex function "smartEnergyManagementPsData".
- 4744 3. The message SHALL be modelled for "restricted function exchange" (this also means the
- 4745 "payload" element "cmd.function" SHALL be set properly; in this case it SHALL be
- 4746 "smartEnergyManagementPsData").

- 4747 4. The “cmdControl” SHALL contain “partial”.
- 4748 5. A (curtailed) function smartEnergyManagementPsData” SHALL be given, containing only the
- 4749 data to modify. The content is defined by Table 117.

4750

Element name	M/O/C (1.4.7)	Explanation
alternatives	M (1..1)	The following restriction to the list size applies: Exactly one occurrence of “alternatives” is permitted.
alternatives. powerSequence	M (1..1)	The following restriction to the list size applies: Exactly one occurrence of “powerSequence” is permitted within its parent element.
alternatives. powerSequence. description	M	
alternatives. powerSequence. description. sequenceId	M	States the sequenceId for this configuration.
alternatives. powerSequence. schedule	O	Its presence depends on its child elements.
alternatives. powerSequence. schedule. startTime	See right, (*1).	If used, SHALL state the relative start time of the whole power sequence. If used with configuration option A, elements alternatives. powerSequence. schedule. endTime and element alternatives. powerSequence. powerTimeSlot shall be omitted, since the whole power sequence is configured. (*1) Element presence: For configuration option A: M B: NV C: If endTime is used: O; Otherwise: M
alternatives. powerSequence. schedule. endTime	See right, (*1).	If used, SHALL state the relative end time of the whole power sequence. If used with configuration option B, elements alternatives. powerSequence. schedule. startTime and alternatives.

		<p>powerSequence. powerTimeSlot SHALL be omitted, since the whole power sequence is configured.</p> <p>(*1) Element presence:</p> <p>For configuration option</p> <p>A: NV</p> <p>B: M</p> <p>C: If startTime is used: O; Otherwise: M</p>
alternatives. powerSequence. powerTimeSlot	See right, (*1).	<p>List of power time slot schedule configuration data.</p> <p>(*1) Element presence:</p> <p>For configuration option</p> <p>A: NV</p> <p>B: NV</p> <p>C: M</p>
alternatives. powerSequence. powerTimeSlot. schedule	See right, (*1).	<p>The power time slot schedule configuration of a slot.</p> <p>(*1) Element presence:</p> <p>For configuration option</p> <p>A: NV</p> <p>B: NV</p> <p>C: M</p>
alternatives. powerSequence. powerTimeSlot. schedule. slotNumber	M	States the slot number for which this configuration applies.
alternatives. powerSequence. powerTimeSlot. schedule. defaultDuration	O	SHALL be present if the slot has a configurable length AND the slot duration shall be changed to this new value.

alternatives. powerSequence. powerTimeSlot. schedule. slotActivated	O	If the optionality of the slot is configurable and its activation status shall be changed to the new value, this element SHALL be present and denote whether the slot shall be used (slotActivated = "true") or not (slotActivated = "false"). Otherwise is SHALL be absent (i.e. the slot is mandatory).
---	---	---

Table 117: Description of "restricted write" operation for powerSequences configuration.

4.3.24.4.4.2 When received

A power sequences server SHALL only process power sequences configuration data received with a "write" classifier and proper restriction elements. Otherwise, a power sequences server SHALL consider this as erroneous receipt.

A power sequences server SHALL only accept power sequences configuration data that fit (with an "acceptable tolerance" as explained below) to the constraints of the according sequenceId. A power sequences server SHALL only accept power sequences configuration that configure all flexibilities that are available (durations, etc.) offered in one "write" operation for one power sequence.

If a power sequences server receives a power sequences configuration data with a configuration this power sequence cannot fulfil, it SHALL consider this as erroneous receipt and – if an acknowledgement message is to be sent – respond with a negative acknowledgement message with error number "7" (command rejected).

On "acceptable tolerance": The transmission of a configuration may be delayed "a bit too much" due to network problems, e.g. In this case the time configuration may violate the constraints given by the server "a bit". In such a case, the server SHOULD tolerate and adjust the received configuration silently as long as the violation is negligible. In general a tolerance of 30 seconds SHOULD be applied on time constraints.

Note: It may happen that a received power sequences configuration is valid according to the rules stated above but leads to no change at all of the sequence schedule (e.g. because received start/end times are identical to the schedule the server already announced). This SHALL NOT be considered an error.

4.3.24.4.4.3 When to send

A power sequences server SHALL never produce this message.

4.3.24.4.4.4 Error considerations

If the power sequences server gets a power sequence configuration containing an invalid sequenceId (a sequenceId which does not exist on this instance of the power sequences server), the power sequences server SHALL consider this as erroneous receipt.

See section 4.3.24.4.3.4 for more error considerations.

4783

4784 *4.3.24.4.5 powerSequences state change*4785 *4.3.24.4.5.1 General data model description and rules*4786 *4.3.24.4.5.1.1 General aspects*

4787 The normal procedure for a power sequences client to control a bound power sequences server's
4788 power sequences is to initiate a powerSequences configuration as described in section 4.3.24.4.4. In
4789 some cases, a more direct way to control power sequences is needed. For this the "powerSequences
4790 state change" can be used. It enables a bound power sequences client to request a change of the
4791 state of a power sequences server's power sequence to "run", "pause" or "invalid" (i.e. terminate),
4792 dependent on the current state of the power sequence and some flags (see section 4.3.24.4.5.1.2 for
4793 details about the rules). This is based upon the definitions of section 4.3.24.3.3.6, among others.

4794 Please note that state changes may occur from other situations as well. Among others, the state of a
4795 currently "chosen sequence" changes if another sequence is configured to become a "chosen
4796 sequence". In this case, the previously "chosen sequence" may get the state "inactive" (e.g.).
4797 However, this is an "indirect" state change which is caused by (server) internal mechanisms
4798 according to this specification's rules. Such "indirect" state changes are NOT subject of this section.

4799

4800 *4.3.24.4.5.1.2 Rules for changing the state*

4801 The state of a power sequences server's power sequence MAY be changed by the bound power
4802 sequence client if the following aspects are considered:

- 4803 1. If the power sequence is in state scheduled AND the flag isPausable is set to true AND the
4804 flag sequenceRemoteControllable is set to true: The client then MAY request for the state
4805 change:
 - 4806 ○ scheduledPaused
- 4807 2. If the power sequence is in state scheduledPaused AND the flag isPausable is set to true AND
4808 the flag sequenceRemoteControllable is set to true: The client then MAY request for the
4809 state change:
 - 4810 ○ scheduled
- 4811 3. If the power sequence is in state running AND the flag isPausable is set to true AND the flag
4812 sequenceRemoteControllable is set to true: The client then MAY request for the state
4813 change:
 - 4814 ○ paused
- 4815 4. If the conditions in item 3 are met and additionally the flag isStoppable is set to true, the
4816 client MAY request for the state changes:
 - 4817 ○ paused
 - 4818 ○ invalid
- 4819 5. If the power sequence is in state paused AND the flag isPausable is set to true AND the flag
4820 sequenceRemoteControllable is set to true, the client MAY request for the state change:
 - 4821 ○ running
- 4822 6. If the conditions in item 5 are met AND additionally the flag isStoppable is set to true, the
4823 client MAY request for the state changes:
 - 4824 ○ running

- 4825 ○ invalid
- 4826 7. If the power sequence is in state running, paused, scheduled, scheduledPaused or inactive
- 4827 AND the flag isStoppable is set to true, the client MAY request for the state change:
- 4828 ○ invalid

4829 In all other cases, a (direct) state change SHALL NOT be requested by the bound power sequences

4830 client.

4831

4832 4.3.24.4.5.1.3 Data model description and rules

4833 The “powerSequences state change” message is modelled with a specific “write” operation to the

4834 power sequences server’s instance of “smartEnergyManagementPsData” (i.e. the server’s

4835 powerSequences information as described in section 4.3.24.4.1). The message SHALL be composed

4836 as follows:

- 4837 1. The classifier SHALL be set to “write”.
- 4838 2. The destination of the message SHALL be the power sequences server’s instance of the
- 4839 complex function “smartEnergyManagementPsData”.
- 4840 3. The message SHALL be modelled for “restricted function exchange” (this also means the
- 4841 “payload” element “cmd.function” SHALL be set properly; in this case it SHALL be
- 4842 “smartEnergyManagementPsData”).
- 4843 4. The “cmdControl” SHALL contain “partial”.
- 4844 5. A (curtailed) function “smartEnergyManagementPsData” SHALL be given, containing the data
- 4845 to modify. The content is defined by Table 118.

Element name	M/O/C (1.4.7)	Explanation
alternatives	M (1..1)	The following restriction to the list size applies: Exactly one occurrence of “alternatives” is permitted.
alternatives. powerSequence	M (1..1)	The following restriction to the list size applies: Exactly one occurrence of “powerSequence” is permitted within its parent element.
alternatives. powerSequence. description	M	
alternatives. powerSequence. description. sequenceId	M	States the sequenceId for this state change.
alternatives. powerSequence. state	M	
alternatives. powerSequence. state. state	M	This element SHALL always be present and denotes the state change. See section 4.3.24.4.5.1.2 for values permitted in this state change.

4846 Table 118: Description of “restricted write” operation for powerSequences state change.

4847

4848 *4.3.24.4.5.2 When received*

4849 If a power sequences server receives a powerSequences state change from its bound power
 4850 sequences client, it SHALL verify whether this state change is allowed within the rules denoted in
 4851 4.3.24.4.5.1.2. If it is not allowed, the power sequences server SHALL consider this as erroneous
 4852 receipt.

4853

4854 *4.3.24.4.5.3 When to send*

4855 A power sequences server SHALL never generate this message.

4856

4857 *4.3.24.4.5.4 Error considerations*

4858 See section 4.3.24.4.5.2.

4859

4860 *4.3.24.4.6 powerSequences price information*

4861 A power sequences server MAY provide information on the cost of one or more of its power
 4862 sequences. The support of this information is optional.

4863 The complex function “smartEnergyManagementPsPriceData” of the complex class

4864 “SmartEnergyManagementPs” is used with the following restrictions and rules:

Element name	M/O/C (1.4.7)	Explanation
price	O	SHALL ONLY be present if proper child elements are present.
price. sequenceId	M	
price. price	M	
price. currency	M	

4865 Table 119: Description of smartEnergyManagementPsPriceData for powerSequences price information.

4866

4867 *4.3.24.4.6.1 When received*

4868 If a power sequences server receives this message with a “notify” or “reply” classifier, it SHALL
 4869 consider this as erroneous receipt. For “write” classifier see section 4.3.24.4.8.

4870

4871 *4.3.24.4.6.2 When to send*

4872 A power sequences server SHALL produce this message with a “notify” classifier and send it to
 4873 subscribed nodes if the price information for one of its power sequences on this server instance
 4874 changes.

4875 A power sequences server SHALL produce a proper message with a “reply” classifier upon a received
 4876 “read” request according to the rules described in section 4.3.24.4.7.2. This reply may then be a
 4877 complete or restricted function exchange.

4878

4879 *4.3.24.4.6.3 Error considerations*

4880 See section 4.3.24.4.7.2.

4881

4882 *4.3.24.4.7 powerSequences price information request*

4883 *4.3.24.4.7.1 General data model description and rules*

4884 A power sequences price information request is a request for the power sequences price information
 4885 of a power sequences server.

4886 A power sequences server SHALL answer to power sequences price information requests if the
 4887 power sequences server support power sequences price information.

4888 The request to get power sequences price information is sent with “read” classifier and an “empty”
 4889 complex class function “smartEnergyManagementPsPriceData”.

4890 Optionally, the request MAY be modelled for “restricted function exchange” using a dedicated
 4891 selectors. This version of the specification RECOMMENDS to support the request for price
 4892 information of a specific power sequence as shown in Table 120.

Element name	M/O/C (1.4.7)	Explanation
smartEnergyManagementPsPriceDataSelectors	O	SHALL ONLY be present if a child element is required. Otherwise it SHALL be absent.
smartEnergyManagementPsPriceDataSelectors. price	M	SHALL ONLY be present if the following child element is required. Otherwise it SHALL be absent.
smartEnergyManagementPsPriceDataSelectors. price. sequenceId	M	SHALL be present and set to the sequenceId of the power sequence whose price information is requested.

4893 *Table 120: Dedicated “selectors” of “smartEnergyManagementPsPriceData” to request a specific powerSequences price*
 4894 *information.*

4895

4896 *4.3.24.4.7.2 When received*

4897 If a power sequences server receives a power sequences price information request with a “read”
 4898 classifier, it SHALL answer with a proper “power sequence price information”.

4899 If the request is modelled for “restricted function exchange” as shown in Table 120, a reply with
 4900 “smartEnergyManagementPsPriceData” (as described in section 4.3.24.4.6) is modified as follows:
 4901 The response SHALL only contain price information of the power sequence with the same sequenceId
 4902 value of the “selectors” content.

4903

4904 *4.3.24.4.7.3 When to send*

4905 A power sequences server SHALL never produce this message.

4906

4907 *4.3.24.4.7.4 Error considerations*

4908 If the power sequences server gets a power sequence request with a sequenceId where no price
4909 information is available, the power sequences server SHALL consider this as erroneous receipt.

4910

4911 *4.3.24.4.8 powerSequences price information change*4912 *4.3.24.4.8.1 General data model description and rules*4913 *4.3.24.4.8.1.1 General aspects*

4914 A power sequences price information change is usually only initiated upon a power sequences price
4915 calculation request call. I.e. once a powerSequences server asked a bound powerSequences client for
4916 a price calculation (see section 4.3.24.4.9), the server awaits receiving a powerSequences price
4917 information change from the client. This information should contain the price for the requested
4918 sequence. Of course, the powerSequences client is then obliged to consider the server's information
4919 on the current sequence schedule (smartEnergyManagementPsData).

4920 **Further rules apply:**

4921 A power sequences price information change can only be considered to reflect a whole sequence's
4922 price if the proper power sequence information is complete. Once a sequence started, a power
4923 sequences server may remove past slots from the power sequences information, resulting in
4924 incomplete sequence information. As a consequence, the request for the price calculation and the
4925 execution and delivery of the price calculation SHOULD occur before the power sequence starts as
4926 otherwise a bound power sequences client may not have sufficient information to calculate a price
4927 for the whole sequence.

4928 Once the price for a whole sequence was calculated and transmitted successfully, it SHOULD also be
4929 possible to request and execute a price calculation after the sequence started. As mentioned above,
4930 this is only feasible if the power sequences server keeps past slots in its sequence information.

4931

4932 *4.3.24.4.8.1.2 Data model description*

4933 The "powerSequences price information change" message is modelled with a specific "write"
4934 operation to the power sequences server's instance of "smartEnergyManagementPsPriceData" (i.e.
4935 the server's powerSequences price information as described in section 4.3.24.4.6). The message
4936 SHALL be composed as follows:

- 4937 1. The classifier SHALL be set to "write".
4938 2. The destination of the message SHALL be the power sequences server's instance of the
4939 complex function "smartEnergyManagementPsPriceData".

3. The message SHALL be modelled for “restricted function exchange” (this also means the “payload” element “cmd.function” SHALL be set properly; in this case it SHALL be “smartEnergyManagementPsPriceData”).
4. The “cmdControl” SHALL contain “partial”.
5. A (curtailed) function “smartEnergyManagementPsPriceData” SHALL be given, containing only the data to modify. The content is defined by Table 121.

Element name	M/O/C (1.4.7)	Explanation
price	M (1..1)	The following restriction to the list size applies: Exactly one occurrence of “price” is permitted.
price. sequenceId	M	The sequenceId whose price shall be modified.
price. price	M	The new price.
price. currency	M	The currency of the price.

Table 121: Description of “restricted write” operation for a power sequence price modification.

4.3.24.4.8.2 When received

If a power sequences server receives a power sequences price information change from a device that is NOT its bound power sequences client, it MAY consider this as erroneous receipt.

A power sequences server SHOULD ONLY accept a power sequences price information change if it was received from its bound power sequences client. Afterwards, further rules apply:

If a power sequences server receives a power sequences price information change as valid message upon its previous power sequence price calculation request, it SHALL evaluate the message accordingly. Afterwards, it SHOULD NOT accept subsequently received power sequences price information changes. Such “additional responses” may denote a price update information. However, instead of evaluating these “additional responses” the power sequences server SHOULD consider this as erroneous receipt, and it MAY initiate a new power sequences price calculation request.

If a power sequences server receives a power sequences price information change without having issued a power sequences price calculation request call before, it SHOULD consider this as erroneous receipt.

If a power sequences server receives a power sequences price information change containing a power sequenceId it has not asked for, it SHALL consider this as erroneous receipt.

Remark (informative): If a power sequences server receives a valid power sequences price information change, the use of this information is manufacturer specific. Examples of use are displaying it to a customer on a device display.

4968 *4.3.24.4.8.3 When to send*

4969 A power sequences server SHALL never generate this message.

4970

4971 *4.3.24.4.8.4 Error considerations*

4972 See section 4.3.24.4.8.2.

4973

4974 *4.3.24.4.9 powerSequences price calculation request call*4975 *4.3.24.4.9.1 General data model description and rules*

4976 A power sequences price calculation request call is a request from a power sequences server to a
 4977 bound power sequences client, in order to ask the power sequences client to calculate a price for a
 4978 certain sequence ID of the power sequences server. The answer is delivered by the power sequences
 4979 client using the “power sequences price calculation response call” message. The sequence needs to
 4980 have schedule information (see 4.3.24.3.3.3) and all slots need to be determined to enable the client
 4981 to calculate a price for it. Furthermore, the request SHOULD be sent “in time” in order to receive the
 4982 price calculation before the sequence starts. See section 4.3.24.4.8.1 for details!

4983 The server’s message uses the complex function

4984 “smartEnergyManagementPsPriceCalculationRequestCall” of the complex class

4985 “SmartEnergyManagementPs” with the following restrictions and rules:

Element name	M/O/C (1.4.7)	Explanation
priceCalculationRequest	M	
priceCalculationRequest. sequenceId	M	

4986 *Table 122: Description of complex class function “smartEnergyManagementPsPriceCalculationRequestCall” for*
 4987 *powerSequences price calculation request call*

4988

4989 *4.3.24.4.9.2 When received*

4990 If a power sequences server receives this message, it SHALL consider this as erroneous receipt.

4991

4992 *4.3.24.4.9.3 When to send*

4993 A power sequences server SHALL produce this message with a “call” classifier if it is interested in the
 4994 price for a configured sequence it offers, i.e. a sequence which has a valid schedule (including
 4995 information about all slot’s durations as stated in 4.3.24.4.9.1). It will get a calculation response
 4996 based on its offered sequence and the according schedule, provided that the bound client provides
 4997 this service and has sufficient information. Please consider also rules described in section
 4998 4.3.24.4.8.1!

4999 Note: A power sequences server SHALL never change the configuration for a power sequence that
 5000 has already been configured by its bound power sequences client, just to calculate different prices.

5001

5002 *4.3.24.4.9.4 Error considerations*

5003 See section 4.3.24.4.9.2.

5004 Please note that calculating prices is an optional feature.

5005 Additionally, if a device sends a power sequences price calculation request call to its bound power
5006 sequences client and the binding partner returns a negative acknowledgement message, this might
5007 happen due to the fact that the power sequences client temporarily cannot calculate prices, e.g.
5008 because the client has not yet been updated with valid prices for the whole period of the sequence.
5009 Thus, a power sequences server SHOULD be aware that it might encounter power sequences clients
5010 which currently cannot calculate prices. For this case, the server SHALL be able to cope with errors
5011 returned to price calculation requests it issues. How exactly it handles these errors is up to the
5012 manufacturer.

5013

5014 **4.3.25 SupplyCondition**

5015 ***4.3.25.1 Dependencies to other Feature Types***

5016 None.

5017

5018 ***4.3.25.2 Introduction***

5019 Practice shows that the supply of electricity, water, etc. can be temporarily impaired. Some possible
5020 causes are announced maintenance work at the supply line, supply network overload, or depletion of
5021 a local energy resource (battery, e.g.). The Supply Condition concept permits indication of the supply
5022 condition by a household's SPINE data models.

5023 The concept is less intended for any automatisms. It is rather intended for indications to users in
5024 order to prepare or apply measures in time.

5025

5026 ***4.3.25.3 Rules***

5027 None.

5028

5029 ***4.3.25.4 supplyConditionListData***

5030 ***4.3.25.4.1 Description***

5031 This function can be used to model the "events", i.e. the information on a specific supply condition.
5032 These events can, e.g., be sent from a DSO to a household. In this case, the household would be the
5033 data owner and the DSO would send (as a client) write commands to this resource.

5034

5035 ***4.3.25.4.2 Rules***

5036 None.

5037

5038 4.3.25.4.3 *Element rules*

Element	Rule	Description
conditionId	SHALL be set as PRIMARY IDENTIFIER.	Identifier of the condition.
timestamp	SHALL be set as SUB IDENTIFIER to the timestamp of the creation of the message.	The timestamp of the event.
eventType	SHALL be set. Value rules of Table 124 SHALL be applied.	The kind of the event.
originator	SHOULD be set. Value rules of Table 125 SHALL be applied.	The originator (in terms of instance, not in terms of address) of the information.
thresholdId	MAY be set as FOREIGN IDENTIFIER to the according thresholdId, if related to a threshold.	The identifier of this event's threshold (if available).
thresholdPercentage	Reached percentage of the threshold value. Only if <i>thresholdId</i> is set, <i>thresholdPercentage</i> MAY be set, too.	Relation of current situation to (configured) threshold (if available).
relevantPeriod	None.	Esp. the announcement of maintenance can make use of this element in order to tell when a maintenance is planned.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the event.
gridCondition	None.	Indicates the condition of the electricity grid.

5039 Table 123: *supplyConditionListData* Element rules5040 Element "**eventType**" value rules:

Value	Rule	Description
thresholdExceeded	None.	The related threshold was exceeded (may be good or bad, see definition of the corresponding <i>thresholdType</i> element in section 5.3.26.4.2 for details). This means that the actual value is greater than the threshold.
fallenBelowThreshold	None.	The related threshold was fallen below (may be good or bad, see definition of the corresponding <i>thresholdType</i> element in section 5.3.26.4.2 for details). This means that the actual value is lower than the threshold.
supplyInterrupt	None.	The supply of the corresponding commodity was interrupted.
releaseOfLimitations	None.	"Back to normal", no limitations or threshold violations.

otherProblem	None.	Some problem with the supply without further details about the problem.
--------------	-------	---

5041 Table 124: Element "eventType" value rules

5042 Element "originator" value rules:

Value	Rule	Description
externDSO	None.	The originator of the current condition "event" is the DSO (distribution system operator).
externSupplier	None.	The originator of the current condition "event" is the energy supplier.
internalLimit	None.	Some internal limit at the premises generated the current condition "event".
internalService	None.	Due to an internal service at the premises, the current condition "event" was generated.
internalUser	None.	A user interaction is the source of the current condition "event".

5043 Table 125: Element "originator" value rules

5044 Element "gridCondition" value rules:

Value	Rule	Description
consumptionRed	None.	The electricity grid condition for consumption from the grid is "red".
consumptionYellow	None.	The electricity grid condition for consumption from the grid is "yellow".
good	None.	The electricity grid condition is "good".
productionYellow	None.	The electricity grid condition for feeding into the grid is "yellow".
productionRed	None.	The electricity grid condition for feeding into the grid is "red".

5045 Table 126: Element "gridCondition" value rules

5046

5047 **4.3.25.5 supplyConditionDescriptionListData**5048 **4.3.25.5.1 Description**

5049 The more static information about a supply condition is modelled within this function.

5050

5051 **4.3.25.5.2 Rules**

5052 None.

5053

5054 **4.3.25.5.3 Element rules**

Element	Rule	Description
conditionId	SHALL be set as PRIMARY IDENTIFIER.	Identifier of the condition.
commodityType	SHOULD be set.	The type of the commodity handled by this condition is stated here.
positiveEnergyDirection	SHOULD be set. The value "consume" expresses that positive values of energy or power denote	The <i>positiveEnergyDirection</i> states whether energy consumption or production will be counted as positive value.

	energy consumption by the premises. The value "produce" expresses that positive values of energy or power denote energy production by the premises.	
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the condition.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the condition.

Table 127: supplyConditionDescriptionListData Element rules

4.3.25.6 supplyConditionThresholdRelationListData**4.3.25.6.1 Description**

Multiple thresholds can be associated to a condition. This leads to the relation

- 1 conditionId -> 0..m thresholdId

The supplyConditionThresholdRelationListData function can be used to model concrete relations.

4.3.25.6.2 Rules

None.

4.3.25.6.3 Element rules

Element	Rule	Description
conditionId	SHALL be set as PRIMARY IDENTIFIER.	Identifier of the condition.
thresholdId (list)	Multiple thresholdIds of thresholds linked to this condition MAY be provided.	Related threshold identifier(s).

Table 128: supplyConditionThresholdRelationListData Element rules

4.3.25.7 Descriptive information and further definitions

None.

4.3.26 TariffInformation

Reserved for future use.

4.3.27 TaskManagement**4.3.27.1 Dependencies to other Feature Types**

- DirectControl

- 5078 - HVAC
- 5079 - LoadControl
- 5080 - PowerSequences
- 5081 - SmartEnergyManagementPs

5082

5083 **4.3.27.2 Introduction**

5084 The TaskManagement Class lists all "tasks" that are currently relevant on an entity. This enables one
 5085 to have a quick look on the tasks of related classes, but not for their modification. Further
 5086 information has to be gathered with the corresponding classes supporting the specific functionality.

5087 In this version of the specification, tasks can be "jobs" like in DirectControl, PowerSequences /
 5088 SmartEnergyManagementPs, etc. (see the related function groups below).

5089 Remark: Future versions of this specification may add other kinds of tasks (i.e. tasks that are not
 5090 covered by "jobs").

5091

5092 **4.3.27.3 Rules**

5093 None.

5094

5095 **4.3.27.4 taskManagementJobListData**

5096 **4.3.27.4.1 Description**

5097 All jobs that are available on the entity are listed here, together with their state and (if available) the
 5098 elapsed and remaining time.

5099

5100 **4.3.27.4.2 Rules**

5101 None.

5102

5103 **4.3.27.4.3 Element rules**

Element	Rule	Description
jobId	SHALL be set as PRIMARY IDENTIFIER.	The <i>jobId</i> is used for indentifying different job information collected by this class.
timestamp	SHOULD be set to the timestamp of the creation of the message.	Point in time, this information was generated.
jobState	SHALL be set. Value rules of Table 50 SHALL be applied. Value rules of Table 70 SHALL be applied. Value rules of section 3.10.1.5.1 SHALL be applied.	Current state of the job.

elapsedTime	If set, this element SHALL contain the duration the job was already active.	The duration, this job was already active (normally, pause times are not counted).
remainingTime	If set, this element SHALL contain the duration the job still needs to be active.	The duration, this job will most likely be active before it ends.

Table 129: taskManagementJobListData Element rules

4.3.27.5 taskManagementJobRelationListData

4.3.27.5.1 Description

To identify where a job originates from, the *taskManagementJobRelationListData* function is used. If the related class allows more than one job the corresponding identifier(s) are denoted.

4.3.27.5.2 Rules

If this function is used, each list entry SHALL contain exactly one relation to a job. This means that it is not allowed to set more than one of the following Elements within one list entry:

- directControlRelated
- hvacRelated
- loadControlRelated
- powerSequencesRelated
- smartEnergyManagementPsRelated

4.3.27.5.3 Element rules

Element	Rule	Description
jobId	SHALL be set as PRIMARY IDENTIFIER.	The <i>jobId</i> is used for indentifying different job information collected by this class.
directControlRelated	None.	Flag that indicates that the job is related to DirectControl.
hvacRelated	None.	Flag that indicates that the job is related to HVAC.
hvacRelated. overrunId	If the job is related to an overrun, this element SHALL be set to the FOREIGN IDENTIFIER of the corresponding overrun.	Identifier for the related overrun.
loadControlRelated	None.	Flag that indicates that the job is related to LoadControl.
loadControlRelated. eventId	If the job is related to an event, this element SHALL be set to the FOREIGN IDENTIFIER of the corresponding event.	Identifier for the related load control event.
powerSequencesRelated	None.	Flag that indicates that the job is related to PowerSequences.
powerSequencesRelated. sequenceId	If the job is related to a power sequence, this element SHALL be	Identifier for the related power sequence.

	set to the FOREIGN IDENTIFIER of the corresponding power sequence.	
smartEnergyManagementPs Related	None.	Flag that indicates that the job is related to SmartEnergyManagementPs.
smartEnergyManagementPs Related. sequenceId	If the job is related to a SmartEnergyManagementPs sequence, this element SHALL be set to the FOREIGN IDENTIFIER of the corresponding sequence.	Identifier for the related power sequence.

Table 130: taskManagementJobRelationListData Element rules

4.3.27.6 taskManagementJobDescriptionListData

4.3.27.6.1 Description

The more static information about the jobs are communicated with this function.

4.3.27.6.2 Rules

None.

4.3.27.6.3 Element rules

Element	Rule	Description
jobId	SHALL be set as PRIMARY IDENTIFIER.	The <i>jobId</i> is used for indentifying different job information collected by this class.
jobSource	SHOULD be set. Value rules of Table 132 SHALL be applied.	Specifies what kind of reason caused this job.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the job.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the job.

Table 131: taskManagementJobDescriptionListData Element rules

Element "**jobSource**" value rules:

Value	Rule	Description
internalMechanism	None.	The job was started due to some device internal mechanism (e.g. a regular cleaning cycle).
userInteraction	None.	The user activated some functionality directly via the UI of the device.
externalConfiguration	None.	The job started, because it was triggered from some external device (e.g. an energy manager).

Table 132: Element "jobSource" value rules

4.3.27.7 taskManagementOverviewData**4.3.27.7.1 Description**

A general overview is given with this function. It specifies whether a remote controlling is permitted at that moment and if currently jobs are active.

4.3.27.7.2 Rules

None.

4.3.27.7.3 Element rules

Element	Rule	Description
remoteControllable	SHOULD be set. If set to "false", write operations on features linked via <i>taskManagementJobRelationListData</i> to this feature SHALL be declined. If absent, a default value of "true" SHALL be applied.	States if remote controlling of the device or entity is permitted.
jobsActive	None.	Denotes whether jobs are active on the device or entity or not.

Table 133: taskManagementOverviewData Element rules

4.3.27.8 Descriptive information and further definitions

The TaskManagement Feature Type enables the brief listing of all tasks that are currently relevant for the feature's entity. A special case is the implementation of the Feature Type *TaskManagement* on an entity with the entity type *DeviceInformation*: In this case the feature covers tasks of all entities that are available on this device.

4.3.28 Threshold**4.3.28.1 Dependencies to other Feature Types**

None.

4.3.28.2 Introduction

Thresholds are helpful in several kind of use cases. E.g. a device or application can be configured to do some specific behaviour if a measured value is above or below a configured threshold. Some measurands need to stay in a certain value range, to ensure proper and safe working of devices. Automatic alarms after reaching a threshold help reacting on the occurrence immediately.

4.3.28.3 Rules

None.

5164

5165 **4.3.28.4 thresholdListData**5166 **4.3.28.4.1 Description**

5167 Each threshold (identified by its *thresholdId*) has a configured value. This function is used to read or
 5168 write the actual value.

5169

5170 **4.3.28.4.2 Rules**

5171 None.

5172

5173 **4.3.28.4.3 Element rules**

Element	Rule	Description
thresholdId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for the threshold.
thresholdValue	SHALL be set.	Value that should not be fallen below or exceeded (depends on the <i>thresholdType</i> , see section 5.3.26.4.2).

5174 *Table 134: thresholdListData Element rules*

5175

5176 **4.3.28.5 thresholdConstraintsListData**5177 **4.3.28.5.1 Description**

5178 The value range from which a threshold value configuration may be chosen from may be restricted
 5179 by a minimum and a maximum value. Between these, not every value, but only ones with a specific
 5180 gap, may be used, specified with the step size.

5181

5182 **4.3.28.5.2 Rules**

5183 If thresholds are writeable, this function SHOULD be supported, otherwise it MAY be supported.

5184

5185 **4.3.28.5.3 Element rules**

Element	Rule	Description
thresholdId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for the threshold.
thresholdRangeMin	If set, the threshold value SHALL NOT be set to a value below this minimum value.	Minimum value that may be used to configure the element <i>thresholdValue</i> (see section 5.3.26.2.2).
thresholdRangeMax	If set, the threshold value SHALL NOT be set to a value above this maximum value.	Maximum value that may be used to configure the element <i>thresholdValue</i> (see section 5.3.26.2.2).
thresholdStepSize	The threshold value SHALL ONLY contain multiples of thresholdStepSize.	Step size for the configuration of the element <i>thresholdValue</i> (see section 5.3.26.2.2).

5186 *Table 135: thresholdConstraintsListData Element rules*

5187

5188 **4.3.28.6 thresholdDescriptionListData**

5189 *4.3.28.6.1 Description*

5190 The rather static information about thresholds is defined within this function.

5191

5192 *4.3.28.6.2 Rules*

5193 None.

5194

5195 *4.3.28.6.3 Element rules*

Element	Rule	Description
thresholdId	SHALL be set as PRIMARY IDENTIFIER.	Identifier for the threshold.
thresholdType	SHALL be set.	The type of the threshold.
unit	SHALL be set, if there could be ambiguity how to interpret the threshold without the unit.	The unit in which the value of the threshold is denoted.
scopeType	None.	Specifies a more detailed meaning of the threshold.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the threshold.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the threshold.

5196 *Table 136: thresholdDescriptionListData Element rules*

5197 Element "**thresholdType**" value rules:

Value	Rule	Description
goodAbove	None.	A positive reaction will follow after the value is greater than the threshold.
badAbove	None.	A negative reaction will follow after the value is greater than the threshold.
goodBelow	None.	A positive reaction will follow after the value is lower than or equal to the threshold.
badBelow	None.	A negative reaction will follow after the value is lower than or equal to the threshold.
minValueThreshold	None.	If the value is lower than this minimum threshold, some warning will be activated.
maxValueThreshold	None.	If the value is greater than this maximum threshold, some warning will be activated.
minValueThresholdExtreme	None.	If the value is lower than this "extreme minimum" threshold, some important warning will be activated.
maxValueThresholdExtreme	None.	If the value is greater than this "extreme maximum" threshold, some important warning will be activated.
sagThreshold	None.	If this sag threshold is underrun, some warning will be activated. Commonly used for electricity voltage sags.

swellThreshold	None.	If this swell threshold is exceeded, some warning will be activated. Commonly used for electricity voltage swells.
----------------	-------	--

Table 137: Element "thresholdType" value rules

Note: All threshold rules SHALL be understood in a mathematical way.

4.3.28.7 Descriptive information and further definitions

None.

4.3.29 TimeInformation

4.3.29.1 Dependencies to other Feature Types

None.

4.3.29.2 Introduction

The exchange of time information is important for most smart grid and home automation systems.

Please note the underlying revision of the SPINE data model does not specify a sophisticated synchronisation mechanism that includes a given precision. This is due to the complex of problems appearing with communication delays in the different technologies and hardware platforms.

However, an application could measure all relevant delays and take care of a real synchronisation in the network using the *TimeInformation* class.

Independent of any synchronisation aspect the "owner concept" applies here as well: The "owner" of any given data (a measured value of a sensor, e.g.) applies its own SPINE time to its own data. This is relevant for the case of a substantial difference between the SPINE time of two nodes.

In order to avoid problems from such differences a typical sequence could be that a new device sends a *timeDistributorEnquiryCall* (with the *isTimeDistributor* element set to true) to all devices in the SPINE network. All time distributors would send a *timeDistributorData* instance as notification to the device. The new device would pick the time distributor with the highest priority and send a read command to the *timeInformationData* function of this distributor. The time distributor would reply with a *timeInformationData* containing its current time.

4.3.29.3 Rules

None.

4.3.29.4 timeInformationData

4.3.29.4.1 Description

The *timeInformationData* command is used for exchanging the SPINE time.

5232 4.3.29.4.2 Rules

5233 None.

5234

5235 4.3.29.4.3 Element rules

Element	Rule	Description
utc	SHALL be set if <i>isTimeDistributor</i> is set "true". Otherwise it SHOULD be set.	The current time in UTC.
utcOffset	None.	The configured offset to UTC (e.g. PT2H [= +2 hours] for Central European Summer Time (CEST)).
dayOfWeek	None.	To model the actual day of week (Monday to Sunday) this element is used. It refers to the local time. If there is no utcOffset given, it is assumed, that the local time is equivalent to UTC-time.
calendarWeek	None.	A value between 1 and 53 stating the current calendar week number. This element refers to the local time. If there is no utcOffset given, it is assumed, that the local time is equivalent to UTC-time.

5236 Table 138: *timeInformationData* Element rules

5237

5238 4.3.29.5 *timeDistributorData*

5239 4.3.29.5.1 Description

5240 One or some devices in a SPINE network should act as time distributor. The information thereof and
 5241 the dedicated priority can be modelled with the *timeDistributorData* command.

5242

5243 4.3.29.5.2 Rules

5244 None.

5245

5246 4.3.29.5.3 Element rules

Element	Rule	Description
isTimeDistributor	If set to "true" the device SHALL be able to act as time distributor. Otherwise it SHALL be omitted or set to "false".	Denotes whether a device can act as a time distributor in the SPINE network or not.
distributorPriority	If <i>isTimeDistributor</i> is set to "false", this element SHOULD be absent. The time distributor with the highest priority SHOULD be used for time synchronization. There MAY be further time distributors with lower priority to have redundant time distributors that can be used as soon	A (configured) priority of the time distributor.

	as the time distributor with the higher priority can currently not be used. Only values greater than zero SHALL be used. A value of "1" SHALL be interpreted as highest priority. The priority decreases with increasing value of distributorPriority.	
--	--	--

Table 139: timeDistributorData Element rules

4.3.29.6 timePrecisionData**4.3.29.6.1 Description**

Information regarding the precision of the time of a device can be modelled using *timePrecisionData*.

4.3.29.6.2 Rules

The element "isSynchronised" SHALL be used with regards to the time distributor with the highest priority. Time should not differ more than 10 seconds from the time retrieved from a commonly used NTP server in the internet (e.g. ptbtime1.ptb.de). If a device has a *clockDrift* element set to 1, it should at least get a sync every 10 days. After that time, it can not be sure to have a correct time information and therefore sets "isSynchronised" to false.

4.3.29.6.3 Element rules

Element	Rule	Description
isSynchronised	SHOULD be set. If omitted, isSynchronised SHALL be interpreted as "false".	This is a Boolean value, stating whether the device's SPINE time information is synchronized with the SPINE network or not, meaning that it could receive the SPINE time from a time distributor and has still a "sufficiently precise time".
lastSyncAt	MAY be used to communicate when the last synchronization occurred.	This is the timestamp of the last synchronisation with a time distributor, or, in case of a time distributor itself, the timestamp of the last official date and time information (e.g. from an NTP server).
clockDrift	MAY be used to communicate the clock drift.	The drift of the internal clock (in seconds per day) is indicated by this element.

Table 140: timePrecisionData Element rules

4.3.29.7 timeDistributorEnquiryCall**4.3.29.7.1 Description**

This function can be used for finding all time distributors in the network. Therefore, it can be sent to all devices in the network that have their *isTimeDistributor* element in the function *timeDistributor* (see section 5.3.27.3) set to true.

5268

5269 *4.3.29.7.2 Rules*

5270 Devices that do not support the function *timeDistributor* SHOULD NOT be used as destination of this
 5271 call.

5272

5273 *4.3.29.7.3 Element rules*

5274 None.

5275

5276 **4.3.29.8 Descriptive information and further definitions**

5277 None.

5278

5279 **4.3.30 TimeSeries**5280 **4.3.30.1 Dependencies to other Feature Types**

5281 None.

5282

5283 **4.3.30.2 Introduction**

5284 TimeSeries is a versatile class that allows to express any kind of time dependent values as a time
 5285 series of values. It can be used to express electricity related values like power constraints or power
 5286 consumption over time, but also other values that are not electricity related like temperature or
 5287 humidity over time.

5288

5289 **4.3.30.3 Rules**

5290 None.

5291

5292 **4.3.30.4 timeSeriesListData**5293 *4.3.30.4.1 Description*

5294 The timeSeriesListData function is used to describe the value of a timeSeries over time.

5295

5296 *4.3.30.4.2 Rules*

5297 None.

5298

5299 *4.3.30.4.3 Element rules*

Element	Rule	Description
---------	------	-------------

timeSeriesId	SHALL be set as PRIMARY IDENTIFIER.	Allows the identification of a timeSeries. Allows linking of the different functions to the same timeSeries. Allows linking of the timeSeries within other Features that are placed in the same entity.
timePeriod	If set, neither timeSeriesSlot.duration nor timeSeriesSlot.timePeriod SHALL define times out of the bound of the overall timeSeries timePeriod. Only relative times SHALL be used (xs:duration). Relative times SHALL be interpreted relative to "now".	Time period of the complete timeSeries.
timeSeriesSlot (list)	A timeSeries SHOULD have at least one timeSeriesSlot.	Allows to define slots of a timeSeries.
timeSeriesSlot. timeSeriesSlotId	SHALL be set as SUB IDENTIFIER, if timeSeriesData contains more than one timeSeriesSlot entry. In case of more than one timeSeriesSlot entry, timeSeriesSlotId SHALL increase matching with the chronological order of the timeSeriesSlots. If timeSeriesSlotId is omitted it SHALL be interpreted as zero.	Allows identification of a timeSeriesSlot within a timeSeries.
timeSeriesSlot. .timePeriod	timeSeriesSlot .timePeriod .startTime SHALL be used in slots that have a time gap to the previous slot. timeSeriesSlot .timePeriod .startTime SHALL be used in the first slot if no startTime is provided in the timePeriod element with the overall time period of the timeSeries. Only relative times SHALL be used (xs:duration). Relative times SHALL be interpreted relative to "now".	Allows to define a timePeriod of a timeSeriesSlot and to model time gaps inbetween slots.
timeSeriesSlot. duration	SHALL be set. SHALL only contain values greater than zero seconds. If timeSeriesSlot. timePeriod. startTime is set, the duration starts from there. If timeSeriesSlot. timePeriod. startTime is not set in the first timeSeriesSlot (slot with the lowest timeSeriesSlotId), the timePeriod. startTime of the overall timeSeries SHALL be set and the duration	Allows to define a duration of a timeSeriesSlot.

	<p>of the first timeSeriesSlot starts from there.</p> <p>If timeSeriesSlot.timePeriod.startTime is not set in a slot that is not the first slot, the duration starts after the previous slot is finished.</p>	
timeSeriesSlot.recurrenceInformation	Only relative times SHALL be used (xs:duration) within recurrenceInformation. Relative times SHALL be interpreted relative to “now”.	If the time series slot is a recurring one, the corresponding information can be found within this element and its sub-elements.
timeSeriesSlot.value	A positive value SHALL relate to consumption and a negative value SHALL relate to production of the server. If minValue is set the content of value SHALL be greater. If maxValue is set the content of value SHALL be smaller.	Defines the expected value during a timeSeriesSlot.
timeSeriesSlot.minValue	A positive value SHALL relate to consumption and a negative value SHALL relate to production of the server. If value or maxValue is set the content of minValue SHALL be smaller.	Defines a lower value limit
timeSeriesSlot.maxValue	A positive value SHALL relate to consumption and a negative value SHALL relate to production of the server. If value or minValue is set the content of maxValue SHALL be greater.	Defines an upper value limit

Table 141: timeSeriesListData Element rules

4.3.30.5 timeSeriesDescriptionListData

4.3.30.5.1 Description

The timeSeriesDescriptionListData function is used to describe a timeSeries.

4.3.30.5.2 Rules

This is the first data that SHOULD initially be read of a timeSeries. If this data matches with the data that the client is expecting, the other functions of timeSeries become relevant.

4.3.30.5.3 Element rules

Element	Rule	Description
timeSeriesId	SHALL be set as PRIMARY IDENTIFIER.	Allows the identification of a timeSeries. Allows linking of the different functions to the same

		timeSeries. Allows linking of the timeSeries within other Features that are placed in the same entity.
timeSeriesType	Value rules of Table 143 SHALL be applied.	Allows to define which timeSeries type is supported.
timeSeriesWriteable	If set to "true", this timeSeriesData SHALL be writeable. If set to false or omitted, this timeSeriesData SHALL NOT be writeable.	Allows to define if the timeSeries is writeable.
updateRequired	With updateRequired the server can request an update of writeable or changeable data related to the same PRIMARY IDENTIFIER from a client. The server SHALL ensure that only one responsible client is able to update the related data. To request an update the server SHALL set updateRequired to "true". Note: In this case, the server expects the responsible client to update the writeable or changeable data related to the same PRIMARY IDENTIFIER. However, also if updateRequired is set to "false" a server SHOULD in general allow updates of the data from the responsible client. The server SHALL set the updateRequired back to "false", as soon as "timeSeriesDescriptionListData" was updated successfully (if writeable or changeable) OR the update of the other writeable or changeable data related to the same PRIMARY IDENTIFIER was successful. Note: The client does not need to stop an ongoing update process (e.g. if multiple functions are written), when updateRequired is set back to "false". The server MAY choose to withdraw the update request at any time by setting updateRequired back to "false".	Request a write to timeSeriesData from the bound client.
measurementId	MAY be set as FOREIGN IDENTIFIER to link data of other Features to the timeSeries.	Allows to link data of other Features to a timeSeries, e.g. Measurement or ElectricalConnection data
currency	If set, the currency SHALL be applied to the value in timeSeriesData	Allows to describe the currency of the value, if the value is a price.

unit	SHALL be set if the corresponding value has a unit. If set, the unit SHALL be applied to the value in timeSeriesData	Allows to describe the unit of the value, if the value has a unit.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	Allows to define user friendly name for the timeSeries.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Allows to define a user friendly description for the timeSeries.
scopeType	None.	Specifies a more detailed meaning of the time series.

Table 142: timeSeriesDescriptionListData Elements

Element "**timeSeriesType**" value rules:

Value	Rule	Description
plan	"plan" SHALL be used to model how a certain value will change over time. value as well as minValue and maxValue MAY be used.	"plan" is used to model how a certain value changes over time.
singleDemand	"singleDemand" SHALL be used to model a single demand with a simple timeSeries that only SHALL use one timeSeriesSlot. value as well as minValue and maxValue MAY be used.	"singleDemand" is used to model a single demand with a simple timeSeries (e.g. this can be used to model a simple energy demand).
constraints	"constraints" SHALL be used to model certain value constraints or limits over time. For this only min- and/or maxValue SHALL be used.	"constraints" is used to model certain value constraints over time with a min- and/or maxValue curve.

Table 143: Element "timeSeriesType" value rules

4.3.30.6 timeSeriesConstraintsListData

4.3.30.6.1 Description

Allows to define constraints for the timeSeries, e.g. what is the maximum amount of slots allowed or similar. This function can be especially valuable if timeSeriesData is writeable but the server has certain constraints. The client can then match its own constraints with the server constraints to form a valid write.

4.3.30.6.2 Rules

None.

4.3.30.6.3 Element rules

Element	Rule	Description
timeSeriesId	SHALL be set as PRIMARY IDENTIFIER.	Allows the identification of a timeSeries. Allows

		linking of the different functions to the same timeSeries. Allows linking of the timeSeries within other Features.
slotCountMin	If set and a corresponding timeSeriesData exists, it SHALL NOT have less identifiable slots.	The minimum amount of slots for a specific time series is stated in this element.
slotCountMax	If set, the corresponding timeSeriesData SHALL NOT have more slots.	The maximum amount of slots for a specific time series is stated in this element. Recurring slots only count as one.
slotDurationMin	If set, the corresponding timeSeriesData SHALL NOT have slots with shorter duration.	If the slots of this time series have a minimum duration, it is denoted here.
slotDurationMax	If set, the corresponding timeSeriesData SHALL NOT have slots with longer duration.	If the slots of this time series have a maximum duration, it is denoted here.
slotDurationStepSize	If set, the duration of a slot SHALL be a multiple of slotDurationStepSize and also slotDurationMin and slotDurationMax SHALL be multiples of slotDurationStepSize.	Slots can have a specific duration step size. This element holds the corresponding value.
earliestTimeSeriesStartTime	If set the start time of a timeSeries SHALL be later than or equal to earliestTimeSeriesStartTime.	This is the earliest start time the corresponding time series is allowed to start.
latestTimeSeriesEndTime	If set, the end time of a timeSeries SHALL be earlier than or equal to latest TimeSeriesEndTime.	This is the latest end time the corresponding times series is allowed to end.
slotValueMin	The value, maxValue or minValue of a slot SHALL be equal to or higher than slotValueMin.	The values within the slots of the corresponding time series are not allowed to be lower than this value.
slotValueMax	The value, maxValue or minValue of a slot SHALL be equal to or lower than slotValueMax.	The values within the slots of the corresponding time series are not allowed to be higher than this value.
slotValueStepSize	<p>If slotValueStepSize is set and slotValueMin as well as slotValueMax is not set, the elements value, maxValue and minValue of a slot SHALL be a multiple of slotValueStepSize.</p> <p>If slotValueStepSize is set and slotValueMin is also set, the elements value, maxValue and minValue of a slot SHALL be</p>	The minimum step size between two different values within the slots of the corresponding time series.

	<p>slotValueMin plus a multiple of slotValueStepSize.</p> <p>If slotValueStepSize is set and slotValueMax is also set, the elements value, max value and min value of a slot SHALL be slotValueMax minus a multiple of slotValueStepSize.</p>	
--	---	--

Table 144: timeSeriesConstraintsListData Element rules

4.3.31 TimeTable

4.3.31.1 Dependencies to other Feature Types

None.

4.3.31.2 Introduction

All time dependent functionality like setpoint-configuration, etc. makes use of this elementary class. It provides a list of time tables, each containing some number of time slots. Each slot has a start and end time. The times can either be absolute (*dateTime*) or recurring (*daysOfWeek* plus *time* and, if required, the recurring interval step elements). The count of slots and the start and end times can be configurable or static (device specific). Other classes that make use of time tables only reference to the *timeTableId*, not to specific time slots.

4.3.31.3 Rules

None.

4.3.31.4 timeTableListData

4.3.31.4.1 Description

The list of time tables and time slots together with the start and end times are modelled in this function.

4.3.31.4.2 Rules

None.

4.3.31.4.3 Element rules

Element	Rule	Description
timeTableId	SHALL be set as PRIMARY IDENTIFIER.	The <i>timeTableId</i> is used for indentifying specific table information in the different functions and for relations in other classes like <i>Setpoint</i> .

timeSlotId	SHALL be set as SUB IDENTIFIER.	Each slot of a time table has its own slot ID.
recurrenceInformation	None.	Information about the recurrence of a slot.
startTime	None.	Start time of one slot.
endTime	None.	End time of one slot.

Table 145: timeTableListData Element rules

4.3.31.5 timeTableConstraintsListData**4.3.31.5.1 Description**

If a time table has some constraints, they are denoted with the help of this function (e.g. the minimum and maximum count of slots (may differ between the time tables)).

4.3.31.5.2 Rules

None.

4.3.31.5.3 Element rules

Element	Rule	Description
timeTableId	SHALL be set as PRIMARY IDENTIFIER.	The <i>timeTableId</i> is used for indentifying specific table information in the different functions and for relations in other classes like <i>Setpoint</i> .
slotCountMin	If set, the time table SHALL NOT have less slots.	The minimum amount of slots for a specific time table is stated in this element. Recurring slots only count as one.
slotCountMax	If set, the time table SHALL NOT have more slots.	The maximum amount of slots for a specific time table is stated in this element. Recurring slots only count as one.
slotDurationMin	If set, the corresponding time table SHALL NOT have slots with lower duration.	If the slots of this time table have a minimum duration, it is denoted here.
slotDurationMax	If set, the corresponding time table SHALL NOT have slots with higher duration.	If the slots of this time table have a maximum duration, it is denoted here.
slotDurationStepSize	If set, the duration of a slot SHALL be a multiple of slotDurationStepSize.	Slots can have a specific duration step size. This element holds the corresponding value.
slotShiftStepSize	If set, the start time of a slot SHALL be firstSlotBeginsAt + a multiple of slotShiftStepSize.	Slots can have a specific shift step size. This element holds the corresponding value.
firstSlotBeginsAt	If set, the start time of the first slot SHALL be firstSlotBeginsAt.	Especially time tables with fixed slot durations probably need this information to specify, where in time the slots are located (i.e. fixed 15 minute slots can start either at

		00:00:00 (default) or e.g. at 00:10:00). This value only defines, when the time pattern (defined by the element <i>slotDurationStepSize</i>) begins on a day.
--	--	---

Table 146: *timeTableConstraintsListData* Element rules**4.3.31.6 timeTableDescriptionListData****4.3.31.6.1 Description**

The more static information about the time tables are described in the *timeTableDescriptionData* function.

4.3.31.6.2 Rules

None.

4.3.31.6.3 Element rules

Element	Rule	Description
timeTableId	SHALL be set as PRIMARY IDENTIFIER.	The <i>timeTableId</i> is used for indentifying specific table information in the different functions and for relations in other classes like <i>Setpoint</i> .
timeSlotCountChangeable	If set to "true", a client SHALL be allowed to add new slots to or remove slots from the timeTable. Otherwise the element SHALL be omitted or set to "false". The timings of added slots SHALL NOT overlap with existing slots.	States whether the amount of slots is changeable by a client for a specific time table or not.
timeSlotTimesChangeable	If set to "true", a client SHALL be able to change timings of existing slots. Otherwise it SHALL be omitted or set to "false".	States whether the times of the slots of a specific time table are changeable by a client or not.
timeSlotTimeMode	Value rules of Table 148 SHALL be applied.	If either <i>absolute</i> or <i>recurring</i> time slots or <i>both</i> modes are supported is stated in this element.
label	Default rules of section 3.10.1.2.1 SHALL be applied.	A short label of the time table.
description	Default rules of section 3.10.1.3.1 SHALL be applied.	Descriptive information on the time table.

Table 147: *timeTableDescriptionListData* Element rules

Element "**timeSlotTimeMode**" value rules:

Value	Rule	Description
absolute	None.	Only absolute times are allowed for the time table.
recurring	None.	Only recurring times are allowed for the time table.
both	None.	Absolute and recurring times are allowed for the time table.

Table 148: Element "timeSlotTimeMode" value rules

4.3.31.7 Descriptive information and further definitions

None.

5 Class descriptions

5.1 Introduction

In this chapter, all Classes defined by the EEBUS Initiative e.V. are described. It should be seen as equivalent to the XSD definitions. Everything that can be found there is described within the sections of this chapter. Descriptions and rules for the Functions and Elements are only described within the Feature Types as they define a more specific meaning for a Class (see section 2.1.2.3).

Please refer to section 2.2 for details on the concepts of Standard Classes and Complex Classes.

5.2 Complex Classes

5.2.1 IncentiveTable

5.2.1.1 Introduction

The IncentiveTable data model allows to define tiers with different incentives and boundaries. In the case the incentive is price based, it also could be called a price table. However, there can also be other incentives, CO₂ Emission for example. Within the boundaries of a tier the incentive of the tier apply. This allows to distribute e.g. different types of power to different power consumers. E.g. a photovoltaic system produces more power than can be consumed and feed into the electricity grid (surplus power), in this case the power is for free and can be distributed from the CEM to consumers that have the flexibility to increase their consumption. When all surplus power is consumed the next tier would be the electricity grid feed-in power, which in this example would be cheaper than the electricity grid power. In most cases the electricity grid power would be more expensive and therefore would be the last tier. In the given example we would have three tiers, 1st surplus power on top of that the 2nd electricity grid feed in power and beyond that the 3rd electricity grid power. However, the electricity grid power can also get cheaper, e.g. in case the electricity grid has an excess of power available an incentive to flexible consumers could be given.

Even though power was used in this example, the boundaries can also be defined by other limiters.

The incentiveTable has three different functions:

1. incentiveTable: holds the actual values of the boundaries and incentives
2. incentiveTableDescription: holds data that describes the values of incentiveTable
3. incentiveTableConstraint: allows a server to define certain constraints regarding the incentiveTable, this is especially important if the IncentiveTable is writeable

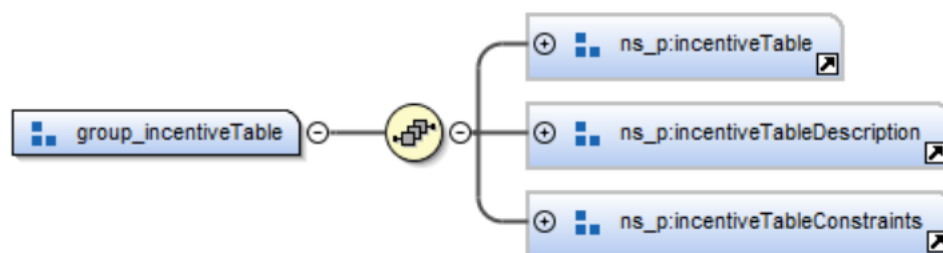


Figure 40: IncentiveTable function-group overview

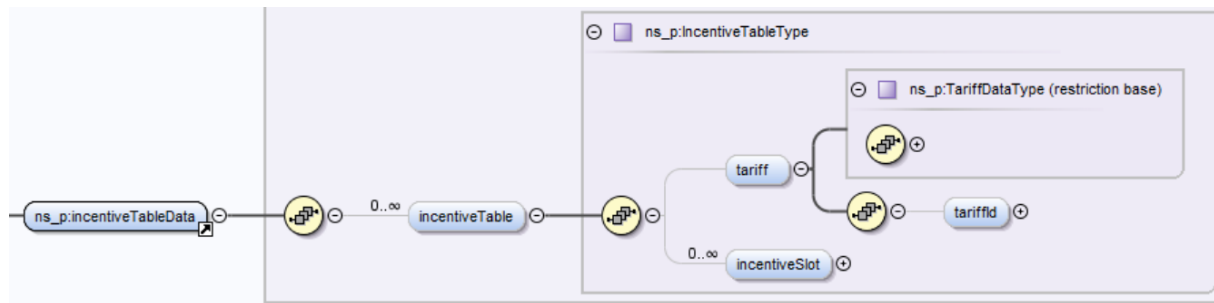
The tariffId is used to identify the data of different IncentiveTables within each function.

5415

5416 **5.2.1.2 incentiveTableData**5417 **5.2.1.2.1 General**

5418 The incentiveTable within the incentiveTableData function holds the values of the IncentiveTable.

5419 This includes the boundary values as well as the incentive values for each tier.



5420

5421 *Figure 41: incentiveTableData function overview*

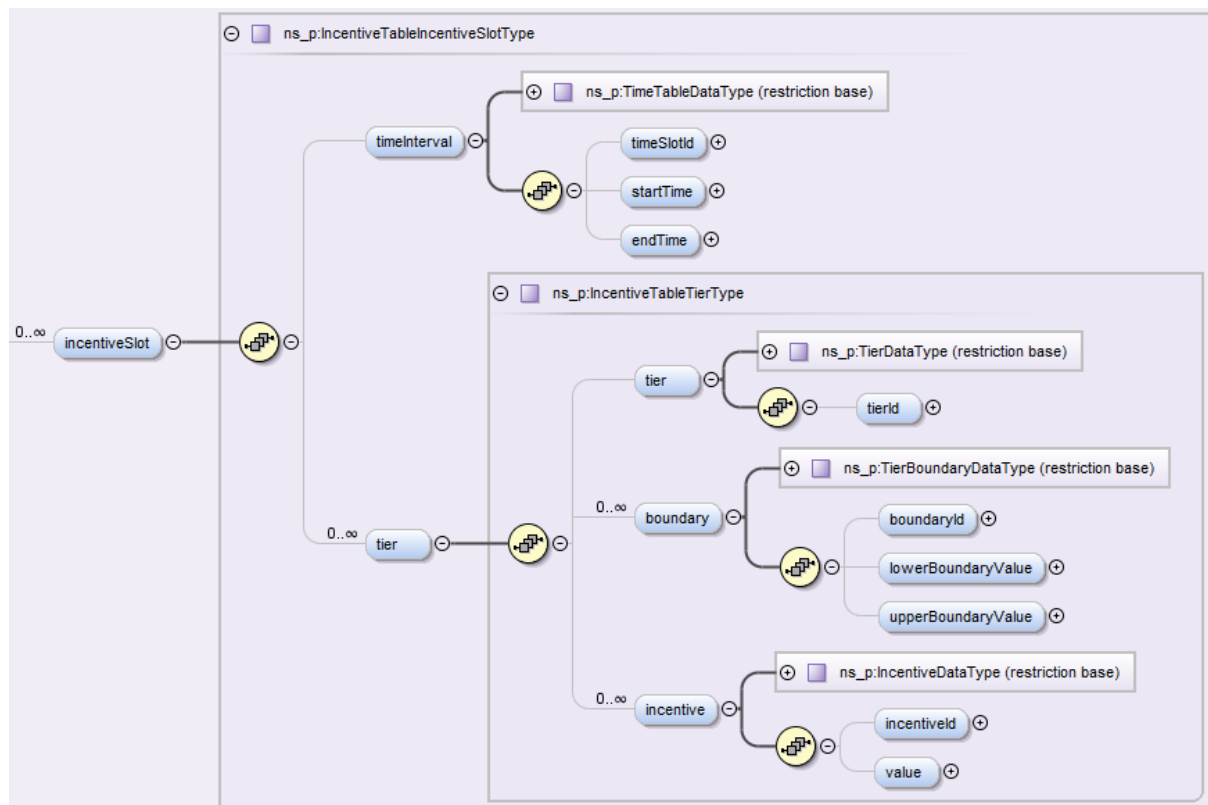
5422

5423 **5.2.1.2.2 Detailed description of elements**

Element	Type	Description
incentiveTable (list)		
incentiveTable. tariff		
incentiveTable. tariff. tariffId	Identifier "TariffIdType" (see Table 339).	Allows the identification of a incentiveTable. Allows linking of the data in the different functions of IncentiveTable. Allows linking of the incentiveTables within other features that are placed in the same entity.
incentiveTable. incentiveSlot (list)	Complex type "IncentiveTableIncentiveSlotType". See Table 150.	An incentiveTable can include different incentiveSlots to describe changes of boundaries or incentives over time. Please refer to the detailed description of <i>IncentiveTableIncentiveSlotType</i> elements in Table 150.

5424 *Table 149: incentiveTable detailed description of elements*

5425 IncentiveTableIncentiveSlotType:



5426

5427 Figure 42: IncentiveTableIncentiveSlotType overview

Element	Type	Description
timeInterval		Time period of the incentiveSlot.
timeInterval. timeSlotId	Identifier "TimeSlotIdType" (see Table 339).	Allows identification of an incentiveSlot within an incentiveTable.
timeInterval. startTime		
timeInterval. startTime. dateTime	xs:dateTime (W3C standard type)	Absolute start time.
timeInterval. startTime. relative	xs:duration (W3C standard type)	Relative start time containing a value relative to now. Positive values are in the future, negative ones are in the past and zero seconds represents "now".
timeInterval. endTime		
timeInterval. endTime. dateTime	xs:dateTime (W3C standard type)	Absolute end time.
timeInterval. endTime. relative	xs:duration (W3C standard type)	Relative end time containing a value relative to now. Positive values are in the future, negative ones are in the past and zero seconds represents "now".
tier (list)		
tier. tier		

tier. tier. tierId	Identifier "TierIdType" (see Table 339).	Allows identification of a tier within a incentiveTable.
tier. boundary (list)		
tier. boundary. boundaryId	Identifier "TierBoundaryIdType" (see Table 339).	Allows identification of a boundary within a tier.
tier. boundary. lowerBoundaryValue	Common data type "ScaledNumberType". See section 3.10.1.8.	This represents the lower end value range boundary.
tier. boundary. upperBoundaryValue	Common data type "ScaledNumberType". See section 3.10.1.8.	This represents the upper end value range boundary.
tier. incentive (list)		
tier. incentive. incentivId	Identifier "IncentivIdType" (see Table 339).	Allows identification of an incentive within a tier.
tier. incentive. value	Common data type "ScaledNumberType". See section 3.10.1.8.	The value of the incentive.

Table 150: IncentiveTableIncentiveSlotType detailed description of elements

5.2.1.2.3 Available selectors

- tariff. tariffId

5.2.1.3 incentiveTableDescriptionData

5.2.1.3.1 General

The incentiveTableDescription within the incentiveTableDescriptionData function holds the description of the IncentiveTable. This includes the tier, boundary and incentive type as well as further elements.

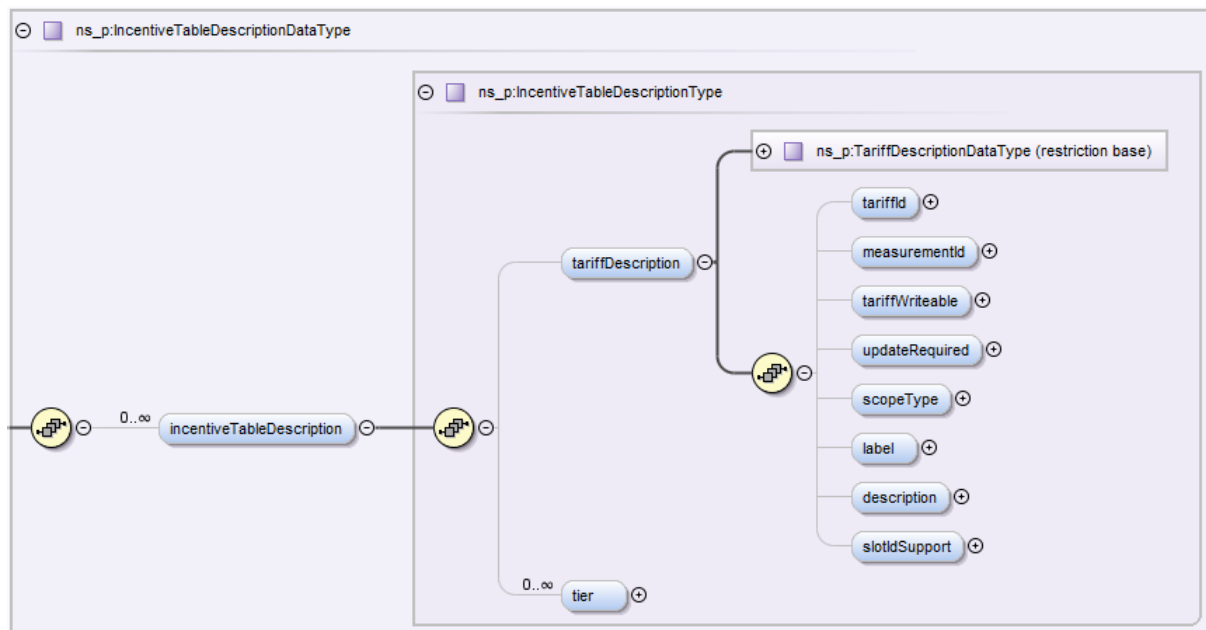


Figure 43: incentiveTableDescriptionData function overview

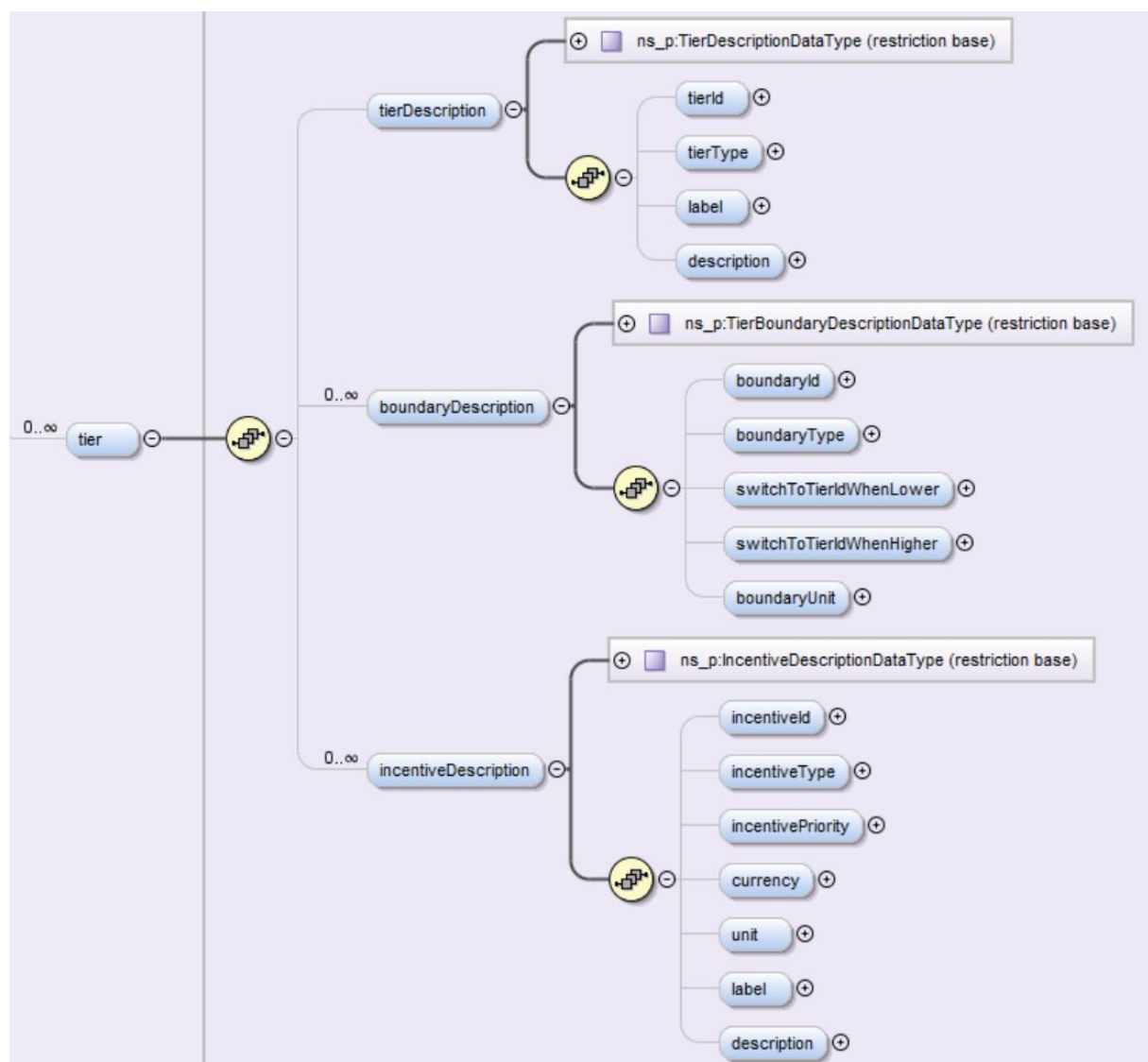
5.2.1.3.2 Detailed description of elements

Element	Type	Description
incentiveTableDescription (list)		
incentiveTableDescription. tariffDescription		
incentiveTableDescription. tariffDescription. tariffId	Identifier "TariffIdType" (see Table 339).	Allows the identification of a incentiveTable. Allows linking of the data in the different functions of IncentiveTable. Allows linking of the incentiveTables within other features that are placed in the same entity.
incentiveTableDescription. tariffDescription. measurementId	Identifier "MeasurementIdType" (see Table 339).	If a measurand is linked to the incentive table, its identifier is denoted here.
incentiveTableDescription. tariffDescription. tariffWriteable	xs:boolean (W3C standard type)	Whether the incentive table is writeable by a client or not can be denoted in this element.
incentiveTableDescription. tariffDescription. updateRequired	xs:boolean (W3C standard type)	Allows to request an update from the bound client.
incentiveTableDescription. tariffDescription. scopeType	Common data type "ScopeTypeType". See section 3.10.1.21.	A scopeType for the incentive table can be stated here.
incentiveTableDescription. tariffDescription. label	Common data type "LabelType". See section 3.10.1.2.	A short label on the incentive table.

incentiveTableDescription. tariffDescription. description	<i>Common data type</i> <i>"DescriptionType". See section 3.10.1.3.</i>	A description for the incentive table.
incentiveTableDescription. tariffDescription. slotIdSupport	xs:boolean (W3C standard type)	Indicates support of timeSlotId within the corresponding tariff in the Function incentiveTableData.
tier (list)	<i>Complex type</i> <i>"IncentiveTableDescriptionTierType". See Table 152.</i>	Allows to describe tiers and the incentives and boundaries that are used by the tier. Please refer to the detailed description of <i>IncentiveTableDescriptionTierType</i> elements.

5442 Table 151: incentiveTableDescriptionData function detailed description of elements

5443 IncentiveTableDescriptionTierType:



5444

5445 Figure 44: IncentiveTableDescriptionTierType overview

Element	Type	Description
tierDescription		

tierDescription. tierId	<i>Identifier "TierIdType" (see Table 339).</i>	Allows identification of a tier within a incentiveTable.
tierDescription. tierType	<i>Union "TierTypeType": - Enum (see Table 153): TierTypeEnumType - EnumExtendType (see section 3.10.1.5)</i>	Allows to define the type of the tier
tierDescription. label	<i>Common data type "LabelType". See section 3.10.1.2.</i>	Allows to define user friendly name for the tier.
tierDescription. description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Allows to define a user friendly description for the tier.
boundaryDescription (list)		
boundaryDescription. boundaryId	<i>Identifier "TierBoundaryIdType" (see Table 339).</i>	Allows identification of a boundary within a tier.
boundaryDescription. boundaryType	<i>Union "TierBoundaryTypeType": - Enum (see Table 154): TierBoundaryTypeEnumType - EnumExtendType (see section 3.10.1.5)</i>	Allows to define the type of the boundary.
boundaryDescription. switchToTierIdWhenLower	<i>Identifier "TierIdType" (see Table 339).</i>	Link to the tier that will become active if the lowerBoundaryValue is underrun.
boundaryDescription. switchToTierIdWhenHigher	<i>Identifier "TierIdType" (see Table 339).</i>	Link to the tier that will become active if the upperBoundaryValue is exceeded.
boundaryDescription. boundaryUnit	<i>Common data type "UnitOfMeasurementType". See section 3.10.1.17.</i>	Allows to define the unit of the boundary values.
incentiveDescription (list)		
incentiveDescription. incentiveId	<i>Identifier "IncentiveIdType" (see Table 339).</i>	Allows identification of an incentive within a tier.
incentiveDescription. incentiveType	<i>Union "IncentiveTypeType": - Enum (see Table 155): IncentiveTypeEnumType - EnumExtendType (see section 3.10.1.5)</i>	Allows to define the type of the incentive.
incentiveDescription. incentivePriority	<i>Simple type "IncentivePriorityType" (restriction of xs:unsignedInt)</i>	Allows to define the priority of the incentive. E.g. the priority that the customer has

		assigned for this incentive.
incentiveDescription. currency	<i>Common data type "CurrencyType". See section 3.10.1.19.</i>	Monetary incentives may need to describe a currency.
incentiveDescription. unit	<i>Common data type "UnitOfMeasurementType". See section 3.10.1.17.</i>	Allows definition of a unit, if the incentive has a unit.
incentiveDescription. label	<i>Common data type "LabelType". See section 3.10.1.2.</i>	Allows to express a user friendly label for the incentive.
incentiveDescription. description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Allows to express a user friendly description for the incentive.

5446 Table 152: IncentiveTableDescriptionTierType detailed description of elements

5447 Enumeration **TierTypeEnumType**:

Value	Rule	Description
fixedCost	The tier has a cost incentive that SHALL NOT vary over time	The fixed cost have to be payed, independent from the consumed energy.
dynamicCost	The tier has a cost incentive that MAY vary over time	The dynamic cost has a (linear) dependency to the consumed energy.

5448 Table 153: Enumeration TierTypeEnumType

5449 Enumeration **TierBoundaryTypeEnumType**:

Value	Rule	Description
powerBoundary	The boundary SHALL be an electrical power. E.g. if the surplus power is consumed, more expensive power has to be used.	Provides boundaries for power.
energyBoundary	The boundary SHALL be an electrical energy. E.g. if the stored self produced energy in a battery is consumed, more expensive energy has to be used.	Provides boundaries for energy.
countBoundary	The boundary SHALL be a count. E.g. after a certain count the price of a good may be reduced.	Provides boundaries for counts.

5450 Table 154: Enumeration TierBoundaryTypeEnumType

5451 Enumeration **IncentiveTypeEnumType**:

Value	Rule	Description
absoluteCost	None.	Absolute costs with a currency and a unit. The total costs can be calculated.
relativeCost	None.	Relative costs that cannot be interpreted as some monetary costs.

renewableEnergyPercentage	None.	Percentage of renewable energy of the total obtained energy. This is an ecological incentive that allows optimizing consumption of renewable energy instead of costs.
co2Emission	None.	CO2 emission related to the obtained energy. This is an ecological incentive that allows optimizing the CO2 emission instead of costs.

Table 155: Enumeration IncentiveTypeEnumType

5.2.1.3.3 Available selectors

- tariffDescription. tariffId
- tariffDescription. scopeType

5.2.1.4 incentiveTableConstraintsData

5.2.1.4.1 General

The incentiveTableConstraints within the incentiveTableConstraintsData function holds additional constraints of the IncentiveTable, regarding how much tiers, boundaries, incentives and slots are allowed.

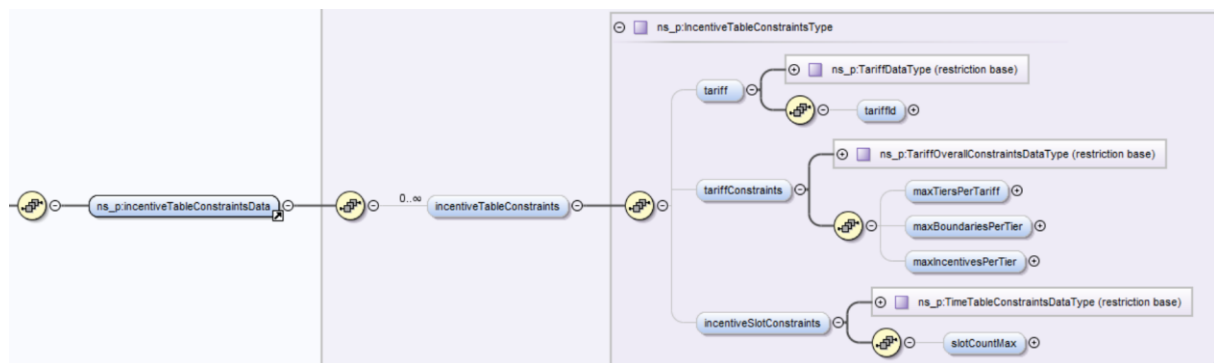


Figure 45: incentiveTableConstraintsData overview

5.2.1.4.2 Detailed description of elements

Element	Type	Description
incentiveTableConstraints (list)		
incentiveTableConstraints. tariff		
incentiveTableConstraints. tariff. tariffId	Identifier "TariffIdType" (see Table 339).	Allows the identification of a incentiveTable. Allows linking of the different functions within the IncentiveTable Feature. Allows linking of the

		incentiveTable within other Features that are placed in the same entity.
incentiveTableConstraints. tariffConstraints		
incentiveTableConstraints. tariffConstraints. maxTiersPerTariff	<i>Simple type "TierCountType" (restriction of "TierIdType" (see Table 339)).</i>	Allows the server to define a maximum tier count for this IncentiveTable. This is especially important if a client wants to write an IncentiveTable.
incentiveTableConstraints. tariffConstraints. maxBoundariesPerTier	<i>Simple type "TierBoundaryCountType" (restriction of "TierBoundaryIdType" (see Table 339)).</i>	Allows the server to define a maximum boundaries count for tiers. This is especially important if a client wants to write an IncentiveTable.
incentiveTableConstraints. tariffConstraints. maxIncentivesPerTier	<i>Simple type "IncentiveCountType" (restriction of "IncentiveIdType" (see Table 339)).</i>	Allows the server to define a maximum incentive count for tiers. This is especially important if a client wants to write an IncentiveTable.
incentiveTableConstraints. incentiveSlotConstraints		
incentiveTableConstraints. incentiveSlotConstraints. slotCountMax	<i>Simple type "TimeSlotCountType" (restriction of "TimeSlotIdType", see Table 328).</i>	Allows the server to define a maximum slot count for this IncentiveTable. This is especially important if a client wants to write an IncentiveTable.

Table 156: incentiveTableConstraints detailed description of elements

5.2.1.4.3 Available selectors

- tariff. tariffId

5.2.2 NodeManagement

See document [ProtocolSpecification], chapter "Functional commissioning" for details.

5.2.3 SmartEnergyManagementPs

5.2.3.1 Introduction

5.2.3.1.1 Scope of this complex class

This complex class provides solutions for "smart energy management" use cases that are typical for domestic homes: On the one hand, energy providers often offer their energy with a price that varies over time (e.g. consumption at night is cheaper than at noon). The availability of renewable energies from an own photovoltaics also varies over time. On the other hand, the residents certainly need to run their white goods (do one's laundry etc.) and other machines, but may have the opportunity to

shift these tasks in order to reduce the overall cost. Such kind of optimisation is hard to achieve manually in order to gain a real benefit. As a consequence, some kind of automation is required.

This complex class provides a solution that can best be expressed with the definition of two roles (i.e. device types) for these roles, and the information exchanged between them. The following roles are assumed:

1. A “smart appliance”:

This kind of device is usually an energy consuming or producing device, i.e. a device whose activity can be worth to be scheduled for a specific time in order to reduce the overall cost. Typical devices are whitegoods, photovoltaics, or battery storages. Such devices need to be “smart” as they have to be able to provide some information on their activity (esp. ahead of time) and (if possible) the capability to shift their activity upon appropriate requests from a role that is described next.

2. An energy management system:

A smart appliance usually cannot have the information required to perform an overall optimization of the cost and possibility for energy consumption and production. Instead, this is the responsibility of an energy management system. Such a system can gather activity information from several appliances of the domestic home. It may even be able to fetch information from an energy provider to get details of the tariff in order to execute up-to-date price calculations. This collection of information can then be used to adjust the activities of the connected appliances.

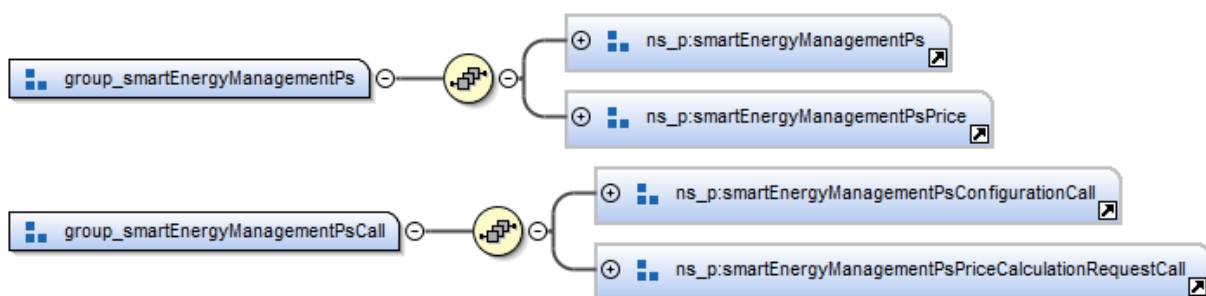


Figure 46: SmartEnergyManagementPs function-group overview

5.2.3.1.2 Re-use of standard classes: PowerSequences etc.

This complex class is mainly based upon definitions and concepts of the class PowerSequences (see section 5.3.19) which are used to express power or energy over time. In fact, this complex class primarily groups definitions from this and some other classes together into “useful collections”. Among others, this helps to avoid many single messages and gives the corresponding data parts a more “direct” relation to each other. This way, the overall structure of the complex class basically still represents power sequences. Therefore, throughout this complex class the term “power sequences” is still used. Section 5.2.3.1.3 briefly introduces the power sequences concept of this complex class.

5.2.3.1.3 Theory of operation (informative)

The power sequences structures of this complex class model the energy consumption / generation / storage as a mathematical graph and additional information, such as constraints for time based shifts.

Throughout this complex class, two terms - from the inherited PowerSequences class - are of major interest: “sequence” and “slot”. A sequence can be used to describe a “task”. It is the most “coarse” view in this class, i.e. a sequence represents all single steps of a whole task. The single steps are represented by slots. Slots are the most “fine” view in this class with regards to the time resolution of a task.

Each sequence and slot can be associated with an identifier or number, respectively. The slot numbers of two sequences should be considered independent from each other. I.e. slot number 7 of sequence 1 describes a different slot than slot number 7 of sequence 2. This basically means a slot is only “uniquely” identified in combination with a sequence ID. As formal notation this leads to the following relation:

- 1 sequence has 1..m slots
- 1 slot belongs to exactly 1 sequence
- 1 pair of a sequenceId and slotNumber uniquely identifies a slot

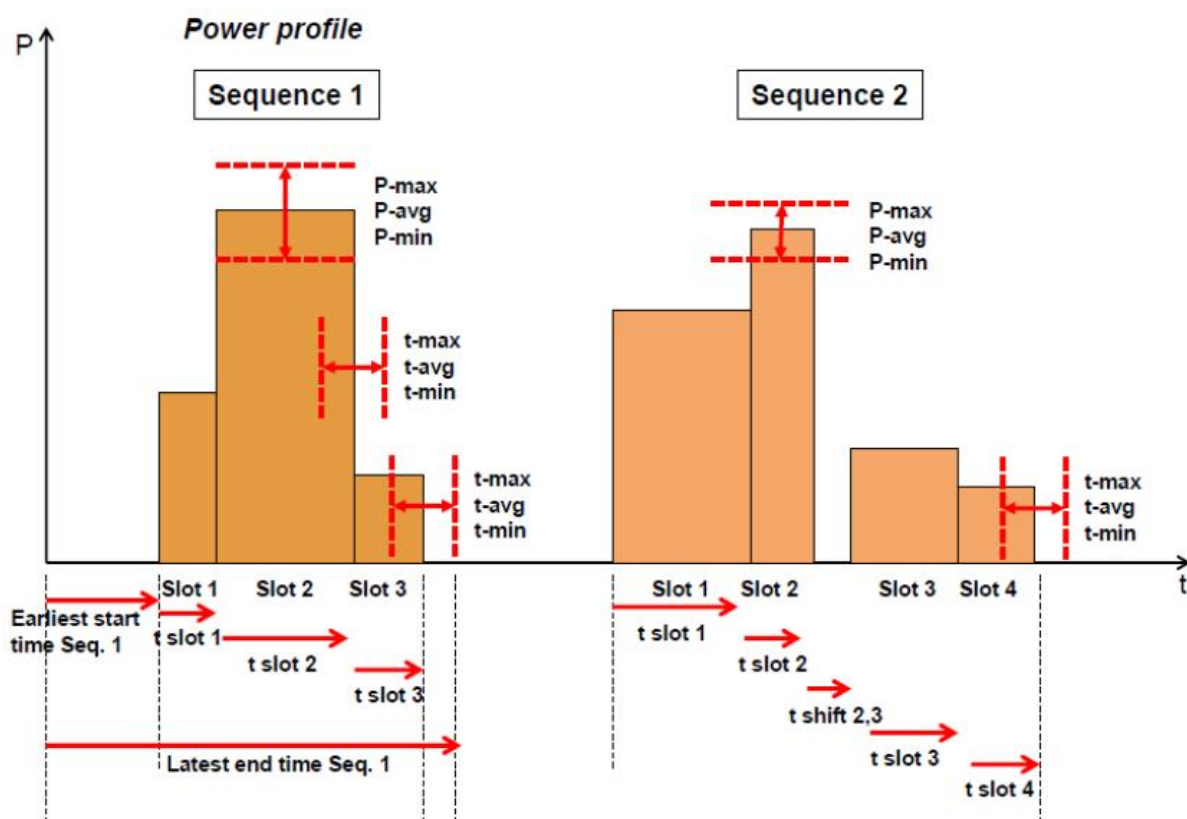


Figure 47: Concept of a power sequence

Each slot is, among other things, characterized by a time it applies for and a list of valid values, which apply for this time, such as average energy consumption, maximum energy consumption and others.

- 5537 In addition to value properties, a slot and a sequence can be described by their constraints.
5538 Constraints state whether a sequence and/or its single slots can be shifted in time. Additionally, if
5539 this is the case, the constraints state the boundaries for time shifts or duration configurations. This
5540 contributes to the definition of a power sequence's flexibility.
- 5541 Some time-related information on a slot shall be considered:
- 5542 - startTime: The currently scheduled start time of the slot.
 - 5543 - endTime: The currently scheduled end time of the slot.
 - 5544 - defaultDuration: The currently scheduled duration of this slot.
 - 5545 - durationUncertainty: The duration may have some uncertainty. The actual duration then
5546 may be in the range from defaultDuration - durationUncertainty to defaultDuration +
5547 durationUncertainty, e.g.
 - 5548 - earliestStartTime: The earliest time a slot could start. This determines the "lowest" boundary
5549 to shift a slot.
 - 5550 - latestEndTime: The latest time a slot has to be finished. This determines the "highest"
5551 boundary to shift a slot.
- 5552 Within a power sequence, there may be mandatory and optional slots. An example for an optional
5553 slot is a pause/intentional gap between 2 process steps of a device, where the pause might either
5554 take place or can be skipped.
- 5555 Within a sequence, the model permits in theory to have a "time gap" between two consecutive slots.
5556 However, for some reason it is usually better to express such a gap (or "pause") as regular slot with
5557 an energy value of "0".
- 5558 Each sequence and slot can be described using textual information, e.g. for displaying it on a
5559 smartphone or an in home display.
- 5560 A power sequences server can also request a price calculation for a sequence, e.g. to display a price
5561 on a device display to the user.
- 5562 In general, a so-called power sequences server (see section 5.2.3.1.4) can offer one or more power
5563 sequences. A power sequence can contain flexibility or can instead be used to forecast fixed
5564 consumption or generation behaviour without any flexibility. The granularity, i.e. the number of slots
5565 and value types provided for the slots a device models per power sequence, is a choice which is up to
5566 the manufacturer.
- 5567 A power sequences server can offer multiple power sequences for different purposes:
- 5568 1. The power sequences can model choices a so-called power sequences client (see 5.2.3.1.4)
5569 can take.
5570 As an example, consider a freezer which can run in normal mode or in deep freezing mode
5571 (which means increased power consumption, but the time frame for the power consumption
5572 is shorter). This choice can be offered to the power sequences client. In this complex class
5573 the power sequences of a choice are combined in a group called "alternatives" (see below).
 - 5574 2. The power sequences can model behaviour over time.

5575 As an example, consider an HVAC system which knows that it has to run in the morning and
5576 in the evening. A power sequences server can publish 2 different power sequences, one for
5577 each time frame.

5578 3. The power sequences can model mixtures of choices and behaviour over time (1 and 2).

5579 The general concept of power sequences is applicable to many kinds of devices. The above
5580 mentioned “alternatives” groups are based on the same concept as the “alternatives” concept of the
5581 PowerSequences class (see section 5.3.19.1.3). Within this complex class an “alternatives” group is
5582 modelled in the root element “alternatives” of function smartEnergyManagementPsData.

5583 Summarizing, there are several kinds of data contributing to the extent of a power sequences
5584 server's flexibility:

- 5585 1. Several flags denoting whether the power sequences server permits configuration from a
5586 bound power sequences client.
- 5587 2. The number of power sequences of an "alternatives" group (i.e. whether a choice from
5588 multiple concurrent power sequences can be taken).
- 5589 3. The possibility for shifting a sequence or its slots (see above for explanation on constraints).

5590

5591 5.2.3.1.3.1 Power sequences application behaviour guidance (informative)

5592 In general, the SmartEnergyManagementPs complex class permits a lot of different use cases or
5593 applications to be realised. Thus, it is rather difficult to give a complete overview for all applications.
5594 Instead, this section gives a guideline, which covers the main aspects that are typical for most
5595 applications.

5596 A power sequence in general is one of the following:

- 5597 • An energy- or power-forecast for a process without any flexibilities (fixed forecast)
- 5598 • An energy- or power-forecast that can be shifted, chosen or configured by a power
5599 sequences client

5600 Anyhow, the basic application guideline is common for both types of power sequences.

5601 We assume a power sequences server and a power sequences client established a communication
5602 channel. At some point in time, a process is going to be started on the device which contains the
5603 power sequences server. The power sequences server thus exposes at least one power sequence for
5604 this process. Depending on the kind of action the device wants to start, the power sequences server
5605 exposes fixed forecast(s), i.e. power sequence(s) without any flexibilities, power sequences with
5606 flexibilities, which might also contain alternative choices, or a mixture of both.

5607 Once a power sequences client receives one or more power sequences from a server, it should start
5608 the evaluation of the received data. The first thing to determine is, whether the power sequences
5609 server has exposed power sequence(s) which allow for configuration, or whether they are fixed
5610 forecasts.

5611 Independent of the type of the power sequence(s), the power sequences server should notify
5612 changes of the power sequence(s) (configuration and state changes, e.g.) according to the current
5613 situations.

5614 If the power sequences contain flexibilities and/or alternatives, the power sequences server can send
5615 a power sequences configuration request call. Once the power sequence client receives this
5616 message, it can evaluate whether it will configure the according power sequences server or ignore
5617 the call (temporarily or permanently). If the power sequences client does not configure the power
5618 sequences server, the power sequences server follows the configuration it has initially exposed, i.e.
5619 its own preference. If the power sequences client wants to configure the server it can achieve this by
5620 writing a power sequences configuration to the power sequence server. If the power sequences
5621 server receives a configuration, it first checks whether the configuration is valid. If the received
5622 configuration is valid, i.e. meets the constraints the power sequences server has laid out, it takes
5623 over the configuration.

5624 During runtime, the power sequences client and server may re-negotiate the configuration multiple
5625 times, even if the according power sequence is already running (if the power sequences server
5626 supports it). However, every time the data on either the power sequence information, configuration
5627 or state changes, the power sequences server should update the subscribed nodes. Of course, this
5628 should also be done at the completion of a power sequence.

5629 If new processes are going to be considered, these mechanisms are then repeated.

5630

5631 *5.2.3.1.3.2 Pausing and resuming power sequences (informative)*

5632 Some devices may permit to pause and resume specific power sequences. Such sequences can be
5633 marked as pausable (via function smartEnergyManagementPsData, "alternatives. powerSequence.
5634 operatingConstraintsInterrupt. isPausable" = "true", eg.). The sequence needs to be remotely
5635 controllable, too (which may be expressed by "alternatives. powerSequence. state.
5636 sequenceRemoteControllable" = "true" in function smartEnergyManagementPsData, e.g.), when a
5637 pause command shall be executed. Of course, the device itself needs to permit power sequences
5638 configuration in general (which may be expressed by "nodeScheduleInformation.
5639 nodeRemoteControllable" = "true" in function smartEnergyManagementPsData, e.g.). The power
5640 sequences client then may pause a running sequence and resume it at a later time (provided that the
5641 constraints are met).

5642 Please note that a device may resume its actions on its own, to meet the configured constraints. It
5643 then SHALL inform all subscribed clients about the changed operation.

5644

5645 *5.2.3.1.4 Role concept*

5646 The SPINE Protocol Specification defines the terms "server" and "client" which apply here as well.
5647 With regards to this Smart Energy Management specification the terms also get a specific semantic
5648 which is briefly described in this section.

5649 A device offering time based shifts for its energy consumption/generation/storage or fixed forecasts
5650 or mode switches (between consumption/generation/storage) SHALL be called a power sequences
5651 server.

5652 A device providing configuration for the flexibilities another device offers and/or understanding fixed
5653 forecasts SHALL be called a power sequences client.

Applications using this complex class may well benefit from binding or subscription mechanisms. Therefore, the following terms are defined:

- A power sequences client which has 1 or more bindings to power sequences servers is called a “bound power sequences client”.
- A power sequences client which does not have a binding to a power sequences server but a subscription instead is called a “subscribed power sequences client”.

5.2.3.2 Function specifications

5.2.3.2.1 smartEnergyManagementPsData

5.2.3.2.1.1 General

This function is modelled to keep all “operational” power sequences information.

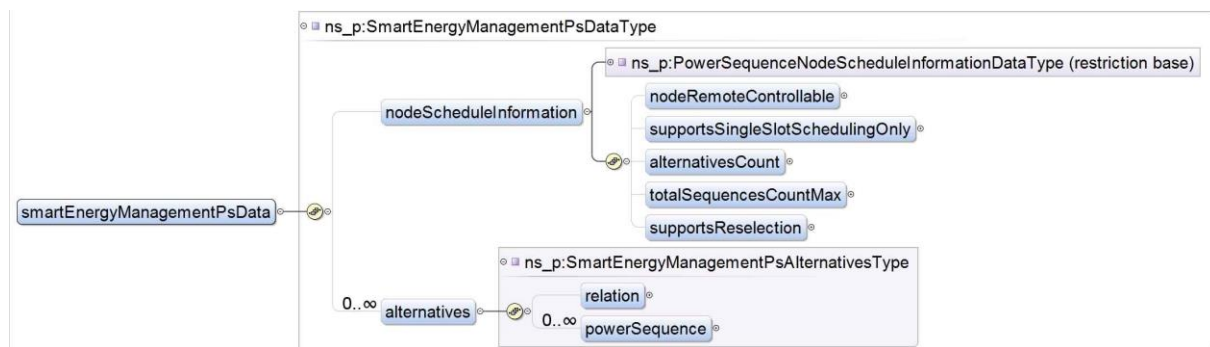


Figure 48: smartEnergyManagementPsData function overview, part 1

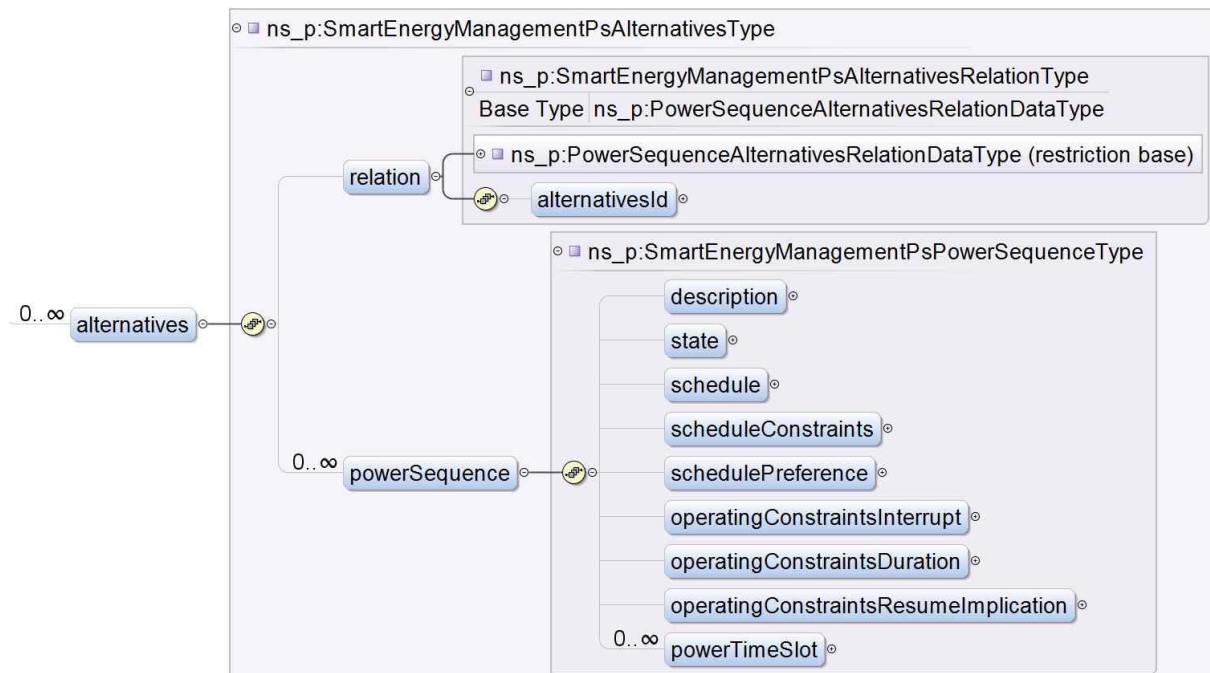


Figure 49: smartEnergyManagementPsData function overview, part 2



Figure 50: smartEnergyManagementPsData function overview, part 3

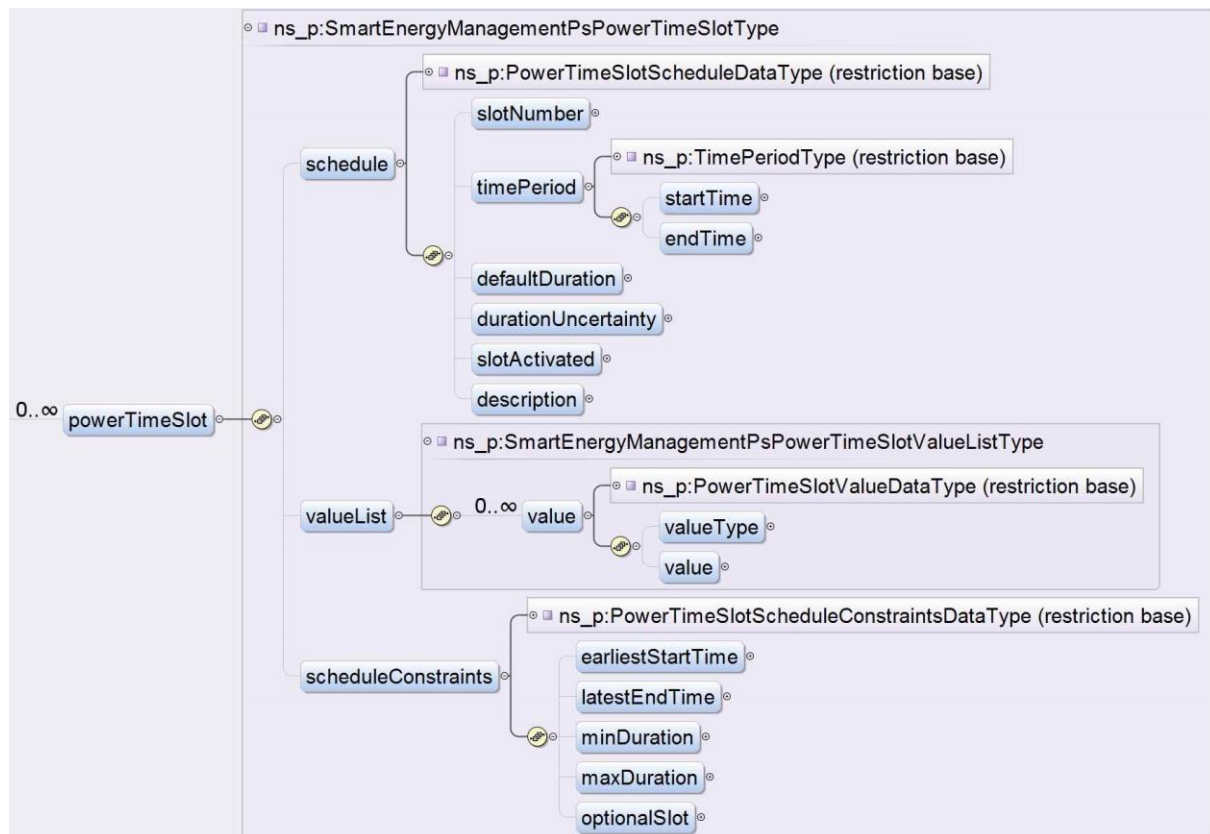


Figure 51: smartEnergyManagementPsData function overview, part 4

5.2.3.2.1.2 Detailed description of elements

The complex function “smartEnergyManagementPsData” of the complex class “SmartEnergyManagementPs” is described in the following table:

Element name	Data type	Explanation
nodeScheduleInformation		Contains “global”/summary information (independent from a specific sequence). Note: This element is derived from the PowerSequences class function “powerSequenceNodeScheduleInformationData”.
nodeScheduleInformation. nodeRemoteControllable	xs:boolean	Indicates whether the power sequences server (e.g., appliance) is configured for remote control (e.g., by an energy management system). This refers to the selection chosen by the user on the remote control feature of the device. If the value is “false”, the remote energy management is disabled, otherwise it is enabled.
nodeScheduleInformation. supportsSingleSlotSchedulingOnly	xs:boolean	If set to “true” the power sequences server does NOT permit the modification of more than one slot per configuration command received from a client. I.e. in

		this case a client needs to submit individual configuration commands for each slot.
nodeScheduleInformation.alternativesCount	xs:unsignedInt	The number of “alternatives” groups. I.e. the number of occurrences of the element “alternatives”.
nodeScheduleInformation.totalSequencesCountMax	xs:unsignedInt	The total number of sequences supported by the power sequences server. I.e. the sum of all power sequences across all “alternatives” group in this message (i.e. also regardless of the sequences’ states).
nodeScheduleInformation.supportsReselection	xs:boolean	If set to “true” the power sequences server does not restrict the number of sequence re-selections by a power sequences client. I.e. within a given “alternatives” group the power sequences client may first choose one sequence, alter the selection by configuring another sequence later on, then alter the selection again, etc. (provided the process rules and data still permit configuration). If supportsReselection is set to “false”, on the other hand, the power sequences server permits the power sequences client to select a sequence of an “alternatives” group only one time.
alternatives (list)		Each occurrence of “alternatives” announces a new group of mutually alternative sequences.
alternatives. relation		Note: This element is derived from the PowerSequences class function “powerSequenceAlternativesRelationData”.
alternatives. relation. alternativesId	<i>Identifier</i> “AlternativesIdType” (see Table 339).	The endpoint-wide unique identifier of the “alternatives” group.
alternatives. powerSequence	List of data group (1..unbounded)	Each occurrence of “powerSequence” announces a new group of power sequences data.
alternatives. powerSequence. description		General descriptions on the power sequence. Note: This element is derived from the PowerSequences class function “powerSequenceDescriptionData”.
alternatives. powerSequence. description. sequenceId	<i>Identifier</i> “PowerSequenceId”	The endpoint-wide unique sequence identifier.

	<i>Type" (see Table 339).</i>	
alternatives. powerSequence. description. description	xs:string (1..60 characters)	If used, should state a textual description for this power sequence.
alternatives. powerSequence. description. powerUnit	<i>Common data type "UnitOfMeasurementType". See section 3.10.1.17.</i>	The unit that applies to all power related values of this function.
alternatives. powerSequence. description. energyUnit	<i>Common data type "UnitOfMeasurementType". See section 3.10.1.17.</i>	The unit that applies to all energy related values of this function.
alternatives. powerSequence. description. valueSource	<i>Union "MeasurementValueSourceType": - Enum (see Table 232): MeasurementValueSourceEnumType - EnumExtendType (see section 3.10.1.5)</i>	The source (origin / foundation) of the forecasted values for this power sequence, such as "calculatedValue". Remark: This element shall express the reliability of the forecast.
alternatives. powerSequence. description. taskIdentifier	xs:unsignedInt	This element is used if a server wants to uniquely identify reoccurring types of sequences. As an example: Specific types of washing cycles with specific parameters should have the same taskIdentifier every time they are offered using power sequences. Note: This element should not be confused with "taskId" of the class TaskManagement.
alternatives. powerSequence. description. repetitionsTotal	xs:unsignedInt	If a power sequence executes its sequence of slots more than one time, the element contains the total number of executions.
alternatives. powerSequence. state		Information about the state of the power sequence. Note: This element is derived from the PowerSequences class function "powerSequenceStateData".

alternatives. powerSequence. state. state	<i>Union</i> "PowerSequenceStateType": - Enum (see Table 269): PowerSequenceStateEnumType - EnumExtendType (see section 3.10.1.5)	The current state of the sequence is stated in this element.
alternatives. powerSequence. state. activeSlotNumber	<i>Identifier</i> "PowerTimeSlotNumberType" (see Table 339).	Contains the currently active slot number.
alternatives. powerSequence. state. elapsedSlotTime	xs:duration	This element contains the time the slot has already been in "running" state (this also means the value remains constant during a "paused" state).
alternatives. powerSequence. state. remainingSlotTime	xs:duration	This element contains the time the slot still needs to be in "running" state (this also means the value remains constant during a "paused" state).
alternatives. powerSequence. state. sequenceRemoteControllable	xs:boolean	Denotes whether the sequence can be configured by a client (value "true") or not (value "false").
alternatives. powerSequence. state. activeRepetitionNumber	xs:unsignedInt	activeRepetitionNumber denotes the number of the current execution of the sequence of slots.
alternatives. powerSequence. state. remainingPauseTime	xs:duration	Denotes the duration the current slot (element activeSlotNumber) permits being paused.
alternatives. powerSequence. schedule		Information about the schedule of the power sequence. Note: This element is derived from the PowerSequences class function "powerSequenceScheduleData".
alternatives. powerSequence. schedule. startTime	xs:duration	The startTime of the power sequence.
alternatives. powerSequence. schedule. endTime	xs:duration	The endTime of the power sequence.
alternatives. powerSequence. scheduleConstraints		Information about the schedule constraints of the power sequence.

		Note: This element is derived from the PowerSequences class function "powerSequenceScheduleConstraintsData".
alternatives. powerSequence. scheduleConstraints. earliestStartTime	xs:duration	The earliest possible start time for this whole power sequence.
alternatives. powerSequence. scheduleConstraints. latestEndTime	xs:duration	The latest possible end time for this whole power sequence.
alternatives. powerSequence. schedulePreference		Contains information on the power sequences server's preference for its scheduling. Note: This element is derived from the PowerSequences class function "powerSequenceSchedulePreferenceData".
alternatives. powerSequence. schedulePreference. greenest	xs:boolean	If present and set to "true": The power sequences client should try to optimise the configuration towards the minimum usage of NON-renewableenergy.
alternatives. powerSequence. schedulePreference. cheapest	xs:boolean	If present and set to "true": The power sequences client should try to apply a configuration that minimises the user's energy bill for this power sequence.
alternatives. powerSequence. operatingConstraintsInterrupt		Information about the interruptibility of the power sequence. Note: This element is derived from the OperatingConstraints class function "operatingConstraintsInterruptData".
alternatives. powerSequence. operatingConstraintsInterrupt. isPausable	xs:boolean	If the sequence is pausable by the bound power sequences client, this element is set to true.
alternatives. powerSequence. operatingConstraintsInterrupt. isStoppable	xs:boolean	If the sequence is stoppable by the bound power sequences client, this element is set to true.
alternatives. powerSequence. operatingConstraintsInterrupt. maxCyclesPerDay	xs:unsignedInt	States the maximum amount of starts the device allows per day.

alternatives. powerSequence. operatingConstraintsDuration		Constraints about minimum or maximum durations of active- or pause-phases. Note: This element is derived from the OperatingConstraints class function "operatingConstraintsDurationData".
alternatives. powerSequence. operatingConstraintsDuration. activeDurationMin	xs:duration	This is the minimum duration the sequence has to run without interruption.
alternatives. powerSequence. operatingConstraintsDuration. activeDurationMax	xs:duration	This is the maximum duration the sequence can run without interruption.
alternatives. powerSequence. operatingConstraintsDuration. pauseDurationMin	xs:duration	This is the minimum duration the sequence has to pause after the end of an activity.
alternatives. powerSequence. operatingConstraintsDuration. pauseDurationMax	xs:duration	This is the maximum duration the sequence can pause after the end of an activity.
alternatives. powerSequence. operatingConstraintsDuration. activeDurationSumMin	xs:duration	This is the minimum duration the sequence has to run in total (summation of all active times).
alternatives. powerSequence. operatingConstraintsDuration. activeDurationSumMax	xs:duration	This is the maximum duration the sequence can run in total (summation of all active times).
alternatives. powerSequence. operatingConstraintsResumeImplication		If resuming an active phase after a pause has implications on energy or price costs, this element and the required sub-elements contain the proper information. Note: This element is derived from the OperatingConstraints class function "operatingConstraintsResumeImplicationData".
alternatives. powerSequence. operatingConstraintsResumeImplication. resumeEnergyEstimated	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Additional energy the device will consume before resuming its normal operation (after a pause). This is only an estimated value which will not be added to the value stated in any slot value information.

alternatives. powerSequence. operatingConstraintsResumeIm plication. energyUnit	<i>Common data type "UnitOfMeasurem entType". See section 3.10.1.17.</i>	Unit for the energy stated in the element resumeEnergyEstimated.
alternatives. powerSequence. operatingConstraintsResumeIm plication. resumeCostEstimated	<i>Common data type "ScaledNumberTyp e". See section 3.10.1.8.</i>	Additional costs for the resumption of a device to its normal operation (after a pause).
alternatives. powerSequence. operatingConstraintsResumeIm plication. currency	<i>Common data type "CurrencyType". See section 3.10.1.19.</i>	Currency for the price stated in the element resumeCostEstimated.
alternatives. powerSequence. powerTimeSlot	List of data group (1..unbounded)	Each occurrence of "powerTimeSlot" announces a new "slot information group".
alternatives. powerSequence. powerTimeSlot. schedule		Information about the schedule of a slot. Note: This element is derived from the PowerSequences class function "powerTimeSlotScheduleData".
alternatives. powerSequence. powerTimeSlot. schedule. slotNumber	<i>Identifier "PowerTimeSlotNu mberType" (see Table 339).</i>	The powerSequenceId-wide unique slot identifier.
alternatives. powerSequence. powerTimeSlot. schedule. timePeriod		
alternatives. powerSequence. powerTimeSlot. schedule. timePeriod. startTime	xs:duration	The start time of the slot.
alternatives. powerSequence. powerTimeSlot. schedule. timePeriod. endTime	xs:duration	The end time of the slot.
alternatives. powerSequence. powerTimeSlot. schedule. defaultDuration	xs:duration	The duration of the slot.
alternatives. powerSequence. powerTimeSlot. schedule. durationUncertainty	xs:duration	The uncertainty of the duration given in the element alternatives. powerSequence. powerTimeSlot. schedule. defaultDuration.

alternatives. powerSequence. powerTimeSlot. schedule. slotActivated	xs:boolean	If the slot is optional, i.e. “alternatives. powerSequence. powerTimeSlot. scheduleConstraints. optionalSlot” is set to “true”, this element reflects the current status of the slot (true = the slot will be executed, false = the slot will not be executed).
alternatives. powerSequence. powerTimeSlot. schedule. description	<i>Common data type "DescriptionType". See section 3.10.1.3. (1..60 characters)</i>	A textual description for this slot.
alternatives. powerSequence. powerTimeSlot. valueList		Parent element of subsequent list.
alternatives. powerSequence. powerTimeSlot. valueList. value	List of “alternatives. powerSequence. powerTimeSlot. valueList. value” (0..unbounded)	Note: This element is derived from the PowerSequences class function “powerTimeSlotValueData”.
alternatives. powerSequence. powerTimeSlot. valueList. value. valueType	<i>Union "PowerTimeSlotVa lueTypeType": - Enum (see Table 263): PowerTimeSlotVal ueTypeEnumType - EnumExtendType (see section 3.10.1.5)</i>	The value type of this value.
alternatives. powerSequence. powerTimeSlot. valueList. value. value	<i>Common data type "ScaledNumberTyp e". See section 3.10.1.8.</i>	A value according to the type expressed in alternatives. powerSequence. powerTimeSlot. valueList. value. valueType. The value's unit is stated in alternatives. powerSequence. description. powerUnit or alternatives. powerSequence. description. energyUnit.
alternatives. powerSequence. powerTimeSlot. scheduleConstraints		Information about constraints for a time slot. Note: This element is derived from the PowerSequences class function “powerTimeSlotScheduleConstraintsData ”.

alternatives. powerSequence. powerTimeSlot. scheduleConstraints. earliestStartTime	xs:duration	The earliest start time for this slot.
alternatives. powerSequence. powerTimeSlot. scheduleConstraints. latestEndTime	xs:duration	The latest possible end time for this slot.
alternatives. powerSequence. powerTimeSlot. scheduleConstraints. minDuration	xs:duration	If the slot has a configurable duration, this element denotes the minimum supported configuration.
alternatives. powerSequence. powerTimeSlot. scheduleConstraints. maxDuration	xs:duration	If the slot has a configurable duration, this element denotes the maximum supported configuration.
alternatives. powerSequence. powerTimeSlot. scheduleConstraints. optionalSlot	xs:boolean	SHALL be present and set to a value of “true” denotes the slot can be omitted.

5677 Table 157: Description of complex class function “smartEnergyManagementPsData”

5678

5679 5.2.3.2.1.3 Identifiers

- 5680 - alternatives. powerSequence. description. sequenceId
- 5681 - alternatives. powerSequence. powerTimeSlot. schedule. slotNumber

5682 Please note: “alternatives. relation. alternativesId” is NOT an identifier as it does not open a further

5683 dimension in the data room. Instead it is kind of a group for one or more sequences.

5684

5685 5.2.3.2.1.4 Available selectors

5686 The selectors definition for function “smartEnergyManagementPsData” is given in the following

5687 table:

Element name	Data range	Explanation
smartEnergyManagementPsDataSelectors		
smartEnergyManagementPsDataSelectors. alternativesRelation		Note: This element is derived from the PowerSequences class selectors definition “powerSequenceAlternativesRel ationListDataSelectors”.

smartEnergyManagementPsDataSelectors. alternativesRelation. alternativesId	Identifier "AlternativesIdType" (see Table 339).	Selector item: The proper alternatives group is selected.
smartEnergyManagementPsDataSelectors. powerSequenceDescription		Note: This element is derived from the PowerSequences class selectors definition "powerSequenceDescriptionListDataSelectors".
smartEnergyManagementPsDataSelectors. powerSequenceDescription. sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	Selector item: The proper power sequence is selected.
smartEnergyManagementPsDataSelectors. powerTimeSlotSchedule		Note: This element is derived from the PowerSequences class selectors definition "powerTimeSlotScheduleListDataSelectors".
smartEnergyManagementPsDataSelectors. powerTimeSlotSchedule. slotNumber	Identifier "PowerTimeSlotNumberType" (see Table 339).	Selector item: The proper slot is selected.
smartEnergyManagementPsDataSelectors. powerTimeSlotValue		Note: This element is derived from the PowerSequences class selectors definition "powerTimeSlotValueListDataSelectors".
smartEnergyManagementPsDataSelectors. powerTimeSlotValue. valueType	Union "PowerTimeSlotValueType": - Enum (see Table 263): PowerTimeSlotValueTypeEnumType - EnumExtendType (see section 3.10.1.5)	Selector item: The kind of the value ("power", "powerMin", "energy", ...).

Table 158: Dedicated "selectors" of "smartEnergyManagementPsData".

5.2.3.2.2 smartEnergyManagementPsConfigurationRequestCall

5.2.3.2.2.1 General

A power sequences server may want to explicitly inform (or remind) a power sequences client if it wants one or "all" of its power sequences to be configured. The server can express this by sending a power sequences configuration request call to the client.



Figure 52: smartEnergyManagementPsConfigurationRequestCall function overview

5.2.3.2.2.2 Detailed description of elements

The complex function “smartEnergyManagementPsConfigurationRequestCall” of the complex class “SmartEnergyManagementPs” is described in the following table:

Element name	Data range	Explanation
scheduleConfigurationRequest		Note: This element is derived from the PowerSequences class function “powerSequenceScheduleConfigurationRequestCall”.
scheduleConfigurationRequest. sequenceId	Identifier “PowerSequenceIdType” (see Table 339).	States the sequence ID of the power sequences server which it prefers for configuration.

Table 159: Description of complex class function “smartEnergyManagementPsConfigurationRequestCall”

5.2.3.2.2.3 Identifiers

None.

5.2.3.2.2.4 Available selectors

None.

5.2.3.2.3 smartEnergyManagementPsPriceData

5.2.3.2.3.1 General

A power sequence may have a price, i.e. information on its cost. The complex function “smartEnergyManagementPsPriceData” of the complex class “SmartEnergyManagementPs” can be used to describe this information.

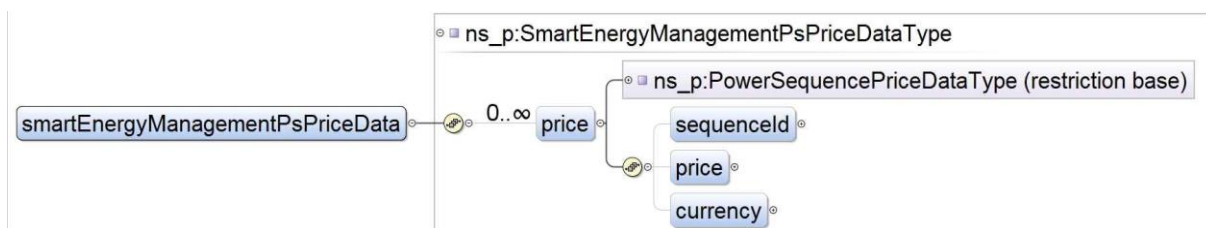


Figure 53: smartEnergyManagementPsPriceData function overview

5.2.3.2.3.2 Detailed description of elements

The complex function “smartEnergyManagementPsPriceData” of the complex class “SmartEnergyManagementPs” is described in the following table:

Element name	Data range	Explanation
price	List of price information (0..unbounded)	Each occurrence of this element contains information on one power sequence's cost. Note: This element is derived from the PowerSequences class function "powerSequencePriceData".
price. sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	The sequenceId the calculated price applies for.
price. price	Common data type "ScaledNumberType". See section 3.10.1.8.	The price (value) which has been calculated.
price. currency	Common data type "CurrencyType". See section 3.10.1.19.	The currency of the price.

Table 160: Description of smartEnergyManagementPsPriceData.

5.2.3.2.3.3 Identifiers

- sequenceId

5.2.3.2.3.4 Available selectors

The selectors definition for function "smartEnergyManagementPsPriceData" is given in the following table:

Element name	Data range	Explanation
smartEnergyManagementPsPriceDataSelectors		
smartEnergyManagementPsPriceDataSelectors. price		Note: This element is derived from the PowerSequences class selectors definition "powerSequencePriceListDataSelectors".
smartEnergyManagementPsPriceDataSelectors. price. sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	Selector item: The price information of the proper power sequence is selected.

Table 161: Dedicated "selectors" of "smartEnergyManagementPsPriceData".

5.2.3.2.4 *smartEnergyManagementPsPriceCalculationRequestCall*

5.2.3.2.4.1 *General*

Similar to the function “powerSequencePriceCalculationRequestCall” of class PowerSequences, the complex function “smartEnergyManagementPsPriceCalculationRequestCall” permits to model a trigger for an external price calculation. I.e. the function is designed for the case that a power sequences server cannot calculate a price for one of its sequences on its own, but knows a power sequences client that is capable to do this. Then, the power sequences server may send a powerSequencePriceCalculationRequestCall with proper data to the client in order to trigger the price calculation at the client. As a result, the client may then write the price to the server. The calculated price can for example be shown on the power sequences server’s device display or stored inside the device to build device internal historical information.



Figure 54: *smartEnergyManagementPsPriceCalculationRequestCall* function overview

5.2.3.2.4.2 *Detailed description of elements*

The complex function “smartEnergyManagementPsPriceCalculationRequestCall” of the complex class “SmartEnergyManagementPs” is described in the following table:

Element name	Data range	Explanation
priceCalculationRequest		Note: This element is derived from the PowerSequences class function “powerSequencePriceCalculationRequestCall”.
priceCalculationRequest. sequenceId	Identifier "PowerSequences sequenceIdType" (see Table 339).	The sequenceId a power sequences server wants to have a price calculated for.

Table 162: Description of complex class function “smartEnergyManagementPsPriceCalculationRequestCall”

5.2.3.2.4.3 *Identifiers*

None.

5.2.3.2.4.4 *Available selectors*

None.

5.3 Standard Classes

5.3.1 ActuatorLevel

5.3.1.1 Introduction

Independent from the *ActuatorSwitch* class, the *ActuatorLevel* class enables a user or application to model *LEVEL* commands (*start*, *up*, *percentageAbsolute*, *relative*, ...). This can be used to dim a light, set the speed of an electric motor, etc. Its simple functions can easily be mapped to other technologies.

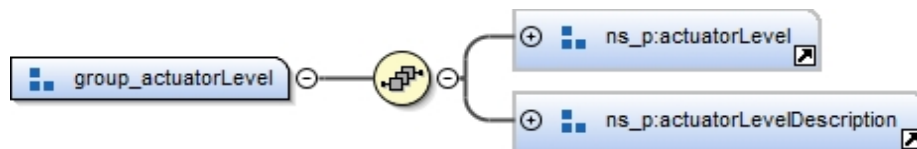


Figure 55: ActuatorLevel function-group overview

5.3.1.2 actuatorLevelData

5.3.1.2.1 General

This function is used for setting the desired operation mode of the node or getting the actual mode respectively.

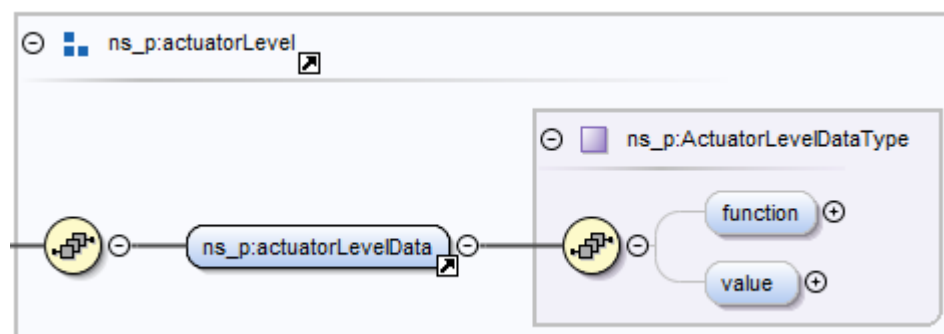


Figure 56: actuatorLevelData function overview

5.3.1.2.2 Detailed description of elements

Element	Type	Description
function	Union "ActuatorLevelFctType": - Enum (see Table 164): ActuatorLevelFctEnumType - EnumExtendType (see section 3.10.1.5)	The function contains the mode.
value	Common data type "ScaledNumberType". See section 3.10.1.8.	The value is only needed for some functions. E.g. setting <i>function</i> to <i>percentageAbsolute</i> just makes sense together with a proper value.

Table 163: actuatorLevelData function detailed description of elements

Enumeration **ActuatorLevelFctEnumType**:

Value	Description
start	Actuator starts.
up	Actuator moves up.
down	Actuator moves down.
stop	Actuator stops.
percentageAbsolute	The percentage position / value of the actuator (e.g. 25 %).
percentageRelative	Changes the current value by some percentage value (e.g. move plus 10 percent).
absolute	The absolute position / value of the actuator (e.g. 0.91 m).
relative	Changes the current value by some value (e.g. move plus 0.23 m).

Table 164: Enumeration ActuatorLevelFctEnumType

5.3.1.2.3 Available selectors

None.

5.3.1.2.4 Examples

5.3.1.2.4.1 Write

To dim a light (50 percent brightness), one could send the following XML together with a write classifier:

```

<actuatorLevelData>
  <function>percentageAbsolute</function>
  <value>
    <number>5</number>
    <scale>1</scale>
  </value>
</actuatorLevelData>

```

5.3.1.3 actuatorLevelDescriptionData

5.3.1.3.1 General

The non-changing descriptive data of the *actuatorLevel* class can contain a default unit, among others. This is useful for some *absolute* commands.

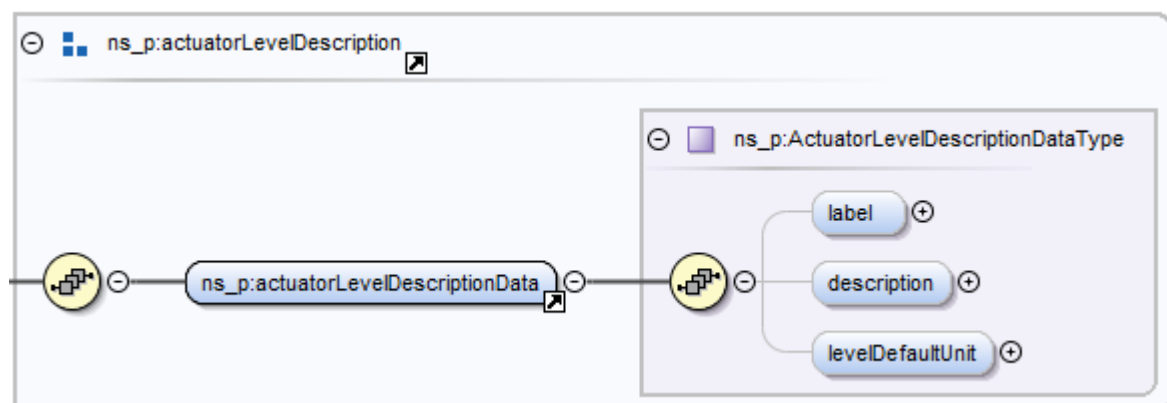


Figure 57: actuatorLevelDescriptionData function overview

5798

5799 **5.3.1.3.2 Detailed description of elements**

Element	Type	Description
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the actuator.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the actuator.
levelDefaultUnit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	If an actuator can specify a unit for its level-controlling, it can be denoted in this element.

5800 *Table 165: actuatorLevelDescriptionData function detailed description of elements*

5801

5802 **5.3.1.3.3 Available selectors**

5803 None.

5804

5805 **5.3.1.3.4 Examples**5806 **5.3.1.3.4.1 Read-reply**

5807 To query the description of an actuator level, one could send a read command to the server. The
 5808 reply could be e.g.:

```

5809 <actuatorLevelDescriptionData>
5810   <label>ActuatorLevel TestDevice</label>
5811   <description>Device for testing SPINE compliance.</description>
5812   <levelDefaultUnit>W</levelDefaultUnit>
5813 </actuatorLevelDescriptionData>

```

5814

5815 **5.3.2 ActuatorSwitch**5816 **5.3.2.1 Introduction**

5817 Basic on/off operations on a simple actuator can be modelled with the *ActuatorSwitch* class.

5818 Whether the function turns a device itself *on* or *off*, or whether it switches a specific feature,
 5819 depends on the implementation. E.g. one could model the super freeze program of a freezer using
 5820 *ActuatorSwitch* class. An *on* command would then activate the super freeze program and an *off*
 5821 command would deactivate it. This example shall just give an idea how *ActuatorSwitch* can be used
 5822 for more purposes than only turning devices on and off.

5823 Please note a device's different features may be modelled with separate implementations of this
 5824 class, each associated to a different node address. Continuing with the example above, a "defrost"
 5825 function may be modelled on another node address with *ActuatorSwitch*.

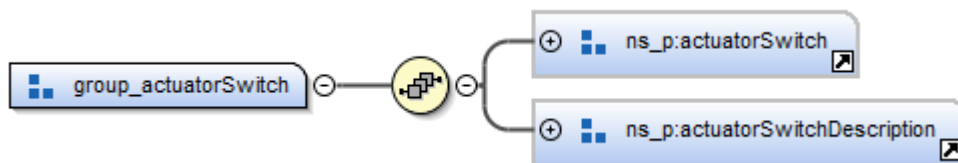


Figure 58: ActuatorSwitch function-group overview

5.3.2.2 actuatorSwitchData

5.3.2.2.1 General

The *actuatorSwitchData* function models the functionality described in section 5.3.2.1.

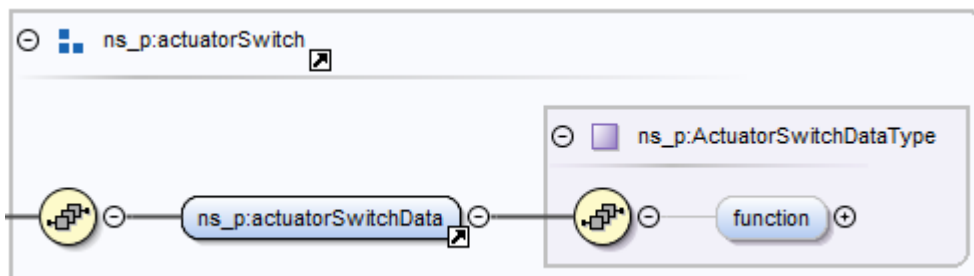


Figure 59: actuatorSwitchData function overview

5.3.2.2.2 Detailed description of elements

Element	Type	Description
function	Union "ActuatorSwitchFctType": - Enum (see Table 167): ActuatorSwitchFctEnumType - EnumExtendType (see section 3.10.1.5)	The only element in this function to model an <i>on</i> , <i>off</i> or <i>toggle</i> command.

Table 166: actuatorSwitchData function detailed description of elements

Enumeration **ActuatorSwitchFctEnumType**:

Value	Description
on	Turn on.
off	Turn off.
toggle	Toggle between on and off.

Table 167: Enumeration ActuatorSwitchFctEnumType

5.3.2.2.3 Available selectors

None.

5.3.2.2.4 Examples

5.3.2.2.4.1 Write

This example could turn on a device upon receipt with a write classifier:

```

5846 <actuatorSwitchData>
5847     <function>on</function>
5848 </actuatorSwitchData>

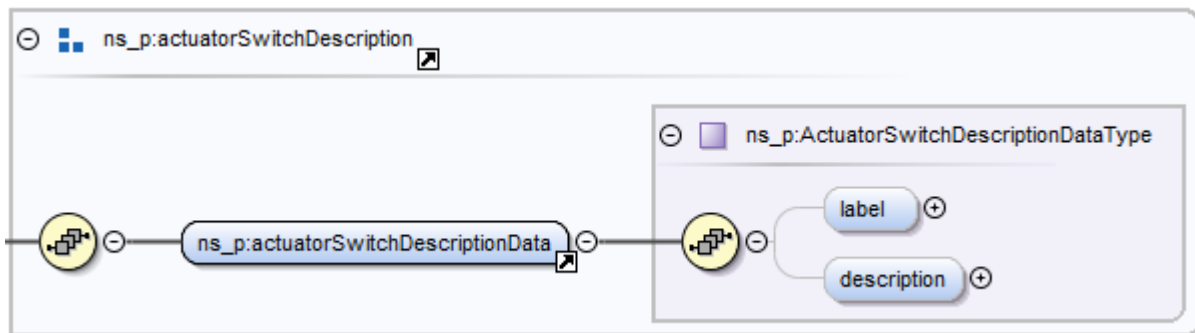
```

5849

5850 5.3.2.3 *actuatorSwitchDescriptionData*

5851 5.3.2.3.1 *General*

5852 The (typically) non-changing information on the actuator is modelled with this function.



5853

5854 Figure 60: *actuatorSwitchDescription* function overview

5855

5856 5.3.2.3.2 *Detailed description of elements*

Element	Type	Description
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the actuator.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the actuator.

5857 Table 168: *actuatorSwitchDescription* function detailed description of elements

5858

5859 5.3.2.3.3 *Available selectors*

5860 None.

5861

5862 5.3.2.3.4 *Examples*

5863 5.3.2.3.4.1 *Read-reply*

5864 One would like to know the contents of the label and description of a device. Therefore, a standard
 5865 read-command is sent to the server. Then a proper reply could be:

```

5866 <actuatorSwitchDescriptionData>
5867     <label>SmartPlug_1</label>
5868     <description>SmartPlug Test Installation 01.04.2012</description>
5869 </actuatorSwitchDescriptionData>

```

5870

5.3.3 Alarm

5.3.3.1 Introduction

There are several reasons for generating an alarm. This might be due to exceeding (in a positive or negative way) a threshold (modelled with the Threshold class, see section 5.3.26), or because of some other occurrence that provokes the generating of an alarm (modelled in a later version of this specification). The Alarm class provides the needed data model.



Figure 61: Alarm function-group overview

5.3.3.2 alarmListData

5.3.3.2.1 General

This function models all the needed data for an alarm. Depending on the type of alarm, different elements are used (see Feature Type "Alarm" for more details; section 4.3.3). It can be used to notify an alarm or as a reply to a read command.

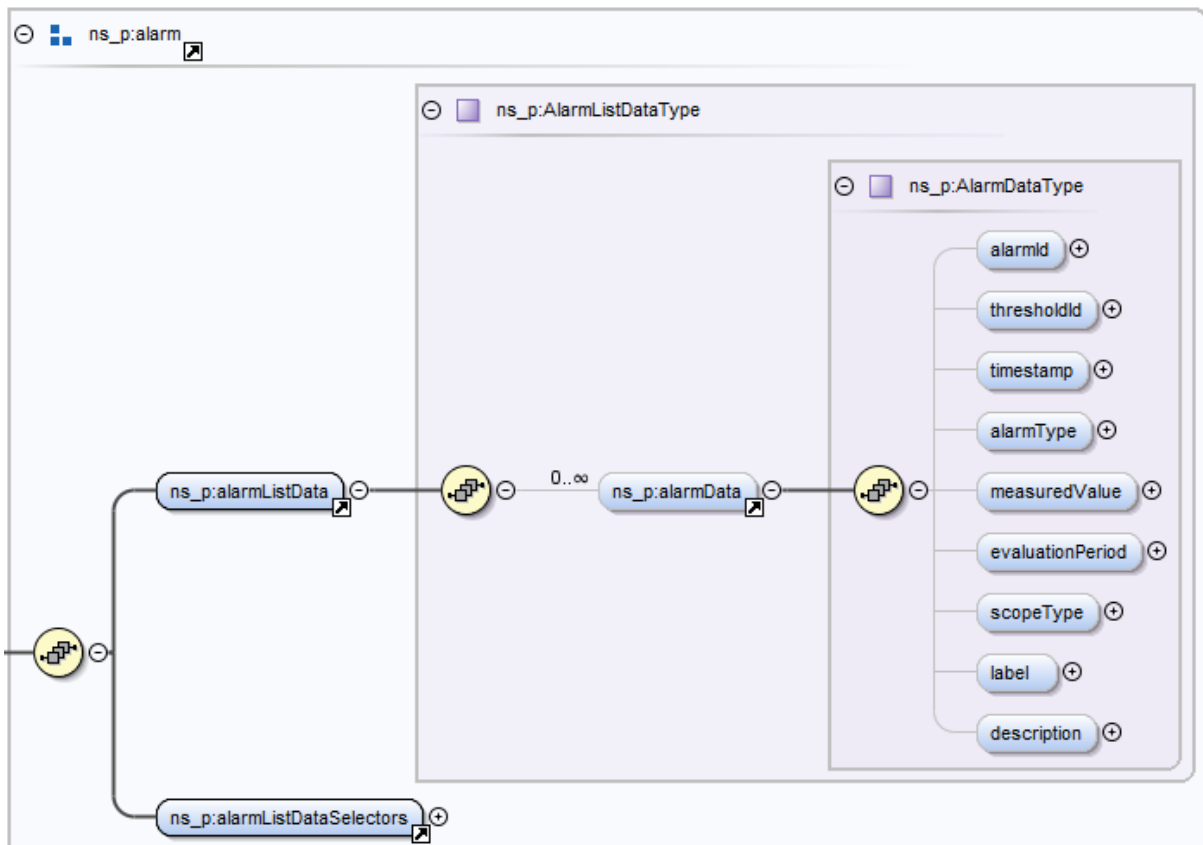


Figure 62: alarmListData function overview

5.3.3.2.2 Detailed description of elements

Element	Type	Description
---------	------	-------------

alarmId	Identifier "AlarmIdType" (see Table 339).	Identifier of the alarm.
thresholdId	Identifier "ThresholdIdType" (see Table 339).	If the alarm relates to a threshold, the corresponding ID can be stated here.
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The time of creation of the alarm.
alarmType	Union "AlarmTypeType": - Enum (see Table 170): AlarmTypeEnumType - EnumExtendType (see section 3.10.1.5)	The type of alarm is given here.
measuredValue	Common data type "ScaledNumberType". See section 3.10.1.8.	The value that was measured which exceeded the threshold is denoted in this element.
evaluationPeriod	Common data type "TimePeriodType". See section 3.10.2.1.	If a time period can be stated in which
scopeType	Common data type "ScopeTypeType". See section 3.10.1.21.	Specifies a more detailed meaning of the alarm (e.g. exceeding of a threshold).
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the alarm.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the alarm.

Table 169: alarmListData function detailed description of elements

Enumeration **AlarmTypeEnumType**:

Value	Description
alarmCancelled	A previously generated alarm is cancelled (e.g. because the value is in a "normal" range again).
underThreshold	The value is under a related threshold.
overThreshold	The value is over a related threshold.

Table 170: Enumeration AlarmTypeEnumType

5.3.3.2.3 Available selectors

- alarmId
- scopeType

5.3.3.2.4 Examples

5.3.3.2.4.1 Notify

When a threshold is reached, the server would send a notification to all subscribed clients, with a content like the following (the "filter/partial" part to announce the "restricted function exchange is simplified by ..." just to improve the readability):

...

```

5903 <alarmListData>
5904   <alarmData>
5905     <alarmId>4</alarmId>
5906     <thresholdId>2</thresholdId>
5907     <timestamp>2015-12-01T18:27:32.4Z</timestamp>
5908     <alarmType>overValue</alarmType>
5909     <measuredValue>
5910       <number>711</number>
5911       <scale>-1</scale>
5912     </measuredValue>
5913     <label>Temperature reached</label>
5914     <description>Desired tea temperature reached by water
5915 boiler.</description>
5916   </alarmData>
5917 </alarmListData>

```

5.3.4 Bill

5.3.4.1 Introduction

In many cases bill will be writeable. To allow the client to perform a single write, billData includes also a lot of descriptive elements. Additionally this concept avoids that data within billDescriptionData is changed by client and server. Therefore if a bill is written by a client, only billData is written by a client, while the other billConstraints and billDescription are never writeable.

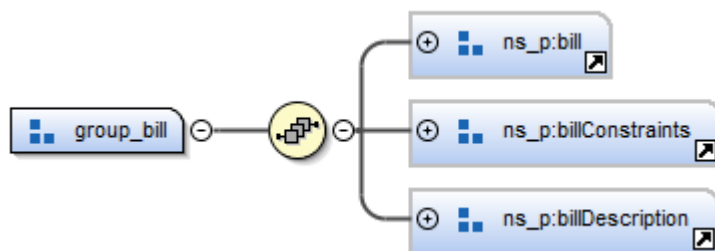


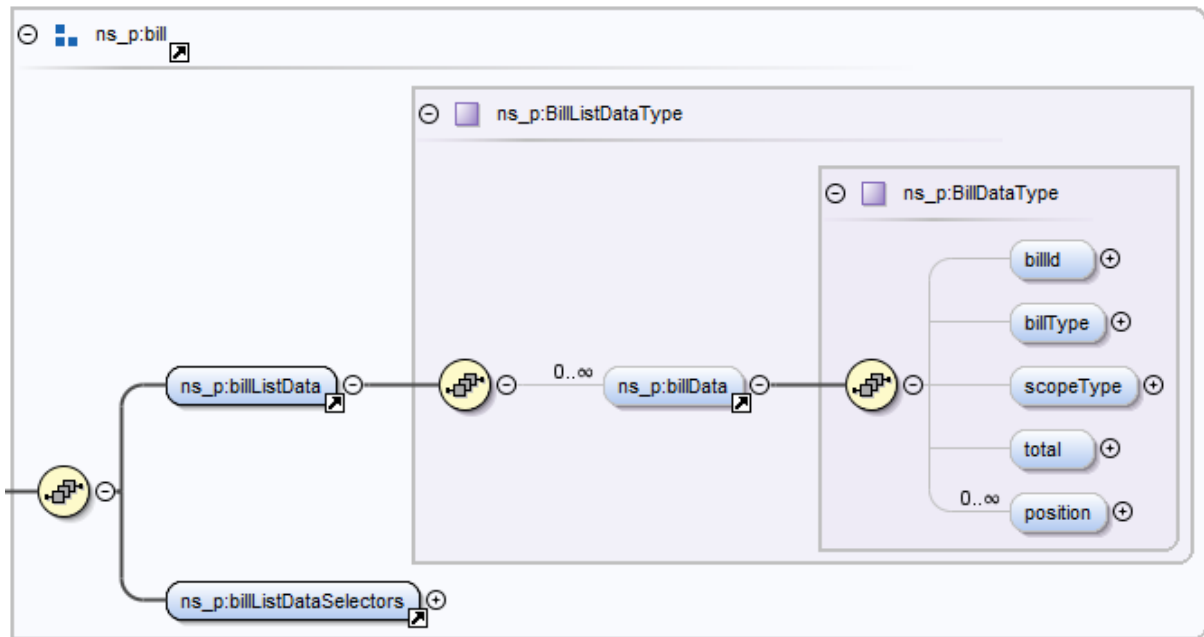
Figure 63: Bill function-group overview

5.3.4.2 billListData

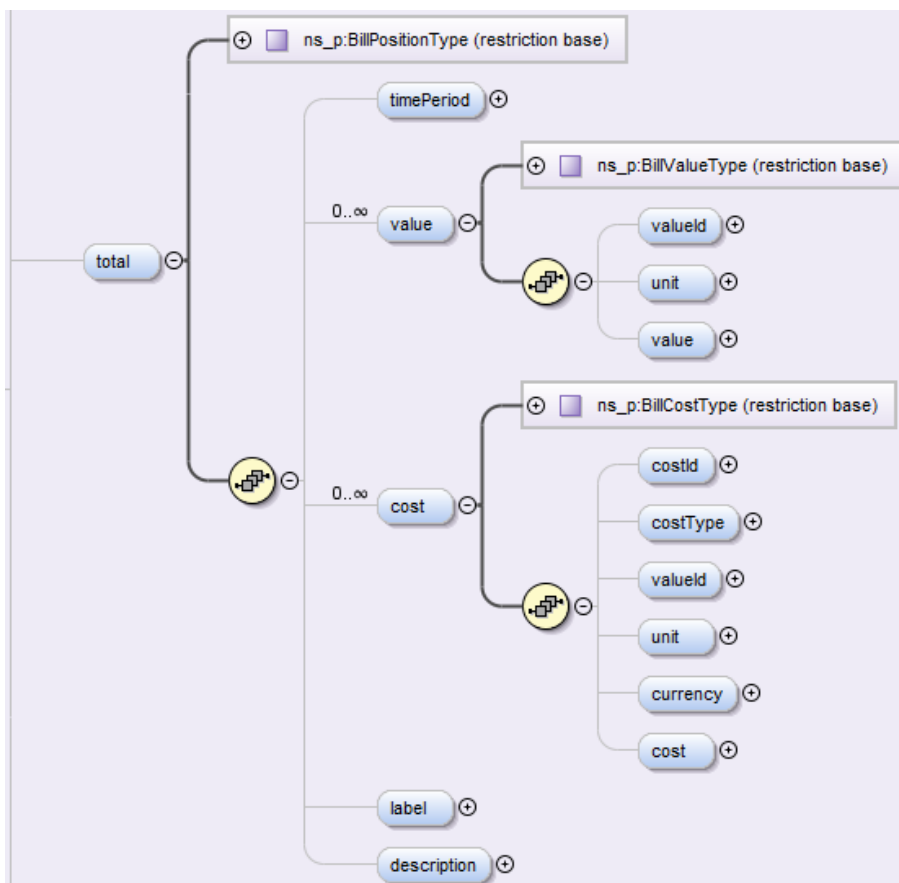
5.3.4.2.1 General

billData holds the bill data, including descriptive elements like currency, unit and billType. This is necessary so the client can write a bill with a single write.

A billData has a "total" element, including total values and corresponding costs in its sub-values. Additionally positions of the total data can be provided, to describe the composition of the total data. To link total costs and values in a position the valuelId and costId are used within a billData. The billId allows to identify the overall bill in other functions of bill as well as outside of the bill resource.



5936

5937 *Figure 64: billListData function overview*

5938

5939 *Figure 65: billListData function detail 1, "total"*

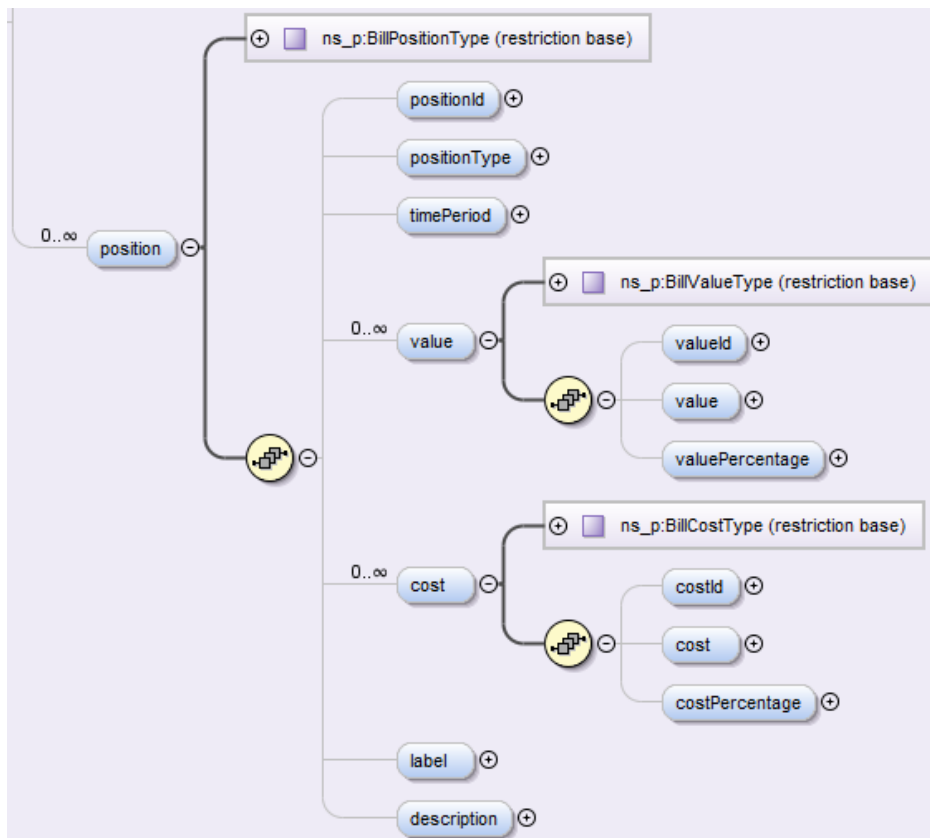


Figure 66: billListData function detail 2, "position"

5.3.4.2.2 Detailed description of elements

Element	Type	Description
billId	Identifier "BillIdType" (see Table 339).	Allows the identification of a bill. Allows linking of the different functions to the same bill. Allows linking of the bill within other features that are placed in the same entity.
billType	Union "BillTypeType": - Enum (see Table 172): BillTypeEnumType - EnumExtendType (see section 3.10.1.5)	The type of the bill.
scopeType	Common data type "ScopeTypeType". See section 3.10.1.21.	Specifies a more detailed meaning of the bill.
total	Restriction of "BillPositionType"	Total information of the bill (in contrast to the positions, see below).
total. timePeriod	Common data type "TimePeriodType". See section 3.10.2.1.	Allows to define a time, or time period for a bill.
total. value (list)	Restriction of "BillValueType"	Different values can be listed for a bill. E.g. energy values or just a number of items.
total. value. valueId	Identifier "BillValueIdType" (see Table 339).	Allows the identification of a value within the bill.

total. value. unit	<i>Common data type "UnitOfMeasurementType". See section 3.10.1.17.</i>	Allows definition of a unit for the value.
total. value. value	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Allows to express the total amount that was bought or sold of a certain commodity during the time or time period given in total.timePeriod.
total. cost (list)	<i>Restriction of "BillCostType"</i>	Different types of costs can be listed for a value. E.g. monetary costs like a price as well as environmental costs as CO2 emission.
total. cost. costId	<i>Identifier "BillCostIdType" (see Table 339).</i>	Allows identification of costs within bill.
total. cost. costType	<i>Union "BillCostTypeType": - Enum (see Table 173): BillCostTypeEnumType - EnumExtendType (see section 3.10.1.5)</i>	Allows to describe the type of costs.
total. cost. valuelId	<i>Identifier "BillValuelIdType" (see Table 339).</i>	Allows to link cost to a value.
total. cost. unit	<i>Common data type "UnitOfMeasurementType". See section 3.10.1.17.</i>	Costs that are not monetary may need to describe a unit (.g. m ³).
total. cost. currency	<i>Common data type "CurrencyType". See section 3.10.1.19.</i>	Monetary costs may need to describe a currency.
total. cost. cost	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Total costs generated during the time or time period given in total.timePeriod.
total. label	<i>Common data type "LabelType". See section 3.10.1.2.</i>	Allows to express a user friendly label for the bill.
total. description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Allows to express a user friendly description for the bill.
position (list)	<i>Restriction of "BillPositionType"</i>	For each total data different positions can be assigned that describe the composition of the total data.
position. positionId	<i>Identifier "BillPositionIdType" (see Table 339).</i>	Identifier of a position within a specific bill.
position. positionType	<i>Union "BillPositionTypeType": - Enum (see Table 174): BillPositionTypeEnumType - EnumExtendType (see section 3.10.1.5)</i>	Allows to describe the type of the a position E.g. different types of energy consumption like self produced or electricity grid energy.
position. timePeriod	<i>Common data type "TimePeriodType". See section 3.10.2.1.</i>	Allow to define a time, or time period for a position if the time or time period is different from the overall time period described in total.
position. value (list)	<i>Restriction of "BillValueType"</i>	Different values can be listed for each position.
position. value. valuelId	<i>Identifier "BillValuelIdType" (see Table 339).</i>	Identifier of a value within a specific bill (see also "total. value. valuelId").

position. value. value	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Allows to describe a value for each position. This can be helpful if some positions have negative while other positions have positive values, as this would can not be express with valuePercentage.
position. value. valuePercentage	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Percentage of the total value (see "total").
position. cost (list)	<i>Restriction of "BillCostType"</i>	Different costs can be listed for each position
position. cost. costId	<i>Identifier "BillCostIdType" (see Table 339).</i>	Identifier of a cost within a specific bill (see also "total. cost. costId").
position. cost. cost	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	The actual costs for this position. This can be helpful if some positions have negative while other positions have positive costs, as this can not be expressed with costPercentage.
position. cost. costPercentage	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Percentage of the total costs (see "total").
position. label	<i>Common data type "LabelType". See section 3.10.1.2.</i>	Allows to provide a user friendly label for each bill position.
position. description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Allows to provide user friendly information for each bill position.

5944 Table 171: billListData function detailed description of elements

5945 Enumeration **BillTypeEnumType**:

Value	Description
chargingSummary	This bill type allows to summarize a charging cycle. In most cases it is used to communicate the total energy that was charged and the total costs, as well as different positions for different types of energy. The summary may also be sent during charging, also multiple times. This allows to keep the customer updated during the charging process.

5946 Table 172: Enumeration BillTypeEnumType

5947 Enumeration **BillCostTypeEnumType**:

Value	Description
absolutePrice	Used for absolute prices.
relativePrice	Used for relative prices, e.g. a price that shows how much cheaper the costs are related to the average tariff in place.
co2Emission	In this case the cost value is used to express how much CO2 emission was emitted.
renewableEnergy	In this case the cost value is used to express how much renewable energy was used.
radioactiveWaste	In this case the cost value is used to express how much radioactive waste was generated.

5948 Table 173: Enumeration BillCostTypeEnumType

5949 Enumeration **BillPositionTypeEnumType**:

Value	Description
-------	-------------

gridElectricEnergy	This position type specifically describes the electricity grid energy portion of the total energy.
selfProducedElectricEnergy	This position type specifically describes the self produced energy portion of the total energy.

Table 174: Enumeration BillPositionTypeEnumType

5.3.4.2.3 Available selectors

- billId
- scopeType

5.3.4.2.4 Examples

5.3.4.2.4.1 Notify

When there is an update for a specific bill, a server would send a notification to all subscribed clients that could look like this:

```

...
<billListData>
  <billData>
    <billId>1</billId>
    <billType>chargingSummary</billType>
    <total>
      <timePeriod>
        <startTime>2017-10-24T13:14:31.0Z</startTime>
        <endTime>2017-10-24T14:11:43.0Z</endTime>
      </timePeriod>
      <value>
        <valueId>1</valueId>
        <unit>Wh</unit>
        <value>
          <number>35</number>
          <scale>3</scale>
        </value>
      </value>
      <cost>
        <costId>1</costId>
        <costType>absolutePrice</costType>
        <valueId>1</valueId>
        <currency>EUR</currency>
        <cost>
          <number>96</number>
          <scale>-1</scale>
        </cost>
      </cost>
    </total>
    <position>
      <positionId>1</positionId>
      <positionType>gridElectricEnergy</positionType>
      <timePeriod>
        <startTime>2017-10-24T13:14:31.0Z</startTime>
        <endTime>2017-10-24T14:11:43.0Z</endTime>
      </timePeriod>
      <value>
        <valueId>1</valueId>
        <valuePercentage>

```

```

5999         <number>73</number>
6000         <scale>0</scale>
6001     </valuePercentage>
6002 </value>
6003 <cost>
6004     <costId>1</costId>
6005     <costPercentage>
6006         <number>95</number>
6007         <scale>0</scale>
6008     </costPercentage>
6009 </cost>
6010 </position>
6011 <position>
6012     <positionId>2</positionId>
6013     <positionType>selfProducedElectricEnergy</positionType>
6014     <timePeriod>
6015         <startTime>2017-10-24T13:14:31.0Z</startTime>
6016         <endTime>2017-10-24T14:11:43.0Z</endTime>
6017     </timePeriod>
6018     <value>
6019         <valueId>1</valueId>
6020         <valuePercentage>
6021             <number>27</number>
6022             <scale>0</scale>
6023         </valuePercentage>
6024     </value>
6025     <cost>
6026         <costId>1</costId>
6027         <costPercentage>
6028             <number>5</number>
6029             <scale>0</scale>
6030         </costPercentage>
6031     </cost>
6032 </position>
6033 </billData>
6034 </billListData>

```

5.3.4.3 *billConstraintsListData*

5.3.4.3.1 *General*

The billConstraintsListData function allows a server to define certain constraints for each linked billListData list entry. This is especially important if a particular billListData list entry is writeable and allows the server to communicate the corresponding constraints before a client performs a write. The client can match the server constraints with own constraints and perform a corresponding write command.

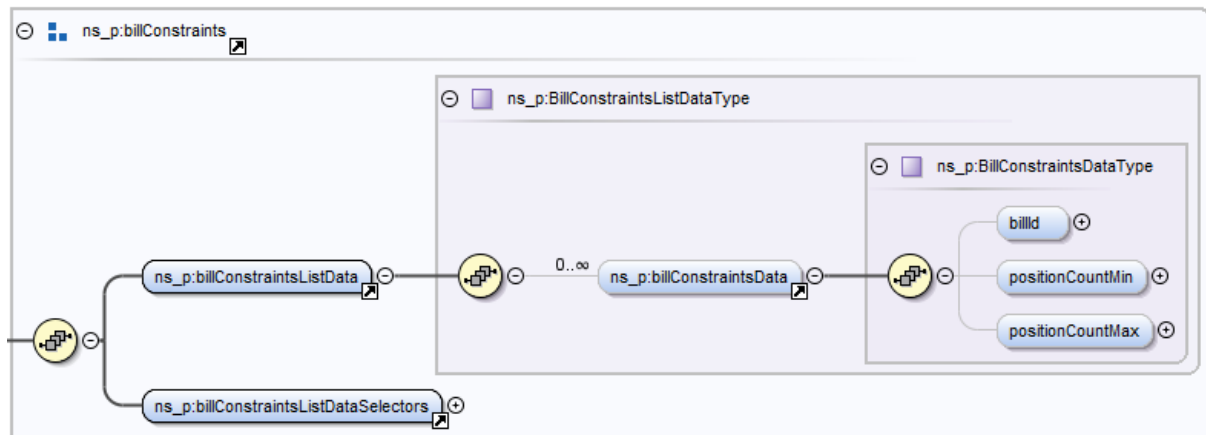


Figure 67: billConstraintsListData function overview

5.3.4.3.2 Detailed description of elements

Element	Type	Description
billId	Identifier "BillIdType" (see Table 339).	Allows the identification of a bill. Allows linking of the different functions to the same bill. Allows linking of the bill within other features that are placed in the same entity.
positionCountMin	Simple type "BillPositionCountType" (restriction of "BillPositionIdType" (see Table 339)).	Minimum amount of possible positions within the specified bill.
positionCountMax	Simple type "BillPositionCountType" (restriction of "BillPositionIdType" (see Table 339)).	Maximum amount of possible positions within the specified bill.

Table 175: billConstraintsListData function detailed description of elements

5.3.4.3.3 Available selectors

- billId

5.3.4.3.4 Examples

5.3.4.3.4.1 Read-reply

If one wants to know the constraints of a specific bill, he could send the following read command:

```

<function>billConstraintsListData</function>
<filter>
  <cmdControl>
    <partial/>
  </cmdControl>
  <billConstraintsListDataSelectors>
    <billId>1</billId>
  </billConstraintsListDataSelectors>
</filter>
<billConstraintsListData/>

```

6065 The reply could look like this:

```

6066 <function>billConstraintsListData</function>
6067 <filter>
6068   <cmdControl>
6069     <partial/>
6070   </cmdControl>
6071 </filter>
6072 <billConstraintsListData>
6073   <billConstraintsData>
6074     <billId>1</billId>
6075     <positionCountMin>2</positionCountMin>
6076     <positionCountMax>4</positionCountMax>
6077   </billConstraintsData>
6078 </billConstraintsListData>

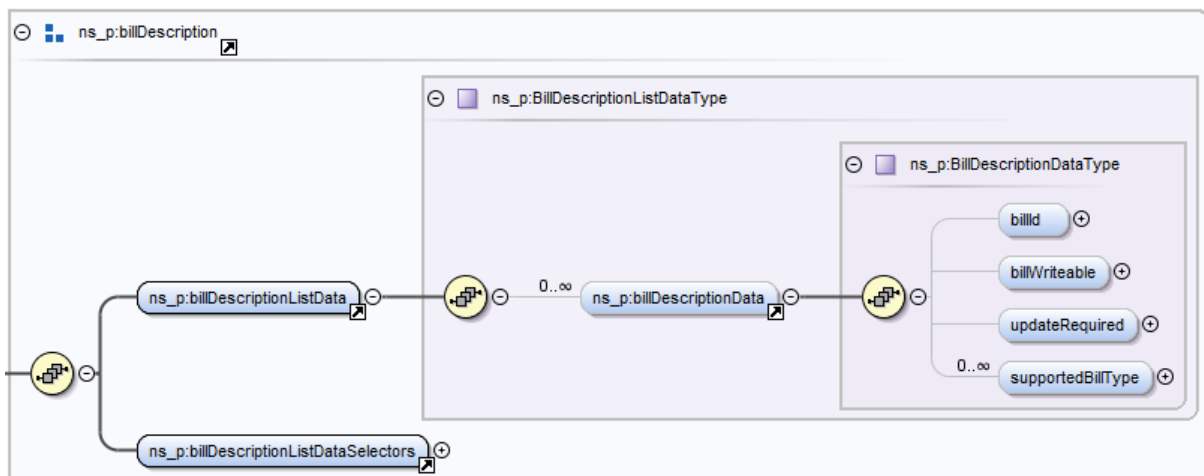
```

6079

6080 5.3.4.4 *billDescriptionListData*

6081 5.3.4.4.1 *General*

6082 The billDescriptionListData function is especially important if a linked billListData list entry is
 6083 writeable and allows a server to define if a bill is writeable and which billTypes are supported for the
 6084 corresponding billId, so that a client can perform corresponding writes. Additionally, the server can
 6085 request updates from a client with the updateRequired flag.



6086

6087 Figure 68: billDescriptionListData function overview

6088

6089 5.3.4.4.2 *Detailed description of elements*

Element	Type	Description
billId	Identifier "BillIdType" (see Table 339).	Allows the identification of a bill. Allows linking of the different functions to the same bill. Allows linking of the bill within other features that are placed in the same entity.
billWriteable	xs:boolean (W3C standard type)	Indicates whether the linked billData entry is writeable.
updateRequired	xs:boolean (W3C standard type)	Indicates whether the server requests an update for the bill.

supportedBillType (list)	Union "BillTypeType": - Enum (see Table 172): BillTypeEnumType - EnumExtendType (see section 3.10.1.5)	List of all supported types for this billId.
-----------------------------	---	---

Table 176: billDescriptionListData function detailed description of elements

5.3.4.4.3 Available selectors

- billId

5.3.4.4.4 Examples

5.3.4.4.4.1 Read-reply

If one is interested in the descriptions of all bills, he could send a standard read command to the server. The reply could look like this:

```
<billDescriptionListData>
  <billDescriptionData>
    <billId>1</billId>
    <billWriteable>true</billWriteable>
    <supportedBillType>chargingSummary</supportedBillType>
  </billDescriptionData>
</billDescriptionListData>
```

5.3.5 BindingManagement

5.3.5.1 Introduction

Bindings are used to simplify communication between devices. Based on matching features, finding suitable communication partners is a lot easier than via manual configuration only.

Each binding exists between a binding client and a binding server. Only clients may initiate new bindings. After a successful establishment of a binding, the binding client may do actions on the binding server, depending on the bound functionality. E.g. a binding server accepts load management configurations by the client and updates the client when changes in its parameters occur. The exact definition of rules and proceedings is out of scope of this documentation.

The *BindingManagement* class provides functions for establishing, reading and deleting bindings.



Figure 69: BindingManagement function-group overview (data)

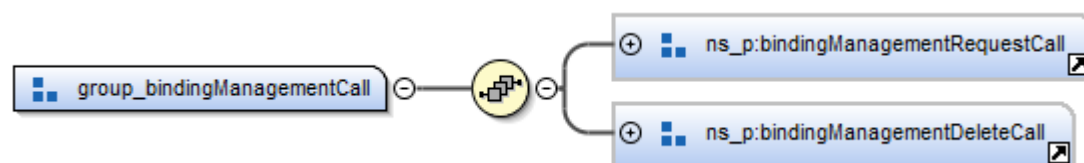
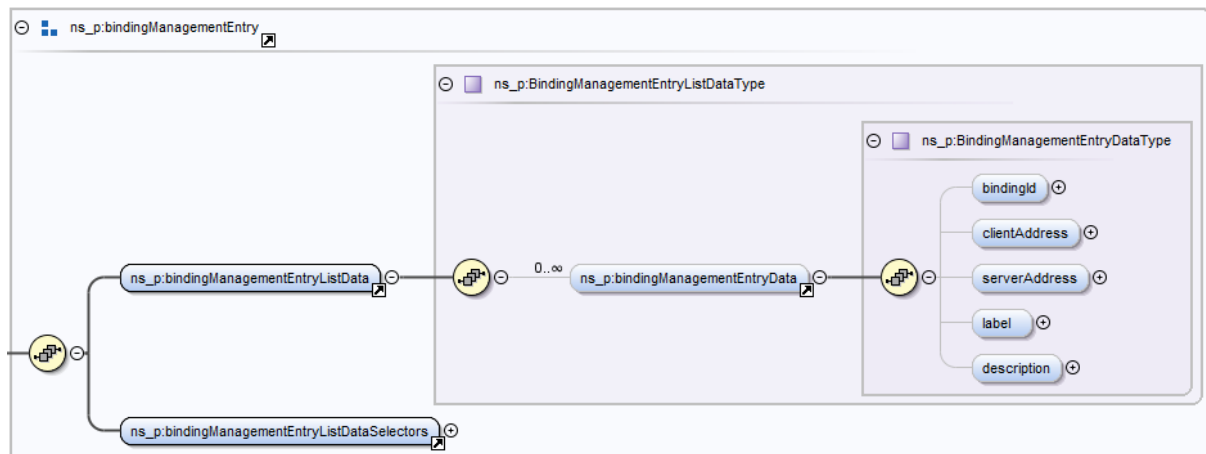


Figure 70: BindingManagement function-group overview (call)

6121

6122 **5.3.5.2 bindingManagementEntryListData**6123 **5.3.5.2.1 General**

6124 All existing bindings can be listed in the *bindingManagementEntryListData* function. It may be used
 6125 to store the entries in the device itself. If another device asks for the list of bindings, only the
 6126 bindings with the requesting device should be replied.



6127

6128 Figure 71: bindingManagementEntryListData function overview

6129

6130 **5.3.5.2.2 Detailed description of elements**

Element	Type	Description
bindingId	Identifier "BindingIdType" (see Table 339).	Identifier of the binding.
clientAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>client</i> in this binding.
serverAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>server</i> in this binding.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the binding.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the binding.

6131 Table 177: bindingManagementEntryListData function detailed description of elements

6132

6133 **5.3.5.2.3 Available selectors**

- 6134 - bindingId
- 6135 - clientAddress
- 6136 - serverAddress

6137

6138 **5.3.5.2.4 Examples**6139 **5.3.5.2.4.1 Read-reply**

6140 If one is interested in all bindings that are active between the own device and another one, he could
 6141 send a standard read function to the device, which could reply something like this:

```

6142 <bindingManagementEntryListData>
6143   <bindingManagementEntryData>
6144     <bindingId>0</bindingId>
6145     <clientAddress>
6146       <device>d:_i:46925_CP\_pqtl-27638162683172637812356813</device>
6147       <entity>2</entity>
6148       <feature>1</feature>
6149     </clientAddress>
6150     <serverAddress>
6151       <device>d:_i:46925_CP\_arqf-64986751356897215468975468</device>
6152       <entity>1</entity>
6153       <feature>1</feature>
6154     </serverAddress>
6155     <label>SwitchLight_01</label>
6156   </bindingManagementEntryData>
6157   <bindingManagementEntryData>
6158     <bindingId>4</bindingId>
6159     <clientAddress>
6160       <device>d:_i:46925_CP\_pqtl-27638162683172637812356813</device>
6161       <entity>2</entity>
6162       <feature>4</feature>
6163     </clientAddress>
6164     <serverAddress>
6165       <device>d:_i:46925_CP\_arqf-64986751356897215468975468</device>
6166       <entity>1</entity>
6167       <feature>2</feature>
6168     </serverAddress>
6169     <label>SwitchLight_02</label>
6170   </bindingManagementEntryData>
6171 </bindingManagementEntryListData>

```

6172

6173 **5.3.5.3 bindingManagementRequestCall**6174 **5.3.5.3.1 General**

6175 To initiate a new binding, an entity capable of node management, sends a
 6176 *bindingManagementRequestCall* to an entity (on another device that should become the binding
 6177 partner) capable of node management, too, with the payload element *clientAddress* set to one of its
 6178 features with the role *client* and the payload element *serverAddress* set to one of the other device's
 6179 feature address with the role *server*. The receiving device checks whether it will accept or decline the
 6180 request and responds with an acknowledgement message (see [ProtocolSpecification], section
 6181 "Acknowledgement message") with the *resultData* function and an according *errorNumber* set to 0
 6182 (accept binding request) or some other value (definition of the error code is out of scope of this
 6183 documentation; decline binding request).

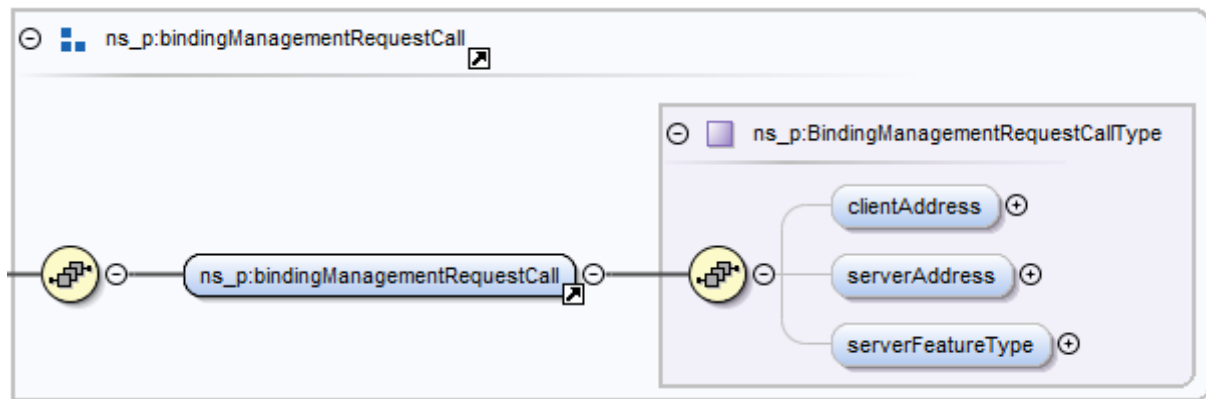


Figure 72: bindingManagementRequestCall function overview

5.3.5.3.2 Detailed description of elements

Element	Type	Description
clientAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>client</i> in the binding request.
serverAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>server</i> in the binding request.
serverFeatureType	Common data type "FeatureTypeType". See section 3.10.1.29.	The <i>Feature Type</i> of the server feature. Stated for verification.

Table 178: bindingManagementRequestCall function detailed description of elements

5.3.5.3.3 Available selectors

None.

5.3.5.3.4 Examples

5.3.5.3.4.1 Call

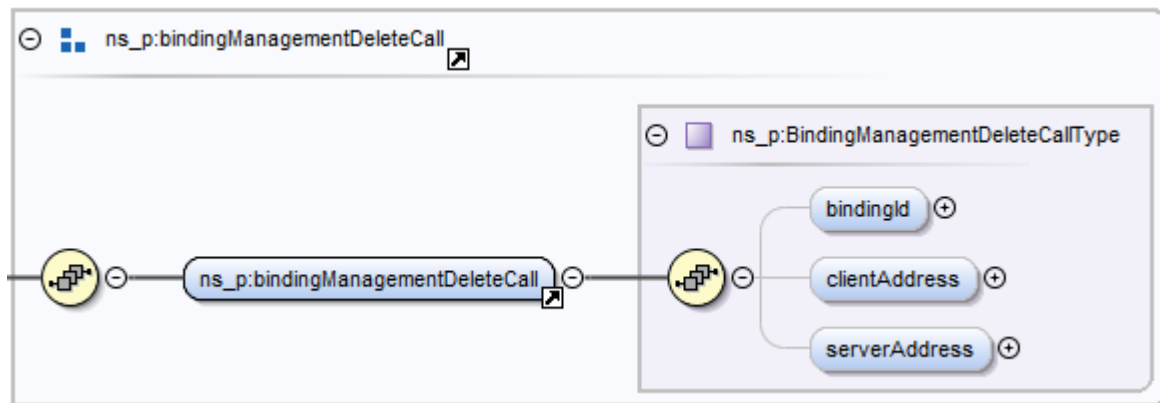
If a new binding shall be initiated, the client device could send a command like this to the server device:

```
<bindingManagementRequestCall>
  <clientAddress>
    <device>d:_i:46925_CP\_pqt1-27638162683172637812356813</device>
    <entity>2</entity>
    <feature>1</feature>
  </clientAddress>
  <serverAddress>
    <device>d:_i:46925_CP\_arqf-64986751356897215468975468</device>
    <entity>1</entity>
    <feature>1</feature>
  </serverAddress>
  <serverFeatureType>ActuatorSwitch</serverFeatureType>
</bindingManagementRequestCall>
```

6210

6211 **5.3.5.4 bindingManagementDeleteCall**6212 **5.3.5.4.1 General**

6213 A device may delete an existing binding with itself and another device. Therefore, this call is used. It
 6214 may be used by clients and servers.



6215

6216 *Figure 73: bindingManagementDeleteCall function overview*

6217

6218 **5.3.5.4.2 Detailed description of elements**

Element	Type	Description
bindingId	Identifier "BindingIdType" (see Table 339).	Identifier of the binding.
clientAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>client</i> in this binding.
serverAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>server</i> in this binding.

6219 *Table 179: bindingManagementDeleteCall function detailed description of elements*

6220

6221 **5.3.5.4.3 Available selectors**

6222 None.

6223

6224 **5.3.5.4.4 Examples**6225 **5.3.5.4.4.1 Call**

6226 If an existing binding (with the bindingId "2") shall be deleted, one device could send a command like
 6227 this to the other device:

```

6228 <bindingManagementDeleteCall>
6229   <bindingId>2</bindingId>
6230   <clientAddress>
6231     <device>d:_i:46925_CP\_pqt1-27638162683172637812356813</device>
6232   </entity>2</entity>
  
```

```

6233         <feature>1</feature>
6234     </clientAddress>
6235     <serverAddress>
6236         <device>d:_i:46925_CP\_arqf-64986751356897215468975468</device>
6237         <entity>1</entity>
6238         <feature>1</feature>
6239     </serverAddress>
6240 </bindingManagementDeleteCall>

```

5.3.6 DataTunneling

5.3.6.1 Introduction

DataTunneling provides a mechanism to submit data of proprietary protocols using the SPINE data models (esp. addressing other nodes). However, the nature of proprietary definitions imposes restrictions:

In the example below an XML makes use of SPINE dataTunneling sub-elements "purposeId", "channelId", "sequenceId", and "payload". Unless stated otherwise, the term "payload" refers to the sub-element of "dataTunnelingCall" within this section (in contrast to the definition of "payload" as sub-element of "datagram" as defined in document [ProtocolSpecification]).

Please note that the use of the above mentioned elements will always be proprietary. The SPINE specification will never define any specific protocol for dataTunneling.

To put it in other words, the SPINE data tunnelling model is not intended for any kind of flexible extension of the SPINE classes. This is mainly because a so-called technology specific mapping could hardly be defined in a standardized way. Another reason is that it shall always be transparent that data tunnelling contains proprietary data. This should not be considered a limitation of SPINE as the EEBus Initiative e.V. committees may define specific classes with functions (like messagingListData, timeInformationData, e.g.) including mappings for specific protocols, if need be.



Figure 74: DataTunneling function-group overview (call)

5.3.6.2 dataTunnelingCall

5.3.6.2.1 General

In general, several conditions have to be fulfilled in order to make use of SPINE data tunnelling. This shall be explained with an example:

A manufacturer "X" of a device "A" created a proprietary extension of a communications protocol (e.g. KNX, ZigBee, LON, ... – we call it "CP" here) in order to configure device "A" with special parameters. We assume there is no specific SPINE function defined so far for these parameters. There is also no mapping defined for it.

6270 Another manufacturer "Y" creates a device "B" that runs "SPINE applications" (i.e. applications
6271 communicating with an internal interface using XMLs of the SPINE data models). The device "B" can
6272 as well communicate via "CP".

6273 Now, manufacturer "Y" wants to make use of the proprietary configuration of device "A". Therefore,
6274 manufacturer "Y" needs to implement the proprietary extension of manufacturer "X"'s
6275 communications protocol as well. I.e. the "CP" implementation of device "B" has to be extended
6276 properly. Furthermore, manufacturer "Y" needs to define how the proprietary protocol shall be
6277 represented as SPINE data tunnelling towards the "SPINE applications" running on device "B" (i.e.
6278 define the proprietary mapping) and of course needs to implement this mapping. Finally, the
6279 applications need to implement the recognition and use of the corresponding SPINE data tunnelling
6280 XMLs.

6281 The scenario above may look "unattractive" or even raise concerns. However, it shall be mentioned
6282 that proprietary extensions of standardized protocols are not unusual. This practice shall not be
6283 judged here.

6284 In the aforementioned scenario it is manufacturer "B" who implements a number of extensions in
6285 order to comply with the extensions of device "A". But it could also be possible that manufacturer
6286 "A" already provided some definitions and implementation for other manufacturers. Then,
6287 manufacturer "B" could integrate this solution and potentially implements less by its own.

6288 Some may find a comparison with Virtual Private Networks helpful (though VPNs have a different
6289 scope): Two computers "A", "B" each have to run a compatible (proprietary) program (i.e.
6290 mapping/implementation) that know how to communicate with each other (i.e. set up the
6291 communication and provide a specific authentication and encryption). There are manufacturers of
6292 VPN gateways that provide client software for different operating systems. This way a proprietary
6293 implementation is made available for others.

6294 The combination { `purposeId`, `channelId`, `sequenceId` } should be considered unique for a
6295 communication from "A" to "B". Within a system (e.g. house) several communications may be used
6296 in parallel. The combinations used in these communications are in general independent of each
6297 other.

6298 Please note that an instance of a `dataTunnelingCall` implementation permits bidirectional
6299 communication in general. I.e. if a `dataTunnelingCall` functionality of "B" submits a message to a
6300 `dataTunnelingCall` functionality of "A", it is also permissive that the `dataTunnelingCall` functionality of
6301 "A" submits `dataTunnelingCall` messages to the `dataTunnelingCall` functionality of "B". However,
6302 similar to a VPN, it may well be that the establishment for such message exchanges is
6303 "unidirectional" to some extent. I.e. it may be that the `dataTunnelingCall` functionality of "A"
6304 determines which kind of messages (or permissive values for "`purposeId`") are accepted on this
6305 functionality and that the functionality first of all acts as some kind of "listening port". This means
6306 the functionality of "A" may wait for an initial incoming message before it also sends
6307 `dataTunnelingCall` messages back.

6308 Please note that the example above does not prevent device "B" from implementing another
6309 `dataTunnelingCall` functionality as "listening port" with own permitted values for "`purposeId`" as well.
6310 In that case device "A" could send a proper initial message to this "listening functionality" of "B".

SPINE data tunnelling is intended to be used for call-operations only. No read-reply or write procedures are supported.

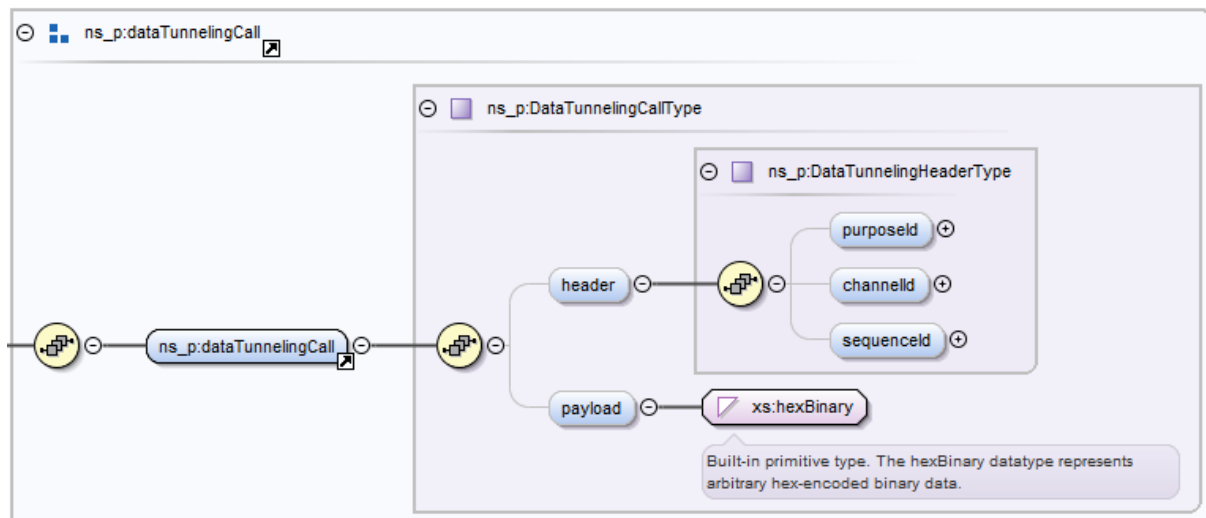


Figure 75: dataTunnelingCall function overview

5.3.6.2.2 Detailed description of elements

Element	Type	Description
header	Complex type "DataTunnelingHeaderType"	A data tunneling specific header.
header. purposeld	Simple type "PurposeldType" (restriction of xs:string)	A string that identifies the purpose / kind of tunnelled data / protocol.
header. channelId	Simple type "ChannelIdType" (restriction of xs:unsignedInt)	Between two nodes there might be several channels used in parallel using the same purposeld. channelId helps to separate these channels. It can also be used to differentiate subsequent (rather than parallel) sessions.
header. sequenceld	xs:unsignedInt (W3C standard type)	This helps to give the payload for each combination of { purposeld, channelId } a chronological order. Note: This "sequenceld" is data tunneling specific and should not be confused with "sequenceld" elements of other classes!
payload	xs:hexBinary (W3C standard type)	The actual proprietary data.

Table 180: dataTunnelingData function detailed description of elements

5.3.6.2.3 Available selectors

None.

5.3.6.2.4 Examples

5.3.6.2.4.1 Call

Device "B" is configured to submit certain proprietary data to device "A". Due to a specific occasion it sends a call classifier together with the following XML:

```
<dataTunnelingCall>
  <header>
    <purposeId>FooManufacturer:BarProtocolWithVersion</purposeId>
    <channelId>7</channelId>
    <sequenceId>6425</sequenceId>
  </header>
  <payload>3c3f786d6c2076657273696f6e3d22312e302220656e636f64696e673d22555446
2d38223f3e0d0a3c6d6573736167654461746120786d6c6e733d22687474703a2f2f646f637
32e65656275732e6f72672f62657461312f312e302f787364223e0d0a202020203c74696d65
7374616d703e323030362d30352d30345431383a31333a35312e305a3c2f74696d657374616
d703e0d0a202020203c6d6573736167654e756d6265723e37323c2f6d6573736167654e756d
6265723e0d0a202020203c747970653e696e666f726d6174696f6e3c2f747970653e0d0a202
020203c746578743e55706461746520636f6d706c657465642e3c2f746578743e0d0a3c2f6d
657373616765446174613e0d0a</payload>
</dataTunnelingCall>
```

5.3.7 DeviceClassification

5.3.7.1 Introduction

Each device can have associated information useful for its identification and which kind of power it requires. This information (mostly plaintext) is bundled in the *DeviceClassification* class. It is intended to describe the physical device, but may be used for logical devices (= *entities*), too.

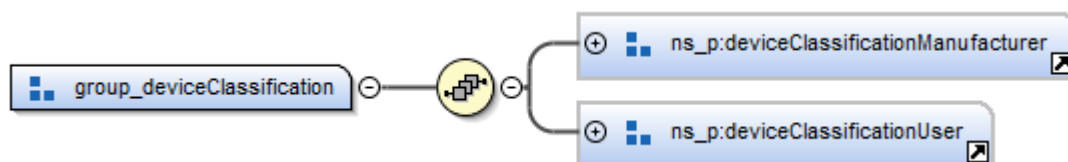


Figure 76: DeviceClassification function-group overview

5.3.7.2 deviceClassificationManufacturerData

5.3.7.2.1 General

Information which is usually NOT writable (hard coded in the device or preconfigured information, e.g.), is modelled with *deviceClassificationManufacturerData*.

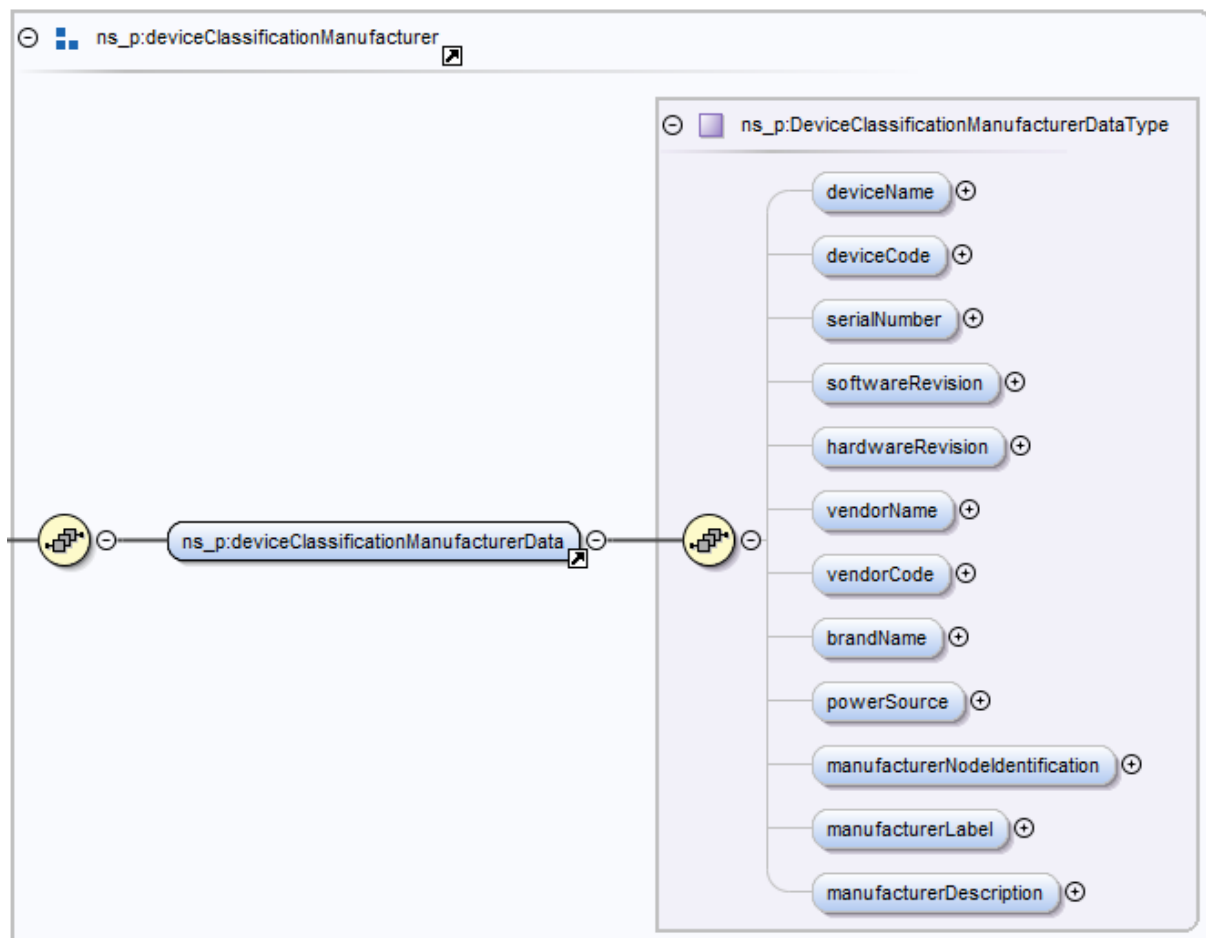


Figure 77: deviceClassificationManufacturerData function overview

5.3.7.2.2 Detailed description of elements

Please note that most of the subsequent elements permit any content. I.e. the underlying SPINE data model usually does not impose any format for the content of these elements. An exception applies if an enumeration simple-type is used (see *powerSource* element).

Element	Type	Description
deviceName	Simple type "DeviceClassificationStringType" (restriction of xs:string)	The name of the (physical or logical) device as defined by the manufacturer.
deviceCode	Simple type "DeviceClassificationStringType" (restriction of xs:string)	A device code for the (physical or logical) device as defined by the manufacturer.
serialNumber	Simple type "DeviceClassificationStringType" (restriction of xs:string)	The serial number of the (physical or logical) device as defined by the manufacturer. Usually the same as printed on the case.
softwareRevision	Simple type "DeviceClassificationStringType" (restriction of xs:string)	The software revision of the (physical or logical) device as

		defined by the manufacturer.
hardwareRevision	<i>Simple type</i> <i>"DeviceClassificationStringType"</i> <i>(restriction of xs:string)</i>	The hardware revision of the (physical or logical) device as defined by the manufacturer.
vendorName	<i>Simple type</i> <i>"DeviceClassificationStringType"</i> <i>(restriction of xs:string)</i>	The name of the vendor of the (physical or logical) device as defined by the manufacturer.
vendorCode	<i>Simple type</i> <i>"DeviceClassificationStringType"</i> <i>(restriction of xs:string)</i>	A code for the vendor of the (physical or logical) device as defined by the manufacturer.
brandName	<i>Simple type</i> <i>"DeviceClassificationStringType"</i> <i>(restriction of xs:string)</i>	The name of the brand. Useful where the name of the brand and the vendor differs.
powerSource	<i>Union "PowerSourceType":</i> <i>- Enum (see Table 182):</i> <i>PowerSourceEnumType</i> <i>- EnumExtendType (see section 3.10.1.5)</i>	This element describes for which kind of primary power source the device is designed for. It is only used for informative purposes. It is NOT used to describe how or to which kind of power source the device is finally connected (this means it is not considered here whether the device's single mains is fed by a house's mains supply or a backup system, e.g.).
manufacturerNodeIdentification	<i>Simple type</i> <i>"DeviceClassificationStringType"</i> <i>(restriction of xs:string)</i>	A node identification for the (physical or logical) device as defined by the manufacturer. This could be used for the identification of a (physical or logical) device, even if it was removed from the network and rejoined later with changed node address.
manufacturerLabel	<i>Common data type</i> <i>"LabelType". See section 3.10.1.2.</i>	A short label of the (physical or logical) device as defined by the manufacturer.
manufacturerDescription	<i>Common data type</i> <i>"DescriptionType". See section 3.10.1.3.</i>	A description for the (physical or logical) device as defined by the manufacturer.

Table 181: deviceClassificationManufacturerData function detailed description of elements

Enumeration **PowerSourceEnumType**:

Value	Description
unknown	Power source not known.
mainsSinglePhase	AC power connection, 1 phase.
mains3Phase	AC power connection, 3 phases.
battery	DC powered with a battery.
dc	DC powered with a constant energy source.

Table 182: Enumeration PowerSourceEnumType

5.3.7.2.3 Available selectors

None.

5.3.7.2.4 Examples

5.3.7.2.4.1 Read-reply

An example of a *deviceClassificationManufacturerData* instance, send as reply:

```
<deviceClassificationManufacturerData>
  <deviceName>SPINETestDevice_A</deviceName>
  <deviceCode>SPINE_TD_A</deviceCode>
  <serialNumber>159753852456</serialNumber>
  <softwareRevision>1.2.0</softwareRevision>
  <hardwareRevision>1.0.1</hardwareRevision>
  <vendorName>EEBus Initiative e.V.</vendorName>
  <vendorCode>4545427573</vendorCode>
  <brandName>TestBrand</brandName>
  <powerSource>mainsSinglePhase</powerSource>
<manufacturerNodeIdentification>SPINE_TD_A_1.2.0_159753852456_A</manufactur
erNodeIdentification>
  <manufacturerLabel>SPINE_TD_A@EEBus_e.V.</manufacturerLabel>
  <manufacturerDescription>SPINETestDevice_A Installation @ EEBus
Initiative e.V.</manufacturerDescription>
</deviceClassificationManufacturerData>
```

5.3.7.3 deviceClassificationUserData

5.3.7.3.1 General

Information which is usually writable (changeable by a user / application) is modelled with *deviceClassificationUserData*.

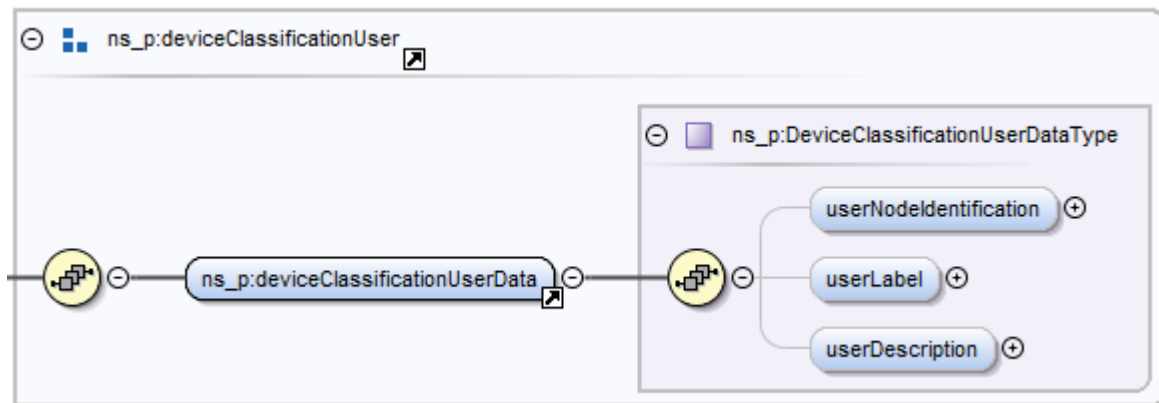


Figure 78: deviceClassificationUserData function overview

5.3.7.3.2 Detailed description of elements

Element	Type	Description
userNodeIdentification	Simple type "DeviceClassificationStringType" (restriction of xs:string)	Similar to <i>manufacturerNodeIdentification</i> but defined by a user or application.
userLabel	Common data type "LabelType". See section 3.10.1.2.	Similar to <i>manufacturerLabel</i> but defined by a user or application.
userDescription	Common data type "DescriptionType". See section 3.10.1.3.	Similar to <i>manufacturerDescription</i> but defined by a user or application.

Table 183: deviceClassificationUserData function detailed description of elements

5.3.7.3.3 Available selectors

None.

5.3.7.3.4 Examples

5.3.7.3.4.1 Read-reply

An example of a *deviceClassificationUserData* instance, send as reply:

```
<deviceClassificationUserData>
  <userNodeIdentification>TD_A_159753852456</userNodeIdentification>
  <userLabel>TD_A of EEBus e.V.</userLabel>
  <userDescription>Test Device A; Install Date:
15.07.13</userDescription>
</deviceClassificationUserData>
```

5.3.8 DeviceConfiguration

5.3.8.1 Introduction

Some information cannot be represented by the "normal" SPINE classes, but are important for the interoperable functionality of a device, hence should not be modelled with the DataTunneling class

(see section 5.3.6) or other proprietary ways. For this purpose, the DeviceConfiguration class should be used. It provides a key-value based modelling of information. Some data, important for interoperability, is already defined by the EEBus Initiative e.V. and listed in Table 187.

The information is split into a rather static part (see section 5.3.8.3) and the actual value (see section 5.3.8.2).

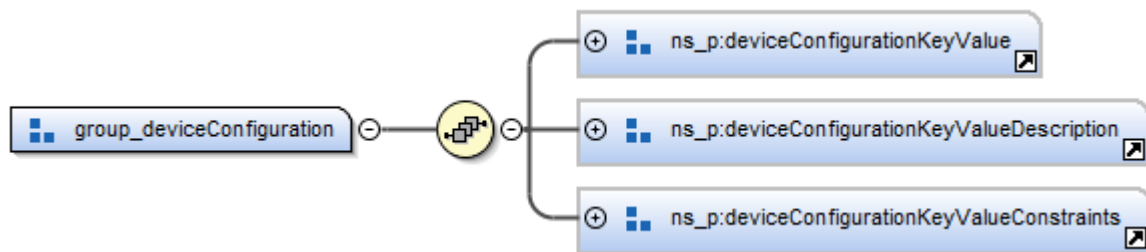


Figure 79: DeviceConfiguration function-group overview

5.3.8.2 deviceConfigurationKeyValueListData

5.3.8.2.1 General

The actual values belonging to a specific key are modelled within this function.

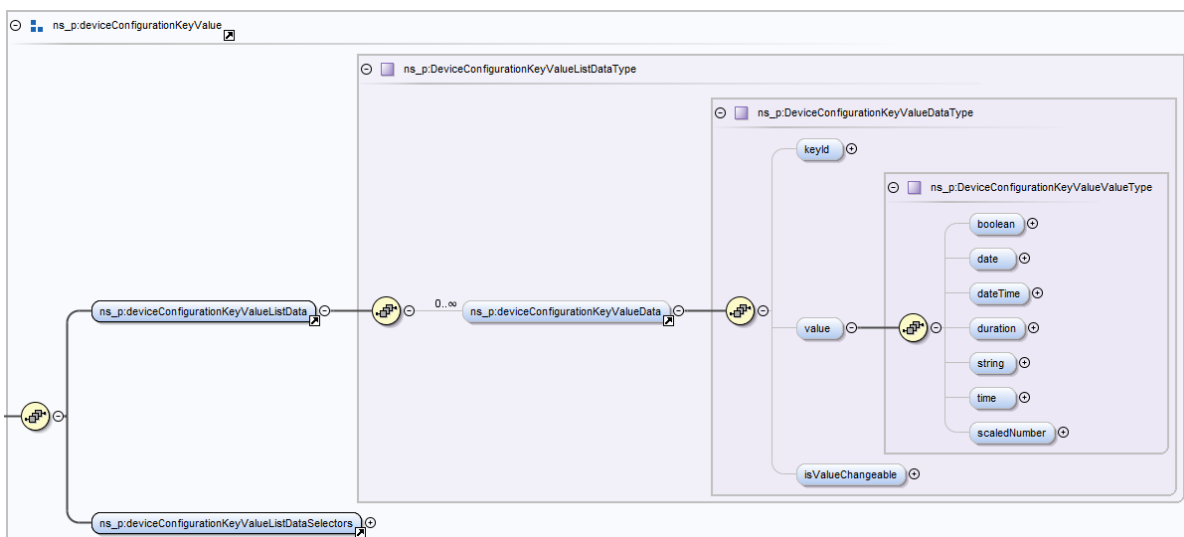


Figure 80: deviceConfigurationKeyValueListData function overview

5.3.8.2.2 Detailed description of elements

Element	Type	Description
keyId	Identifier "DeviceConfigurationKeyIdType" (see Table 339).	Enables the identification of different keys on one SPINE feature.
value	Complex type "DeviceConfigurationKeyValueValueType". See Table 185.	The actual value belonging to the keyId.

isValueChangeable	xs:boolean (W3C standard type)	States whether the value belonging to the <i>keyId</i> is changeable by a client or not.
-------------------	--------------------------------	--

6430 Table 184: deviceConfigurationKeyValueListData function detailed description of elements

6431 Complex type **DeviceConfigurationKeyValueValueType**:

Element	Type	Description
boolean	xs:boolean (W3C standard type)	Used, if the <i>value</i> is of type <i>boolean</i> .
date	xs:date (W3C standard type)	Used, if the <i>value</i> is of type <i>date</i> .
dateTime	xs:dateTime (W3C standard type)	Used, if the <i>value</i> is of type <i>dateTime</i> .
duration	xs:duration (W3C standard type)	Used, if the <i>value</i> is of type <i>duration</i> .
string	Simple type "DeviceConfigurationKeyValueStringType" (restriction of xs:string)	Used, if the <i>value</i> is of type <i>string</i> .
time	xs:time (W3C standard type)	Used, if the <i>value</i> is of type <i>time</i> .
scaledNumber	Common data type "ScaledNumberType". See section 3.10.1.8.	Used, if the <i>value</i> is of type <i>scaledNumber</i> .

6432 Table 185: Complex type DeviceConfigurationKeyValueValueType

6433

6434 5.3.8.2.3 Available selectors

6435 - keyId

6436

6437 5.3.8.2.4 Examples

6438 5.3.8.2.4.1 Read-reply

6439 If one is interested in the value of a specific key, he could send the following read command:

```

6440 <function>deviceConfigurationKeyValueListData</function>
6441 <filter>
6442   <cmdControl>
6443     <partial/>
6444   </cmdControl>
6445   <deviceConfigurationKeyValueListDataSelectors>
6446     <keyId>2</keyId>
6447   </deviceConfigurationKeyValueListDataSelectors>
6448 </filter>
6449 <deviceConfigurationKeyValueListData/>

```

6450 If the receiving device has a proper value for this *keyId*, it would respond with the following function:

```

6451 <deviceConfigurationKeyValueListData>
6452   <deviceConfigurationKeyValueData>
6453     <keyId>2</keyId>
6454     <value>
6455       <scaledNumber>
6456         <number>18</number>
6457         <scale>1</scale>
6458       </scaledNumber>
6459     </value>

```

```

6460         <isValueChangeable>false</isValueChangeable>
6461     </deviceConfigurationKeyValueData>
6462 </deviceConfigurationKeyValueListData>

```

6463

6464 5.3.8.2.4.2 Notify

6465 If a value changes, it will be notified to all subscribed clients:

```

6466 <deviceConfigurationKeyValueListData>
6467     <deviceConfigurationKeyValueData>
6468         <keyId>2</keyId>
6469         <value>
6470             <scaledNumber>
6471                 <number>32</number>
6472                 <scale>2</scale>
6473             </scaledNumber>
6474         </value>
6475         <isValueChangeable>false</isValueChangeable>
6476     </deviceConfigurationKeyValueData>
6477 </deviceConfigurationKeyValueListData>

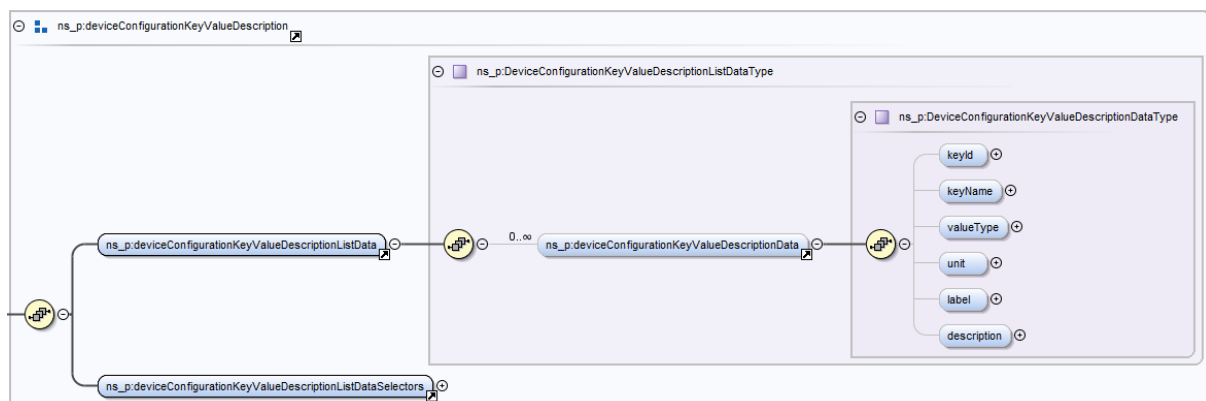
```

6478

6479 5.3.8.3 deviceConfigurationKeyValueDescriptionListData

6480 5.3.8.3.1 General

6481 The rather static part of the device configuration keys is modelled with this function, including an
 6482 identifier for the key, a name that defines the meaning of the key and further additional elements.



6483

6484 Figure 81: deviceConfigurationKeyValueDescriptionListData function overview

6485

6486 5.3.8.3.2 Detailed description of elements

Element	Type	Description
keyId	Identifier "DeviceConfigurationKeyIdType" (see Table 339).	The key identifier.
keyName	Union "DeviceConfigurationKeyNameType": - Enum (see Table 187): DeviceConfigurationKeyNameEnumType	A certain key name. If one of the keys defined by the EEBus Initiative e.V. is used, the meaning is defined in Table 187. Vendors may define own key names (but then have to mark them

	- EnumExtendType (see section 3.10.1.5)	as vendor specific, see [ProtocolSpecification], section "Rules for vendor specific extensions").
valueType	Enumeration DeviceConfigurationKeyValueDataTypeType (see Table 188).	Different value types are possible, but only one may be chosen. It is denoted in this element.
unit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	The unit in which the value of the key is given. Not all keys need the unit element.
label	Common data type "LabelType". See section 3.10.1.2.	A label for this key.
description	Common data type "DescriptionType". See section 3.10.1.3.	A description for this key.

6487 Table 186: deviceConfigurationKeyValueDescriptionListData function detailed description of elements

6488 Enumeration **DeviceConfigurationKeyNameEnumType**:

Value	Description
peakPowerOfPvSystem	The percentage of the peak power of the pv system that is permitted to be fed into the electricity grid through the electricity grid connection point.
pvCurtailementLimitFactor	The nominal peak power of the production of the installed PV System.
asymmetricChargingSupported	This configuration parameter is used by the Entity EV to indicate the Entity SECC if it is possible to charge with different current per phase.
communicationsStandard	Defines the used communications standard for the entity where the DeviceConfiguration feature is located.

6489 Table 187: Enumeration DeviceConfigurationKeyNameEnumType

6490 Enumeration **DeviceConfigurationKeyValueDataTypeType**:

Value	Description
boolean	The corresponding key value is of type xs:boolean.
date	The corresponding key value is of type xs:date.
dateTime	The corresponding key value is of type xs:dateTime.
duration	The corresponding key value is of type xs:duration.
string	The corresponding key value is of type xs:string.
time	The corresponding key value is of type xs:time.
scaledNumber	The corresponding key value is of type "ScaledNumberType" (see section 3.10.1.8).

6491 Table 188: Enumeration DeviceConfigurationKeyValueDataTypeType

6492

6493 5.3.8.3.3 Available selectors

6494 - keyId

6495 - keyName

6496

5.3.8.3.4 Examples

5.3.8.3.4.1 Read-reply

If one is interested in the rather static information of the device configuration keys, he could request the information with a read command. The device replies with information like this:

```
<deviceConfigurationKeyValueDescriptionListData>
  <deviceConfigurationKeyValueDescriptionData>
    <keyId>1</keyId>
    <keyName>pvCurtailmentLimitFactor</keyName>
    <valueType>scaledNumber</valueType>
    <unit>pct</unit>
    <label>PV Curtailment Limit Factor</label>
    <description>The nominal peak power of the production of
the installed pv system</description>
  </deviceConfigurationKeyValueDescriptionData>
  <deviceConfigurationKeyValueDescriptionData>
    <keyId>2</keyId>
    <keyName>peakPowerOfPvSystem</keyName>
    <valueType>scaledNumber</valueType>
    <unit>W</unit>
    <label>Peak Power of PV System</label>
    <description>The percentage of the peak power of the pv system that
is permitted to be fed into the electricity grid through the electricity
grid connection point.</description>
  </deviceConfigurationKeyValueDescriptionData>
</deviceConfigurationKeyValueDescriptionListData>
```

5.3.8.4 deviceConfigurationKeyValueConstraintsListData

5.3.8.4.1 General

In cases where a configuration value has some range, it could make sense to specify the upper and lower bounds that are possible for this value as well as a step size. Therefore, the deviceConfigurationKeyValueConstraintsListData function can be used.

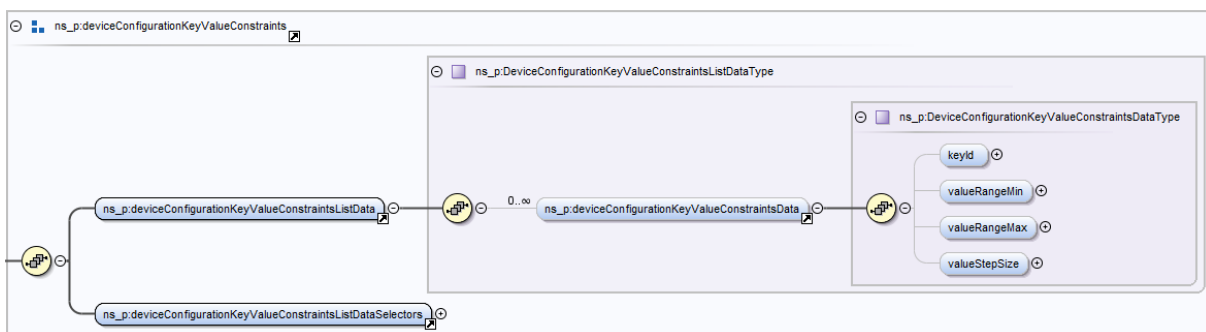


Figure 82: deviceConfigurationKeyValueConstraintsListData function overview

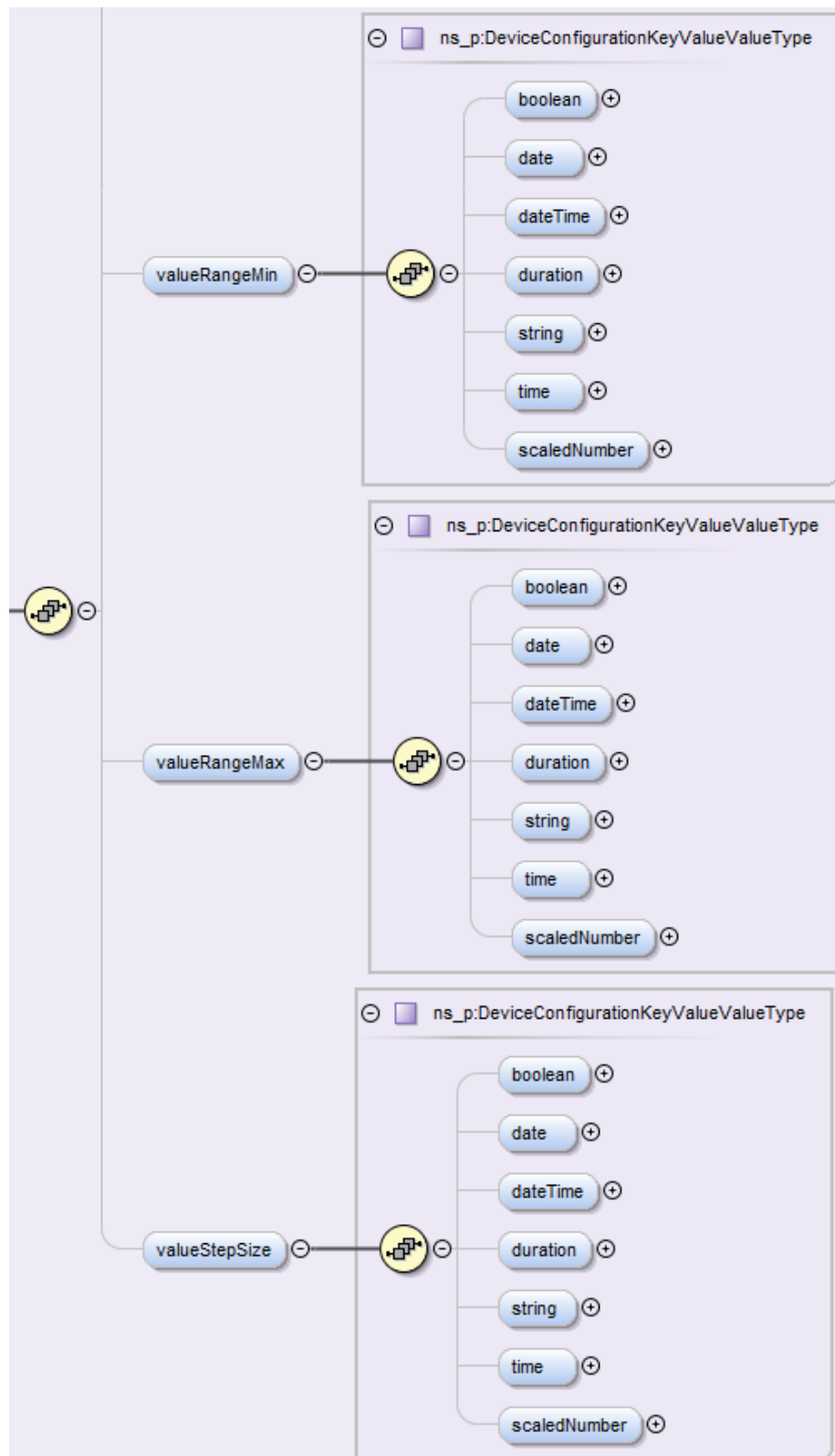


Figure 83: `deviceConfigurationKeyValueConstraintsListData` function detail

5.3.8.4.2 Detailed description of elements

Element	Type	Description
---------	------	-------------

keyId	Identifier "DeviceConfigurationKeyIdType" (see Table 339).	Enables the identification of different keys on one SPINE feature.
valueRangeMin	Complex type "DeviceConfigurationKeyValueValueType". See Table 185.	The lower bound that is possible for the related value.
valueRangeMax	Complex type "DeviceConfigurationKeyValueValueType". See Table 185.	The upper bound that is possible for the related value.
valueStepSize	Complex type "DeviceConfigurationKeyValueValueType". See Table 185.	The step size that is possible for the related value.

Table 189: deviceConfigurationKeyValueListData function detailed description of elements

5.3.8.4.3 Available selectors

- keyId

5.3.8.4.4 Examples

5.3.8.4.4.1 Read-reply

If one is interested in the constraints of all device configuration keys, he could request the information with a read command. The device replies with information like this:

```

<deviceConfigurationKeyValueConstraintsListData>
  <deviceConfigurationKeyValueConstraintsData>
    <keyId>1</keyId>
    <valueRangeMin>
      <scaledNumber>
        <number>0</number>
      </scaledNumber>
    </valueRangeMin>
    <valueRangeMax>
      <scaledNumber>
        <number>1</number>
        <scale>2</scale>
      </scaledNumber>
    </valueRangeMax>
    <valueStepSize>
      <scaledNumber>
        <number>5</number>
        <scale>-1</scale>
      </scaledNumber>
    </valueStepSize>
  </deviceConfigurationKeyValueConstraintsData>
  <deviceConfigurationKeyValueConstraintsData>
    <keyId>2</keyId>
    <valueRangeMin>
      <scaledNumber>
        <number>5</number>
        <scale>3</scale>
      </scaledNumber>
    </valueRangeMin>
    <valueRangeMax>
      <scaledNumber>

```

```

6574         <number>7</number>
6575         <scale>3</scale>
6576     </scaledNumber>
6577 </valueRangeMax>
6578 <valueStepSize>
6579     <scaledNumber>
6580         <number>1</number>
6581     </scaledNumber>
6582 </valueStepSize>
6583 </deviceConfigurationKeyValueConstraintsData>
6584 </deviceConfigurationKeyValueConstraintsListData>

```

5.3.9 DeviceDiagnosis

5.3.9.1 Introduction

This class contains information about the state of a device, heartbeat counting and service data.

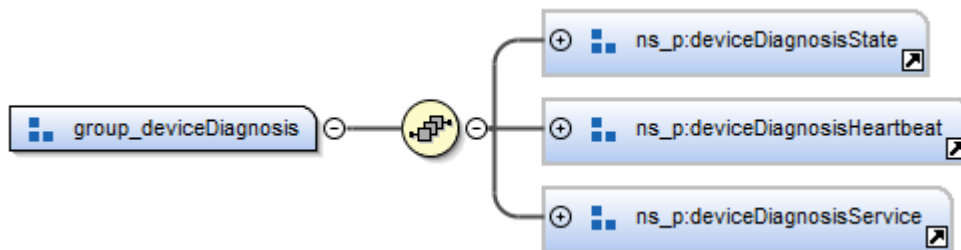


Figure 84: DeviceDiagnosis function-group overview

5.3.9.2 deviceDiagnosisStateData

5.3.9.2.1 General

The *deviceDiagnosisStateData* function provides information about dynamically changing state information.

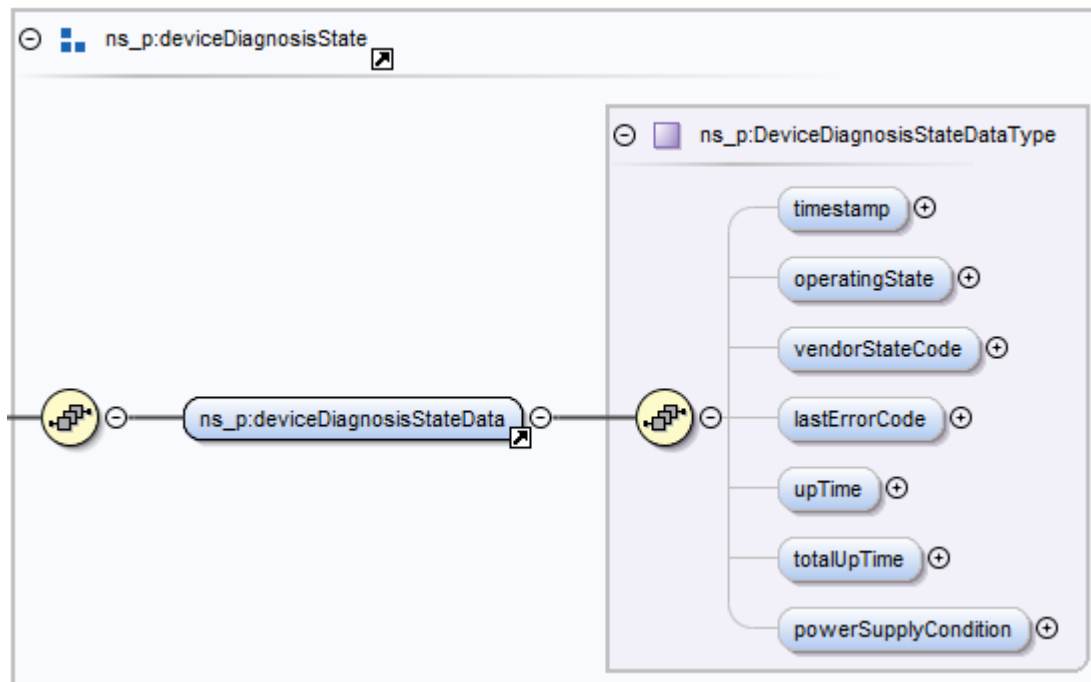


Figure 85: deviceDiagnosisStateData function overview

5.3.9.2.2 Detailed description of elements

Element	Type	Description
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The time of creation of the state.
operatingState	Union "DeviceDiagnosisOperatingStateType": - Enum (see Table 191): DeviceDiagnosisOperatingStateEnumType - EnumExtendType (see section 3.10.1.5)	The current operating state of the device.
vendorStateCode	Simple type "VendorStateCodeType" (restriction of xs:string)	A vendor specific state. May contain any state, the vendor uses for its devices.
lastErrorCode	Simple type "LastErrorCodeType" (restriction of xs:string)	The last error code that occurred is stored in this element. Even if the device is in <i>normalOperation</i> again, the error code remains here.
upTime	xs:duration (W3C standard type)	The duration the functionality is running since the last start.
totalUpTime	xs:duration (W3C standard type)	The total duration the functionality is running since installation (see <i>installationTime</i> element in <i>deviceDiagnosisServiceData</i> function).

powerSupplyCondition	<i>Union</i> "PowerSupplyConditionType": - Enum (see Table 192): PowerSupplyConditionEnumType - EnumExtendType (see section 3.10.1.5)	The condition of the power supply can be stated here.
----------------------	--	---

6600 Table 190: deviceDiagnosisStateData function detailed description of elements

6601 Enumeration **DeviceDiagnosisOperatingStateEnumType**:

Value	Description
normalOperation	The device does its normal operation without any errors or limitations.
standby	The device is in standby mode and does nothing but waits for user interaction and listens on its communications protocol for external commands.
failure	A failure occurred and the device cannot run as desired. Its normal functionality is likely not given.
serviceNeeded	The device needs some kind of service. Possibly the normal functionality is not given.
overrideDetected	The device detected an override. Normal operation may be given or the device is in some safe mode.
inAlarm	An alarm (or emergency) occurred and user interaction is needed. Normal operation may be given or the device is in some safe mode.
notReachable	The device is not reachable via its non-SPINE communications protocol. This operating state can only be set by some technology gateway device that handles the normal communication to the device.
finished	The device has finished its temporary operation. As long as it remains in state <i>finished</i> , the normal operation is not given.

6602 Table 191: Enumeration DeviceDiagnosisOperatingStateEnumType

6603 Enumeration **PowerSupplyConditionEnumType**:

Value	Description
good	The power supply is in a good condition.
low	The power supply has low capacity. It would make sense to check it (e.g. change battery).
critical	The power supply is in critical state. If it is supplied by a battery, a change is advised.
unknown	The power supply condition is unknown. A notification to a user could make sense.
error	The power supply is in error condition.

6604 Table 192: Enumeration PowerSupplyConditionEnumType

6605

6606 5.3.9.2.3 Available selectors

6607 None.

6608

5.3.9.2.4 Examples

5.3.9.2.4.1 Notify

The current state of a device is notified to all subscribed clients:

```
<deviceDiagnosisStateData>
  <timestamp>2014-04-28T17:43:14.0Z</timestamp>
  <operatingState>normalOperation</operatingState>
  <vendorStateCode>heating up</vendorStateCode>
  <lastErrorCode>13</lastErrorCode>
  <upTime>P3DT4H7M14S</upTime>
  <totalUpTime>P2Y8M15DT7H32M17S</totalUpTime>
  <powerSupplyCondition>good</powerSupplyCondition>
</deviceDiagnosisStateData>
```

5.3.9.3 deviceDiagnosisHeartbeatData

5.3.9.3.1 General

The *deviceDiagnosisHeartbeatData* function enables sending a specific signal, which indicates a device is still running.

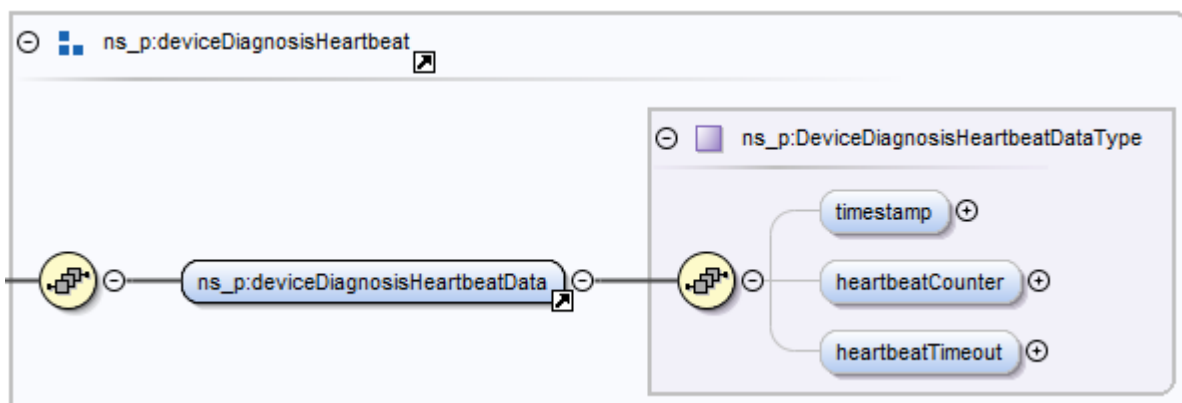


Figure 86: deviceDiagnosisHeartbeatData function overview

5.3.9.3.2 Detailed description of elements

Element	Type	Description
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The time of creation of the data.
heartbeatCounter	xs:unsignedLong (W3C standard type)	An incrementing counter of the heartbeat.
heartbeatTimeout	xs:duration (W3C standard type)	The heartbeatTimeout element contains the period, in which the deviceDiagnosisHeartbeatData function is sent by the device (NOT the remaining time till the next sending).

Table 193: deviceDiagnosisHeartbeatData function detailed description of elements

5.3.9.3.3 Available selectors

None.

5.3.9.3.4 Examples

5.3.9.3.4.1 Read-reply

One is interested in the heartbeat information of a device and sends a standard read command to the device which responds with a *deviceDiagnosisHeartbeatData* function:

```
<deviceDiagnosisHeartbeatData>
  <timestamp>2014-04-28T17:54:38.0Z</timestamp>
  <heartbeatCounter>238</heartbeatCounter>
  <heartbeatTimeout>PT15M</heartbeatTimeout>
</deviceDiagnosisHeartbeatData>
```

5.3.9.4 deviceDiagnosisServiceData

5.3.9.4.1 General

The *deviceDiagnosisServiceData* function informs about service information related to a device's installation or maintenance.

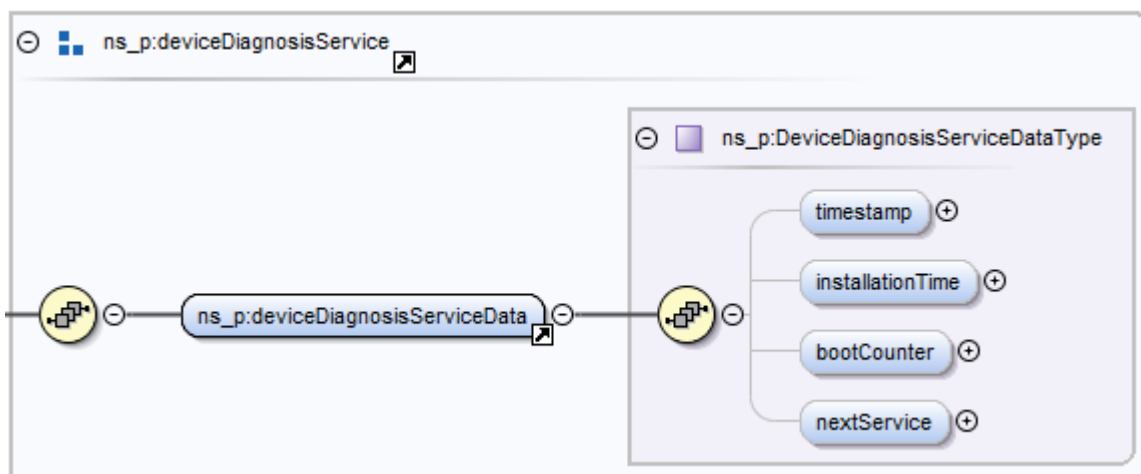


Figure 87: deviceDiagnosisServiceData function overview

5.3.9.4.2 Detailed description of elements

Element	Type	Description
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The time of creation of the data.
installationTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The point in time the device was installed (commissioned).
bootCounter	xs:unsignedLong (W3C standard type)	The count of boot processes the device completed since <i>installationTime</i> .

nextService	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The point in time, the next service should be done, is stated by this element. Alternatively, the decrementing duration till the next service can be indicated.
-------------	--	--

Table 194: deviceDiagnosisServiceData function detailed description of elements

5.3.9.4.3 Available selectors

None.

5.3.9.4.4 Examples

5.3.9.4.4.1 Read-reply

One is interested in the service information of a device and sends a standard read command to the device which responds with a *deviceDiagnosisServiceData* function:

```
<deviceDiagnosisServiceData>
  <timestamp>2014-04-28T17:59:05.0Z</timestamp>
  <installationTime>2011-08-15T11:05:08.0Z</installationTime>
  <bootCounter>83</bootCounter>
  <nextService>2015-08-15T11:05:08.0Z</nextService>
</deviceDiagnosisServiceData>
```

5.3.10 DirectControl

5.3.10.1 Introduction

Some kind of electricity consuming or producing devices require (in contrast to the PowerSequences Class, see section 5.3.19) a rather instantaneous energy consumption or generation control. This can be modelled with two primary information sets:

3. Notification of instantaneous consumption/generation as well as instantaneous control. This is covered by DirectControl and discussed in this section.
4. Information on the power/energy constraints and implications. This is covered by OperatingConstraints and discussed in section 5.3.18.

Information on and control of energy/power can be modelled using directControlActivity. directControlDescription is used to model some general constant data.

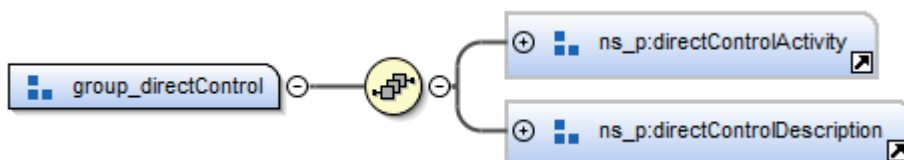


Figure 88: DirectControl function-group overview

5.3.10.2 *directControlActivityListData*

5.3.10.2.1 General

The function *directControlActivityListData* combines some information on the activity of a given time. Apart from information on state, power and energy it can contain information whether power, energy, etc. can be modified with a proper (partial) write command.

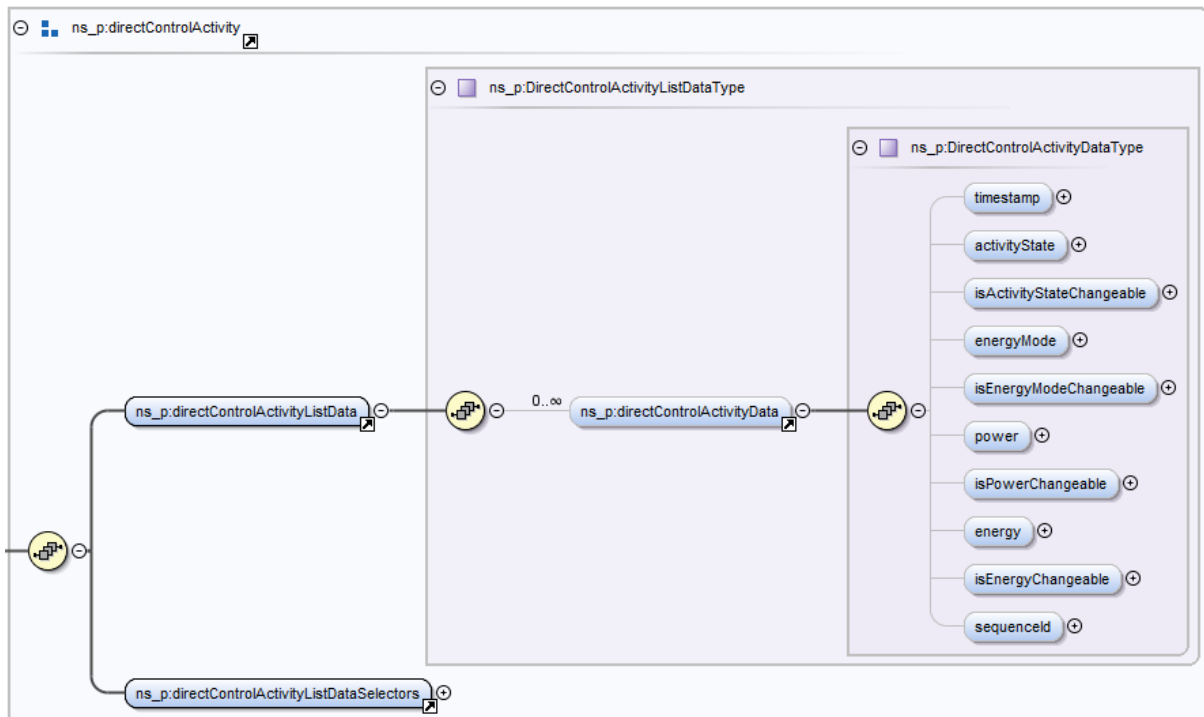


Figure 89: `directControlActivityListData` function overview

5.3.10.2.2 Detailed description of elements

Element	Type	Description
timestamp	<i>Common data type</i> <i>"AbsoluteOrRelativeTimeType". See section 3.10.2.3.</i>	The timestamp of this function instance.
activityState	<i>Union</i> <i>"DirectControlActivityStateType":</i> <i>- Enum (see Table 196):</i> <i>DirectControlActivityStateEnumType</i> <i>- EnumExtendType (see section 3.10.1.5)</i>	The state of the activity.
isActivityStateChangeable	xs:boolean (W3C standard type)	Indicates whether the "activityState" can be modified by a client or not.
energyMode	<i>Common data type</i> <i>"EnergyModeType". See section 3.10.1.15.</i>	Energy mode of the device.
isEnergyModeChangeable	xs:boolean (W3C standard type)	Indicates whether the "energyMode" can be modified by a client or not.

power	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	The overall electrical power value belonging to this activity information. Please note: Phase-specific information is not given here.
isPowerChangeable	xs:boolean (W3C standard type)	Indicates whether the "power" can be modified by a client or not.
energy	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	The electrical energy value belonging to this activity information.
isEnergyChangeable	xs:boolean (W3C standard type)	Indicates whether the "energy" can be modified by a client or not.
sequenceId	xs:unsignedInt (W3C standard type)	The related power sequence identifier (if available).

6692 Table 195: directControlActivityListData function detailed description of elements

6693 Enumeration **DirectControlActivityStateEnumType**:

Value	Description
running	The activity is running.
paused	The activity is paused.
inactive	The activity is inactive.

6694 Table 196: Enumeration DirectControlActivityStateEnumType

6695

6696 5.3.10.2.3 Available selectors

6697 - timestampInterval

6698

6699 5.3.10.2.4 Examples

6700 5.3.10.2.4.1 Notify

6701 When the activity state changes, a server will notify the new information to all subscribed devices
 6702 with a message like this (the "filter/partial" part to announce the "restricted function exchange is
 6703 simplified by "..." just to improve the readability"):

```

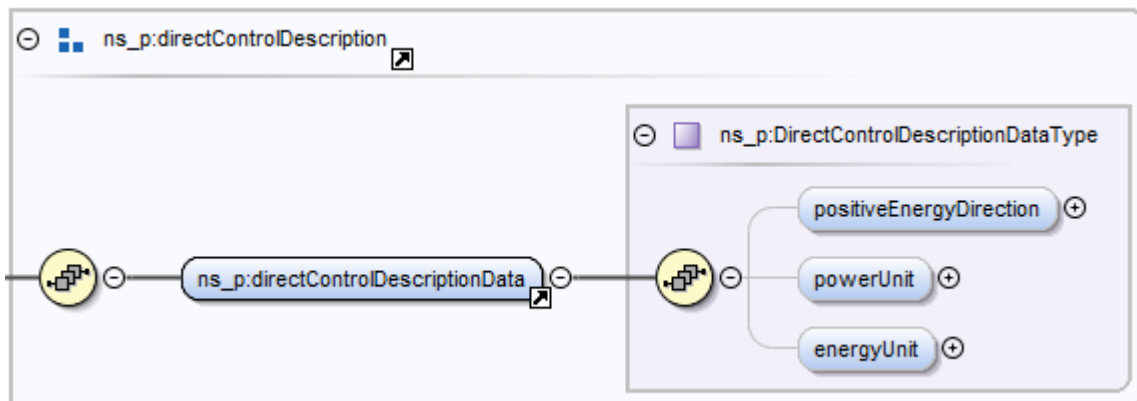
6704 ...
6705 <directControlActivityListData>
6706   <directControlActivityData>
6707     <timestamp>2015-11-23T12:05:04.3Z</timestamp>
6708     <activityState>running</activityState>
6709     <isActivityStateChangeable>true</isActivityStateChangeable>
6710     <energyMode>consume</energyMode>
6711     <isEnergyModeChangeable>>false</isEnergyModeChangeable>
6712     <power>
6713       <number>2</number>
6714       <scale>3</scale>
6715     </power>
6716     <isPowerChangeable>>false</isPowerChangeable>
6717     <sequenceId>1</sequenceId>
6718   </directControlActivityData>
6719 </directControlActivityListData>

```

6720

6721 **5.3.10.3 directControlDescriptionData**6722 *5.3.10.3.1 General*

6723 This function contains the rather static information available in this class.



6724

6725 *Figure 90: directControlDescriptionData function overview*

6726

6727 *5.3.10.3.2 Detailed description of elements*

Element	Type	Description
positiveEnergyDirection	Common data type "EnergyDirectionType". See section 3.10.1.13.	The <i>positiveEnergyDirection</i> states whether energy consumption or production will be counted as positive value.
powerUnit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	This is the unit for all power values related to consumption or production.
energyUnit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	This is the unit for all energy values related to consumption or production.

6728 *Table 197: directControlDescriptionData function detailed description of elements*

6729

6730 *5.3.10.3.3 Available selectors*

6731 None.

6732

6733 *5.3.10.3.4 Examples*6734 *5.3.10.3.4.1 Read-reply*

6735 If one is interested in the rather static information of the direct control, he could request the
 6736 information with a read command. The device replies with information like this:

```

6737 <directControlDescriptionData>
6738   <positiveEnergyDirection>consume</positiveEnergyDirection>
6739   <powerUnit>W</powerUnit>
6740   <energyUnit>Wh</energyUnit>

```

6741 </directControlDescriptionData>

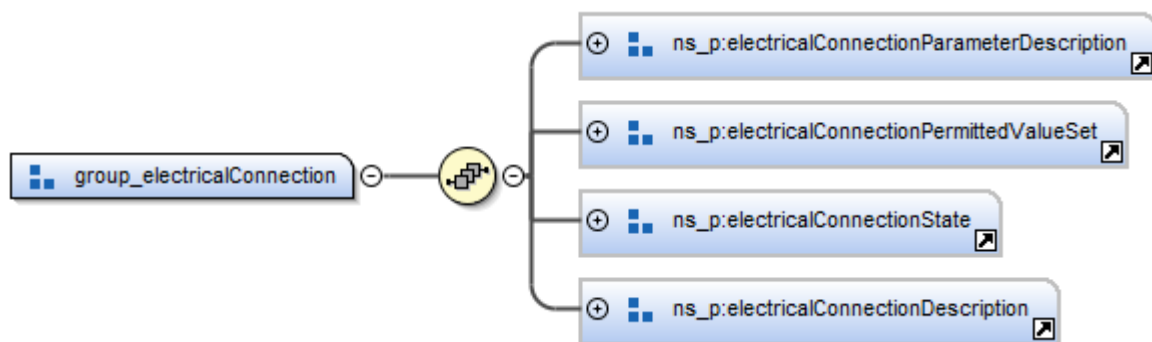
6742

6743 5.3.11 ElectricalConnection

6744 5.3.11.1 Introduction

6745 Electrical connections have some more complex parameters (e.g. which AC phase is measured,
6746 real/reactive/apparent measurement, amplitude/rms, number of measured harmonic, etc.), which
6747 are modelled with this class. Some electrical connection specific state information (e.g. AC or DC,
6748 number of connected phases (only AC), positive energy direction, label, etc.) is modelled, too.

6749 Measuring electrical values is still done with the Measurement class (see section 5.3.15). But the
6750 additional parameters are only described within the ElectricalConnection class. The *measurementId*
6751 from the Measurement class is stated in the *electricalConnectionParameterDescriptionListData*
6752 function, to have a proper reference.



6753

6754 Figure 91: ElectricalConnection function-group overview

6755

6756 5.3.11.2 electricalConnectionParameterDescriptionListData

6757 5.3.11.2.1 General

6758 The aforementioned parameters for electrical connection measurement are described in this
6759 function. The value itself is modelled in the Measurement class (section 5.3.15). Only a reference is
6760 given in this function.

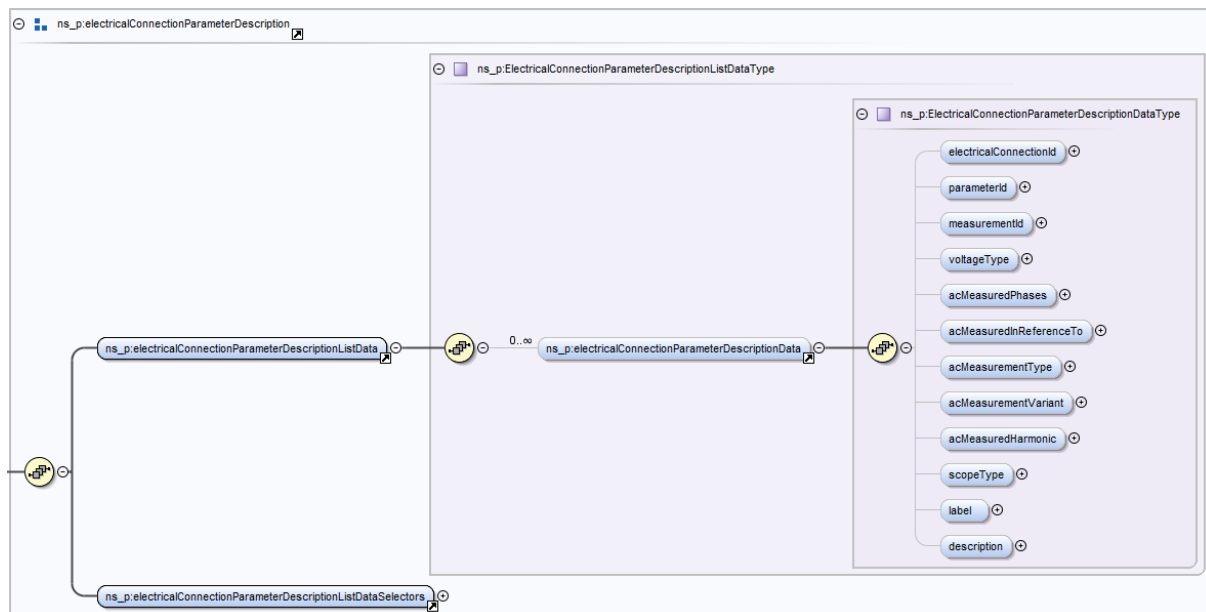


Figure 92: electricalConnectionParameterDescriptionListData function overview

5.3.11.2.2 Detailed description of elements

Element	Type	Description
electricalConnectionId	Identifier "ElectricalConnectionIdType" (see Table 339).	Reference to the electrical connection, described in the <i>electricalConnectionDescriptionListData</i> function (see section 5.3.11.5).
parameterId	Identifier "ElectricalConnectionParameterIdType" (see Table 339).	One electrical connection can have multiple parameter combinations, which can be measured. The <i>parameterId</i> helps to distinguish them.
measurementId	Identifier "MeasurementIdType" (see Table 339).	Reference to the Measurement class (see section 5.3.15), which contains the actual measured values for this parameter combination.
voltageType	Union "ElectricalConnectionVoltageTypeType": - Enum (see Table 199): ElectricalConnectionVoltageTypeEnumType - EnumExtendType (see section 3.10.1.5)	Specifies which kind of electricity is measured ("ac" or "dc").
acMeasuredPhases	Union "ElectricalConnectionPhaseNameType": - Enum (see Table 200): ElectricalConnectionPhaseNameEnumType - EnumExtendType (see section 3.10.1.5)	In case of <i>powerSupplyType</i> = "ac", this element states the phase, which is measured. Combinations of phases (e.g. "ab") are possible.
acMeasuredInReferenceTo	Union "ElectricalConnectionPhaseNameType":	In case of <i>powerSupplyType</i> = "ac", this element states the

	- Enum (see Table 200): ElectricalConnectionPhaseNameEnumType - EnumExtendType (see section 3.10.1.5)	phase, which <i>acMeasuredPhases</i> is measured against (e.g. "neutral").
acMeasurementType	Union "ElectricalConnectionAcMeasurementType": - Enum (see Table 201): ElectricalConnectionAcMeasurementTypeEnumType - EnumExtendType (see section 3.10.1.5)	In case of <i>powerSupplyType</i> = "ac", this element states the kind of ac measurement is done (e.g. "real").
acMeasurementVariant	Union "ElectricalConnectionMeasurandVariantType": - Enum (see Table 202): ElectricalConnectionMeasurandVariantEnumType - EnumExtendType (see section 3.10.1.5)	In case of <i>powerSupplyType</i> = "ac", this element states the variation of the ac measurement (e.g. "amplitude").
acMeasuredHarmonic	xs:unsignedByte (W3C standard type)	In case of <i>powerSupplyType</i> = "ac", this element states the harmonic, which is measured. A value of "1" indicates that the normal value is measured.
scopeType	Common data type "ScopeTypeType". See section 3.10.1.21.	A certain meaning of the parameter combination.
label	Common data type "LabelType". See section 3.10.1.2.	A label for this electrical connection parameter.
description	Common data type "DescriptionType". See section 3.10.1.3.	A description for this electrical connection parameter.

6765 Table 198: electricalConnectionParameterDescriptionListData function detailed description of elements

6766 Enumeration **ElectricalConnectionVoltageTypeEnumType**:

Value	Description
ac	Alternating current.
dc	Direct current.

6767 Table 199: Enumeration ElectricalConnectionVoltageTypeEnumType

6768 Enumeration **ElectricalConnectionPhaseNameEnumType**:

Value	Description
a	Phase a (or L1).
b	Phase b (or L2).
c	Phase c (or L3).
ab	Phase a and b (or L1 and L2).
bc	Phase b and c (or L2 and L3).
ac	Phase a and c (or L1 and L3).
abc	Phase a, b and c (or L1, L2 and L3).
neutral	The neutral conductor of a connection.
ground	The ground conductor of a connection.
none	No phase specified or available.

6769 Table 200: Enumeration ElectricalConnectionPhaseNameEnumType

6770 Enumeration **ElectricalConnectionAcMeasurementTypeEnumType**:

Value	Description
real	AC measurement of the real part.
reactive	AC measurement of the reactive part.
apparent	AC measurement of the apparent part.
phase	AC measurement of the phase.

6771 *Table 201: Enumeration ElectricalConnectionAcMeasurementTypeEnumType*

6772 Enumeration **ElectricalConnectionMeasurandVariantEnumType**:

Value	Description
amplitude	Measurement of the amplitude.
rms	Measurement of the RMS (Root Mean Square).
instantaneous	Measurement of the instantaneous value.
angle	Measurement of the angle.
cosPhi	Measurement of the cosine phi.

6773 *Table 202: Enumeration ElectricalConnectionMeasurandVariantEnumType*

6774

6775 *5.3.11.2.3 Available selectors*

- 6776 - electricalConnectionId
- 6777 - parameterId
- 6778 - measurementId
- 6779 - scopeType

6780

6781 *5.3.11.2.4 Examples*

6782 *5.3.11.2.4.1 Read-reply*

6783 If one only knows the *measurementId* of a specific measurand and is interested in the electrical
6784 connection parameters, he could send the following read command:

```

6785 <function>electricalConnectionParameterDescriptionListData</function>
6786 <filter>
6787   <cmdControl>
6788     <partial/>
6789   </cmdControl>
6790   <electricalConnectionParameterDescriptionListDataSelectors>
6791     <measurementId>6</measurementId>
6792   </electricalConnectionParameterDescriptionListDataSelectors>
6793 </filter>
6794 <electricalConnectionParameterDescriptionListData/>

```

6795 If the receiving device has a proper parameter description for this *measurementId*, it would respond
6796 with the following function:

```

6797 <function>electricalConnectionParameterDescriptionListData</function>
6798 <filter>
6799   <cmdControl>
6800     <partial/>
6801   </cmdControl>
6802 </filter>
6803 <electricalConnectionParameterDescriptionListData>

```

```

6804     <electricalConnectionParameterDescriptionData>
6805         <electricalConnectionId>1</electricalConnectionId>
6806         <parameterId>3</parameterId>
6807         <measurementId>6</measurementId>
6808         <voltageType>ac</voltageType>
6809         <acMeasuredPhases>a</acMeasuredPhases>
6810         <acMeasuredInReferenceTo>neutral</acMeasuredInReferenceTo>
6811         <acMeasurementType>real</acMeasurementType>
6812         <acMeasurementVariant>rms</acMeasurementVariant>
6813         <acMeasuredHarmonic>1</acMeasuredHarmonic>
6814         <scopeType>acYieldTotal</scopeType>
6815     </electricalConnectionParameterDescriptionData>
6816 </electricalConnectionParameterDescriptionListData>

```

5.3.11.3 electricalConnectionPermittedValueSetListData

5.3.11.3.1 General

If an electrical connection parameter has some defined value(s) or value range(s) that are allowed or possible, they can be modelled with this function. It enables the server to state single values, value ranges or combinations of both.

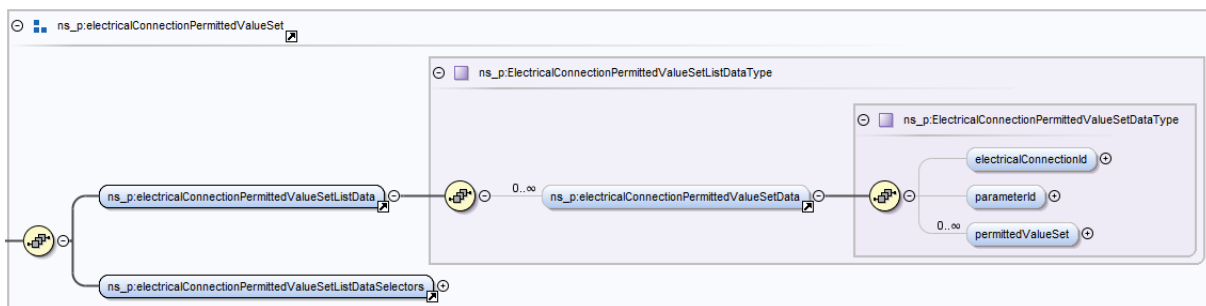


Figure 93: electricalConnectionPermittedValueSetListData function overview

5.3.11.3.2 Detailed description of elements

Element	Type	Description
electricalConnectionId	Identifier "ElectricalConnectionIdType" (see Table 339).	Electrical connection, this static information belongs to.
parameterId	Identifier "ElectricalConnectionParameterIdType" (see Table 339).	Relation to the according parameter.
permittedValuesSet (list)	Common data type "ScaledNumberSetType". See section 3.10.1.9.	The permittedValueSet allows to define an arbitrary set of permitted data. Compared to a simple value range the permittedValueSet also may have gaps. Please refer to the ScaledNumberSetType for more details.

Table 203: electricalConnectionPermittedValueSetListData function detailed description of elements

5.3.11.3.3 Available selectors

- electricalConnectionId
- parameterId

5.3.11.3.4 Examples

5.3.11.3.4.1 Read-reply

If one is interested in all electrical connection permitted value sets of a device, he could send the read command to the device, which will respond with something like the following command:

```
<electricalConnectionPermittedValueSetListData>
  <electricalConnectionPermittedValueSetData>
    <electricalConnectionId>1</electricalConnectionId>
    <parameterId>2</parameterId>
    <permittedValueSet>
      <range>
        <min>
          <number>0</number>
        </min>
        <max>
          <number>35</number>
          <scale>2</scale>
        </max>
      </range>
    </permittedValueSet>
  </electricalConnectionPermittedValueSetData>
  <electricalConnectionPermittedValueSetData>
    <electricalConnectionId>1</electricalConnectionId>
    <parameterId>3</parameterId>
    <permittedValueSet>
      <value>
        <number>0</number>
      </value>
      <range>
        <min>
          <number>5</number>
          <scale>1</scale>
        </min>
        <max>
          <number>2</number>
          <scale>3</scale>
        </max>
      </range>
    </permittedValueSet>
  </electricalConnectionPermittedValueSetData>
</electricalConnectionPermittedValueSetListData>
```

5.3.11.4 electricalConnectionStateListData

5.3.11.4.1 General

Some electrical connection specific state information is modelled in this function.

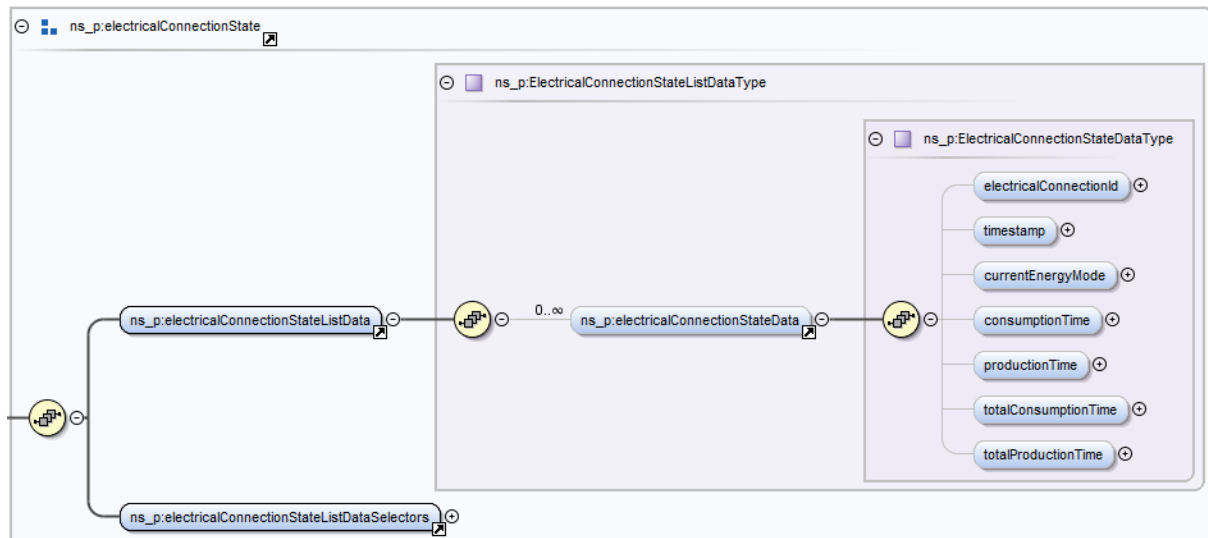


Figure 94: electricalConnectionStateListData function overview

5.3.11.4.2 Detailed description of elements

Element	Type	Description
electricalConnectionId	Identifier "ElectricalConnectionIdType" (see Table 339).	Electrical connection, this state information belongs to.
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	Point in time, the state information was generated.
currentEnergyMode	Common data type "EnergyModeType". See section 3.10.1.15.	Whether a device is consuming, producing or idle is stated by this element.
consumptionTime	xs:duration (W3C standard type)	The duration, the device is in energy mode "consume" since the last reboot (see section 5.3.9.2).
productionTime	xs:duration (W3C standard type)	The duration, the device is in energy mode "produce" since the last reboot (see section 5.3.9.2).
totalConsumptionTime	xs:duration (W3C standard type)	The duration, the device is in energy mode "consume" since its <i>installationTime</i> (see section 5.3.9.4).
totalProductionTime	xs:duration (W3C standard type)	The duration, the device is in energy mode "producing" since its <i>installationTime</i> (see section 5.3.9.4).

Table 204: electricalConnectionStateListData function detailed description of elements

5.3.11.4.3 Available selectors

- electricalConnectionId

6886 5.3.11.4.4 Examples

6887 5.3.11.4.4.1 Read-reply

6888 The state of a specific electrical connection can be requested with this read command:

```
6889 <function>electricalConnectionStateListData</function>
6890 <filter>
6891     <cmdControl>
6892         <partial/>
6893     </cmdControl>
6894     <electricalConnectionStateListDataSelectors>
6895         <electricalConnectionId>1</electricalConnectionId>
6896     </electricalConnectionStateListDataSelectors>
6897 </filter>
6898 <electricalConnectionStateListData/>
```

6899 If the requested device has proper information, it will respond them with the following command:

```
6900 <function>electricalConnectionStateListData</function>
6901 <filter>
6902     <cmdControl>
6903         <partial/>
6904     </cmdControl>
6905 </filter>
6906 <electricalConnectionStateListData>
6907     <electricalConnectionStateData>
6908         <electricalConnectionId>1</electricalConnectionId>
6909         <timestamp>2014-05-12T17:55:30.0Z</timestamp>
6910         <currentEnergyMode>consume</currentEnergyMode>
6911         <consumptionTime>PT2H10M15S</consumptionTime>
6912         <productionTime>PT0S</productionTime>
6913         <totalConsumptionTime>P1Y3M2DT21H14M57S</totalConsumptionTime>
6914         <totalProductionTime>PT0S</totalProductionTime>
6915     </electricalConnectionStateData>
6916 </electricalConnectionStateListData>
```

6917

6918 5.3.11.5 electricalConnectionDescriptionListData

6919 5.3.11.5.1 General

6920 Some electrical connection specific information, that are rather static, are modelled within this
6921 function.

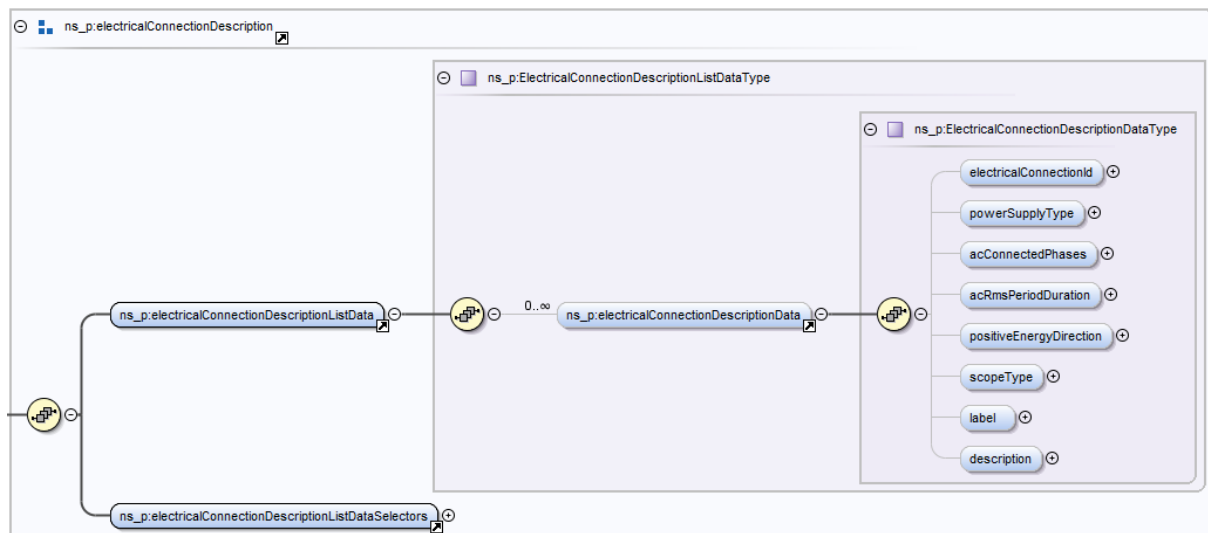


Figure 95: electricalConnectionStateDescriptionData function overview

5.3.11.5.2 Detailed description of elements

Element	Type	Description
electricalConnectionId	Identifier "ElectricalConnectionIdType" (see Table 339).	Electrical connection, this static information belongs to.
powerSupplyType	Union "ElectricalConnectionVoltageTypeType": - Enum (see Table 199): ElectricalConnectionVoltageTypeEnumType - EnumExtendType (see section 3.10.1.5)	States whether the electrical connection is of type "ac" or "dc".
acConnectedPhases	xs:unsignedInt (W3C standard type)	If powerSupplyType="ac", this element contains the number of phases, this electrical connection has (typically 1, 2 or 3).
acRmsPeriodDuration	xs:duration (W3C standard type)	If the "rms" value of a parameter combination (see element <i>acMeasurementVariant</i> in section 5.3.11.2.2) is measured, this element states the duration, the rms value is determined in.
positiveEnergyDirection	Common data type "EnergyDirectionType". See section 3.10.1.13.	This element states whether energy consumption or production will be counted as positive value.
scopeType	Common data type "ScopeTypeType". See section 3.10.1.21.	A certain meaning of the electrical connection.
label	Common data type "LabelType". See section 3.10.1.2.	A label for this electrical connection.

description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	A description for this electrical connection.
-------------	--	---

Table 205: electricalConnectionDescriptionListData function detailed description of elements

5.3.11.5.3 Available selectors

- electricalConnectionId
- scopeType

5.3.11.5.4 Examples

5.3.11.5.4.1 Read-reply

If one is interested in all electrical connection descriptions of a device, he could send the read command to the device, which will respond with something like the following command:

```
<electricalConnectionDescriptionListData>
  <electricalConnectionDescriptionData>
    <electricalConnectionId>1</electricalConnectionId>
    <powerSupplyType>ac</powerSupplyType>
    <acConnectedPhases>1</acConnectedPhases>
    <acRmsPeriodDuration>PT0.1S</acRmsPeriodDuration>
    <positiveEnergyDirection>consume</positiveEnergyDirection>
    <label>mains connection</label>
    <description>1-phase mains connection of the device</description>
  </electricalConnectionDescriptionData>
  <electricalConnectionDescriptionData>
    <electricalConnectionId>2</electricalConnectionId>
    <powerSupplyType>dc</powerSupplyType>
    <positiveEnergyDirection>consume</positiveEnergyDirection>
    <label>battery connection</label>
    <description>Backup battery connection of the device</description>
  </electricalConnectionDescriptionData>
</electricalConnectionDescriptionListData>
```

5.3.12 HVAC

5.3.12.1 Introduction

Heating, Ventilation and Air Conditioning (HVAC) devices have some special functionality which is covered by this class.

All system functions (like heating, cooling, domestic hot water preparation (dhw), ventilation, etc.) have their specific HVAC operation modes (like on, off, auto, etc.). Relations between these are modelled as well as relations to (optional) power sequences, which represent a concrete run of a device's program.

HVAC overruns (like an one time domestic-hot-water (DHW) preparation), which do have an impact on the normal operation mode of a HVAC system function, are modelled in further SPINE functions of the HVAC Class.

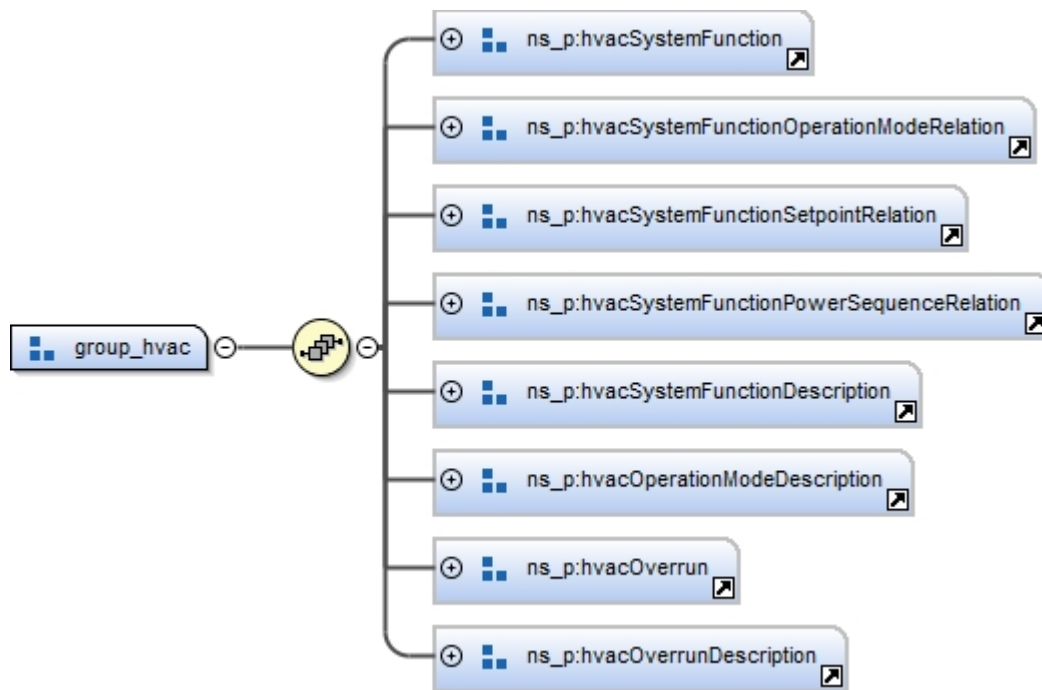


Figure 96: HVAC function-group overview

5.3.12.2 hvacSystemFunctionListData

5.3.12.2.1 General

The hvacSystemFunctionListData function announces the current status of the HVAC system functions, including the current HVAC operation mode, a flag that indicates, if the HVAC operation mode can be changed by some external communications partner (client) and a flag that indicates whether an HVAC overrun is currently active or not.

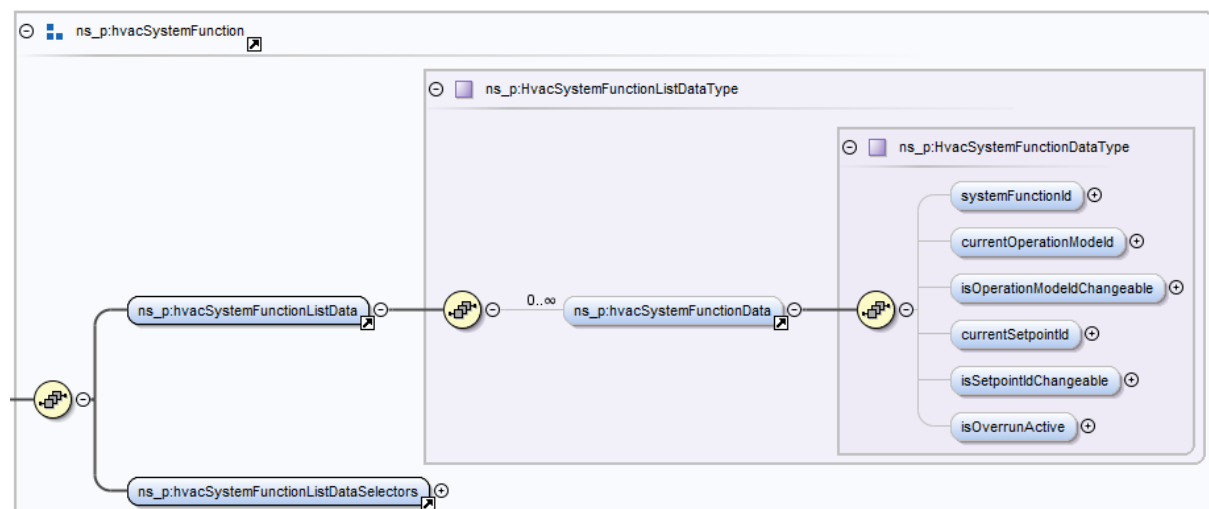


Figure 97: hvacSystemFunctionListData function overview

5.3.12.2.2 Detailed description of elements

Element	Type	Description
---------	------	-------------

systemFunctionId	<i>Identifier</i> <i>"HvacSystemFunctionIdType"</i> (see Table 339).	Identifier for a specific HVAC system function.
currentOperationModeId	<i>Identifier</i> <i>"HvacOperationModeIdType"</i> (see Table 339).	States which HVAC operation mode is currently active for this specific HVAC system function
isOperationModeIdChangeable	xs:boolean (W3C standard type)	Denotes whether the current HVAC operation mode identifier is changeable (by some external communications partner) or not.
currentSetpointId	<i>Identifier "SetpointIdType"</i> (see Table 339).	States which setpoint is active for this specific HVAC system function
isSetpointIdChangeable	xs:boolean (W3C standard type)	Denotes whether the current setpoint identifier is changeable (by some external communications partner) or not.
isOverrunActive	xs:boolean (W3C standard type)	If an HVAC overrun is currently active, this element is set to <i>true</i> . The HVAC overrun itself is modelled with other functions (see section 5.3.12.8 and 5.3.12.9) and may be located on a different feature or even a different entity or device.

Table 206: hvacSystemFunctionListData function detailed description of elements

5.3.12.2.3 Available selectors

- systemFunctionId

5.3.12.2.4 Examples

5.3.12.2.4.1 Notify

When the *currentOperationMode* of a HVAC system function changes, the server sends a notify to all its subscribed clients (the “filter/partial” part to announce the “restricted function exchange is simplified by “...” just to improve the readability”):

```

...
<hvacSystemFunctionListData>
  <hvacSystemFunctionData>
    <systemFunctionId>2</systemFunctionId>
    <currentOperationModeId>3</currentOperationModeId>
  </hvacSystemFunctionData>
</hvacSystemFunctionListData>

```

5.3.12.3 hvacSystemFunctionOperationModeRelationListData

5.3.12.3.1 General

Each HVAC system function has its own relation to all for this HVAC system function available HVAC operation modes. E.g. there may be three HVAC operation modes available on the feature: *on*, *off*, *auto* (see section 5.3.12.7). HVAC System function 1 only supports HVAC operation mode 1 (*on*) and 2 (*off*). HVAC System function 2 supports all three HVAC operation modes. The enumeration of the available HVAC system functions is described in section 5.3.12.6.2.

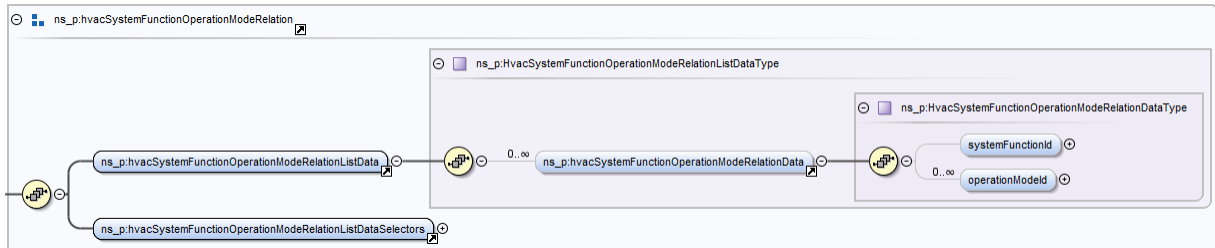


Figure 98: hvacSystemFunctionOperationModeRelationListData function overview

5.3.12.3.2 Detailed description of elements

Element	Type	Description
systemFunctionId	Identifier "HvacSystemFunctionIdType" (see Table 339).	Identifier for a specific HVAC system function.
operationModeId (list)	Identifier "HvacOperationModeIdType" (see Table 339).	Lists the HVAC operation modes that are available (selectable) for a specific HVAC system function.

Table 207: hvacSystemFunctionOperationModeRelationListData function detailed description of elements

5.3.12.3.3 Available selectors

- systemFunctionId

5.3.12.3.4 Examples

5.3.12.3.4.1 Read-reply

If one is interested in the supported HVAC operation modes of all HVAC system functions, he could send a standard read command to the server. The reply could look like this:

```
<hvacSystemFunctionOperationModeRelationListData>
  <hvacSystemFunctionOperationModeRelationData>
    <systemFunctionId>1</systemFunctionId>
    <operationModeId>1</operationModeId>
    <operationModeId>2</operationModeId>
    <operationModeId>3</operationModeId>
  </hvacSystemFunctionOperationModeRelationData>
  <hvacSystemFunctionOperationModeRelationData>
    <systemFunctionId>2</systemFunctionId>
    <operationModeId>1</operationModeId>
    <operationModeId>2</operationModeId>
```

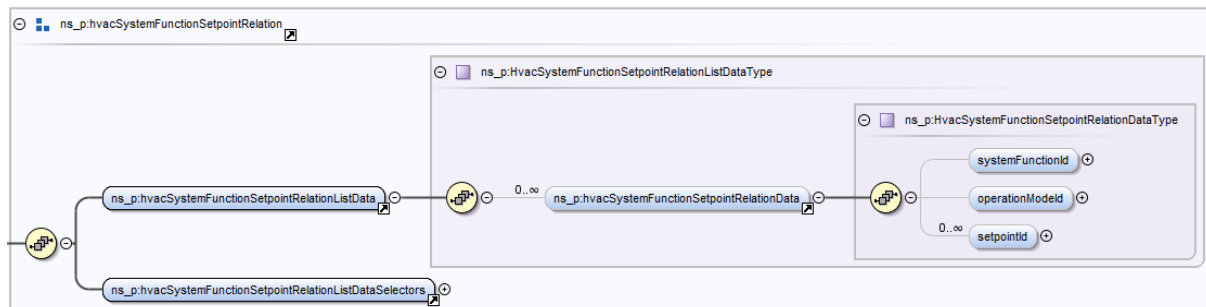

7028 </hvacSystemFunctionOperationModeRelationData>
 7029 </hvacSystemFunctionOperationModeRelationListData>

7030

7031 **5.3.12.4 hvacSystemFunctionSetpointRelationListData**

7032 5.3.12.4.1 General

7033 The setpoint class (see section 5.3.21) may be used to model some HVAC relevant setpoints like
 7034 desired room temperature. An HVAC system function may relate to some of these setpoints. The
 7035 relation is modelled with this SPINE function.



7036

7037 *Figure 99: hvacSystemFunctionSetpointRelationListData function overview*

7038

7039 5.3.12.4.2 Detailed description of elements

Element	Type	Description
systemFunctionId	Identifier "HvacSystemFunctionIdType" (see Table 339).	Identifier for a specific HVAC system function.
operationModelId	Identifier "HvacOperationModelIdType" (see Table 339).	Identifier for a specific HVAC operation mode.
setpointId (list)	Identifier "SetpointIdType" (see Table 339).	List of setpoints that are related with a specific HVAC system function.

7040 *Table 208: hvacSystemFunctionSetpointRelationListData function detailed description of elements*

7041

7042 5.3.12.4.3 Available selectors

- 7043 - systemFunctionId
- 7044 - operationModelId

7045

7046 5.3.12.4.4 Examples

7047 5.3.12.4.4.1 Read-reply

7048 If one is interested in the related setpoints of all HVAC system functions, he could send a standard
 7049 read command to the server. The reply could look like this:

7050 <hvacSystemFunctionSetpointRelationListData>
 7051 <hvacSystemFunctionSetpointRelationData>

```

7052     <systemFunctionId>1</systemFunctionId>
7053     <operationModeId>1</operationModeId>
7054     <setpointId>1</setpointId>
7055 </hvacSystemFunctionSetpointRelationData>
7056 <hvacSystemFunctionSetpointRelationData>
7057     <systemFunctionId>1</systemFunctionId>
7058     <operationModeId>2</operationModeId>
7059     <setpointId>2</setpointId>
7060     <setpointId>3</setpointId>
7061 </hvacSystemFunctionSetpointRelationData>
7062 <hvacSystemFunctionSetpointRelationData>
7063     <systemFunctionId>1</systemFunctionId>
7064     <operationModeId>3</operationModeId>
7065     <setpointId>4</setpointId>
7066 </hvacSystemFunctionSetpointRelationData>
7067 </hvacSystemFunctionSetpointRelationListData>

```

5.3.12.5 hvacSystemFunctionPowerSequenceRelationListData

5.3.12.5.1 General

Power sequences (see section 5.3.19) describe tasks of a functionality in detail. An HVAC system function may relate to some of these power sequences. The relation is modelled with this function.

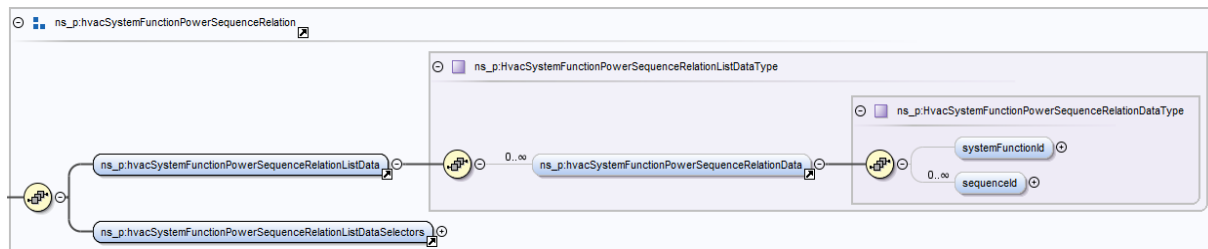


Figure 100: hvacSystemFunctionPowerSequenceRelationListData function overview

5.3.12.5.2 Detailed description of elements

Element	Type	Description
systemFunctionId	Identifier "HvacSystemFunctionIdType" (see Table 339).	Identifier for a specific HVAC system function.
sequenceId (list)	Identifier "PowerSequenceIdType" (see Table 339).	List of power sequences that are related with a specific HVAC system function.

Table 209: hvacSystemFunctionPowerSequenceRelationListData function detailed description of elements

5.3.12.5.3 Available selectors

- systemFunctionId

5.3.12.5.4 Examples

5.3.12.5.4.1 Read-reply

If one is interested in the related power sequences of all HVAC system functions, he could send a standard read command to the server. The reply could look like this:

```
<hvacSystemFunctionPowerSequenceRelationListData>
  <hvacSystemFunctionPowerSequenceRelationData>
    <systemFunctionId>1</systemFunctionId>
    <sequenceId>1</sequenceId>
    <sequenceId>2</sequenceId>
    <sequenceId>11</sequenceId>
  </hvacSystemFunctionPowerSequenceRelationData>
  <hvacSystemFunctionPowerSequenceRelationData>
    <systemFunctionId>2</systemFunctionId>
    <sequenceId>5</sequenceId>
  </hvacSystemFunctionPowerSequenceRelationData>
</hvacSystemFunctionPowerSequenceRelationListData>
```

5.3.12.6 hvacSystemFunctionDescriptionListData

5.3.12.6.1 General

The rather static information about the HVAC system functions are modelled within this SPINE function.

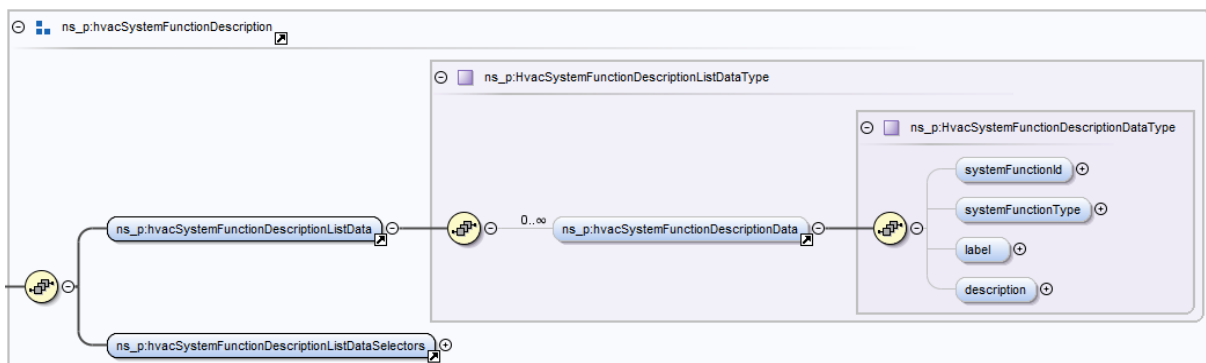


Figure 101: hvacSystemFunctionDescriptionListData function overview

5.3.12.6.2 Detailed description of elements

Element	Type	Description
systemFunctionId	Identifier "HvacSystemFunctionIdType" (see Table 339).	Identifier for a specific HVAC system function.
systemFunctionType	Union "HvacSystemFunctionTypeType": - Enum (see Table 211): HvacSystemFunctionTypeEnumType - EnumExtendType (see section 3.10.1.5)	Denotes the type of the HVAC system function.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the HVAC system function.

description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Descriptive information on the HVAC system function.
-------------	--	--

7107 *Table 210: hvacSystemFunctionDescriptionListData function detailed description of elements*

7108 Enumeration **HvacSystemFunctionTypeEnumType**:

Value	Description
heating	HVAC system function for space heating.
cooling	HVAC system function for space cooling.
ventilation	HVAC system function for ventilation.
dhw	HVAC system function for domestic hot water preparation.

7109 *Table 211: Enumeration HvacSystemFunctionTypeEnumType*

7110

7111 *5.3.12.6.3 Available selectors*

7112 - systemFunctionId

7113

7114 *5.3.12.6.4 Examples*

7115 *5.3.12.6.4.1 Read-reply*

7116 If one is interested in the description of all HVAC system functions, he could send a standard read
7117 command to the server. The reply could look like this:

```

7118 <hvacSystemFunctionDescriptionListData>
7119   <hvacSystemFunctionDescriptionData>
7120     <systemFunctionId>1</systemFunctionId>
7121     <systemFunctionType>heating</systemFunctionType>
7122     <label>Heating function</label>
7123   </hvacSystemFunctionDescriptionData>
7124   <hvacSystemFunctionDescriptionData>
7125     <systemFunctionId>2</systemFunctionId>
7126     <systemFunctionType>dhw</systemFunctionType>
7127     <label>Domestic hot water</label>
7128   </hvacSystemFunctionDescriptionData>
7129 </hvacSystemFunctionDescriptionListData>

```

7130

7131 **5.3.12.7 hvacOperationModeDescriptionListData**

7132 *5.3.12.7.1 General*

7133 HVAC operation modes are used to describe which behaviour the HVAC system functions have. Each
7134 HVAC operation mode may be used by several HVAC system functions.

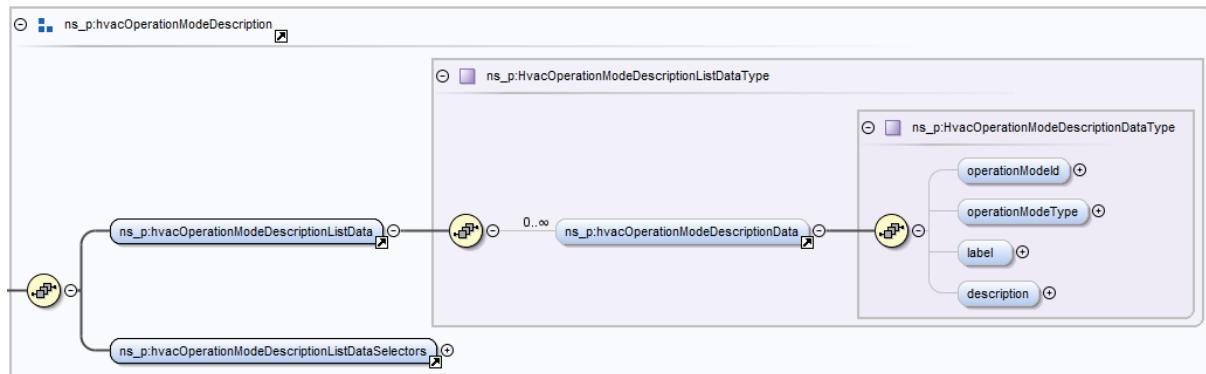


Figure 102: hvacOperationModeDescriptionListData function overview

5.3.12.7.2 Detailed description of elements

Element	Type	Description
operationModelId	Identifier "HvacOperationModelIdType" (see Table 339).	Identifier for the HVAC operation mode.
operationModeType	Union "HvacOperationModeTypeType": - Enum (see Table 213): HvacOperationModeTypeEnumType - EnumExtendType (see section 3.10.1.5)	Denotes the type of the HVAC operation mode.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the HVAC operation mode.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the HVAC operation mode.

Table 212: hvacOperationModeDescriptionListData function detailed description of elements

Enumeration HvacOperationModeTypeEnumType:

Value	Description	
auto	HVAC System Function	Meaning
	heating / cooling / dhw / ventilation	HVAC System function is controlled depending on a time table (i.e. the current setpoint for the HVAC system function is changed depending on a time table)
on	HVAC System Function	Meaning
	heating / cooling / dhw / ventilation	HVAC System function is controlled (e.g. Setpoint) independant from time tables (i.e. the current setpoint for the HVAC system function is not depending on any time table).
off	HVAC System Function	Meaning
	heating / dhw	Protection mode against freezing (no physically turning off). The (temperature) setpoint for freezing protection may be stated with a setpointId.
	cooling	Space cooling function not active.
eco	ventilation	Ventilation function not active (protection functions like humidity protection may be active).
	HVAC System Function	Meaning

	Heating / cooling / dhw / ventilation	HVAC System function is controlled (e.g. current Setpoint) independant from time tables (i.e. the current setpoint for the HVAC system function is not depending on any time table). HVAC System function is controlled by a different setpoint as for “on”
--	---------------------------------------	--

Table 213: Enumeration *HvacOperationModeTypeEnumType*

5.3.12.7.3 Available selectors

- operationModeId

5.3.12.7.4 Examples

5.3.12.7.4.1 Read-reply

If one is interested in the description of all HVAC operation modes, he could send a standard read command to the server. The reply could look like this:

```
<hvacOperationModeDescriptionListData>
  <hvacOperationModeDescriptionData>
    <operationModeId>1</operationModeId>
    <operationModeType>on</operationModeType>
  </hvacOperationModeDescriptionData>
  <hvacOperationModeDescriptionData>
    <operationModeId>2</operationModeId>
    <operationModeType>off</operationModeType>
  </hvacOperationModeDescriptionData>
  <hvacOperationModeDescriptionData>
    <operationModeId>3</operationModeId>
    <operationModeType>auto</operationModeType>
  </hvacOperationModeDescriptionData>
</hvacOperationModeDescriptionListData>
```

5.3.12.8 hvacOverrunListData

5.3.12.8.1 General

The normal HVAC operation may be overridden by some HVAC specific functionality. There are some HVAC overruns defined by the EEBus Initiative e.V., but vendors may add further vendor-specific HVAC overruns, too (which are then out of interoperability scope).

When the HVAC overrun is active, some (not all! See section 5.3.12.9) HVAC system functions have a specific behaviour, other than the normal operation would be. After the HVAC overrun is inactive again, the HVAC system returns to its normal operation.

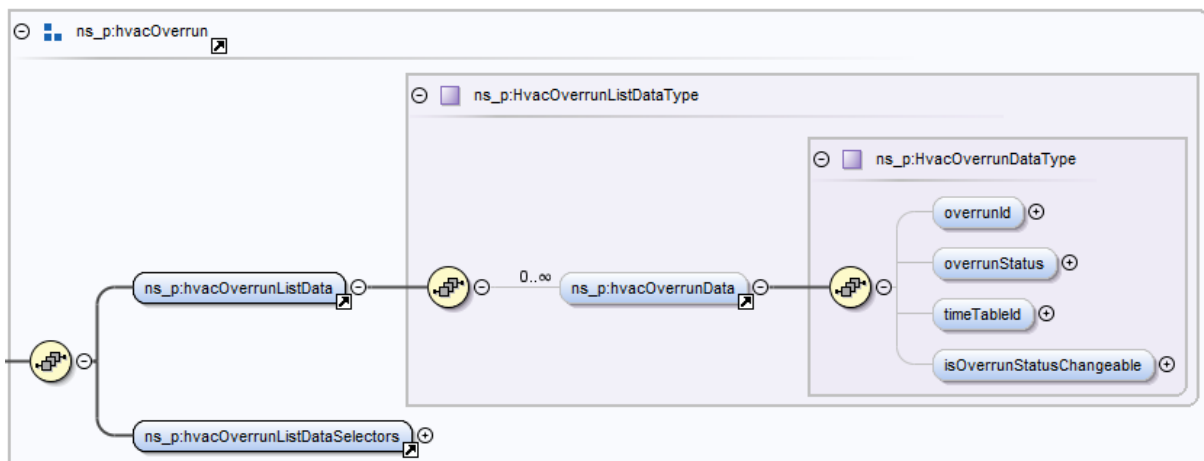


Figure 103: hvacOverrunListData function overview

5.3.12.8.2 Detailed description of elements

Element	Type	Description
overrunId	Identifier "HvacOverrunIdType" (see Table 339).	Identifier for the HVAC overrun.
overrunStatus	Union "HvacOverrunStatusType": - Enum (see Table 215): HvacOverrunStatusEnumType - EnumExtendType (see section 3.10.1.5)	The current status of the HVAC overrun.
timeTableId	Identifier "TimeTableIdType" (see Table 339).	Reference to a time table. Needed, if an overrun has some start and end point(s).
isOverrunStatusChangeable	xs:boolean (W3C standard type)	Whether the overrun status is changeable by a client or not is denoted in this element.

Table 214: hvacOverrunListData function detailed description of elements

Enumeration HvacOverrunStatusEnumType:

Value	Description
active	HVAC overrun triggered, but not started yet.
running	HVAC overrun running.
finished	HVAC overrun finished.
inactive	HVAC overrun inactive.

Table 215: Enumeration HvacOverrunStatusEnumType

5.3.12.8.3 Available selectors

- overrunId

5.3.12.8.4 Examples

5.3.12.8.4.1 Read-reply

If one is interested in the status of all HVAC overruns, he could send a standard read command to the server. The reply could look like this:

```
<hvacOverrunListData>
  <hvacOverrunData>
    <overrunId>1</overrunId>
    <overrunStatus>inactive</overrunStatus>
    <timeTableId>5</timeTableId>
  </hvacOverrunData>
  <hvacOverrunData>
    <overrunId>2</overrunId>
    <overrunStatus>active</overrunStatus>
  </hvacOverrunData>
</hvacOverrunListData>
```

5.3.12.9 hvacOverrunDescriptionListData

5.3.12.9.1 General

Each HVAC overrun is defined by its description part. Here, the HVAC overrun type is specified as well as the list of HVAC system functions that are affected by the HVAC overrun (if this information is available).

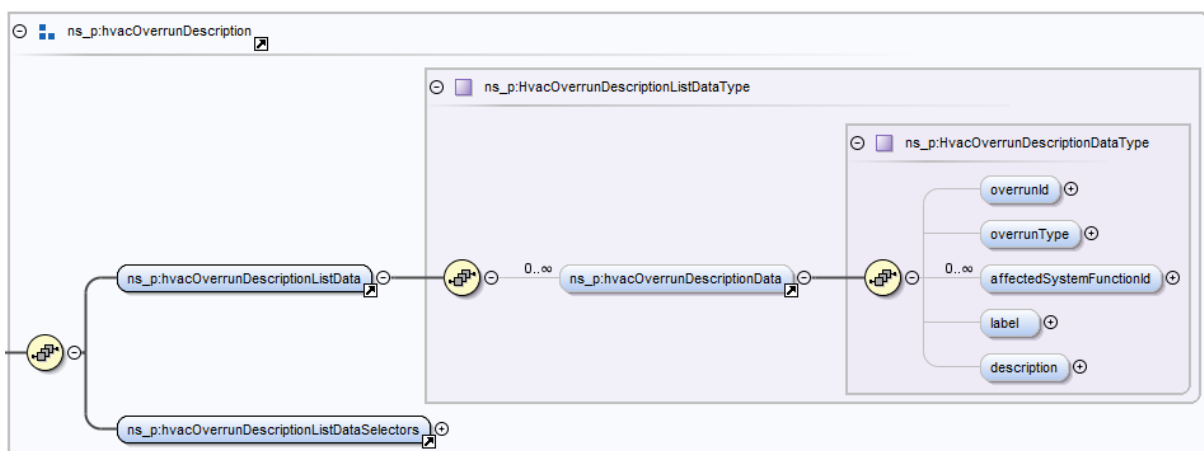


Figure 104: hvacOverrunDescriptionListData function overview

5.3.12.9.2 Detailed description of elements

Element	Type	Description
overrunId	Identifier "HvacOverrunIdType" (see Table 339).	Identifier for the HVAC overrun.
overrunType	Union "HvacOverrunTypeType": - Enum (see Table 217): HvacOverrunTypeEnumType - EnumExtendType (see section 3.10.1.5)	The type of the overrun.

affectedSystemFunctionId (list)	Identifier "HvacSystemFunctionIdType" (see Table 339).	Identifiers of all HVAC system functions that are affected by this HVAC overrun.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the HVAC overrun.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the HVAC overrun.

7209 Table 216: hvacOverrunDescriptionListData function detailed description of elements

7210 Enumeration **HvacOverrunTypeEnumType**:

Value	Description
oneTimeDhw	A one-time domestic hot water loading (independend from current HVAC operation mode or time table).
party	During a party, the HVAC system will stay in the normal day-operation mode and not turn off at the usual time. Example: HVAC system function "heating" is controlled by the day setpoint, independend from operation mode and time table, until HVAC overrun "party" is finished.
sgReadyCondition1	The SGReady condition 1 will be applied. Please consider [SGReady].
sgReadyCondition3	The SGReady condition 3 will be applied. Please consider [SGReady].
sgReadyCondition4	The SGReady condition 4 will be applied. Please consider [SGReady].
oneDayAway	Overrides the normal daily routine to a out-of-home configuration (e.g. eco-mode the whole day).
oneDayAtHome	Overrides the normal daily routine to an at-home configuration (e.g. on-mode the whole day).
oneTimeVentilation	Activates the ventilation once, independent from the configured routine.
hvacSystemOff	Deactivates the HVAC system (frost protection is still active and the system is in a network-standby mode, in which it is reachable for re-configuration).

7211 Table 217: Enumeration HvacOverrunTypeEnumType

7212

7213 5.3.12.9.3 Available selectors

7214 - overrunId

7215

7216 5.3.12.9.4 Examples

7217 5.3.12.9.4.1 Read-reply

7218 If one is interested in the description of all HVAC overruns, he could send a standard read command
7219 to the server. The reply could look like this:

```

7220 <hvacOverrunDescriptionListData>
7221   <hvacOverrunDescriptionData>
7222     <overrunId>1</overrunId>
7223     <overrunType>party</overrunType>
7224     <affectedSystemFunctionId>1</affectedSystemFunctionId>
7225     <affectedSystemFunctionId>2</affectedSystemFunctionId>
7226     <label>label0</label>
7227     <description>description0</description>
7228   </hvacOverrunDescriptionData>
7229   <hvacOverrunDescriptionData>
```

```

7230         <overrunId>2</overrunId>
7231         <overrunType>oneTimeDhw</overrunType>
7232         <affectedSystemFunctionId>2</affectedSystemFunctionId>
7233         <label>label1</label>
7234         <description>description1</description>
7235     </hvacOverrunDescriptionData>
7236 </hvacOverrunDescriptionListData>

```

5.3.13 Identification

5.3.13.1 Introduction

The Identification Class can be used to express certain identification values of the parent Entity.



Figure 105: Identification function-group overview

5.3.13.2 identificationListData

5.3.13.2.1 General

The identificationListData function allows to express certain identification values of the corresponding Entity (the parent Entity of the Identification Feature).

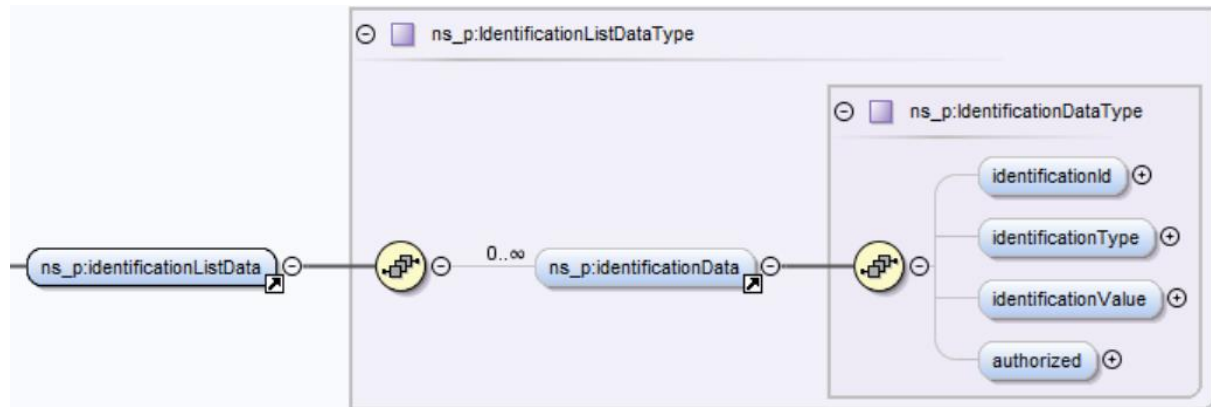


Figure 106: identificationListData function overview

5.3.13.2.2 Detailed description of elements

Element	Type	Description
identificationId	Identifier "IdentificationIdType" (see Table 339).	The identificationId is used to identify the different list entries within IdentificationListData.
identificationType	Union "IdentificationTypeType": - Enum (see Table 219): IdentificationTypeEnumType - EnumExtendType (see section 3.10.1.5)	The identificationType allows to describe a certain type of identification, e.g. a MAC address or an RFID tag.

identificationValue	<i>Simple type</i> <i>"IdentificationValueType"</i> <i>(restriction of "xs:string")</i>	This holds the value of the identification and therefore is the essential element of IdentificationData.
authorized	xs:boolean	The "authorized" element can be used to directly authorize an identification value with a write command.

7252 Table 218: identificationListData Element rules

7253 Enumeration **IdentificationTypeEnumType**:

Value	Description
eui48	eui48 MAC address
eui64	eui64 MAC address
userRfidTag	RFID Tag

7254 Table 219: Enumeration IdentificationTypeEnumType

7255

7256 5.3.13.2.3 Available selectors

7257 - identificationId

7258

7259 5.3.13.2.4 Examples

7260 5.3.13.2.4.1 Notify

7261 If the content of a list entries elements changes, the information is sent to all subscribed resources:

```

7262 <identificationListData>
7263   <identificationData>
7264     <identificationId>1</identificationId>
7265     <identificationType>eui64</identificationType>
7266     <identificationValue>12-AB-34-CD-56-EF-78-90</identificationValue>
7267     <authorized>true</authorized>
7268   </identificationData>
7269 </identificationListData>

```

7270

7271 5.3.14 LoadControl

7272 5.3.14.1 Introduction

7273 The LoadControl class has two different approaches to control the load of an appliance:

- 7274 - Event-based load control with "loadControlEvents" (see section 5.3.14.3) and
- 7275 "loadControlState" (see section 5.3.14.4)
- 7276 - Limit-based load control with "loadControlLimit" (see section 5.3.14.5) and
- 7277 "loadControlLimitDescription" (see section 5.3.14.7)

7278 The event-based load control is used to change the energy demand (load) or production of a

7279 household by some external contract partner or energy manager. The household or device (owner)

7280 may reject the change commands, but then may have to pay more to the external partner or receive

7281 less reward. The concept defines two function-groups. One (*loadControlEvents*) that keeps the

received change commands (a household or device may store more than one command, which may be active in different time periods; these commands are requestable by some other client side) and another one (*loadControlState*) for the status of these commands (where more than one could be requestable, too). Possible commands are: pausing / resuming the demand or production, decreasing / increasing it, an emergency mode (where the electricity grid or local house power supply is in a critical condition and only an immediate substantial reduction of the energy consumption or feeding of energy into the electricity grid can prevent a blackout situation) and the "back-to-normal" command.

The limit-based load control is used where limits (changeable by a client) shall affect the behaviour of a node automatically. E.g. the power demand of an appliance could be limited to a certain value to protect the household of an overload.

Additionally with the information given in *loadControlNode* (see section 5.3.14.2), a node can define whether it is (at the moment) remote controllable or not.

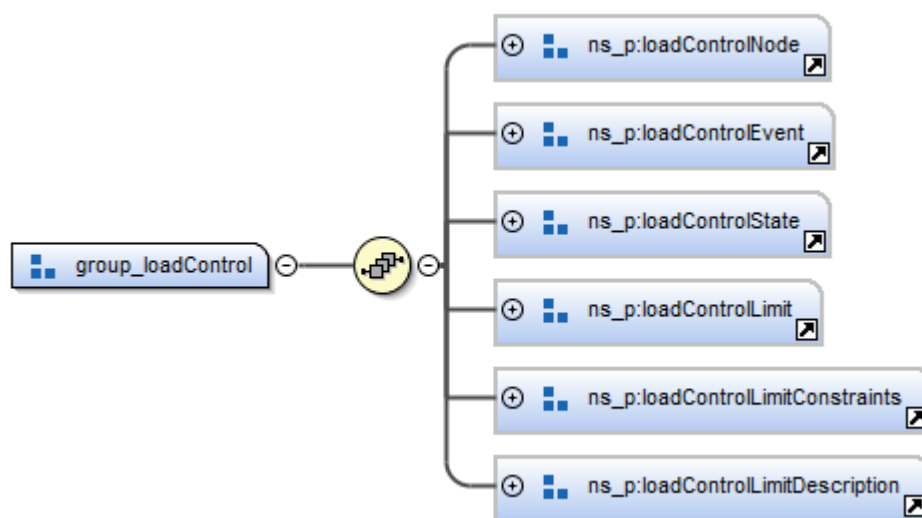


Figure 107: LoadControl function-group overview

5.3.14.2 loadControlNodeData

5.3.14.2.1 General

The *loadControlNodeData* includes node-wide information related to load control (e.g. whether the node is remote controllable).

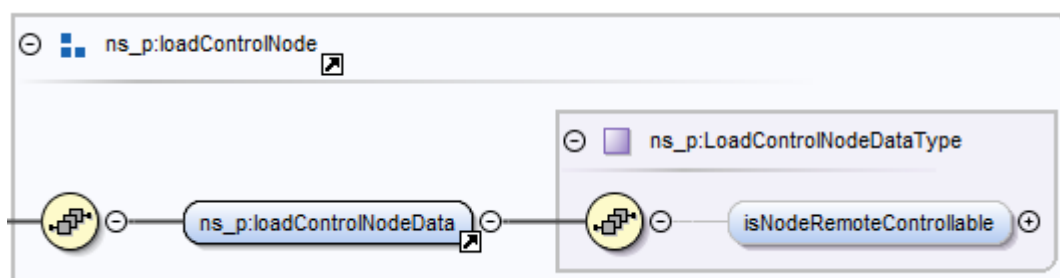


Figure 108: loadControlNodeData function overview

7304

7305 *5.3.14.2.2 Detailed description of elements*

Element	Type	Description
isNodeRemoteControllable	xs:boolean (W3C standard type)	Defines whether the node is (at the moment) remote controllable by a client, or not.

7306 *Table 220: loadControlNodeData function detailed description of elements*

7307

7308 *5.3.14.2.3 Available selectors*

7309 None.

7310

7311 *5.3.14.2.4 Examples*7312 *5.3.14.2.4.1 Notify*

7313 If the content of the element isNodeRemoteControllable changes (e.g. due to a configuration change
7314 by the user), the information is sent to all subscribed resources:

```
7315 <loadControlNodeData>
7316   <isNodeRemoteControllable>true</isNodeRemoteControllable>
7317 </loadControlNodeData>
```

7318

7319 **5.3.14.3 loadControlEventListData**7320 *5.3.14.3.1 General*

7321 With the loadControlEventListData function, a household or device can show the commands (events)
7322 it received to change its load demand or energy production (or both). These commands contain
7323 rather simple instructions like "reduce", "increase", "pause", "resume", etc. This function can also be
7324 used to model a single command from another point (external contract partner or energy manager,
7325 e.g.) to the household or device (this could be a "write" operation to add a new
7326 "loadControlEventData" to the loadControlEventListData instance, e.g.).

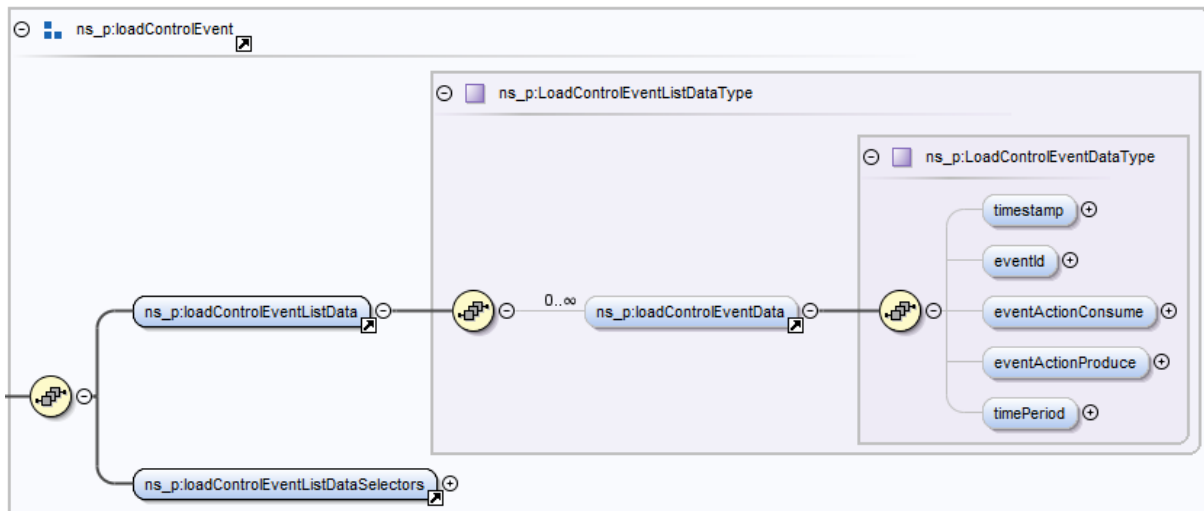


Figure 109: loadControlEventListData function overview

5.3.14.3.2 Detailed description of elements

Element	Type	Description
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The time when the message was generated.
eventId	Identifier "LoadControlEventIdType" (see Table 339).	An incrementing number, enabling the linking of a <i>loadControlState</i> command to a <i>loadControlEvent</i> command.
eventActionConsume	Union "LoadControlEventActionType": - Enum (see Table 222): LoadControlEventActionEnumType - EnumExtendType (see section 3.10.1.5)	The action for consuming that shall be applied by the receiving device (feature server).
eventActionProduce	Union "LoadControlEventActionType": - Enum (see Table 222): LoadControlEventActionEnumType - EnumExtendType (see section 3.10.1.5)	The action for producing that shall be applied by the receiving device (feature server).
timePeriod	Common data type "TimePeriodType". See section 3.10.2.1.	The period of time, the event shall be active (e.g. from 12:00 to 14:00 or with relative times, from 1 hour in the future to 3 hours in the future).

Table 221: loadControlEventListData function detailed description of elements

Enumeration LoadControlEventActionEnumType:

Value	Description
pause	Demanding the pausing of consumption / production.
resume	Demanding the resumption of consumption / production.
reduce	Demanding the reduction of consumption / production.
increase	Demanding the increasing of consumption / production.

emergency	Demanding the reduction of consumption to the absolutely minimum level that is possible without risking anything. Not to be used for production.
normal	Demanding the return to the normal operation.

7333 Table 222: Enumeration LoadControlEventActionEnumType

7334

7335 5.3.14.3.3 Available selectors

7336 - timestampInterval

7337 - eventId

7338

7339 5.3.14.3.4 Examples

7340 5.3.14.3.4.1 Write (partial)

7341 If an energy provider wants a household to reduce its consumption and increase its production for a
7342 certain period of time, it could send the following command to the CEM:

```

7343 <function>loadControlEventListData</function>
7344 <filter>
7345   <cmdControl>
7346     <partial/>
7347   </cmdControl>
7348 </filter>
7349 <loadControlEventListData>
7350   <loadControlEventData>
7351     <timestamp>2015-10-15T18:40:23.2Z</timestamp>
7352     <eventId>1</eventId>
7353     <eventActionConsume>reduce</eventActionConsume>
7354     <eventActionProduce>increase</eventActionProduce>
7355     <timePeriod>
7356       <startTime>PT1H10M</startTime>
7357       <endTime>PT3H15M</endTime>
7358     </timePeriod>
7359   </loadControlEventData>
7360 </loadControlEventListData>

```

7361

7362 5.3.14.4 loadControlStateListData

7363 5.3.14.4.1 General

7364 A device that supports being controlled by LoadControl mechanisms (see function
7365 "loadControlEventListData") should express the states it currently assigns to each event.

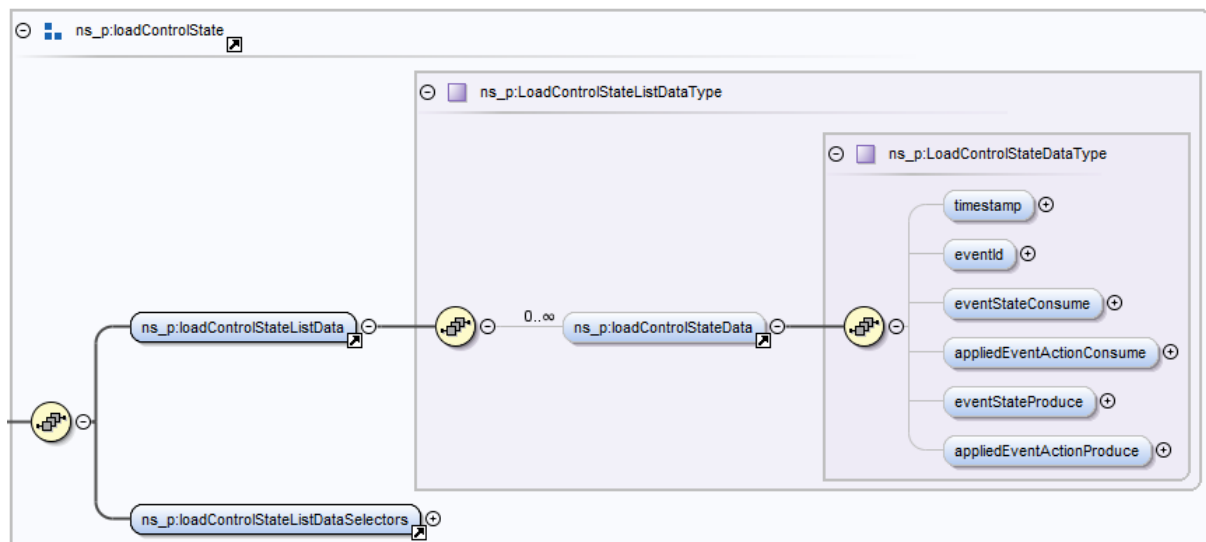


Figure 110: loadControlStateListData function overview

5.3.14.4.2 Detailed description of elements

Element	Type	Description
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The time when the message was generated.
eventId	Identifier "LoadControlEventIdType" (see Table 339).	An incrementing number, enabling the linking of a <i>loadControlState</i> command to a <i>loadControlEvent</i> command.
eventStateConsume	Union "LoadControlEventStateType": - Enum (see Table 224): LoadControlEventStateEnumType - EnumExtendType (see section 3.10.1.5)	Current state for consumption of the event.
appliedEventActionConsume	Union "LoadControlEventActionType": - Enum (see Table 222): LoadControlEventActionEnumType - EnumExtendType (see section 3.10.1.5)	The event action for consumption, actually applied by the server.
eventStateProduce	Union "LoadControlEventStateType": - Enum (see Table 224): LoadControlEventStateEnumType - EnumExtendType (see section 3.10.1.5)	Current state for production of the event.
appliedEventActionProduce	Union "LoadControlEventActionType": - Enum (see Table 222): LoadControlEventActionEnumType - EnumExtendType (see section 3.10.1.5)	The event action for production, actually applied by the server.

7370 *Table 223: loadControlStateListData function detailed description of elements*

7371 Enumeration **LoadControlEventStateEnumType**:

Value	Description
eventAccepted	The server accepted the event command. It has not started yet!
eventStarted	The server started the event, meaning that, e.g., the server has started to reduce its consumption or is increasing its production.
eventStopped	The server stopped the event due to reaching the end time, of the event (stated in the <i>timePeriod</i> element of the <i>loadControlEvent</i> command).
eventRejected	The server rejected the command due to some reason, not stated here. It depends on the contract of the household owner how often he may reject a <i>loadControlEvent</i> command.
eventCancelled	The server cancelled the event due to some reason, not stated here.
eventError	An error occurred during executing the event.

7372 *Table 224: Enumeration LoadControlEventStateEnumType*

7373

7374 *5.3.14.4.3 Available selectors*

7375 - timestampInterval

7376 - eventId

7377

7378 *5.3.14.4.4 Examples*

7379 *5.3.14.4.4.1 Notify*

7380 As a result of the load-reducing-command example from section 5.3.14.3.4.1, the CEM could inform
7381 the DSO with the following (complete) notification:

```

7382 <loadControlStateListData>
7383   <loadControlStateData>
7384     <timestamp>2015-10-15T18:40:23.9Z</timestamp>
7385     <eventId>1</eventId>
7386     <eventStateConsume>eventAccepted</eventStateConsume>
7387     <appliedEventActionConsume>reduce</appliedEventActionConsume>
7388     <eventStateProduce>eventAccepted</eventStateProduce>
7389     <appliedEventActionProduce>increase</appliedEventActionProduce>
7390   </loadControlStateData>
7391 </loadControlStateListData>

```

7392 Following this example, exact one hour and ten minutes later, the CEM would send the following
7393 (complete) notification to the DSO:

```

7394 <loadControlStateListData>
7395   <loadControlStateData>
7396     <timestamp>2015-10-15T19:50:23.9Z</timestamp>
7397     <eventId>1</eventId>
7398     <eventStateConsume>eventStarted</eventStateConsume>
7399     <appliedEventActionConsume>reduce</appliedEventActionConsume>
7400     <eventStateProduce>eventStarted</eventStateProduce>
7401     <appliedEventActionProduce>increase</appliedEventActionProduce>
7402   </loadControlStateData>
7403 </loadControlStateListData>

```

7404

5.3.14.5 loadControlLimitListData

5.3.14.5.1 General

The loadControlLimitListData function provides the value of the limit (and an optional time period for that limit) and further information whether the limit is active or not and if it is changeable by a client or not.

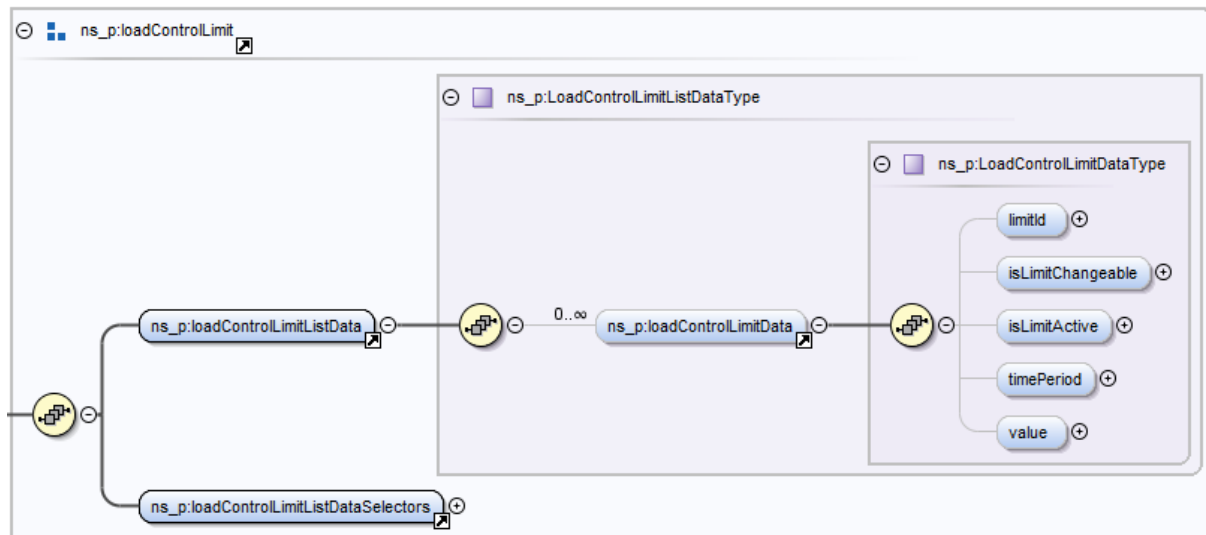


Figure 111: loadControlLimitListData function overview

5.3.14.5.2 Detailed description of elements

Element	Type	Description
limitId	Identifier "LoadControlLimitIdType" (see Table 339).	Identifier for the limit.
isLimitChangeable	xs:boolean (W3C standard type)	States whether the limit may be changed by a client or not.
isLimitActive	xs:boolean (W3C standard type)	Indicates whether the limit is currently active or not.
timePeriod	Common data type "TimePeriodType". See section 3.10.2.1.	The period where the limit shall be active.
value	Common data type "ScaledNumberType". See section 3.10.1.8.	The actual limit.

Table 225: loadControlLimitListData function detailed description of elements

5.3.14.5.3 Available selectors

- limitId

5.3.14.5.4 Examples

5.3.14.5.4.1 Read-reply

If one is interested in the current value of a specific limit, he could send the following read command:

```

7422 <function>loadControlLimitListData</function>
7423 <filter>
7424     <cmdControl>
7425         <partial/>
7426     </cmdControl>
7427     <loadControlLimitListDataSelectors>
7428         <limitId>2</limitId>
7429     </loadControlLimitListDataSelectors>
7430 </filter>
7431 <loadControlLimitListData/>

```

7432 If the receiving device has a proper value for this *keyId*, it would respond with the following function:

```

7433 <loadControlLimitListData>
7434     <loadControlLimitData>
7435         <limitId>2</limitId>
7436         <isLimitActive>true</isLimitActive>
7437         <isLimitChangeable>true</isLimitChangeable>
7438         <timePeriod>
7439             <startTime>-PT13M30S</startTime>
7440             <endTime>PT16M30S</endTime>
7441         </timePeriod>
7442         <value>
7443             <number>15</number>
7444             <scale>1</scale>
7445         </value>
7446     </loadControlLimitData>
7447 </loadControlLimitListData>

```

7448

7449 5.3.14.5.4.2 Write (partial)

7450 If an client wants to limit the consumption of an appliance to a certain value that shall begin in 15
7451 minutes and end in 2 hours, it could send the following command to the load control resource:

```

7452 <function>loadControlLimitListData</function>
7453 <filter>
7454     <cmdControl>
7455         <partial/>
7456     </cmdControl>
7457 </filter>
7458 <loadControlLimitListData>
7459     <loadControlLimitData>
7460         <limitId>2</limitId>
7461         <timePeriod>
7462             <startTime>PT15M</startTime>
7463             <endTime>PT2H</endTime>
7464         </timePeriod>
7465         <value>
7466             <number>1</number>
7467             <scale>2</scale>
7468         </value>
7469     </loadControlLimitData>
7470 </loadControlLimitListData>

```

7471

5.3.14.6 loadControlLimitConstraintsListData

5.3.14.6.1 General

Constraints that shall be held when trying to change a limit, are modelled with the loadControlLimitConstraintsListData function, including a minimum value, a maximum value and the stepsize of the values between.

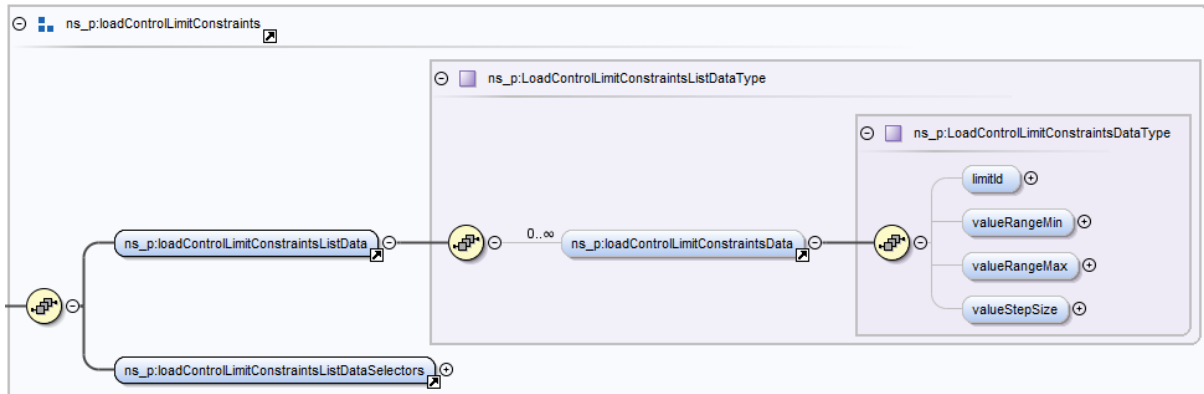


Figure 112: loadControlLimitConstraintsListData function overview

5.3.14.6.2 Detailed description of elements

Element	Type	Description
limitId	Identifier "LoadControlLimitIdType" (see Table 339).	Identifier for the limit.
valueRangeMin	Common data type "ScaledNumberType". See section 3.10.1.8.	This element allows to define a minimum limit for the element value within loadControlLimitListData.
valueRangeMax	Common data type "ScaledNumberType". See section 3.10.1.8.	This element allows to define a maximum limit for the element value within loadControlLimitListData.
valueStepSize	Common data type "ScaledNumberType". See section 3.10.1.8.	The minimum step size between two different limit values.

Table 226: loadControlLimitConstraintsListData function detailed description of elements

5.3.14.6.3 Available selectors

- limitId

5.3.14.6.4 Examples

5.3.14.6.4.1 Read-reply

If one is interested in the value constraints of a specific limit, he could send the following read command:

```
<function>loadControlLimitConstraintsListData</function>
<filter>
```

```

7492     <cmdControl>
7493         <partial/>
7494     </cmdControl>
7495     <loadControlLimitConstraintsListDataSelectors>
7496         <limitId>2</limitId>
7497     </loadControlLimitConstraintsListDataSelectors>
7498 </filter>
7499 <loadControlLimitConstraintsListData/>

```

7500 If the receiving device has a proper value for this *limitId*, it would respond with the following
7501 function:

```

7502 <loadControlLimitConstraintsListData>
7503     <loadControlLimitConstraintsData>
7504         <limitId>2</limitId>
7505         <valueRangeMin>
7506             <scaledNumber>
7507                 <number>0</number>
7508             </scaledNumber>
7509         </valueRangeMin>
7510         <valueRangeMax>
7511             <scaledNumber>
7512                 <number>36</number>
7513                 <scale>2</scale>
7514             </scaledNumber>
7515         </valueRangeMax>
7516         <valueStepSize>
7517             <scaledNumber>
7518                 <number>5</number>
7519                 <scale>-1</scale>
7520             </scaledNumber>
7521         </valueStepSize>
7522     </loadControlLimitConstraintsData>
7523 </loadControlLimitConstraintsListData>

```

7524

7525 **5.3.14.7 loadControlLimitDescriptionListData**

7526 **5.3.14.7.1 General**

7527 The rather static information about the available limit(s) are modelled within the
7528 loadControlLimitDescriptionListData function.

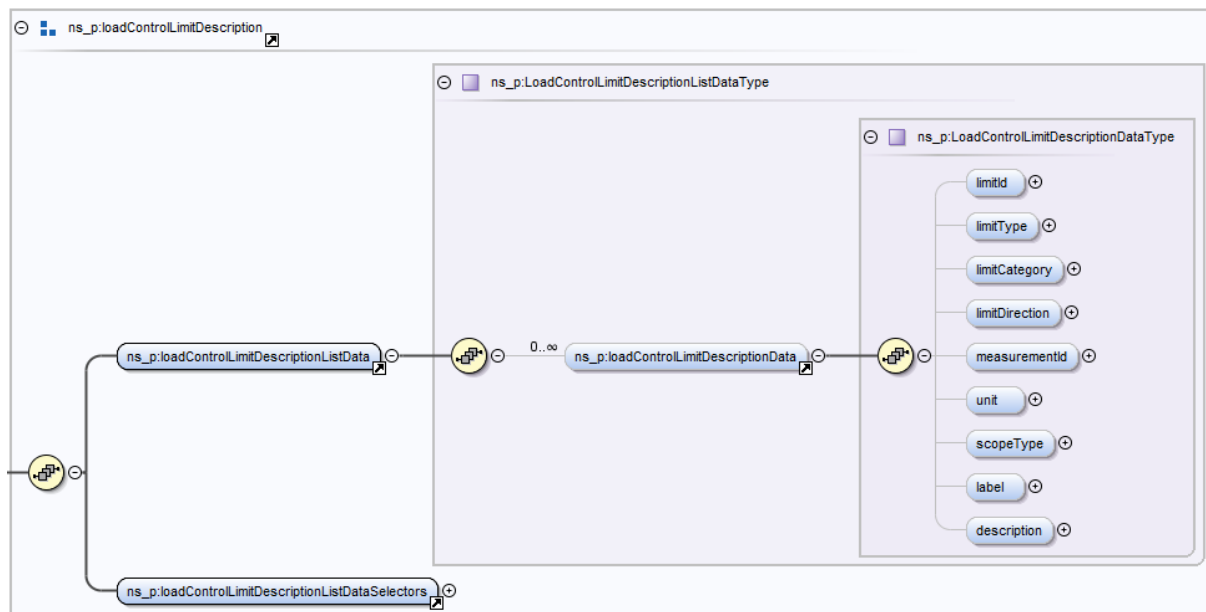


Figure 113: loadControlLimitDescriptionListData function overview

5.3.14.7.2 Detailed description of elements

Element	Type	Description
limitId	Identifier "LoadControlLimitIdType" (see Table 339).	Identifier for the limit.
limitType	Union "LoadControlLimitTypeType": - Enum (see Table 228): LoadControlLimitTypeEnumType - EnumExtendType (see section 3.10.1.5)	Whether the limit shall be used as a minimum or maximum is denoted within this element.
limitCategory	Union "LoadControlLimitCategoryType": - Enum (see Table 229): LoadControlLimitCategoryEnumType - EnumExtendType (see section 3.10.1.5)	There are different levels of how important the limit is (see enumeration description).
limitDirection	Common data type "EnergyDirectionType". See section 3.10.1.13.	If a device supports both, energy consumption and energy production, the direction which shall be limited is stated in this element (e.g. if only the consumption of a battery shall be limited, the enumeration "consume" will be set; if both, consumption and production shall be limited, two limits have to be set).
measurementId	Identifier "MeasurementIdType" (see Table 339).	If a measurand is linked to the limit, its identifier is denoted here.

unit	<i>Common data type "UnitOfMeasurementType". See section 3.10.1.17.</i>	The unit, which is used for the limit value, is denoted with this element.
scopeType	<i>Common data type "ScopeTypeType". See section 3.10.1.21.</i>	A certain meaning of the limit.
label	<i>Common data type "LabelType". See section 3.10.1.2.</i>	A label for the limit.
description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	A description for the limit.

7533 Table 227: loadControlLimitDescriptionListData function detailed description of elements

7534 Enumeration **LoadControlLimitTypeEnumType**:

Value	Description
minValueLimit	The limit set in the "value" shouldn't be underrun (value > minValueLimit).
maxValueLimit	The limit set in the "value" shouldn't be overrun (value < maxValueLimit).

7535 Table 228: Enumeration LoadControlLimitTypeEnumType

7536 Enumeration **LoadControlLimitCategoryEnumType**:

Value	Description
obligation	Used for limits that are absolutely important to be hold. A loss of comfort could happen and will be accepted. The normal operation will be interrupted for keeping the limit (if needed).
recommendation	Used for limits that are quite important for the energy management. A loss of comfort could happen and will be accepted for a short period of time. The normal operation should not be interrupted.
optimization	Used for limits that just optimize the energy management. No loss of comfort will be accepted. The normal operation shall not be interrupted.

7537 Table 229: Enumeration LoadControlLimitCategoryEnumType

7538

7539 5.3.14.7.3 Available selectors

- 7540 - limitId
- 7541 - limitType
- 7542 - limitDirection
- 7543 - measurementId
- 7544 - scopeType

7545

7546 5.3.14.7.4 Examples

7547 5.3.14.7.4.1 Read-reply

7548 A loadControlLimitDescriptionListData instance sent as reply to a standard read command:

```

7549 <loadControlLimitDescriptionListData>
7550   <loadControlLimitDescriptionData>
7551     <limitId>1</limitId>
7552     <limitType>maxValueLimit</limitType>
```

```

7553         <limitCategory>obligation</limitCategory>
7554         <limitDirection>consume</limitDirection>
7555         <measurementId>3</measurementId>
7556         <unit>W</unit>
7557     </loadControlLimitDescriptionData>
7558     <loadControlLimitDescriptionData>
7559         <limitId>2</limitId>
7560         <limitType>maxValueLimit</limitType>
7561         <limitCategory>obligation</limitCategory>
7562         <limitDirection>produce</limitDirection>
7563         <measurementId>5</measurementId>
7564         <unit>W</unit>
7565     </loadControlLimitDescriptionData>
7566 </loadControlLimitDescriptionListData>

```

5.3.15 Measurement

5.3.15.1 Introduction

The SPINE Measurement class provides functionality for transmitting measured, calculated or empirically obtained values from sensors. Mainly these are sensors that provide some kind of information about the environment or physical parameters. Clients can use the information for further actions or just visualize them for the information of a user.

Due to the nature of measurement, the measured values are only readable. Some control systems permit a user to select a preferred value (e.g. a desired room temperature). Such “desired” values obviously require writable fields. This can be achieved with the "Setpoint" class (see section 5.3.21).

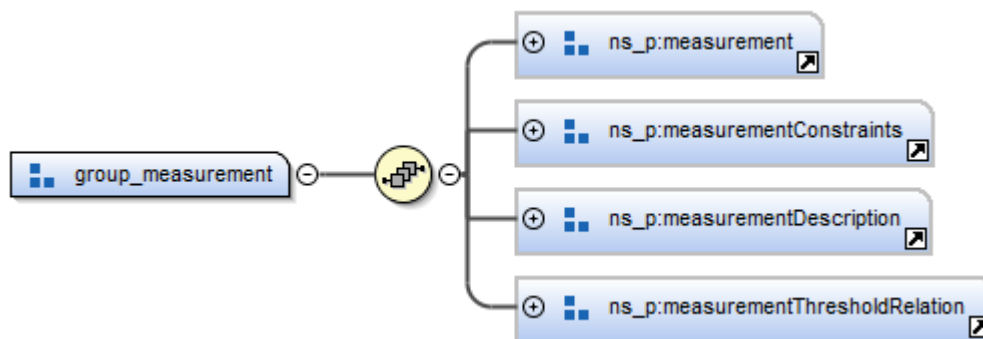


Figure 114: Measurement function-group overview

5.3.15.2 measurementListData

5.3.15.2.1 General

The *measurementListData* command is used to communicate measurement values. Typically, a sensor sends this command as notification as soon as it changes more than a specific threshold. Of course, it may be readable, too.

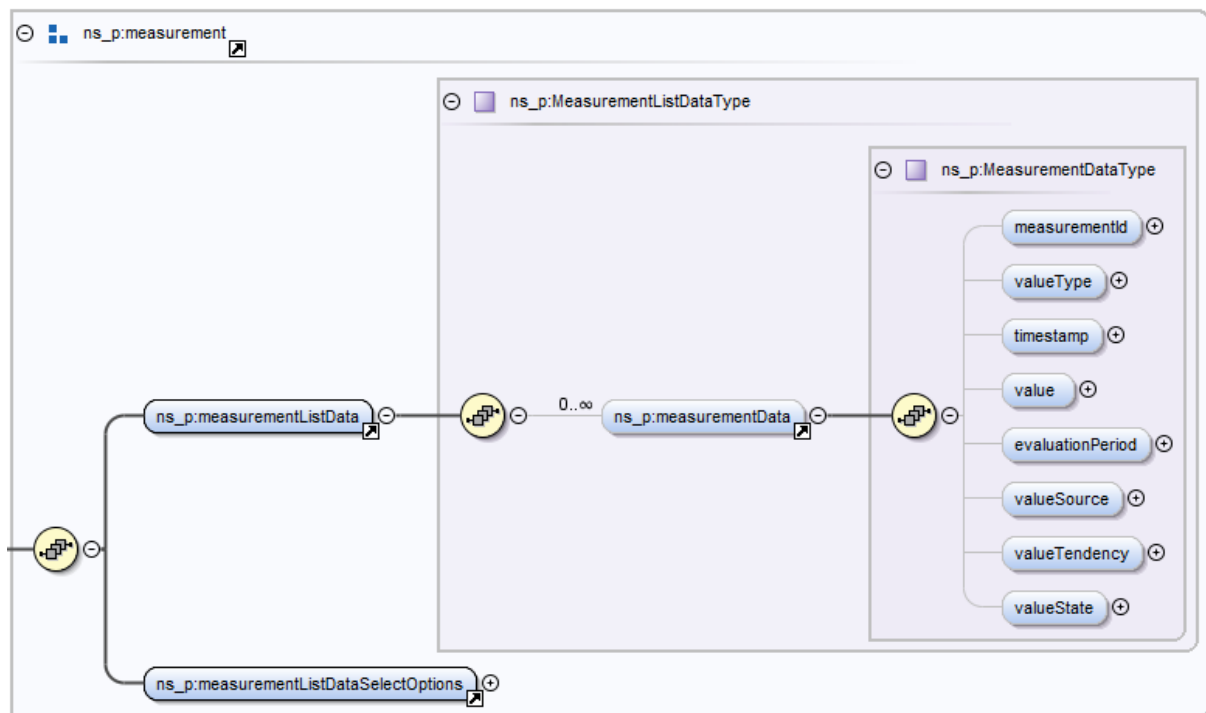


Figure 115: measurementListData function overview

5.3.15.2.2 Detailed description of elements

Element	Type	Description
measurementId	Identifier "MeasurementIdType" (see Table 339).	Enables the identification of different sensors on one SPINE feature.
valueType	Union "MeasurementValueTypeType": - Enum (see Table 231): MeasurementValueTypeEnumType - EnumExtendType (see section 3.10.1.5)	It is possible to model different types of measurement values. For example "value", "averageValue", "minValue", etc. (see the SPINE data model for all possible value types).
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The time of creation of a measurement value element (i.e. the time it was measured, calculated, ...). Some sensors and communications technologies may already provide this information natively. In other cases the technology interface may set timestamp to the time of creation of a measurementData instance.
value	Common data type "ScaledNumberType". See section 3.10.1.8.	The measurement value itself, i.e. the magnitude according to "valueType".
evaluationPeriod	Common data type "TimePeriodType". See section 3.10.2.1.	Some <i>valueType</i> values can be detailed or even require information about the time period. For example, an <i>averageValue</i> is

		usually computed from measured values of a finite time period. For smart devices, the start and end time of the <i>evaluationPeriod</i> can be filled in directly. For bare ("simple") sensors, such "derived"/"accompanying" values (average, min, max, ...) could be calculated (if desired) by the technology interface component together with the corresponding <i>evaluationPeriod</i> .
valueSource	<i>Union</i> "MeasurementValueSourceType": - Enum (see Table 232): MeasurementValueSourceEnumType - EnumExtendType (see section 3.10.1.5)	Whether element "value" is measured, calculated or empirically obtained can be denoted with this element.
valueTendency	<i>Union</i> "MeasurementValueTendencyType": - Enum (see Table 233): MeasurementValueTendencyEnumType - EnumExtendType (see section 3.10.1.5)	This elements states whether the tendency is <i>rising</i> , <i>stable</i> or <i>falling</i> .
valueState	<i>Union</i> "MeasurementValueStateType": - Enum (see Table 234): MeasurementValueStateEnumType - EnumExtendType (see section 3.10.1.5)	A measurand may have a state.

7589 Table 230: measurementListData function detailed description of elements

7590 Enumeration **MeasurementValueTypeEnumType**:

Value	Description
value	The normal measurement value.
averageValue	The average measurement value over the last time, stated in the element evaluationPeriod.
minValue	The minimum measurement value of the last time, stated in the element evaluationPeriod.
maxValue	The maximum measurement value of the last time, stated in the element evaluationPeriod.
standardDeviation	Used to quantify the amount of variation or dispersion of a set of values.

7591 Table 231: Enumeration MeasurementValueTypeEnumType

7592 Enumeration **MeasurementValueSourceEnumType**:

Value	Description
measuredValue	The measurement value was measured with a dedicated sensor. The value is rather exact.
calculatedValue	The measurement value was calculated with the values from some other parameters (e.g. water temperature, outside temperature, etc.) that have a calculable impact on the measurand. This value is not as exact as a measured value, but it differs not too much from the real value.

empiricalValue	The measurement value is only an empirical one that can differ a lot from the real value. It can rely on some other parameters that have a known value, but there is no exactly calculable relation between them.
----------------	---

Table 232: Enumeration MeasurementValueSourceEnumType

Enumeration **MeasurementValueTendencyEnumType**:

Value	Description
rising	The average measurement value is currently rising.
stable	The average measurement value is currently stable.
falling	The average measurement value is currently falling.

Table 233: Enumeration MeasurementValueTendencyEnumType

Enumeration **MeasurementValueStateEnumType**:

Value	Description
normal	The measurement value could be measured / calculated / empirically determined as expected.
outOfRange	The measurement value is currently out of range of the sensor (as defined by "valueRangeMin" and "valueRangeMax", see section 5.3.15.3.2).
error	An error occurred while determining the measurement value.

Table 234: Enumeration MeasurementValueStateEnumType

5.3.15.2.3 Available selectors

- measurementId
- valueType
- timestampInterval

5.3.15.2.4 Examples

5.3.15.2.4.1 Read-reply

A *measurementListData* command with a measured value, sent as reply with only one entry (device is not storing history values):

```

<measurementListData>
  <measurementData>
    <measurementId>1</measurementId>
    <valueType>value</valueType>
    <timestamp>2013-07-15T12:00:00.0Z</timestamp>
    <value>
      <number>255</number>
      <scale>-1</scale>
    </value>
    <valueSource>measuredValue</valueSource>
    <valueTendency>rising</valueTendency>
    <valueState>normal</valueState>
  </measurementData>
</measurementListData>

```

7623 **5.3.15.2.4.2 Notify**

7624 A *measurementListData* command with a calculated average value (no other value type supported by
7625 the device), sent as (complete) notification:

```
7626 <measurementListData>
7627   <measurementData>
7628     <measurementId>1</measurementId>
7629     <valueType>averageValue</valueType>
7630     <timestamp>2013-07-15T12:13:14.5Z</timestamp>
7631     <value>
7632       <number>23</number>
7633     </value>
7634     <evaluationPeriod>
7635       <startTime>2013-07-15T11:00:00.0Z</startTime>
7636       <endTime>2013-07-15T12:00:00.0Z</endTime>
7637     </evaluationPeriod>
7638     <valueSource>calculatedValue</valueSource>
7639     <valueTendency>stable</valueTendency>
7640     <valueState>normal</valueState>
7641   </measurementData>
7642 </measurementListData>
```

7643

7644 **5.3.15.2.4.3 Read-reply**

7645 In this example the requested node supports only values of type "value" and offers this reply:

```
7646 <measurementListData>
7647   <measurementData>
7648     <measurementId>1</measurementId>
7649     <valueType>value</valueType>
7650     <timestamp>2013-07-15T12:10:00.0Z</timestamp>
7651     <value>
7652       <number>257</number>
7653       <scale>-1</scale>
7654     </value>
7655     <valueSource>measuredValue</valueSource>
7656     <valueTendency>falling</valueTendency>
7657     <valueState>normal</valueState>
7658   </measurementData>
7659   <measurementData>
7660     <measurementId>1</measurementId>
7661     <valueType>value</valueType>
7662     <timestamp>2013-07-15T12:20:00.0Z</timestamp>
7663     <value>
7664       <number>261</number>
7665       <scale>-1</scale>
7666     </value>
7667     <valueSource>measuredValue</valueSource>
7668     <valueTendency>stable</valueTendency>
7669     <valueState>normal</valueState>
7670   </measurementData>
7671 </measurementListData>
```

7672

7673 **5.3.15.3 measurementConstraintsListData**7674 **5.3.15.3.1 General**

7675 Some constraints concerning the possible values of a measurand are modelled within this function.

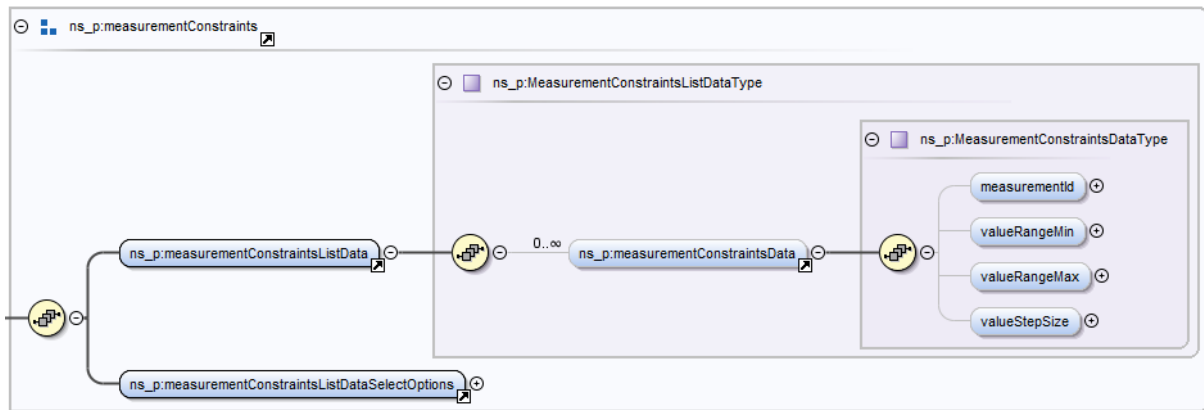


Figure 116: measurementConstraintsListData function overview

5.3.15.3.2 Detailed description of elements

Element	Type	Description
measurementId	Identifier "MeasurementIdType" (see Table 339).	Enables the identification of different sensors on one SPINE feature.
valueRangeMin	Common data type "ScaledNumberType". See section 3.10.1.8.	The minimum possible measurement value measurable by the feature.
valueRangeMax	Common data type "ScaledNumberType". See section 3.10.1.8.	The maximum possible measurement value measurable by the feature.
valueStepSize	Common data type "ScaledNumberType". See section 3.10.1.8.	The minimum step size between two different measurement values.

Table 235: measurementConstraintsListData function detailed description of elements

5.3.15.3.3 Available selectors

- measurementId

5.3.15.3.4 Examples

5.3.15.3.4.1 Read-reply

A measurementConstraintsListData instance from a temperature sensor, sent as reply to a standard read command:

```

<measurementConstraintsListData>
  <measurementConstraintsData>
    <measurementId>1</measurementId>
    <valueRangeMin>
      <number>-1</number>
      <scale>1</scale>
    </valueRangeMin>
    <valueRangeMax>
      <number>4</number>
      <scale>1</scale>
  </measurementConstraintsData>
</measurementConstraintsListData>

```

```

7699         </valueRangeMax>
7700         <valueStepSize>
7701             <number>1</number>
7702             <scale>-1</scale>
7703         </valueStepSize>
7704     </measurementConstraintsData>
7705 </measurementConstraintsListData>

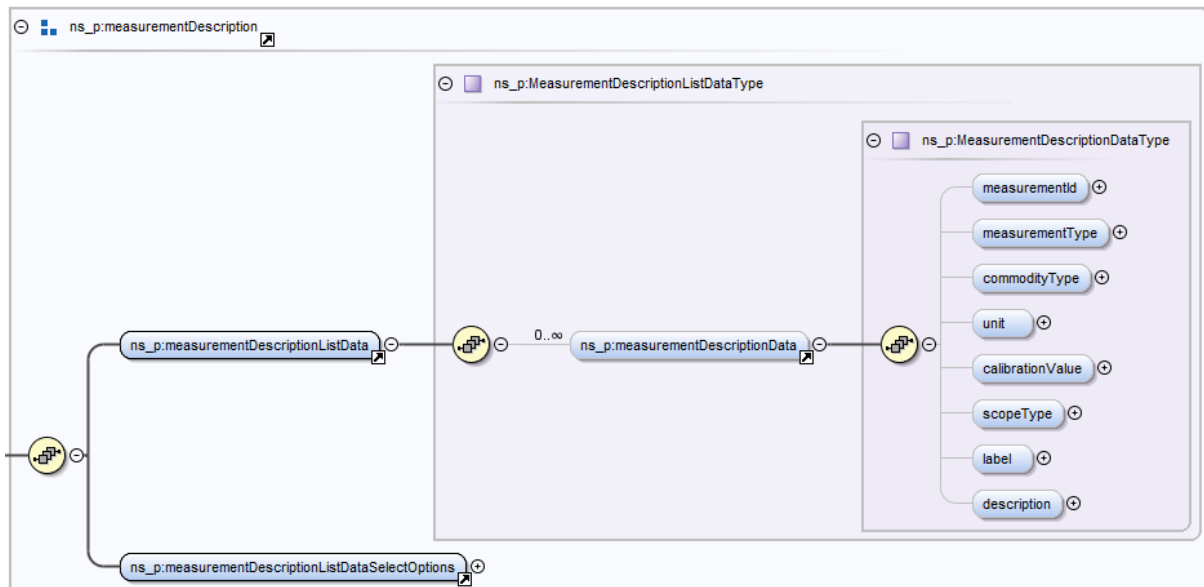
```

7706

7707 5.3.15.4 *measurementDescriptionListData*

7708 5.3.15.4.1 *General*

7709 Descriptive information on a specific measurement type is modelled with a
7710 *measurementDescriptionListData* function. This information is rather constant in comparison to
7711 measurement values. Typically, it is requested once at the beginning, when a node (an external
7712 display, e.g.) is configured to present or use a specific measurement value. As usual, for such
7713 scenarios it is recommended the the server notifies changes to all relevant communication partners,
7714 because a server cannot expect clients to periodically request this information.



7715

7716 Figure 117: *measurementDescriptionListData* function overview

7717

7718 5.3.15.4.2 *Detailed description of elements*

Element	Type	Description
measurementId	Identifier "MeasurementIdType" (see Table 339).	Enables the identification of different measurands on one SPINE feature.
measurementType	Union "MeasurementTypeType": - Enum (see Table 237): MeasurementTypeEnumType - EnumExtendType (see section 3.10.1.5)	To identify which type of measurand is measured (for example "temperature").

commodityType	<i>Common data type "CommodityTypeType". See section 3.10.1.9.</i>	If a measurand of a commodity is measured, the type of commodity is stated here.
unit	<i>Common data type "UnitOfMeasurementType". See section 3.10.1.17.</i>	The unit, which is used for the value. It is always related to the normal value. Other <i>valueType</i> values (e.g. <i>averageDerivativeValue</i>) can have different units (derived from <i>unit</i>).
calibrationValue	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Some measurement sensors need a calibration value (e.g. set during production phase). It is automatically added to the internal measured value. I.e. the value stated in the <i>value</i> element in the <i>measurementListData</i> function is the already corrected value.
scopeType	<i>Common data type "ScopeTypeType". See section 3.10.1.21.</i>	Specifies a more detailed meaning of the measurand (e.g. <i>outsideTemperature</i>).
label	<i>Common data type "LabelType". See section 3.10.1.2.</i>	A (short) label can be useful to identify or group specific sensors.
description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	A (longer) description of a measurement device can be deposited here.

7719 Table 236: *measurementDescriptionListData* function detailed description of elements7720 Enumeration **MeasurementTypeEnumType**:

Value	Description
acceleration	The acceleration is measured.
angle	An angle is measured.
angularVelocity	The angular velocity is measured.
area	An area is measured.
atmosphericPressure	The atmospheric pressure is measured.
capacity	The capacity is measured.
concentration	A concentration of a substance is measured.
count	A count is measured.
current	The current is measured.
density	The density is measured.
distance	A distance is measured.
electricField	The electric field is measured.
energy	The energy is measured.
force	The force is measured.
frequency	The frequency is measured.
harmonicDistortion	The harmonic distortion is measured.
heat	The heat is measured.
heatFlux	The heat flux is measured.
illuminance	The illuminance is measured.
impulse	An impulse is measured.
level	A level is measured.
magneticField	The magnetic field is measured.
mass	A mass is measured.

massFlow	The mass flow is measured.
particles	The quantity of particles is measured.
percentage	The percentage is measured.
power	The power is measured.
powerFactor	The power factor is measured.
pressure	The pressure is measured.
radonActivity	The radon activity is measured.
relativeHumidity	The relative humidity is measured.
resistance	The resistance is measured.
solarRadiation	The solar radiation is measured.
speed	The speed is measured.
temperature	The temperature is measured.
time	The time is measured.
torque	The torque is measured.
unknown	An unknown measurand is measured.
velocity	The velocity is measured.
voltage	The voltage is measured.
volume	The volume is measured.
volumetricFlow	The volumetric flow is measured.

7721 *Table 237: Enumeration MeasurementTypeEnumType*

7722

7723 5.3.15.4.3 Available selectors

- 7724 - measurementId
- 7725 - measurementType
- 7726 - commodityType
- 7727 - scopeType

7728

7729 5.3.15.4.4 Examples

7730 5.3.15.4.4.1 Read-reply

7731 A *measurementDescriptionListData* instance from a temperature sensor, sent as reply to a standard
7732 read command:

```

7733 <measurementDescriptionListData>
7734   <measurementDescriptionData>
7735     <measurementId>1</measurementId>
7736     <measurementType>temperature</measurementType>
7737     <commodityType>air</commodityType>
7738     <unit>degC</unit>
7739     <calibrationValue>
7740       <number>-48</number>
7741       <scale>-1</scale>
7742     </calibrationValue>
7743     <scopeType>outsideTemperature</scopeType>
7744     <label>Outside_Temp.</label>
7745     <description>Outside Air Temperature</description>
7746   </measurementDescriptionData>
7747 </measurementDescriptionListData>

```

7748

5.3.15.5 *measurementThresholdRelationListData*

5.3.15.5.1 *General*

The relation between a measurand and dedicated thresholds (see section 5.3.26) is modelled with this function.

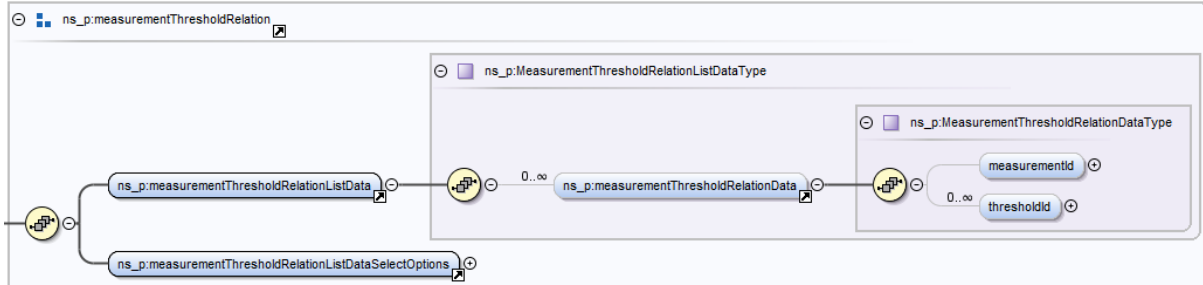


Figure 118: *measurementThresholdRelationListData* function overview

5.3.15.5.2 *Detailed description of elements*

Element	Type	Description
measurementId	Identifier "MeasurementIdType" (see Table 339).	Enables the identification of different sensors on one SPINE address.
thresholdId (list)	Identifier "ThresholdIdType" (see Table 339).	Related threshold identifier(s).

Table 238: *measurementThresholdRelationListData* function detailed description of elements

5.3.15.5.3 *Available selectors*

- measurementId
- thresholdId

5.3.15.5.4 *Examples*

5.3.15.5.4.1 *Read-reply*

A *measurementThresholdRelationListData* sent as a reply to a standard read command:

```

<measurementThresholdRelationListData>
  <measurementThresholdRelationData>
    <measurementId>1</measurementId>
    <thresholdId>2<thresholdId>
    <thresholdId>3<thresholdId>
    <thresholdId>6<thresholdId>
  </measurementThresholdRelationData>
</measurementThresholdRelationListData>

```

5.3.16 Messaging

5.3.16.1 Introduction

Messaging is basically intended for simple textual information transfer for displays and users. The textual information is not intended to be used for any automatisms. It should be used for textual information that are for example event based or otherwise will change over time.

Some devices may keep a list of messages rather than (or in addition to) a single message.



Figure 119: Messaging function-group overview

5.3.16.2 messagingListData

5.3.16.2.1 General

The messagingListData function contains the necessary elements to model a simple text message.

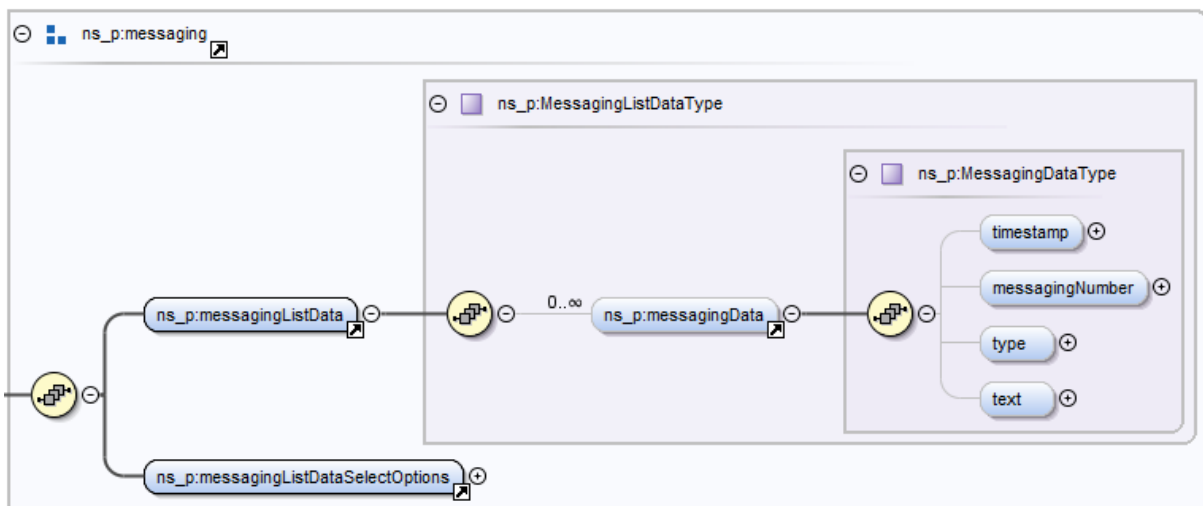


Figure 120: messagingListData function overview

5.3.16.2.2 Detailed description of elements

Element	Type	Description
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The time when the message was generated.
messagingNumber	Identifier "MessagingNumberType" (see Table 339).	An identifier for one specific message. If a message is marked as obsolete (see below) this number can be used to identify the original message.
type	Union "MessagingTypeType": - Enum (see Table 240): MessagingTypeEnumType	The type of the message.

	- EnumExtendType (see section 3.10.1.5)	
text	<i>Simple type</i> <i>"MessagingDataTextType"</i> <i>(restriction of xs:string)</i>	The message itself.

7791 Table 239: messagingListData function detailed description of elements

7792 Enumeration **MessagingTypeEnumType**:

Value	Description
logging	Used for messages that are stored in a log file.
information	Messages that are presented to the customer on a display; lower priority.
warning	Messages that are presented to the customer on a display; medium priority.
alarm	Messages that are presented to the customer on a display; high priority.
emergency	Messages that are presented to the customer on a display; very high priority.
obsolete	Used to mark previously sent messages as obsolete (so that they are not displayed anymore).

7793 Table 240: Enumeration MessagingTypeEnumType

7794

7795 5.3.16.2.3 Available selectors

7796 - timestampInterval

7797 - messagingNumber

7798

7799 5.3.16.2.4 Examples

7800 5.3.16.2.4.1 Notify

7801 A washing machine is configured to send messages to an external display. After a firmware update it
7802 sends this (complete) notification:

```

7803 <messagingListData>
7804   <messagingData>
7805     <timestamp>2006-05-04T18:13:51.0Z</timestamp>
7806     <messagingNumber>72</messagingNumber>
7807     <type>information</type>
7808     <text>Update completed.</text>
7809   </messagingData>
7810 </messagingListData>

```

7811 Other possible reasons for messages to a display: Notification of end of washing cycle; notification of
7812 problem with water supply; etc.

7813

7814 5.3.16.2.4.2 Notify

7815 An alarm device keeps a list of its events of the last month. It is configured to send new messages to
7816 an external display every 2 minutes (the “filter/partial” part to announce the “restricted function
7817 exchange is simplified by “...” just to improve the readability”):

```

7818 ...
7819 <messagingListData>
7820   <messagingData>

```

```

7821         <timestamp>2006-05-04T18:13:51.0Z</timestamp>
7822         <messagingNumber>2014</messagingNumber>
7823         <type>information</type>
7824         <text>Window 1st floor closed.</text>
7825     </messagingData>
7826     <messagingData>
7827         <timestamp>2006-05-04T18:14:23.0Z</timestamp>
7828         <messagingNumber>2015</messagingNumber>
7829         <type>information</type>
7830         <text>Alarm engaged.</text>
7831     </messagingData>
7832 </messagingListData>

```

7833

7834 5.3.16.2.4.3 Read-reply

7835 The alarm device is able to provide its complete message list upon request.

7836 A user selects the alarm device on an external display and queries its messages. The alarm device
 7837 replies with a proper messagingListData. Extract with just the first and the last message:

```

7838 <messagingListData>
7839     <messagingData>
7840         <timestamp>2006-04-05T03:23:17.0Z</timestamp>
7841         <messagingNumber>1716</messagingNumber>
7842         <type>alarm</type>
7843         <text>Window ground floor opened.</text>
7844     </messagingData>
7845     ...
7846     <messagingData>
7847         <timestamp>2006-05-04T18:14:23.0Z</timestamp>
7848         <messagingNumber>2015</messagingNumber>
7849         <type>information</type>
7850         <text>Alarm engaged.</text>
7851     </messagingData>
7852 </messagingListData>

```

7853

7854 5.3.16.2.4.4 Read-reply

7855 The alarm device supports the selector *timestampInterval*. A user selects the alarm device on an
 7856 external display and queries its messages of the last hour. The display sends a read classifier together
 7857 with the corresponding filter:

```

7858 <function>messagingListData</function>
7859 <filter>
7860     <cmdControl>
7861         <partial/>
7862     </cmdControl>
7863     <messagingListDataSelectors>
7864         <timestampInterval>
7865             <startTime>-PT1H</startTime>
7866             <endTime>PT0S</endTime>
7867         </timestampInterval>
7868     </messagingListDataSelectors>
7869 </filter>
7870 <messagingListData/>

```

7871 The alarm device replies with a proper `messagingListData`. Extract with just the first and the last
7872 message:

```
7873 <function>messagingListData</function>
7874 <filter>
7875     <cmdControl>
7876         <partial/>
7877     </cmdControl>
7878 </filter>
7879 <messagingListData>
7880     <messagingData>
7881         <timestamp>2006-05-04T17:55:19.0Z</timestamp>
7882         <messagingNumber>2012</messagingNumber>
7883         <type>information</type>
7884         <text>Front door closed.</text>
7885     </messagingData>
7886     ...
7887     <messagingData>
7888         <timestamp>2006-05-04T18:14:23.0Z</timestamp>
7889         <messagingNumber>2015</messagingNumber>
7890         <type>information</type>
7891         <text>Alarm engaged.</text>
7892     </messagingData>
7893 </messagingListData>
```

7894

7895 5.3.16.2.4.5 Read-reply

7896 The alarm device supports the select option *messagingNumber*. A user selects the alarm device on an
7897 external display and queries the messages with `messagingNumber` "2011". The display sends a read
7898 classifier together with the select options:

```
7899 <function>messagingListData</function>
7900 <filter>
7901     <cmdControl>
7902         <partial/>
7903     </cmdControl>
7904     <messagingListDataSelectors>
7905         <messagingNumber>2011</messagingNumber>
7906     </messagingListDataSelectors>
7907 </filter>
7908 <messagingListData/>
```

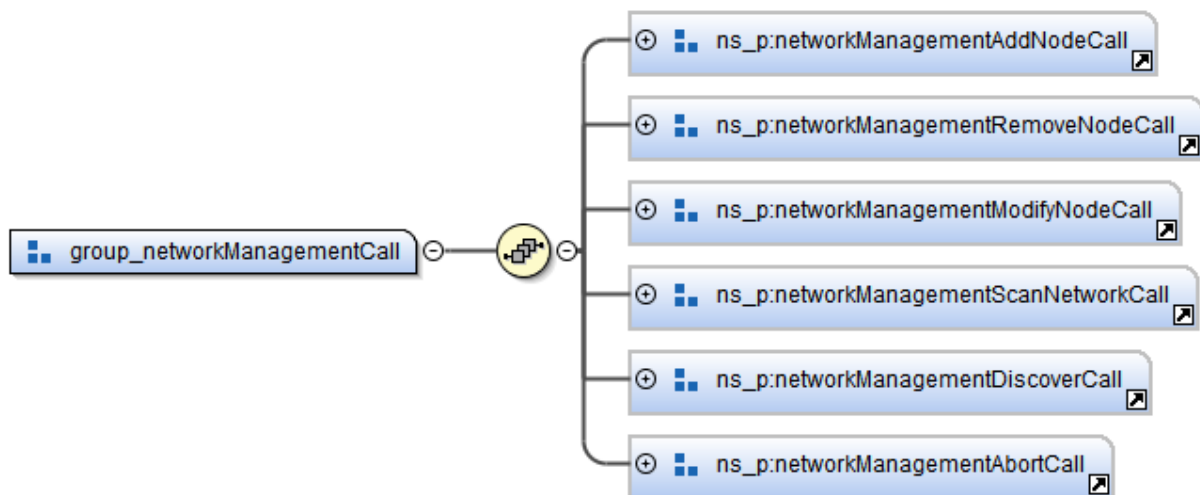
7909 The alarm device replies with a proper `messagingListData`:

```
7910 <function>messagingListData</function>
7911 <filter>
7912     <cmdControl>
7913         <partial/>
7914     </cmdControl>
7915 </filter>
7916 <messagingListData>
7917     <messagingData>
7918         <timestamp>2006-05-04T16:10:12.0Z</timestamp>
7919         <messagingNumber>2011</messagingNumber>
7920         <type>information</type>
7921         <text>Alarm disengaged.</text>
7922     </messagingData>
7923 </messagingListData>
```

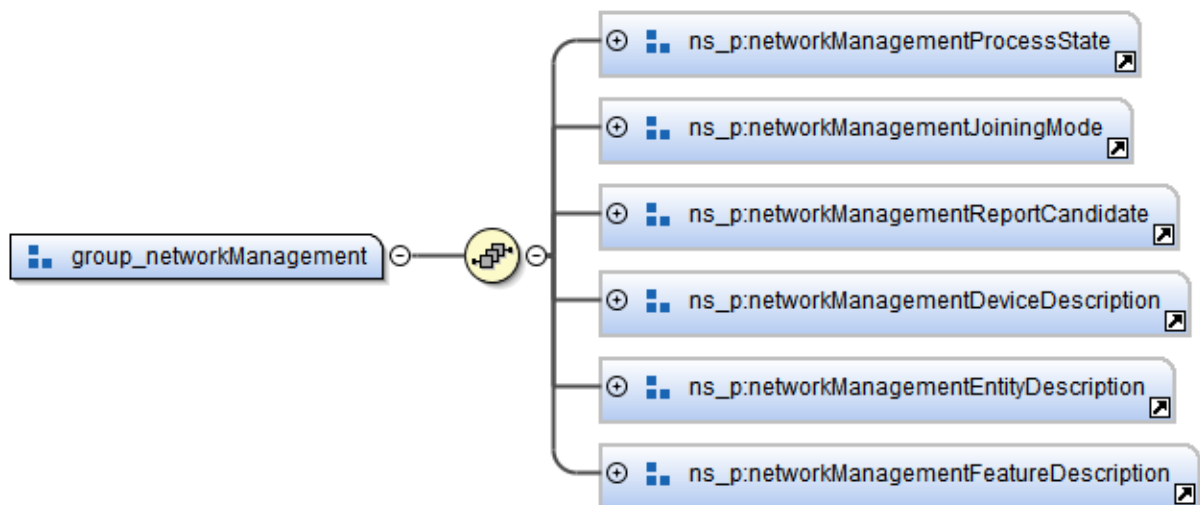
7924

7925 **5.3.17 NetworkManagement**7926 **5.3.17.1 Introduction**

7927 This class permits to model certain "knowledge" of other devices as well as the "modification" of this
 7928 knowledge. In terms of a communications technology these models can be used to add remote
 7929 nodes (devices, endpoints) to the own (SPINE) network, to remove them from the network, etc.



7930

7931 *Figure 121: NetworkManagement function-group overview (call)*

7932

7933 *Figure 122: NetworkManagement function-group overview (data)*

7934

7935 **5.3.17.1.1 SPINE address components, hierarchy, terminology**

7936 A SPINE address can contain these address elements:

- 7937 1. device
- 7938 2. entity
- 7939 3. feature

7940 The element “entity” can appear multiple times (i.e. as list). This is used to model an entity hierarchy,
7941 starting with a root entity as first item of the list. The consecutive list of “entity” elements finally
7942 denotes a specific entity.

7943 Altogether, the address elements are used to determine the proper functional component. Each
7944 device can contain several entities. Each entity can contain several features. Each addressable
7945 component is in general called a “node”.

7946

7947 *5.3.17.1.2 Network Management Responsible*

7948 A so-called "Network Management Responsible" is defined in this class. This is an instance
7949 responsible to perform SPINE network management tasks for other nodes. The Network
7950 Management Responsible itself is considered a node with node address as well. A brief example
7951 explains this: A sophisticated "SPINE gateway" permits software applications to exchange SPINE
7952 XMLs with an (internal) interface that provides access to devices of different communications
7953 technologies. For each communications technology there is an (internal) implementation that
7954 realizes the translation between SPINE XML (from/to the aforementioned internal interface) and the
7955 dedicated communications technology interface/protocol (i.e. it makes use of the so-called
7956 "mapping"). We assume this "mapping implementation" is responsible to "find" devices accessible
7957 via its communications technology. In general, the implementation is responsible to control whether
7958 data can be exchanged between software applications and devices or not. This means these
7959 implementations are responsible for network management activities concerning their attached or
7960 accessible devices.

7961 Please note this example is just to explain the term "Network Management Responsible". The
7962 architecture sketched in the example shall not be considered to be the only permitted architecture.

7963

7964 *5.3.17.1.3 Use of “energy management gateway” in examples*

7965 Some communications technologies follow a “centralised” approach, whereas others are “de-
7966 centralised”. The class “NetworkManagement” does not impose which approach has to be followed.
7967 In the interest of simplification, subsequent examples often assume an “energy management
7968 gateway” which is capable to connect with other SPINE devices. However, please be aware that these
7969 examples are neither a mandatory feature of an “energy management gateway”, nor will a
7970 communications architecture be derived from it within this class description.

7971

7972 **5.3.17.2 networkManagement...Call**

7973 The *networkManagement...Calls* are used to add, modify or remove nodes (usually on Gateways) or
7974 to tell a node, that it shall perform a network scan, discover a specific node or abort the current task.

7975

7976 **5.3.17.3 networkManagementAddNodeCall**

7977 *5.3.17.3.1 General*

7978 This function can be used to add another node (device) to the own SPINE network.

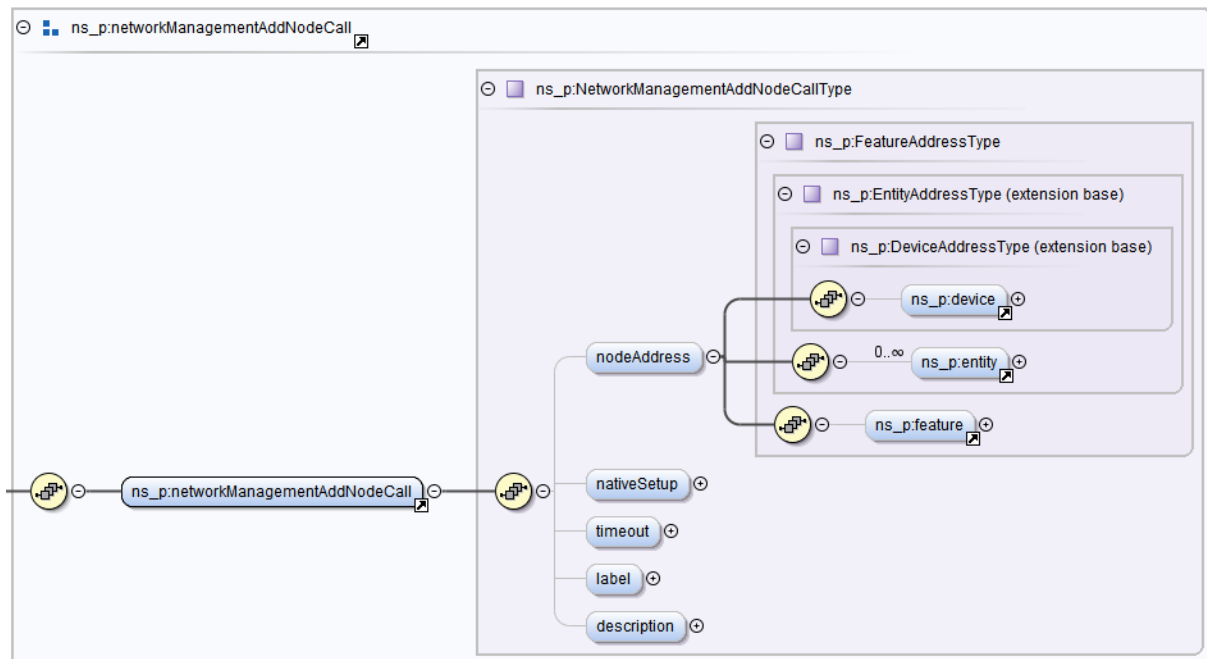


Figure 123: networkManagementAddNodeCall function overview

5.3.17.3.2 Detailed description of elements

Element	Type	Description
nodeAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	The node address of the node that shall be added. Contains elements for <i>device</i> , <i>entity</i> and <i>feature</i> .
nodeAddress.device	See section 3.10.3.6.	Device address part.
nodeAddress.entity (list)	See section 3.10.3.6.	Entity (list) address part.
nodeAddress.feature	See section 3.10.3.6.	Feature address part.
nativeSetup	Simple type "NetworkManagementNativeSetupType" (restriction of xs:string)	Technology dependent or implementation dependent configuration to identify (and maybe configure) the new node.
timeout	xs:duration (W3C standard type)	The procedure of adding the node shall be aborted latest after this duration.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the new node.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the new node.

Table 241: networkManagementAddNodeCall function element description

5.3.17.3.3 Available selectors

None.

7988 5.3.17.3.4 Examples

7989 5.3.17.3.4.1 Call

7990 The user interface of an energy management gateway permits adding a display that is accessible via a
 7991 specific communications technology denoted here as "CP". In this example the native address of the
 7992 display in "CP" is known in advance ("21.6.5.3.11", e.g.). Furthermore it is assumed that "CP" is a
 7993 rather low-level protocol and the energy management gateway's interface to "CP" has to be
 7994 configured in order to communicate with the display properly (configuration of profiles, e.g.). There
 7995 is information available how to translate this configuration into "nativeSetup" properly. We assume
 7996 the vendor of the energy management gateway provided a sophisticated user interface that
 7997 simplifies this procedure for the user. Thus, the user just selects the technology interface to "CP",
 7998 enters the native address and the label and then selects the right profile for a display for the
 7999 interface to "CP". The user interface creates the subsequent XML and provides it to the Network
 8000 Management Responsible of "CP":

```
8001 <networkManagementAddNodeCall>
8002   <nativeSetup>21.6.5.3.11;R-2.7;at-42.344;pqtl-
8003 27638162683172637812356813</nativeSetup>
8004   <timeout>PT40S</timeout>
8005   <label>Display</label>
8006 </networkManagementAddNodeCall>
```

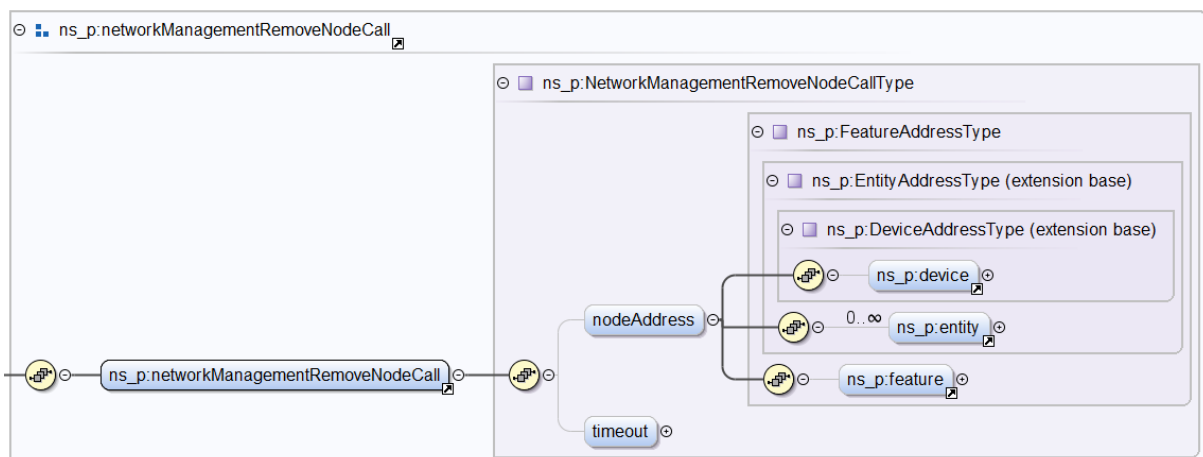
8007 In this example the *nodeAddress* is unset to let the Network Management Responsible of "CP" assign
 8008 a proper address. Upon the successful completion of the process the communications technology
 8009 interface provides a proper *networkManagementProcessStateData* notification to the user interface.
 8010 It furthermore creates proper *networkManagementDeviceDescriptionListData*,
 8011 *networkManagementEntityDescriptionListData* and *networkManagementFeatureDescriptionListData*
 8012 notifications for internal purposes (information for further applications installed on the energy
 8013 management gateway: notification of a new node).

8014

8015 5.3.17.4 networkManagementRemoveNodeCall

8016 5.3.17.4.1 General

8017 This function can be used to remove a node from the own SPINE network.



8018

8019 Figure 124: networkManagementRemoveNodeCall function overview

8020

8021 5.3.17.4.2 Detailed description of elements

Element	Type	Description
nodeAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	The node address of the node that will be removed. Contains elements for <i>device</i> , <i>entity</i> and <i>feature</i> .
nodeAddress.device	See section 3.10.3.6.	Device address part.
nodeAddress.entity (list)	See section 3.10.3.6.	Entity (list) address part.
nodeAddress.feature	See section 3.10.3.6.	Feature address part.
timeout	xs:duration (W3C standard type)	The procedure of removing the node will be aborted latest after this duration.

8022 Table 242: networkManagementRemoveNodeCall function element description

8023

8024 5.3.17.4.3 Available selectors

8025 None.

8026

8027 5.3.17.4.4 Examples

8028 5.3.17.4.4.1 Call

8029 We assume a display belongs to the SPINE network of an energy management gateway and is
 8030 assigned the device address "d:_i:46925_CP_pqtl-27638162683172637812356813" (as a result of
 8031 the example in 5.3.17.3.4, e.g.). The user interface of the energy management gateway permits
 8032 removing the display from the gateway's network. The user selects the display on the user interface
 8033 and requests its removal. The user interface creates the subsequent XML and provides it to the
 8034 Network Management Responsible of this node:

```

8035 <networkManagementRemoveNodeCall>
8036   <nodeAddress>
8037     <device>d:_i:46925_CP\_pqtl-27638162683172637812356813</device>
8038   </nodeAddress>
8039   <timeout>PT30S</timeout>
8040 </networkManagementRemoveNodeCall>

```

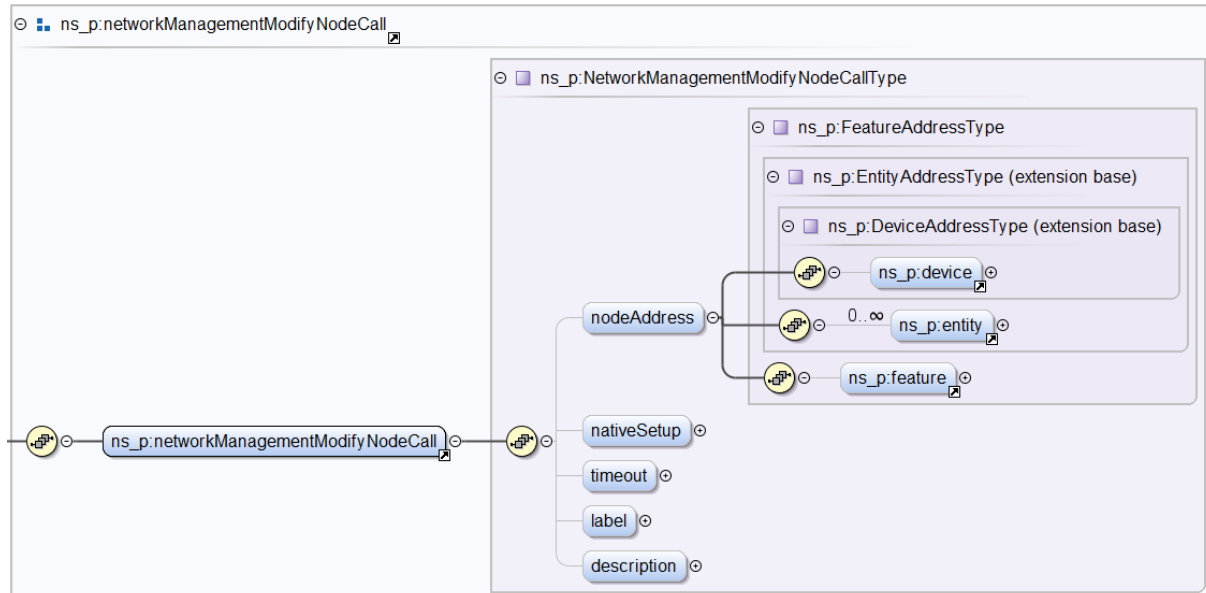
8041 Because the whole *device* will be removed (and not a specific *entity* or *feature*) only the *device*-part
 8042 of the address is to be filled.

8043 Upon the successful completion of the process the Network Management Responsible provides a
 8044 proper *networkManagementProcessStateData* notification to the user interface. It furthermore
 8045 creates a proper *networkManagementDeviceDescriptionListData* notification for internal purposes
 8046 (information for further applications installed on the energy management gateway: notification of a
 8047 removed node).

8048

8049 **5.3.17.5 networkManagementModifyNodeCall**8050 **5.3.17.5.1 General**

8051 This function can be used to modify an already "present" node of the own SPINE network. Here,
 8052 "present" implies the node has a node address.



8053

8054 *Figure 125: networkManagementModifyNodeCall function overview*

8055

8056 **5.3.17.5.2 Detailed description of elements**

Element	Type	Description
nodeAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	The node address of the node that will be modified. Contains elements for <i>device</i> , <i>entity</i> and <i>feature</i> .
nodeAddress.device	See section 3.10.3.6.	Device address part.
nodeAddress.entity (list)	See section 3.10.3.6.	Entity (list) address part.
nodeAddress.feature	See section 3.10.3.6.	Feature address part.
nativeSetup	Simple type "NetworkManagementNativeSetupType" (restriction of xs:string)	Similar to <i>nativeSetup</i> of <i>networkManagementAddNodeCall</i> . But for <i>networkManagementModifyNodeCall</i> this element is not used to identify which node has to be modified (see element <i>nodeAddress</i> for this purpose).
timeout	xs:duration (W3C standard type)	The procedure of modifying the node will be aborted latest after this duration.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the node.

description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Descriptive information on the node.
-------------	--	--------------------------------------

Table 243: *networkManagementModifyNodeCall* function element description**5.3.17.5.3 Available selectors**

None.

5.3.17.5.4 Examples**5.3.17.5.4.1 Call**

We assume a display belongs to the SPINE network of an energy management gateway and is assigned the device address "d:_i:46925_CP_pqtl-27638162683172637812356813" (as a result of the example in 5.3.17.3.4, e.g.). The user interface of the energy management gateway permits modifying the display's label. The user selects the display on the user interface and enters the new label "Kitchen display". The user interface creates the subsequent XML and provides it to its own interface to the Network Management Responsible of this node:

```
<networkManagementModifyNodeCall>
  <nodeAddress>
    <device>d:_i:46925_CP\_pqtl-27638162683172637812356813</device>
  </nodeAddress>
  <nativeSetup>21.6.5.3.11;R-2.7;at-42.344;pqtl-
27638162683172637812356813</nativeSetup>
  <timeout>PT15S</timeout>
  <label>Kitchen display</label>
  <description></description>
</networkManagementModifyNodeCall>
```

Because the label of the physical device will be changed (and not of a specific *entity* or *feature*) only the *device*-part of the address is to be filled.

Upon the successful completion of the process the Network Management Responsible provides a proper *networkManagementProcessStateData* notification to the user interface. It furthermore creates a proper *networkManagementDeviceDescriptionListData* notification for internal purposes (information for further applications installed on the energy management gateway: notification of a modified node).

5.3.17.6 networkManagementScanNetworkCall**5.3.17.6.1 General**

Some communications technologies provide measures to scan which other devices are accessible. The function *networkManagementScanNetworkCall* can be used to trigger such an action, if supported by technology and implementation. This function is not intended to add discovered devices directly to the own SPINE network. Instead, it is intended just to get a report. This report can be modelled by *networkManagementReportCandidateData*. Of course, these reports may then be used to add discovered nodes.

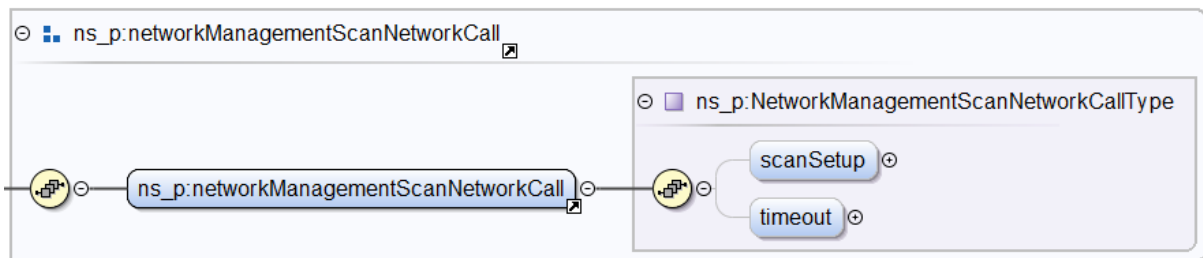


Figure 126: networkManagementScanNetworkCall function overview

5.3.17.6.2 Detailed description of elements

Element	Type	Description
scanSetup	Simple type "NetworkManagementScanSetupType" (restriction of xs:string)	Technology dependent or implementation dependent configuration for the scanning procedure. This might contain parameters to search for specific types of devices, e.g.
timeout	xs:duration (W3C standard type)	The procedure of scanning will be aborted latest after this duration.

Table 244: networkManagementScanNetworkCall function element description

5.3.17.6.3 Available selectors

None.

5.3.17.6.4 Examples

5.3.17.6.4.1 Call

An energy management gateway includes (among others) a specific communications technology (denoted here as "CP"). The implementation provides scanning other devices. The user selects "CP" in the gateway's user interface and triggers "Scan". The user interface then creates the subsequent XML and provides it to the Network Management Responsible of "CP":

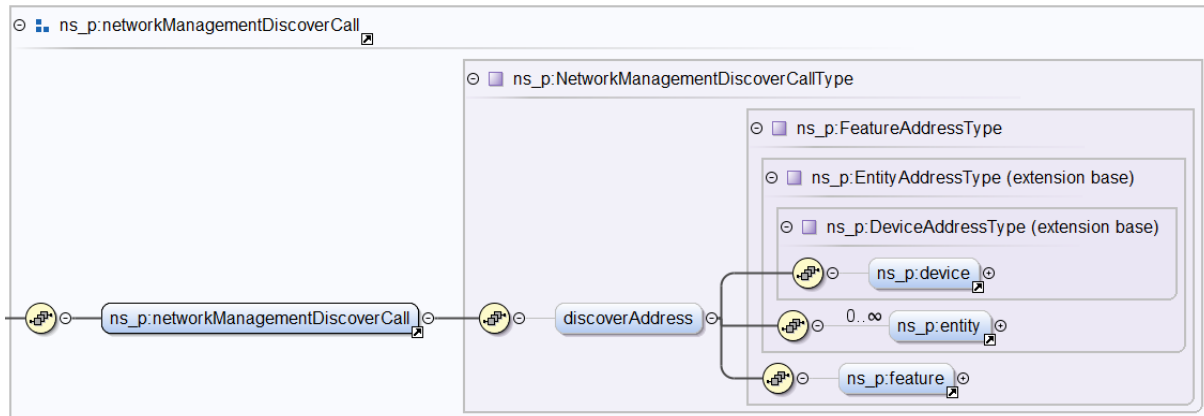
```
<networkManagementScanNetworkCall>
  <timeout>PT1M</timeout>
</networkManagementScanNetworkCall>
```

In this example "CP" is assumed to not require "scanSetup". During the scan the Network Management Responsible reports discovered devices as described above. Upon the successful completion of the process the Network Management Responsible also provides a proper *networkManagementProcessStateData* notification to the user interface.

8119 5.3.17.7 *networkManagementDiscoverCall*

8120 5.3.17.7.1 *General*

8121 If a node changed its functionality, it can announce the changes with all details to all nodes
 8122 interested in that information. This approach can generate a lot of unneeded traffic as not every
 8123 recipient requires knowledge of all changes. Hence, the *networkManagementDiscoverCall* can be
 8124 sent instead to those nodes to tell them, which address to discover (again). Then the nodes can still
 8125 decide to query for details on the changes.



8126
8127 Figure 127: *networkManagementDiscoverCall* function overview

8128

8129 5.3.17.7.2 *Detailed description of elements*

Element	Type	Description
discoverAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	The SPINE address of the node that will be discovered in detail. Contains elements for device, entity and feature.
discoverAddress.device	See section 3.10.3.6.	Device address part.
discoverAddress.entity (list)	See section 3.10.3.6.	Entity (list) address part.
discoverAddress.feature	See section 3.10.3.6.	Feature address part.

8130 Table 245: *networkManagementDiscoverCall* function element description

8131

8132 5.3.17.7.3 *Available selectors*

8133 None.

8134

8135 5.3.17.7.4 *Examples*

8136 5.3.17.7.4.1 *Call*

8137 If the functionality of a node has changed, it can send a *networkManagementDiscoverCall* to ask
 8138 other nodes to discover the changed node:

```

8139 <networkManagementDiscoverCall>
8140   <discoverAddress>
8141     <device>d:_i:46925_CP\_pqt1-27638162683172637812356813</device>
8142     <entity>3</entity>
  
```

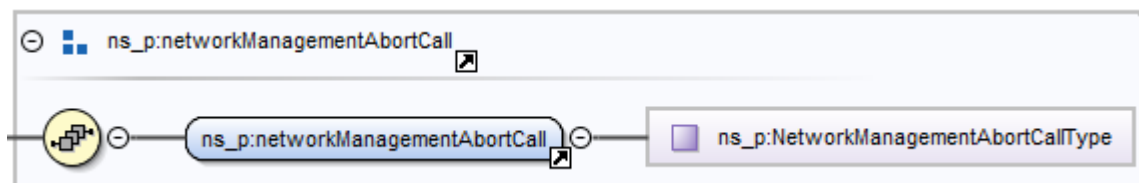
8143 </discoverAddress>
 8144 </networkManagementDiscoverCall>

8145

8146 **5.3.17.8 networkManagementAbortCall**

8147 *5.3.17.8.1 General*

8148 This function can be used to model the request to abort all ongoing processes triggered by
 8149 *networkManagementAddNodeCall*, *networkManagementRemoveNodeCall*,
 8150 *networkManagementModifyNodeCall*, *networkManagementScanNetworkCall* or
 8151 *networkManagementDiscoverCall*.



8152

8153 *Figure 128: networkManagementAbortCall function overview*

8154

8155 *5.3.17.8.2 Detailed description of elements*

8156 In this version of the specification the function *networkManagementAbortCall* does not contain child
 8157 elements.

8158

8159 *5.3.17.8.3 Available selectors*

8160 None.

8161

8162 *5.3.17.8.4 Examples*

8163 *5.3.17.8.4.1 Call*

8164 We assume a user triggered a scan of the communications technology "CP" as described in
 8165 5.3.17.6.4. Furthermore, we assume the implementation provides the possibility to abort this
 8166 process. 25 seconds after the user initiated the scan he decides to abort the process and selects the
 8167 "Abort" button on the gateway's user interface. The user interface then creates the subsequent XML
 8168 and provides it to the Network Management Responsible of "CP":

8169 <networkManagementAbortCall/>

8170 Upon the abortion of the process the Network Management Responsible provides a proper
 8171 *networkManagementProcessStateData* notification to the user interface.

8172

8173 **5.3.17.9 networkManagement...Data**

8174 The *networkManagement...Data* functions inform about the process of actions triggered with the
 8175 aforementioned calls, deliver or change the current joining mode or report new candidates (found

via the *networkManagementScanNetworkCall*). The device, entity and feature descriptions deliver detailed information about nodes (in the meaning of a node-graph including all devices, entities and features) and specific information about entities and features.

5.3.17.10 *networkManagementProcessStateData*

5.3.17.10.1 *General*

This function can be used to model the state of a process triggered by *networkManagementAddNodeCall*, *networkManagementRemoveNodeCall*, *networkManagementModifyNodeCall*, *networkManagementScanNetworkCall* or *networkManagementDiscoverCall*.

The underlying revision of the SPINE data model supports only notification of this function (i.e. read, write, or reply mechanisms are not supported).

Please note this function does not support the identification of a specific process. This should be considered carefully in case an implementation supports parallel processes.

It is advised to notify an instance of *networkManagementProcessStateData* only to the originator of the related network management process. Considering the architecture example of section 5.3.17.1.2 this would mean only the software application that triggered a certain process would be notified by the "Network Management Responsible".

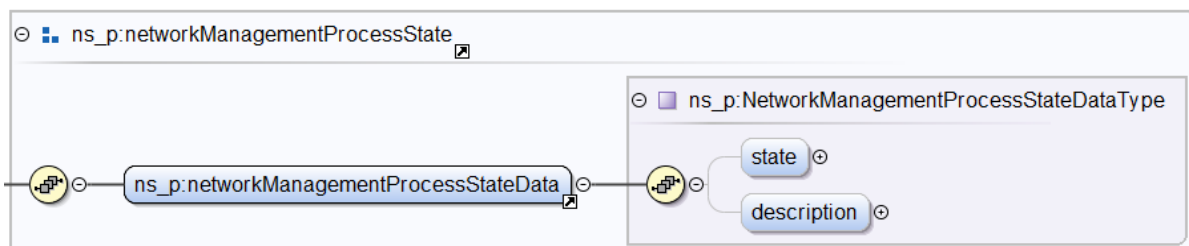


Figure 129: *networkManagementProcessStateData* function overview

5.3.17.10.2 *Detailed description of elements*

Element	Type	Description
state	Enum (see Table 247): <i>NetworkManagementProcessStateStateType</i>	The state of the process.
description	Common data type " <i>DescriptionType</i> ". See section 3.10.1.3.	Descriptive information.

Table 246: *networkManagementProcessStateData* function element description

Enumeration **NetworkManagementProcessStateStateType**:

Value	Description
succeeded	The process completed successfully.
failed	The process failed.
aborted	The process was aborted.

Table 247: Enumeration *NetworkManagementProcessStateStateType*

8201

8202 *5.3.17.10.3 Available selectors*

8203 None.

8204

8205 *5.3.17.10.4 Examples*8206 *5.3.17.10.4.1 Notify*

8207 Section 5.3.17.3.4 described the creation of a *networkManagementProcessStateData* notification as
 8208 a result of the process of adding a display. This XML could look like this:

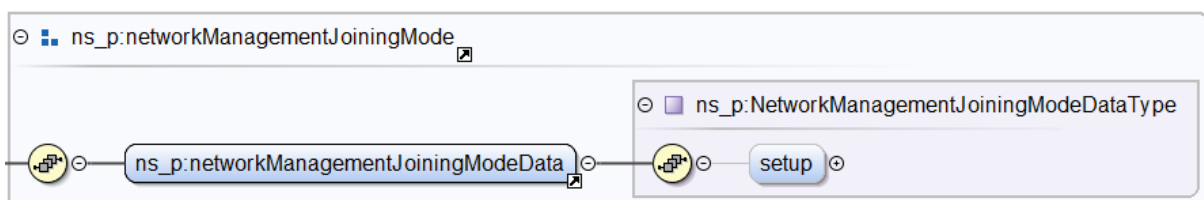
```
8209 <networkManagementProcessStateData>
8210     <state>succeeded</state>
8211     <description>Added node.</description>
8212 </networkManagementProcessStateData>
```

8213

8214 **5.3.17.11 networkManagementJoiningModeData**8215 *5.3.17.11.1 General*

8216 Some communications technologies or implementations permit a temporary or permanent
 8217 acceptance of other devices or at least a report of discovery. I.e. upon an incoming joining request of
 8218 another device the local device may add it automatically to its list of accepted devices. In terms of
 8219 the SPINE network management this means a local device would be added automatically to the own
 8220 SPINE network. As mentioned before, the new device also might just be reported instead of
 8221 automatically added. The function *networkManagementJoiningModeData* permits to model the
 8222 configuration of the local node's joining behaviour.

8223 Please note joining does not apply to a "scan network" process (see 5.3.17.6). But an "add node"
 8224 process initiated on another device might lead to an incoming joining request on the local device,
 8225 e.g. Depending on the joining mode configuration this request can be skipped (rejected), reported
 8226 (see 5.3.17.12) or automatically added (see 5.3.17.13.4.2).



8227

8228 *Figure 130: networkManagementJoiningModeData function overview*

8229

8230 *5.3.17.11.2 Detailed description of elements*

Element	Type	Description
setup	Simple type "NetworkManagementSetupType" (restriction of xs:string)	Technology dependent or implementation dependent configuration of the joining behaviour. In addition to the kind of joining procedure this could also contain parameters for

		a temporary enabling only, acceptance of specific devices only, etc., e.g.
--	--	--

Table 248: *networkManagementJoiningModeData* function element description

5.3.17.11.3 Available selectors

None.

5.3.17.11.4 Examples

5.3.17.11.4.1 Write

An energy management gateway includes (among others) a specific communications technology (denoted here as "CP"). The implementation provides so-called pairing with another device. In this example it will not be discussed which kind of joining procedure applies (adding the new device or just reporting it). But we assume "CP" requires some proprietary parameters.

The user chooses "CP" in the gateway's user interface and selects "Enable pairing". The user interface then creates the subsequent XML and provides it together with a *write* classifier to the Network Management Responsible of "CP" which configures its joining behaviour:

```
<networkManagementJoiningModeData>
  <setup>pairingOn;T=40;v=130.255</setup>
</networkManagementJoiningModeData>
```

5.3.17.12 networkManagementReportCandidateData

5.3.17.12.1 General

As already described in 5.3.17.6.1 and 5.3.17.11.1 some processes might require the report of a potential (but not yet accepted/added) device. The function *networkManagementReportCandidateData* can be used to model such a report.

Please note the reported data strongly depends on the implementation and possibilities of the related communications technology.

The underlying revision of the SPINE data model supports only notifications of this function (i.e. read, write and reply mechanisms are not supported).

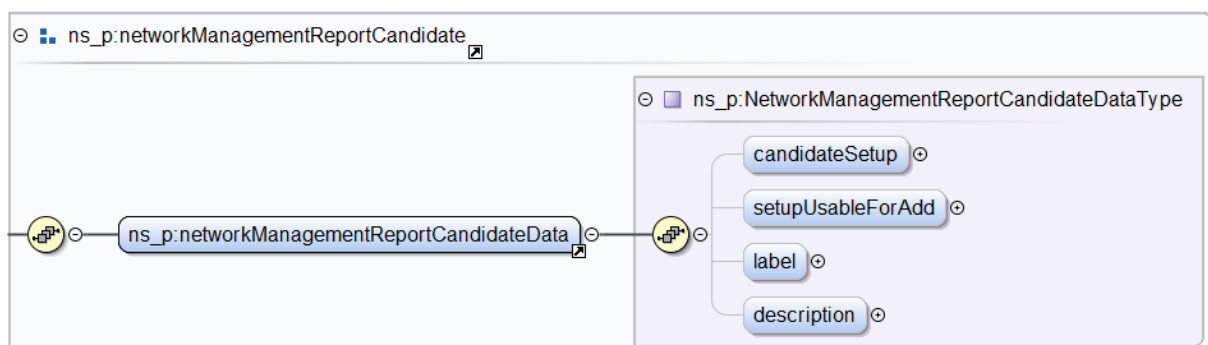


Figure 131: *networkManagementReportCandidateData* function overview

8260

8261 5.3.17.12.2 Detailed description of elements

Element	Type	Description
candidateSetup	Simple type "NetworkManagementCandidateSetupType" (restriction of xs:string)	Technology dependent or implementation dependent information on the reported node. This element is intended to provide information usable for an "add node" process. Whether its content could be copied unchanged into an element <i>nativeSetup</i> of <i>networkManagementAddNodeCall</i> depends on the element <i>setupUsableForAdd</i> .
setupUsableForAdd	xs:boolean (W3C standard type)	Only if this element is set to <i>true</i> the content of <i>candidateSetup</i> could be used for <i>nativeSetup</i> of <i>networkManagementAddNodeCall</i> .
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the reported node.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the reported node.

8262 Table 249: networkManagementReportCandidateData function element description

8263

8264 5.3.17.12.3 Available selectors

8265 None.

8266

8267 5.3.17.12.4 Examples

8268 5.3.17.12.4.1 Notify

8269 We assume the "scan network" example of 5.3.17.6.4 occurs on a device with implementation and
 8270 communications technology capable of providing sufficient details of reported devices. Among
 8271 others, the Network Management Responsible provides this XML as notification to the user interface
 8272 due to a discovered device:

```

8273 <networkManagementReportCandidateData>
8274   <candidateSetup>21.6.5.3.11;R-2.7;at-42.344;pqtl-
8275 27638162683172637812356813</candidateSetup>
8276   <setupUsableForAdd>true</setupUsableForAdd>
8277   <label>Display</label>
8278   <description>Device type: Display/Generic</description>
8279 </networkManagementReportCandidateData>

```

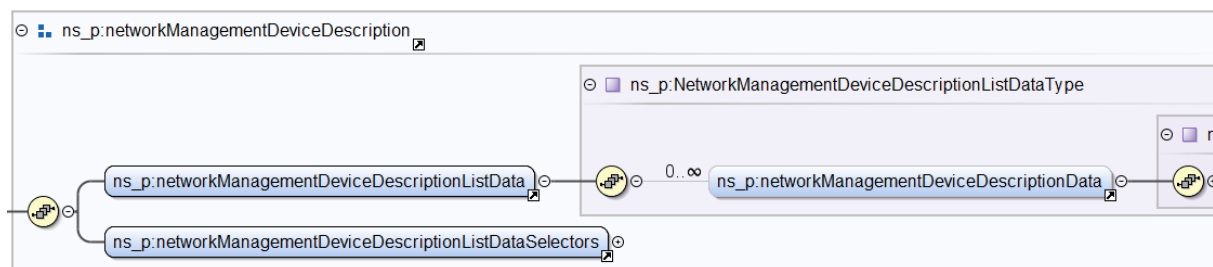
8280

8281 **5.3.17.13 networkManagementDeviceDescriptionListData**

8282 **5.3.17.13.1 General**

8283 This function can be used to model SPINE network management relevant detailed information about
8284 the "device level" of a node.

8285 It is recommended to notify an instance of *networkManagementDeviceDescriptionListData* to all
8286 "parts" requiring knowledge of "any" added/removed/modified device. Considering the architecture
8287 example of section 5.3.17.1.2 this could mean all those software applications that want to react
8288 dynamically on such changes could be notified by the "Network Management Responsible". Thus,
8289 each of these applications could verify whether the reported change affects its own tasks.



8290

8291 *Figure 132: networkManagementDeviceDescriptionListData function overview, part 1*

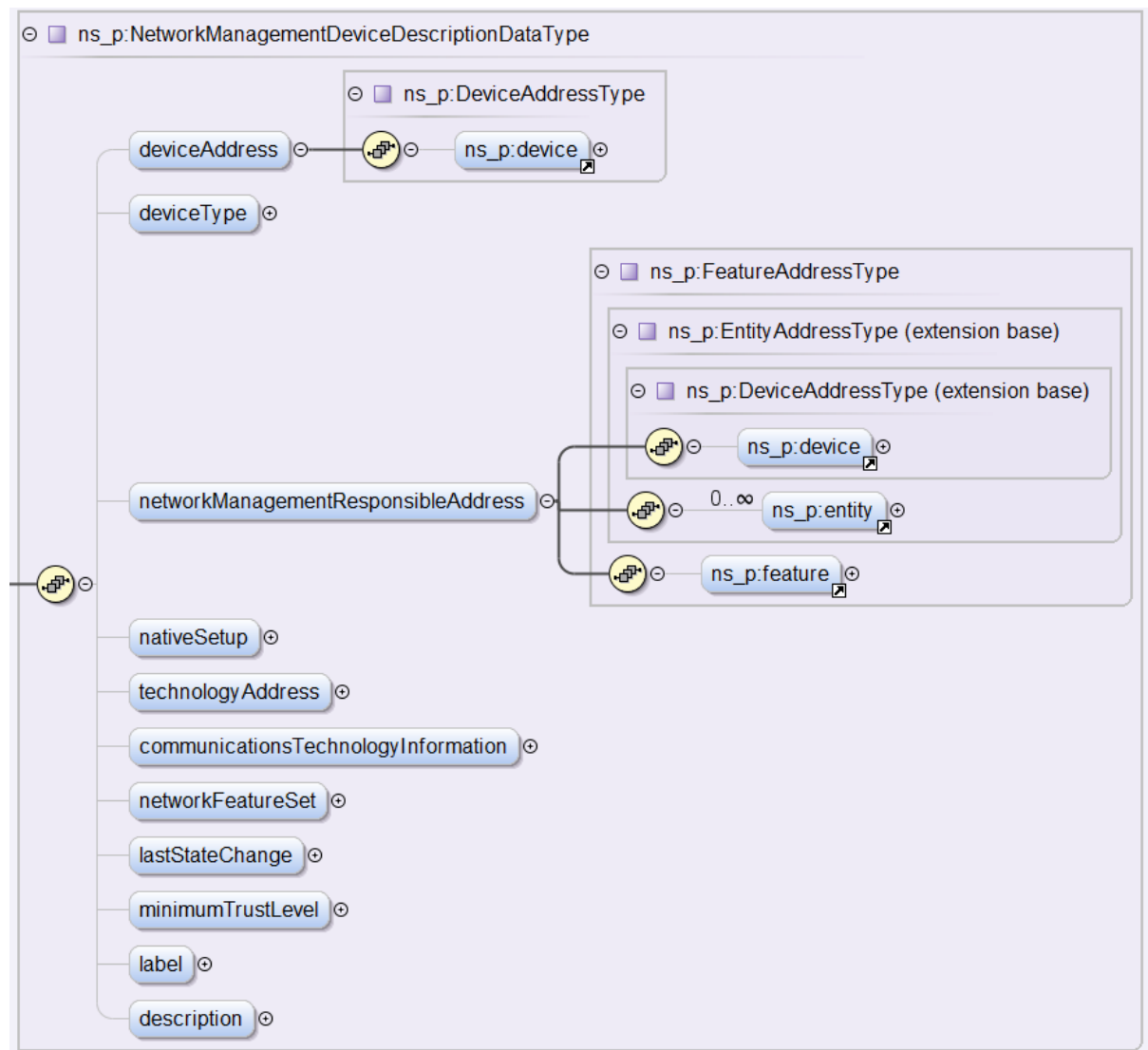


Figure 133: networkManagementDeviceDescriptionListData function overview, part 2

5.3.17.13.2 Detailed description of elements

Element	Type	Description
deviceAddress	Common data type "DeviceAddressType". See section 3.10.3.4.	The node address of the node. Contains elements for <i>device</i> .
deviceAddress.device	See section 3.10.3.4.	Device address part.
deviceType	Common data type "DeviceTypeType". See section 3.10.1.25.	The type of the device. E.g. Washer, Dryer, ...
networkManagementResponsibleAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	The address of the so-called "Network Management Responsible" (see section 5.3.17.1.2) of <i>deviceAddress</i> . Contains elements for <i>device</i> , <i>entity</i> and <i>feature</i> .
networkManagementResponsibleAddress.device	See section 3.10.3.6.	Device address part.
networkManagementResponsibleAddress.entity (list)	See section 3.10.3.6.	Entity (list) address part.

networkManagementResponseAddress.feature	See section 3.10.3.6.	Feature address part.
nativeSetup	Simple type "NetworkManagementNativeSetupType" (restriction of xs:string)	Technology dependent or implementation dependent configuration to identify (and maybe configure) the device.
technologyAddress	Simple type "NetworkManagementTechnologyAddressType" (restriction of xs:string)	The address the device has in its own communications technology.
communicationsTechnologyInformation	Simple type "NetworkManagementCommunicationsTechnologyInformationType" (restriction of xs:string)	Technology dependent or implementation dependent information on the device with focus on communications aspects. <i>communicationsTechnologyInformation</i> is intended to be rather "read only" (but may change dependent on configuration, e.g.).
networkFeatureSet	Enum (see Table 251): NetworkManagementFeatureSetType	Specifies which network capabilities the device has (simple, smart, router, ...).
lastStateChange	Enum (see Table 252): NetworkManagementStateChangeType	This element can contain the last change of its network management state (<i>added</i> , <i>removed</i> , etc.).
minimumTrustLevel	Simple type "NetworkManagementMinimumTrustLevelType" (restriction of xs:string)	Contains the lowest trust level needed, to be allowed to communicate with this node.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the device.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the device.

8296 Table 250: networkManagementDeviceDescriptionData function element description

8297 Enumeration **NetworkManagementFeatureSetType**:

Value	Description
gateway	Connects to devices with another communications technology; keeps a list of its "attached devices" and provides routing to them.
router	Reserved for future use.
smart	Keeps a list of devices of interest for its communication and may permit routing to them.
simple	Can only communicate with directly reachable devices (i.e. no communication via an intermediate SPINE device). Does not provide routing to other devices.

8298 Table 251: Enumeration NetworkManagementFeatureSetType

8299 Enumeration **NetworkManagementStateChangeType**:

Value	Description
added	The device was added. See also 5.3.17.3.
removed	The device was removed. See also 5.3.17.4.
modified	The device was modified. See also 5.3.17.5.

Table 252: Enumeration *NetworkManagementStateChangeType*

8301

8302 *5.3.17.13.3 Available selectors*

8303 - deviceAddress.device

8304

8305 *5.3.17.13.4 Examples*8306 *5.3.17.13.4.1 Notify (change of a "FridgeFreezer")*

8307 After the information of a device has changed, the updated device information is notified to all
8308 subscribed instances (the "filter/partial" part to announce the "restricted function exchange is
8309 simplified by "..." just to improve the readability"):

```

8310 ...
8311 <networkManagementDeviceDescriptionListData>
8312   <networkManagementDeviceDescriptionData>
8313     <deviceAddress>
8314       <device>d:_i:46925_CP\_irto-74629867356210956382438562</device>
8315     </deviceAddress>
8316     <deviceType>FridgeFreezer</deviceType>
8317     <technologyAddress>21.6.5.3.11</technologyAddress>
8318 <communicationsTechnologyInformation>CP;addr=21.6.5.3.11;shortAddr=114</com
8319 municationsTechnologyInformation>
8320   <networkFeatureSet>smart</networkFeatureSet>
8321 </networkManagementDeviceDescriptionData>
8322 </networkManagementDeviceDescriptionListData>

```

8323

8324 *5.3.17.13.4.2 Notify (added "Display")*

8325 We assume a user triggered adding a display as described in section 5.3.17.3.4. That section
8326 mentions the notification of a proper *networkManagementDeviceDescriptionListData* instance. In
8327 this example it is assumed the Network Management Responsible of "CP" has the device address
8328 "d:_i:46925_CP_A873C6BC09EF4E50" and the display has been assigned the node address
8329 "d:_i:46925_CP_pqtl-27638162683172637812356813". Thus, the subsequent XML could be created
8330 (the "filter/partial" part to announce the "restricted function exchange is simplified by "..." just to
8331 improve the readability"):

```

8332 ...
8333 <networkManagementDeviceDescriptionListData>
8334   <networkManagementDeviceDescriptionData>
8335     <deviceAddress>
8336       <device>d:_i:46925_CP\_pqtl-27638162683172637812356813</device>
8337     </deviceAddress>
8338     <deviceType>Generic</deviceType>
8339     <networkManagementResponsibleAddress>
8340       <device>d:_i:46925_CP\_A873C6BC09EF4E50</device>
8341       <entity>0</entity>
8342     </networkManagementResponsibleAddress>

```

```

8343         <nativeSetup>21.6.5.3.11;R-2.7;at-42.344;pqtl-
8344 27638162683172637812356813</nativeSetup>
8345         <lastStateChange>added</lastStateChange>
8346         <minimumTrustLevel>8</minimumTrustLevel>
8347         <label>Display</label>
8348     </networkManagementDeviceDescriptionData>
8349 </networkManagementDeviceDescriptionListData>

```

8350

8351 *5.3.17.13.4.3 Read-reply (from gateway with full device list information)*

8352 We assume a user added a display as described in sections 5.3.17.3.4 and 5.3.17.13.4.2. In this
8353 example, "CP" provides a resource with aggregated information on all "device description details" of
8354 its attached devices and itself. We also assume only the display is currently attached via "CP". The
8355 user wants to see the list of devices from the described resource. He selects this option on the user
8356 interface. The user interface then creates a proper request to the Network Management
8357 Responsible's aggregating resource of "CP" and finally gets a proper response where the Network
8358 Management Responsible itself and the display are described:

```

8359 <networkManagementDeviceDescriptionListData>
8360     <networkManagementDeviceDescriptionData>
8361         <deviceAddress>
8362             <device>d:_i:46925_CP\_A873C6BC09EF4E50</device>
8363         </deviceAddress>
8364         <networkFeatureSet>gateway</networkFeatureSet>
8365         <lastStateChange>modified</lastStateChange>
8366         <minimumTrustLevel>8</minimumTrustLevel>
8367         <label>CP mapping</label>
8368     </networkManagementDeviceDescriptionData>
8369     <networkManagementDeviceDescriptionData>
8370         <deviceAddress>
8371             <device>d:_i:46925_CP\_pqtl-27638162683172637812356813</device>
8372         </deviceAddress>
8373         <deviceType>Generic</deviceType>
8374         <networkManagementResponsibleAddress>
8375             <device>d:_i:46925_CP\_A873C6BC09EF4E50</device>
8376             <entity>0</entity>
8377         </networkManagementResponsibleAddress>
8378         <nativeSetup>21.6.5.3.11;R-2.7;at-42.344;pqtl-
8379 27638162683172637812356813</nativeSetup>
8380         <lastStateChange>added</lastStateChange>
8381         <minimumTrustLevel>8</minimumTrustLevel>
8382         <label>Display</label>
8383     </networkManagementDeviceDescriptionData>
8384 </networkManagementDeviceDescriptionListData>

```

8385

8386 **5.3.17.14 networkManagementEntityDescriptionListData**

8387 *5.3.17.14.1 General*

8388 In addition to the *networkManagementDeviceDescriptionListData* function, this functions can be
8389 used to model SPINE network management relevant detailed information about an "entity level" of a
8390 node.

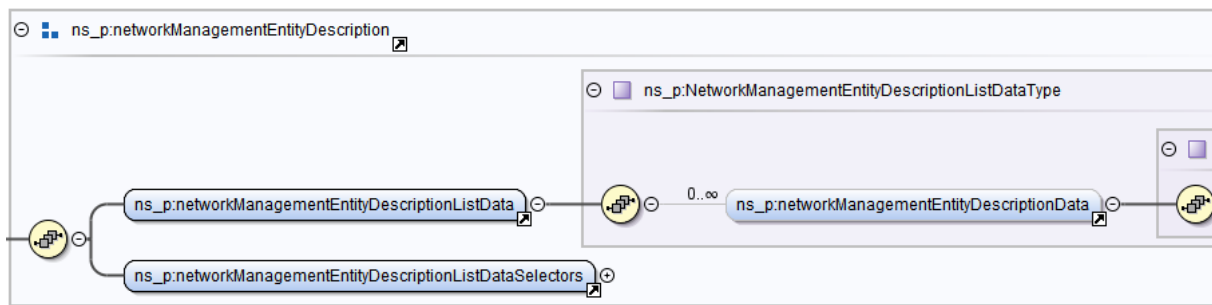


Figure 134: networkManagementEntityDescriptionListData function overview, part 1

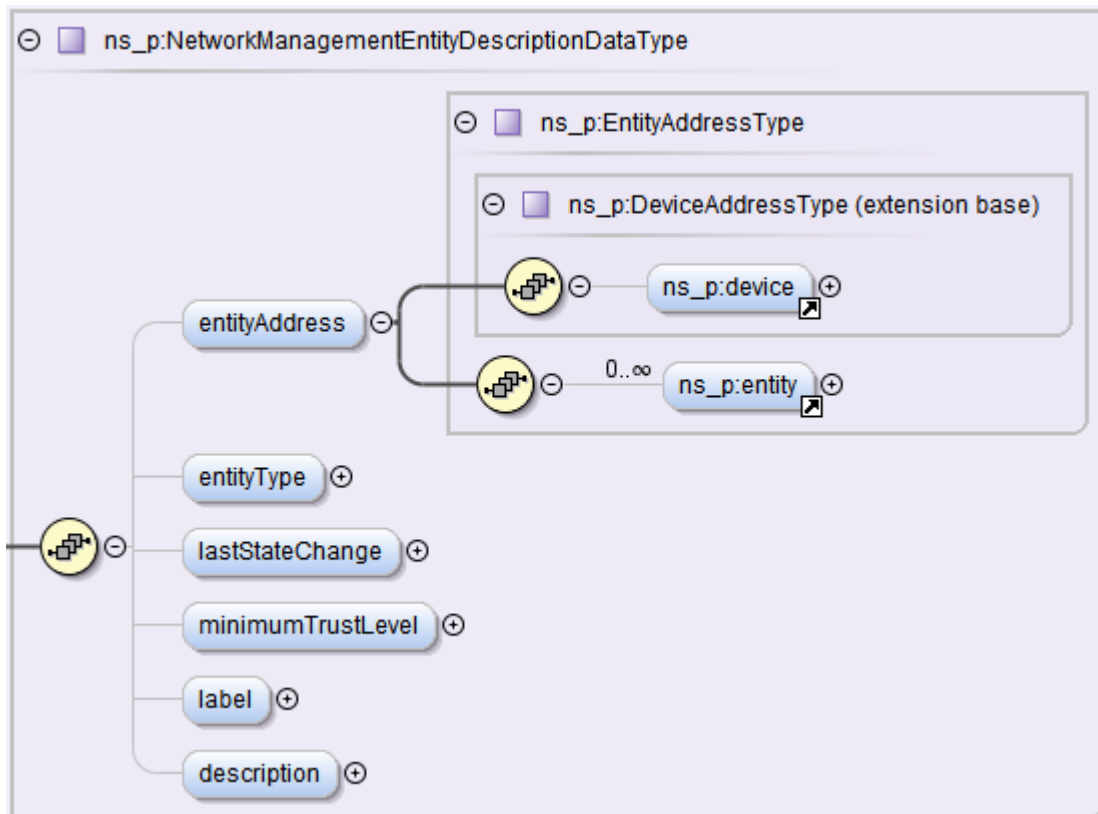


Figure 135: networkManagementEntityDescriptionListData function overview, part 2

5.3.17.14.2 Detailed description of elements

Element	Type	Description
entityAddress	Common data type "EntityType". See section 3.10.3.5.	The node address of the node. Contains elements for <i>device</i> and <i>entity</i> .
entityAddress.device	See section 3.10.3.5.	Device address part.
entityAddress.entity (list)	See section 3.10.3.5.	Entity (list) address part.
entityType	Common data type "EntityType". See section 3.10.1.27.	The type of the entity. E.g. Battery, Fan, ...
lastStateChange	Enum (see Table 252): NetworkManagement StateChangeType	This element can contain the last change of its network management state (<i>added</i> , <i>removed</i> , etc.).

minimumTrustLevel	<i>Simple type "NetworkManagementMinimumTrustLevelType" (restriction of xs:string)</i>	Contains the lowest trust level needed, to be allowed to communicate with this node.
label	<i>Common data type "LabelType". See section 3.10.1.2.</i>	A short label of the entity.
description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Descriptive information on the entity.

Table 253: networkManagementEntityDescriptionData function element description

5.3.17.14.3 Available selectors

- entityAddress

5.3.17.14.4 Examples

5.3.17.14.4.1 Notify

After updating information on its entities, a device notifies the corresponding information to all subscribed instances (the “filter/partial” part to announce the “restricted function exchange is simplified by “...” just to improve the readability”):

```

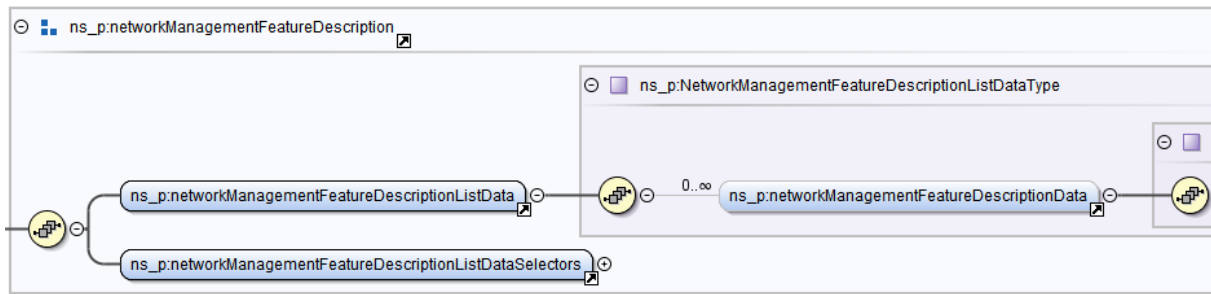
...
<networkManagementEntityDescriptionListData>
  <networkManagementEntityDescriptionData>
    <nodeAddress>
      <device>d:_i:46925_CP\_irto-74629867356210956382438562</device>
      <entity>1</entity>
    </nodeAddress>
    <entityType>Fridge</entityType>
    <label>Big fridge</label>
  </networkManagementEntityDescriptionData>
  <networkManagementEntityDescriptionData>
    <nodeAddress>
      <device>d:_i:46925_CP\_irto-74629867356210956382438562</device>
      <entity>2</entity>
    </nodeAddress>
    <entityType>Freezer</entityType>
    <label>Small freezer compartment</label>
  </networkManagementEntityDescriptionData>
</networkManagementEntityDescriptionListData>

```

5.3.17.15 networkManagementFeatureDescriptionListData

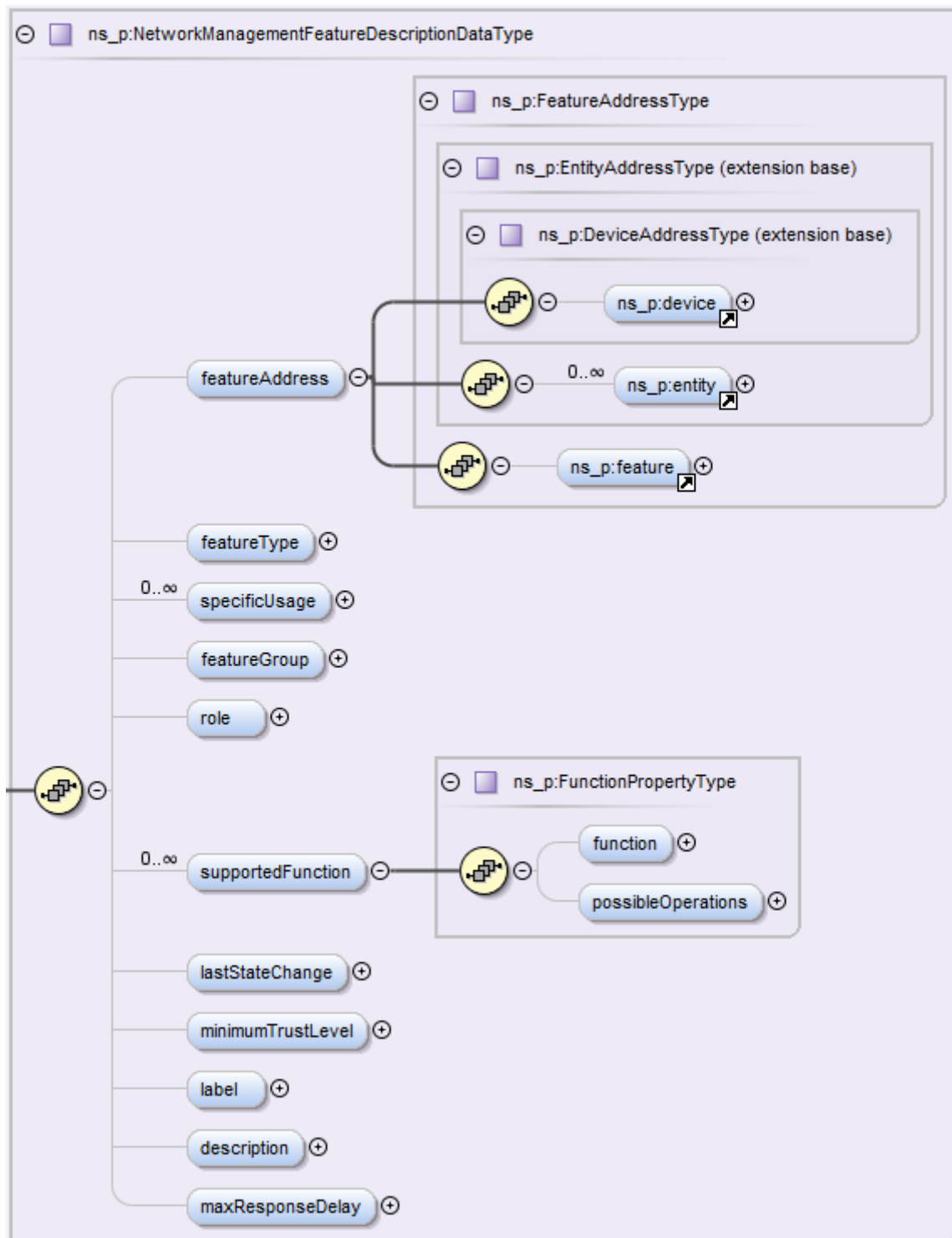
5.3.17.15.1 General

In addition to the *networkManagementDeviceDescriptionListData* function, this functions adds feature-specific information which is only needed for nodes on address hierarchy level *feature*.



8431

8432 *Figure 136: networkManagementFeatureDescriptionListData function overview, part 1*



8433

8434 *Figure 137: networkManagementFeatureDescriptionListData function overview, part 2*

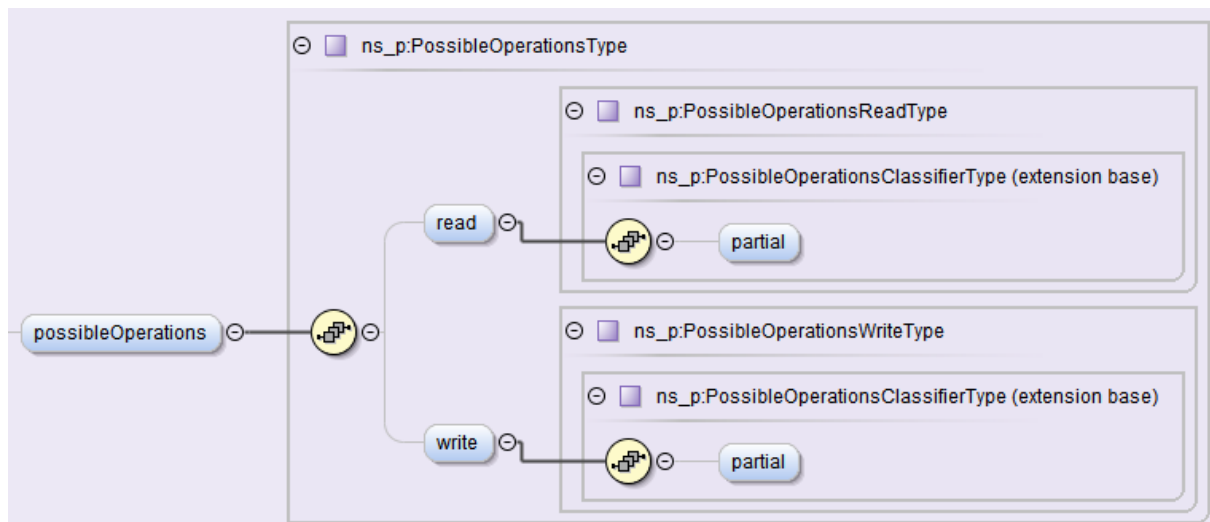


Figure 138: networkManagementFeatureDescriptionListData function overview, part 3

5.3.17.15.2 Detailed description of elements

Element	Type	Description
featureAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	The node address of the node. Contains elements for <i>device</i> , <i>entity</i> and <i>feature</i> .
featureAddress.device	See section 3.10.3.6.	Device address part.
featureAddress.entity (list)	See section 3.10.3.6.	Entity (list) address part.
featureAddress.feature	See section 3.10.3.6.	Feature address part.
featureType	Common data type "FeatureTypeType". See section 3.10.1.29.	The type of the feature. E.g. ActuatorSwitch, Sensing, ...
specificUsage (list)	Common data type "FeatureSpecificUsageType". See section 3.10.1.31.	Each occurrence of "specificUsage" denotes a specific usage (see section 2.1.5) that is implemented for this Feature Type. Please consider the proper definition of "featureType" for possible values.
featureGroup	Common data type "FeatureGroupType". See section 3.10.1.24.	Information to "link" to related features of this entity. See section 2.1.4 for details.
role	Common data type "RoleType". See section 3.10.1.23.	The functional role of this feature (i.e. "client" or "server" or "special").
supportedFunction (list)	Common data type "FunctionPropertyType". See section 3.10.1.41.	Each occurrence contains information on a function that is supported on this feature with the given role. Note: This element is most useful for features with the role "server" or "special" to express which functionality is offered for clients. But it can also be used if the role is "client" in order to express client functionality.

supportedFunction.function	See section 3.10.1.41.	Contains the name of a supported function.
supportedFunction.possibleOperations	See section 3.10.1.41.	Can be used to express some details on operations supported by the given function (which also means the element "function" has to be set properly) if the role is "server" or "special".
supportedFunction.possibleOperations.read	See section 3.10.1.41.	If present, it denotes the feature can receive "read" commands for the given function and creates proper replies.
supportedFunction.possibleOperations.read.partial	See section 3.10.1.41.	If present, it denotes the feature permits the creation of somehow "curtailed" responses (in contrast to full copies of its function instance).
supportedFunction.possibleOperations.write	See section 3.10.1.41.	If present, it denotes the feature can receive "write" commands for the given function and adjusts its own function instance accordingly.
supportedFunction.possibleOperations.write.partial	See section 3.10.1.41.	If present, it denotes the feature accepts also somehow "curtailed" "write" commands (in contrast to a full replacement of its own data).
lastStateChange	Enum (see Table 252): NetworkManagementStateChangeType	This element can contain the last change of its network management state (<i>added, removed, etc.</i>).
minimumTrustLevel	Simple type "NetworkManagementMinimumTrustLevelType" (restriction of xs:string)	Contains the lowest trust level needed, to be allowed to communicate with this node.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the feature.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the feature.
maxResponseDelay	Common data type "MaxResponseDelayType". See section 3.10.1.42.	The duration the Server usually needs to generate a proper reply.

Table 254: networkManagementFeatureDescriptionListData function element description

5.3.17.15.3 Available selectors

- featureAddress

8444 **5.3.17.15.4 Examples**8445 **5.3.17.15.4.1 Notify**

8446 After the functionality of an entity has changed, the new feature information is notified to all bound
 8447 instances (the “filter/partial” part to announce the “restricted function exchange is simplified by “...”
 8448 just to improve the readability”):

```

8449 ...
8450 <networkManagementFeatureDescriptionListData>
8451   <networkManagementFeatureDescriptionData>
8452     <featureAddress>
8453       <device>d:_i:46925_CP\_pqtl-27638162683172637812356813</device>
8454       <entity>1</entity>
8455       <feature>1</feature>
8456     </featureAddress>
8457     <featureType>ActuatorLevel</featureType>
8458     <role>server</role>
8459     <supportedFunction>
8460       <function>actuatorLevelData</function>
8461       <possibleOperations>
8462         <read/>
8463         <write/>
8464       </possibleOperations>
8465     </supportedFunction>
8466     <supportedFunction>
8467       <function>actuatorLevelDescriptionData</function>
8468       <possibleOperations>
8469         <read/>
8470       </possibleOperations>
8471     </supportedFunction>
8472     <lastStateChange>modified</lastStateChange>
8473     <label>Illumination</label>
8474     <maxResponseDelay>PT20S</maxResponseDelay>
8475   </networkManagementFeatureDescriptionData>
8476 </networkManagementFeatureDescriptionListData>

```

8477

8478 **5.3.18 OperatingConstraints**8479 **5.3.18.1 Introduction**

8480 This class is designed to model some constraints of an appliance (or functionality) with regards to its
 8481 energy consumption or production. I.e. this class is NOT intended to control a device. Rather, it
 8482 enables an appliance to express what needs to be considered especially for the case it permits being
 8483 controlled by another device. An example is a central energy management system that uses the class
 8484 DirectControl to control the appliance; in this case the energy management system needs to be
 8485 aware of the appliance’s operating constraints.

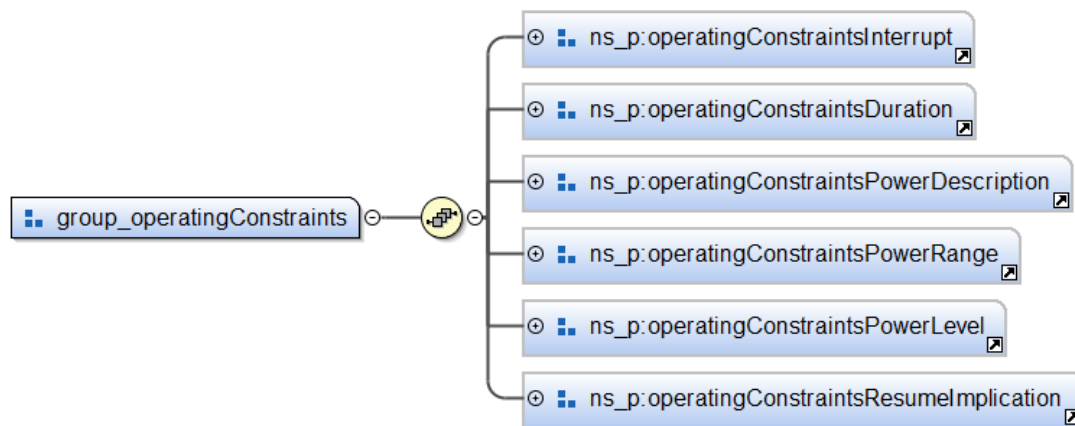


Figure 139: OperatingConstraints function-group overview

In general the class OperatingConstraints can be used with an without relation to the class PowerSequences. As can be seen in the following functions this is usually achieved with the (optional) element “sequenceld”. In fact, most functions of this class are list-based, hence provide a possibility to model also constraints per power sequence.

Without a relation to a power sequence the OperatingConstraints functions usually relate to “now”, i.e. the current state of the functionality. With the relation to a power sequence the corresponding constraints apply for the time of the related power sequence. Care should be taken if some constraints depend on the power sequence and other constraints are rather “global”, i.e. independent from a power sequence. In every case the use of a feature with such a function should be clearly defined in order to avoid ambiguities. The following brief recommendations may be helpful to achieve this:

1. Avoid a mixture of operatingConstraintsInterruptData items with and without sequenceld within operatingConstraintsInterruptListData.
2. Within operatingConstraintsInterruptListData NO child element of operatingConstraintsInterruptData should depend on sequenceld in some operatingConstraintsInterruptData items and be independent of sequenceld in other operatingConstraintsInterruptData items.

5.3.18.2 operatingConstraintsInterruptListData

5.3.18.2.1 General

This function models basic constraints for an interrupt of a device's current task.

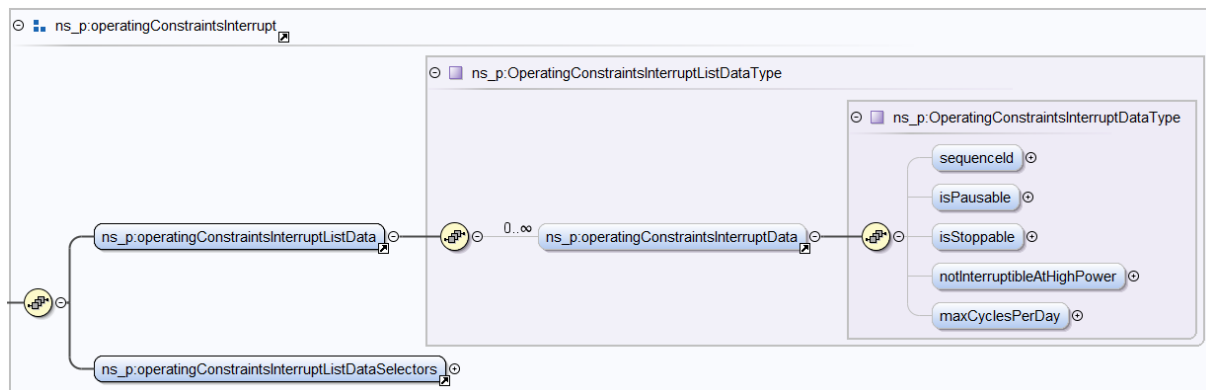


Figure 140: operatingConstraintsInterruptListData function overview

5.3.18.2.2 Detailed description of elements

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	If interrupt constraints are related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
isPausable	xs:boolean (W3C standard type)	States whether a functionality or task is pausable (true) or not (false).
isStoppable	xs:boolean (W3C standard type)	States whether a functionality or task is stoppable (true) or not (false).
notInterruptibleAtHighPower	xs:boolean (W3C standard type)	States whether a functionality or task is NOT interruptible while it is consuming (or producing) energy (true) or whether it can be interrupted nevertheless (false).
maxCyclesPerDay	xs:unsignedInt (W3C standard type)	If a functionality or task may be started only a maximum number of times per day, this value is denoted here.

Table 255: operatingConstraintsInterruptListData function element description

5.3.18.2.3 Available selectors

- sequenceId

5.3.18.2.4 Examples

5.3.18.2.4.1 Read-reply

We assume an energy management system wants to control an appliance. We also assume the appliance expresses details on its process with a power sequence (with sequenceId "1" in this example). The energy management system queries the appliance's operatingConstraintsInterrupt information and gets the following response:

```
<operatingConstraintsInterruptListData>
  <operatingConstraintsInterruptData>
    <sequenceId>1</sequenceId>
    <isPausable>true</isPausable>
```

```

8528         <notPausableAtHighPower>true</notPausableAtHighPower>
8529     </operatingConstraintsInterruptData>
8530 </operatingConstraintsInterruptListData>

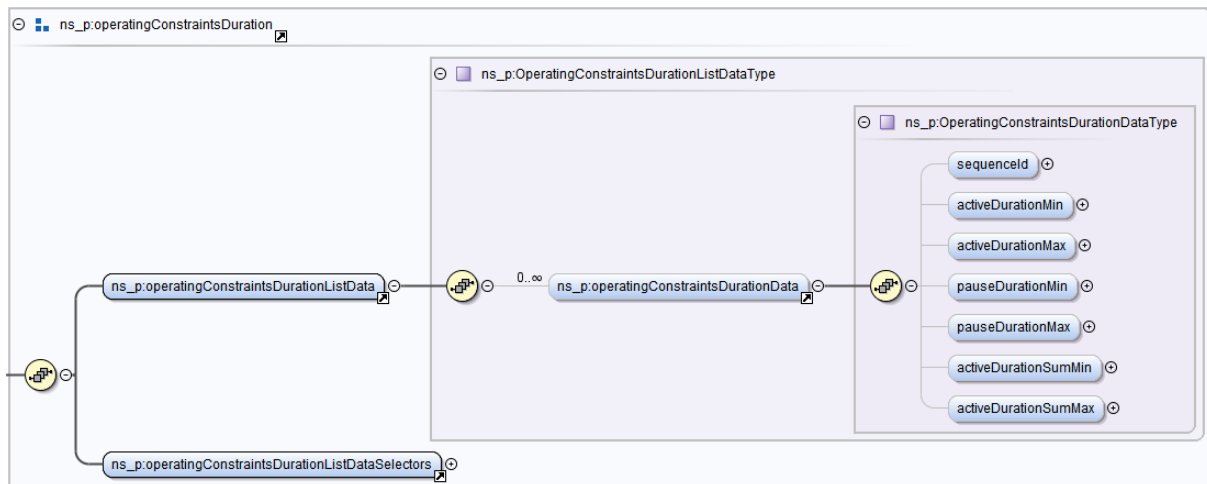
```

8531

8532 5.3.18.3 *operatingConstraintsDurationListData*

8533 5.3.18.3.1 *General*

8534 This function models constraints on the minimum/maximum duration of active ("on") and pause
 8535 ("off") phases. Example: A compressor can be damaged if it is started and then switched off too early.



8536

8537 Figure 141: *operatingConstraintsDurationListData* function overview

8538

8539 5.3.18.3.2 *Detailed description of elements*

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	If duration constraints are related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
activeDurationMin	xs:duration (W3C standard type)	This is the minimum duration a device has to run without interruption.
activeDurationMax	xs:duration (W3C standard type)	This is the maximum duration a device can run without interruption.
pauseDurationMin	xs:duration (W3C standard type)	This is the minimum duration a device has to pause after the end of an activity.
pauseDurationMax	xs:duration (W3C standard type)	This is the maximum duration a device can pause after the end of an activity.
activeDurationSumMin	xs:duration (W3C standard type)	This is the minimum duration a device has to run in total for a given task (summation of all active times).
activeDurationSumMax	xs:duration (W3C standard type)	This is the maximum duration a device can run in total for a given task (summation of all active times).

8540 Table 256: *operatingConstraintsDurationListData* function element description

8541

5.3.18.3.3 Available selectors

- `sequenceId`

5.3.18.3.4 Examples

5.3.18.3.4.1 Read-reply

This example continues the example of section 5.3.18.2.4.1. The energy management system queries for the appliance's duration constraints and gets the following response:

```
<operatingConstraintsDurationListData>
  <operatingConstraintsDurationData>
    <sequenceId>1</sequenceId>
    <activeDurationMin>PT10M</activeDurationMin>
    <activeDurationMax>PT1H</activeDurationMax>
    <pauseDurationMin>PT5M30S</pauseDurationMin>
    <pauseDurationMax>P1D</pauseDurationMax>
    <activeDurationSumMin>PT3H15M</activeDurationSumMin>
    <activeDurationSumMax>PT5H30M</activeDurationSumMax>
  </operatingConstraintsDurationData>
</operatingConstraintsDurationListData>
```

5.3.18.4 operatingConstraintsPowerDescriptionListData

5.3.18.4.1 General

This function models basic information relevant for *operatingConstraintsPowerRangeListData* and *operatingConstraintsPowerLevelListData*.

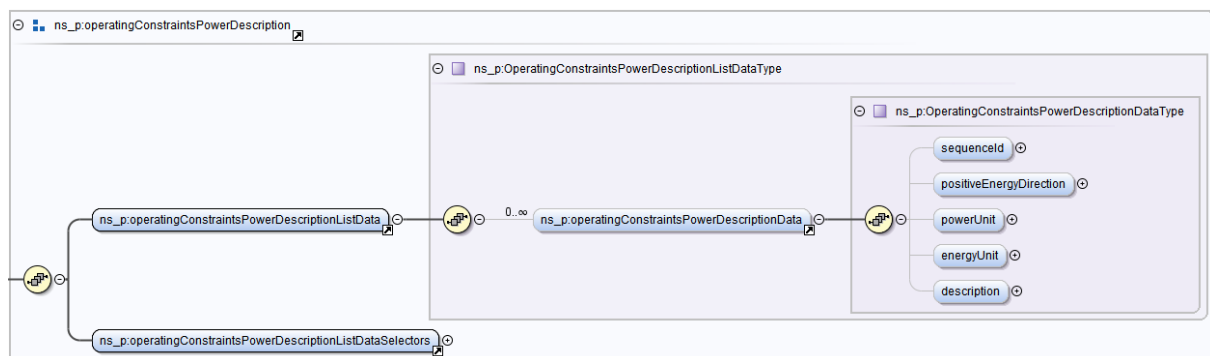


Figure 142: *operatingConstraintsPowerDescriptionListData* function overview

5.3.18.4.2 Detailed description of elements

Element	Type	Description
<code>sequenceId</code>	Identifier "PowerSequenceIdType" (see Table 339).	If function elements are related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
<code>positiveEnergyDirection</code>	Common data type "EnergyDirectionType". See section 3.10.1.13.	The <i>positiveEnergyDirection</i> states whether energy consumption or production is counted as positive value.

powerUnit	<i>Common data type "UnitOfMeasurement Type". See section 3.10.1.17.</i>	The unit for all power values in operatingConstraintsPowerRangeListData and operatingConstraintsPowerLevelListData.
energyUnit	<i>Common data type "UnitOfMeasurement Type". See section 3.10.1.17.</i>	The unit for all energy values in operatingConstraintsPowerRangeListData and operatingConstraintsPowerLevelListData.
description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Descriptive information.

Table 257: operatingConstraintsPowerDescriptionListData function element description

5.3.18.4.3 Available selectors

- sequenceId

5.3.18.4.4 Examples

5.3.18.4.4.1 Read-reply

This example continues the example of section 5.3.18.2.4.1. The energy management system queries for the appliance's description on constraints and gets the following response:

```
<operatingConstraintsPowerDescriptionListData>
  <operatingConstraintsPowerDescriptionData>
    <sequenceId>1</sequenceId>
    <positiveEnergyDirection>consume</positiveEnergyDirection>
    <powerUnit>W</powerUnit>
    <energyUnit>Wh</energyUnit>
    <description>Constraints for sequence 1</description>
  </operatingConstraintsPowerDescriptionData>
</operatingConstraintsPowerDescriptionListData>
```

5.3.18.5 operatingConstraintsPowerRangeListData

5.3.18.5.1 General

This function can be used to model a power/energy range(!) a device can operate. An alternative model for power/energy constraints is operatingConstraintsPowerLevelListData.

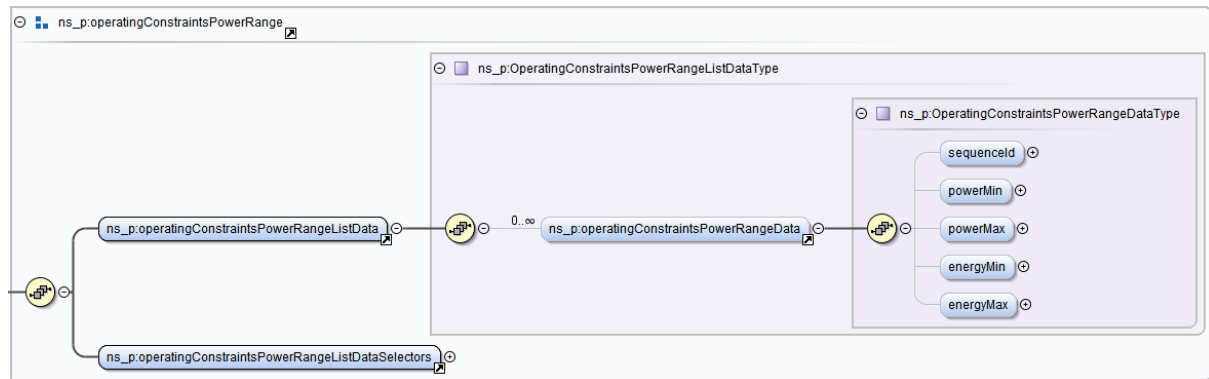


Figure 143: operatingConstraintsPowerRangeListData function overview

5.3.18.5.2 Detailed description of elements

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	If the power range is related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
powerMin	Common data type "ScaledNumberType". See section 3.10.1.8.	The minimum power a device can consume/produce.
powerMax	Common data type "ScaledNumberType". See section 3.10.1.8.	The maximum power a device can consume/produce.
energyMin	Common data type "ScaledNumberType". See section 3.10.1.8.	The minimum energy a device can consume/produce.
energyMax	Common data type "ScaledNumberType". See section 3.10.1.8.	The maximum energy a device can consume/produce.

Table 258: operatingConstraintsPowerRangeListData function element description

5.3.18.5.3 Available selectors

- sequenceId

5.3.18.5.4 Examples

5.3.18.5.4.1 Read-reply

This example continues the example of section 5.3.18.4.4.1. The energy management system queries for the appliance's power range constraints and gets the following response:

```

<operatingConstraintsPowerRangeListData>
  <operatingConstraintsPowerRangeData>
    <sequenceId>1</sequenceId>
    <powerMin>
      <number>0</number>
    </powerMin>
    <powerMax>
      <number>2</number>
      <scale>3</scale>
    </powerMax>
    <energyMin>
      <number>5</number>

```

```

8617         <scale>2</scale>
8618     </energyMin>
8619     <energyMax>
8620         <number>21</number>
8621         <scale>2</scale>
8622     </energyMax>
8623 </operatingConstraintsPowerRangeData>
8624 </operatingConstraintsPowerRangeListData>

```

5.3.18.6 operatingConstraintsPowerLevelListData

5.3.18.6.1 General

This function can be used to model a list of possible power values a device can operate. An alternative model for power constraints is operatingConstraintsPowerRangeListData, where a full range (not only discrete levels) can be modelled.

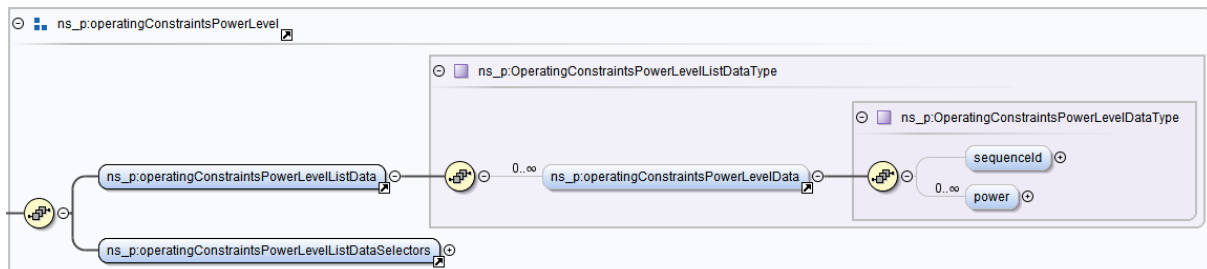


Figure 144: operatingConstraintsPowerLevelListData function overview

5.3.18.6.2 Detailed description of elements

Element	Type	Description
sequenceld	Identifier "PowerSequenceldType" (see Table 339).	If the power levels are related to a power sequence, the appropriate <i>sequenceld</i> is stated in this element.
power (list)	Common data type "ScaledNumberType". See section 3.10.1.8.	Each "power" item contains a possible power level.

Table 259: operatingConstraintsPowerLevelListData element description

5.3.18.6.3 Available selectors

- sequenceld

5.3.18.6.4 Examples

5.3.18.6.4.1 Read-reply

This example continues the example of section 5.3.18.4.4.1. The energy management system queries for the appliance's power level constraints and gets the following response:

```

8644 <operatingConstraintsPowerLevelListData>
8645     <operatingConstraintsPowerLevelData>

```

```

8646         <sequenceId>1</sequenceId>
8647         <power>
8648             <number>5</number>
8649             <scale>2</scale>
8650         </power>
8651         <power>
8652             <number>15</number>
8653             <scale>2</scale>
8654         </power>
8655         <power>
8656             <number>25</number>
8657             <scale>2</scale>
8658         </power>
8659     </operatingConstraintsPowerLevelData>
8660 </operatingConstraintsPowerLevelListData>

```

5.3.18.7 operatingConstraintsResumeImplicationListData

5.3.18.7.1 General

Pausing a device might be permitted according to above mentioned (duration) constraints. Nevertheless, it can be expected that it has implications if a device's task is paused temporarily and then resumed to continue the task. One possible implication is an additional mechanical stress which reduces the device's lifetime and could be interpreted as additional cost. Such additional costs or additional energy consumption can be modelled with this function in order to be considered for energy management. However, these are typically empirical (thus rather constant) data, therefore should not be considered to be precise.

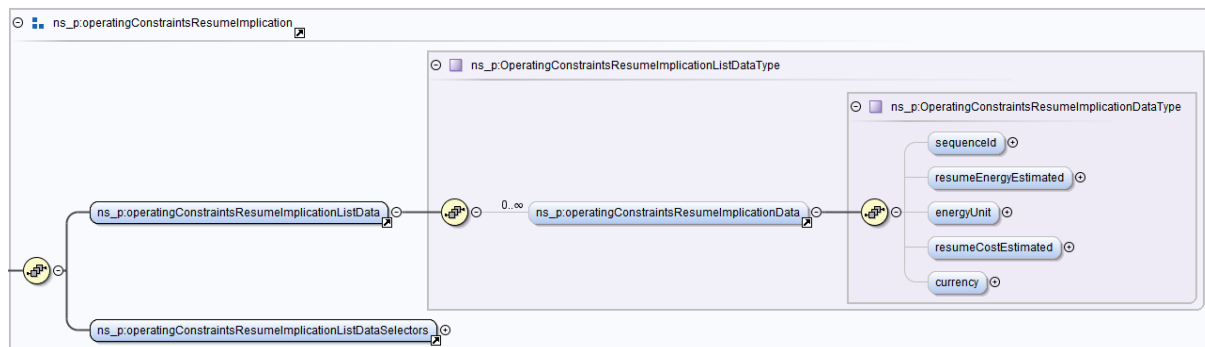


Figure 145: operatingConstraintsResumeImplicationListData function overview

5.3.18.7.2 Detailed description of elements

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	If the resume implications are related to a power sequence, the appropriate <i>sequenceId</i> is stated in this element.
resumeEnergyEstimated	Common data type "ScaledNumberType". See section 3.10.1.8.	Additional energy a device consumes before resuming to its normal operation (after a pause).
energyUnit	Common data type "UnitOfMeasurement"	Unit for the energy stated in the element <i>resumeEnergyEstimated</i> .

	<i>Type". See section 3.10.1.17.</i>	
resumeCostEstimated	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Additional costs for the resumption of a device to its normal operation.
currency	<i>Common data type "CurrencyType". See section 3.10.1.19.</i>	Currency for the price stated in the element resumeCostEstimated.

Table 260: operatingConstraintsResumeImplicationListData function element description

5.3.18.7.3 Available selectors

- sequenceId

5.3.18.7.4 Examples

5.3.18.7.4.1 Read-reply

This example continues the example of section 5.3.18.2.4.1. The energy management system queries for the appliance's resume implications and gets the following response:

```

<operatingConstraintsResumeImplicationListData>
  <operatingConstraintsResumeImplicationData>
    <sequenceId>1</sequenceId>
    <resumeEnergyEstimated>
      <number>5</number>
      <scale>1</scale>
    </resumeEnergyEstimated>
    <energyUnit>Wh</energyUnit>
    <resumeCostEstimated>
      <number>2</number>
      <scale>-2</scale>
    </resumeCostEstimated>
    <currency>EUR</currency>
  </operatingConstraintsResumeImplicationData>
</operatingConstraintsResumeImplicationListData>

```

5.3.19 PowerSequences

5.3.19.1 Introduction

This class primarily permits to model curves of power and energy over time. This is considered in the sub-class PowerTimeSlot (see 5.3.19.2). Secondly, the class provides definitions for the modelling of power scheduling including alternative plans.

5.3.19.1.1 Basic concepts

Throughout this class two terms are of major interest: *Sequence* and *Slot*. A sequence can be used to describe a "task". It is the most "coarse" view in this class, i.e. a sequence represents all single steps of a whole task. The single steps are represented by slots. Slots are the most "fine" view in this class with regards to the time resolution of a task.

Each sequence and slot can be associated an identifier or number, respectively. The slot numbers of two sequences should be considered independent from each other. I.e. slot number 7 of sequence 1 describes a different slot than slot number 7 of sequence 2. This basically means a slot is only "uniquely" identified in combination with a sequence ID. As formal notation this leads to the following relation:

- 1 sequence has 0..m slots
- 1 slot belongs to 0..1 sequence
- 1 pair of { sequenceId, slotNumber } uniquely identifies 1 slot

These relations are quite general as they permit modelling sequences with no slot and also the use of slots without sequences. However, for the most applications it can be expected that constraints are required where a sequence has at least 1 slot and a slot belongs to exactly one sequence.

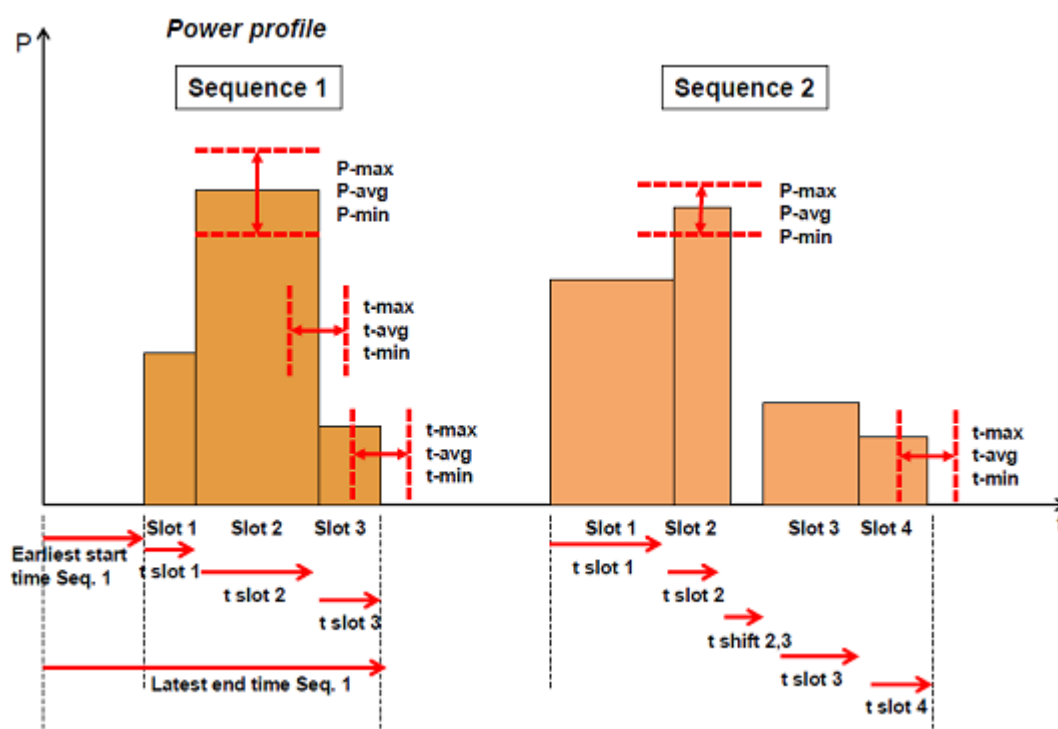
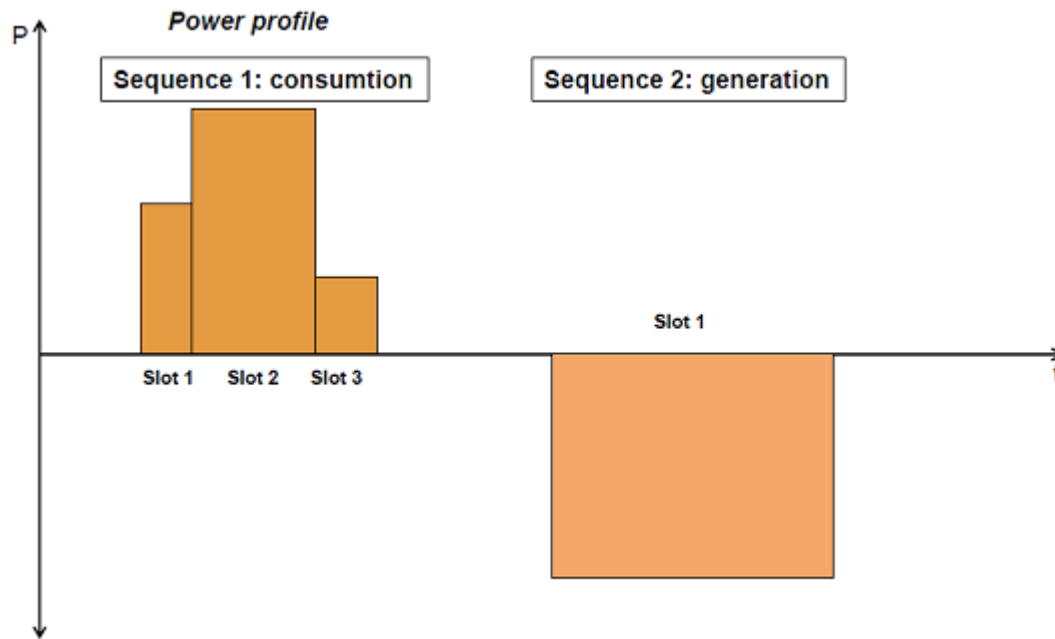


Figure 146: Concept overview of power sequences: Power P over time t ; sequences are constituted of slots

Irrespective of whether the Smart Device is a consumer or producer it needs to announce a kind of expected energy consumption or generation (power profile) to allow energy allocation within smart premises.

For example, a heat pump can ask for allocating 2 sequences per day, once in the morning, once in the afternoon, each sequence with different phases of power consumption. Alternatively, a smart device can offer multiple sequences as alternatives to each other (for the same time period, e.g.). An example for such alternative plans can be a battery pack offering charging or discharging. A customer energy management (CEM) system may then choose between these possibilities. Some more details on alternative plans will be described later on.

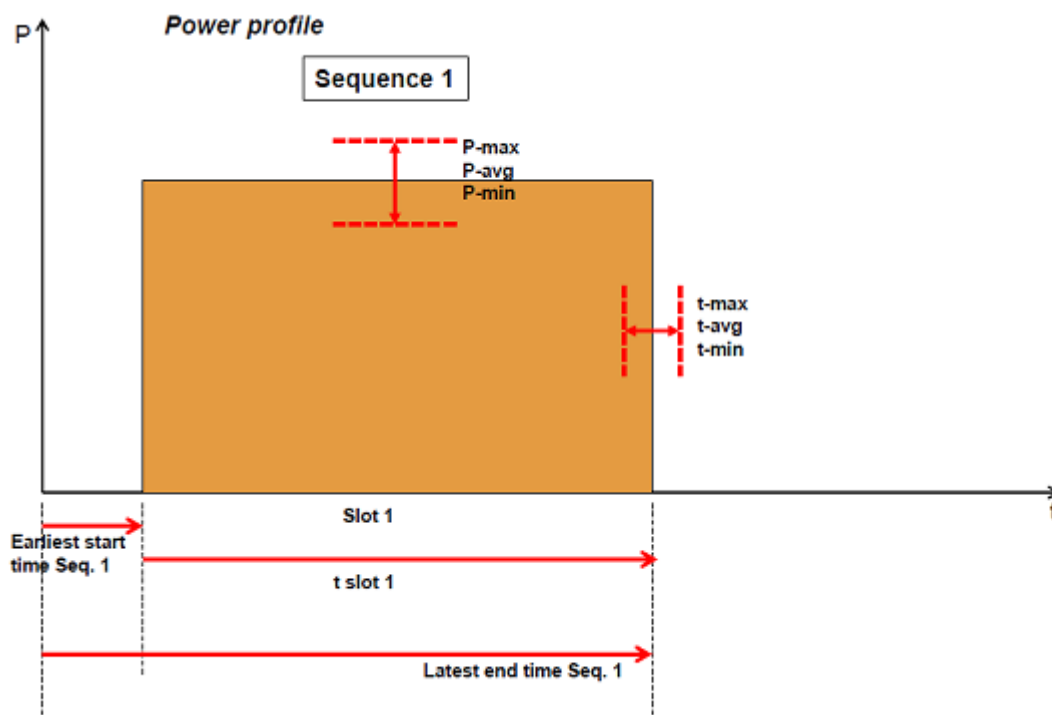
Power sequences may exist for energy consumption or production.



8734

8735 Figure 147: Example of a device that first consumes and later on generates power. In this case “consumption” is assigned a
 8736 positive value.

8737 The easiest way to express the expected energy consumption / generation is one sequence with one
 8738 timeframe with one power value over timeslot.



8739

8740 Figure 148: Example of a sequence with only one slot with some possible parameter boundaries.

8741

8742 *5.3.19.1.2 Repetition concept*

8743 In its most basic form, a sequence can execute its explicitly announced slots one after the other and
8744 is completed when the last of these slots was completed. However, the PowerSequences also
8745 permits a “repetition concept” where a sequence is executed a given number of times.

8746 As example we consider a sequence with three explicitly announced slots. As each slot represents a
8747 “phase” of the sequence, we denote the three phases “A”, “B”, and “C”. In this example the
8748 sequence shall be executed two times. This finally means that this sequence of phases is executed:
8749 “A”, “B”, “C”, “A”, “B”, “C”.

8750

8751 *5.3.19.1.3 Alternative sequences*

8752 As mentioned above, the class permits the modelling of “alternative plans”. This means two or more
8753 sequences may be offered by an appliance to an energy management system (e.g.) to select none or
8754 exactly one of the alternatives for the execution. Among others, the so-called “state” of a power
8755 sequence can be useful to distinguish the selected sequence from the remaining alternatives (see
8756 section 5.3.19.9).

8757 Some examples describe subsequent tasks (an appliance executes first sequence 1 and some time
8758 later it executes sequence 2, e.g.). The PowerSequences class permits as well the combination of
8759 alternatives and sequential plans. I.e. alternatives are grouped together and the groups are
8760 considered sequentially. Section 5.3.19.7 is useful to model groups of alternatives.

8761

8762 *5.3.19.1.4 Overview on the class modules*

8763 The SPINE data model of this class follows the design principle of modular definitions. As a result the
8764 underlying revision of the SPINE data model contains almost no relation to slots within the sub-class
8765 PowerSequence. Thus, in theory it is possible to use PowerSequence without PowerTimeSlot.
8766 However, this is not the primary use case of PowerSequence and is not considered further in this
8767 document.

8768 PowerTimeSlot, on the other hand, uses *sequenceId* in many functions, e.g. Therefore, using
8769 PowerTimeSlot without PowerSequences is not considered in detail.

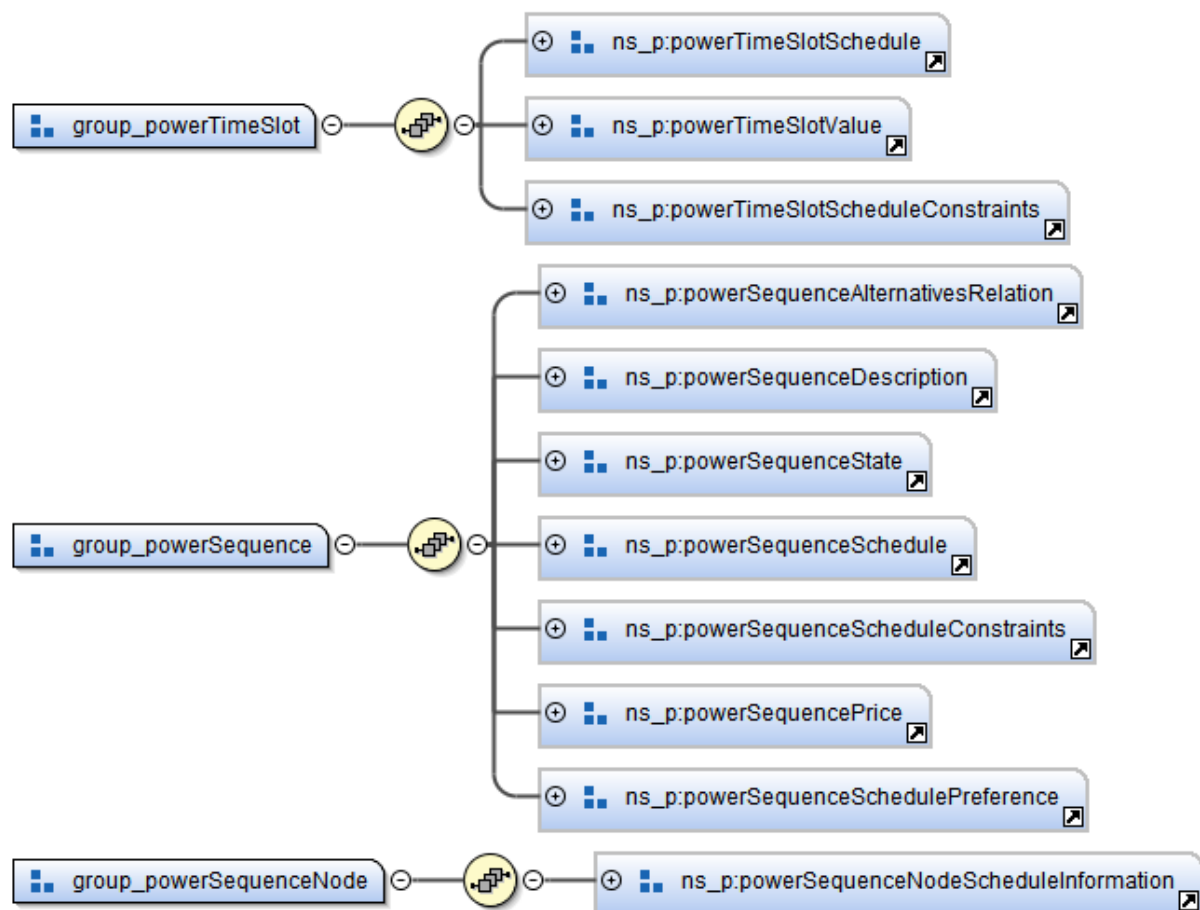


Figure 149: PowerSequences function-group overview (data)

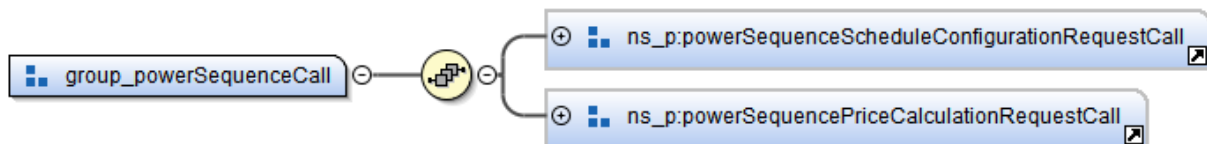


Figure 150: PowerSequences function-group overview (call)

5.3.19.2 Sub-class PowerTimeSlot – Power values, facets, curves

The root of this sub-class is the consideration of single power or energy values. Each value can be "enriched" with miscellaneous facets like a timestamp, durations, dependency to other values, information on its "precision", etc. The combination of the single values or facets into lists permits modelling power curves over time, or modelling of "variants" at a given time, to give two examples.

With regards to the time and dependency facets it is possible to model scheduling of periods or whole curves. Constraints on scheduling can be modelled, too.

Please note this class basically supports modelling curves for "any purpose". One purpose might be the announcement of a device's planned energy consumption. Another purpose might be the presentation of a calculated or potential power availability. Esp. information from the *PowerSequence* class may provide further information on the use of slots and curves.

5.3.19.3 powerTimeSlotScheduleListData

5.3.19.3.1 General

This function can be used to model the schedule of a single slot. This information is intended to be used as actual information rather than schedule constraints (see 5.3.19.5). Together with a proper "write" classifier it can also be used to model the configuration of a slot schedule.

As already mentioned in 5.3.19.2 the "purpose" or applicability of a slot might depend on further information.

The function distinguishes between slots with "absolute" and "differential" schedule. The "absolute" schedule applies the element "timePeriod" (see below) and should be considered the most relevant information and the most common use. It can be used for curves with known or "already decided" time information, e.g. As timePeriod's sub-elements make use of the type AbsoluteOrRelativeTimeType it is possible to express all slot's time information with UTC or relative to "now" (or from a different point of view: "absolute" with "now" as reference time). A differential schedule can be used to model theoretical curves, e.g. The latter is useful for curves having no absolute schedule yet.

Of course, the slots of a curve shall not overlap in time. On the other hand, subsequent slots may well be separated by a gap in time. Mathematically spoken the slots of a sequence shall be disjoint and may be non-contiguous with regards to their schedule.

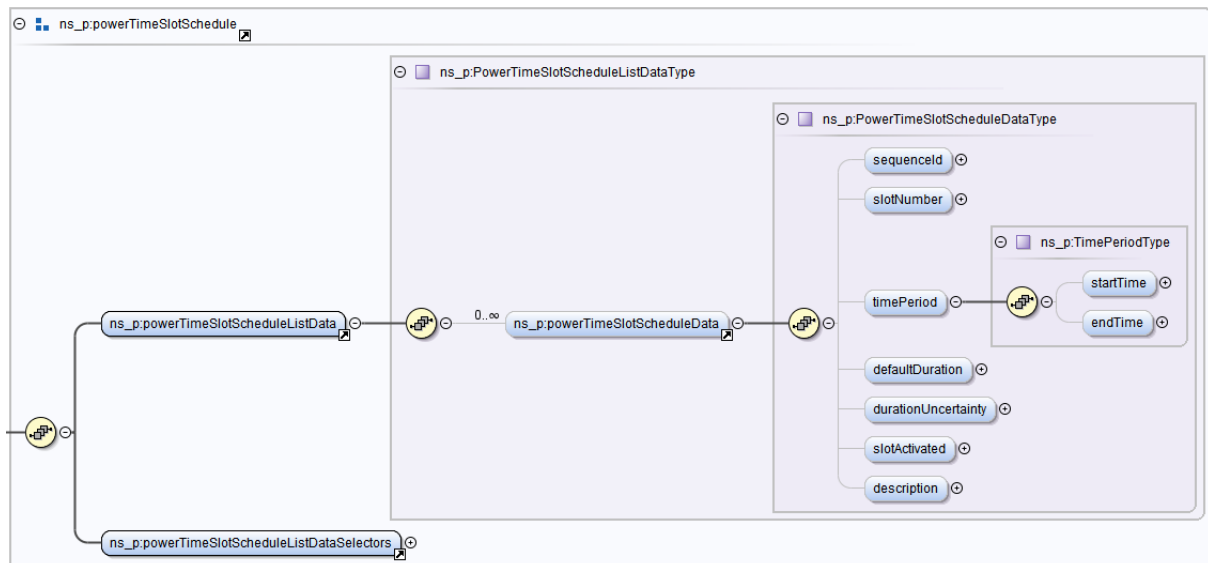


Figure 151: powerTimeSlotScheduleListData function overview

5.3.19.3.2 Detailed description of elements

Element	Type	Description
sequencedId	Identifier "PowerSequencedType" (see Table 339).	The sequence identifier.
slotNumber	Identifier "PowerTimeSlotNumb"	The number of the slot.

	<i>erType</i> " (see Table 339).	
timePeriod	<i>Common data type "TimePeriodType". See section 3.10.2.1.</i>	Time information on the slot.
timePeriod.startTime	<i>See section 3.10.2.1.</i>	Start time of the slot.
timePeriod.endTime	<i>See section 3.10.2.1.</i>	End time of the slot.
defaultDuration	xs:duration (W3C standard type)	The duration of this slot.
durationUncertainty	xs:duration (W3C standard type)	The defaultDuration may have some uncertainty. The actual duration then may be in the range from <i>defaultDuration - durationUncertainty</i> to <i>defaultDuration + durationUncertainty</i> .
slotActivated	xs:boolean (W3C standard type)	This element permits modelling the state of an "optional slot". Whether a slot is activated or not is specified here. The value "true" denotes the slot is used, whereas "false" means it is not used. See also element "optionalSlot" of function "powerTimeSlotScheduleConstraintsList Data" (see Table 264).
description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Descriptive information on this slot.

Table 261: powerTimeSlotScheduleListData function element description

5.3.19.3.3 Available selectors

- sequenceId
- slotNumber

5.3.19.3.4 Examples

5.3.19.3.4.1 Read-reply

An energy management gateway was notified by a fridge about an upcoming task, referenced by sequence 2. The energy management gateway queries for powerTimeSlotScheduleListData schedule details on sequence 2 and receives a proper reply from the fridge:

```

<powerTimeSlotScheduleListData>
  <powerTimeSlotScheduleData>
    <sequenceId>2</sequenceId>
    <slotNumber>1</slotNumber>
    <timePeriod>
      <startTime>2006-05-04T16:15:00.0Z</startTime>
      <endTime>2006-05-04T16:22:00.0Z</endTime>
    </timePeriod>
    <defaultDuration>PT7M</defaultDuration>
  </powerTimeSlotScheduleData>
</powerTimeSlotScheduleListData>

```

```

8829         <durationUncertainty>PT0S</durationUncertainty>
8830         <slotActivated>true</slotActivated>
8831         <description>Super freeze</description>
8832     </powerTimeSlotScheduleData>
8833     <powerTimeSlotScheduleData>
8834         <sequenceId>2</sequenceId>
8835         <slotNumber>2</slotNumber>
8836         <timePeriod>
8837             <startTime>2006-05-04T16:25:00.0Z</startTime>
8838             <endTime>2006-05-04T16:29:30.0Z</endTime>
8839         </timePeriod>
8840         <defaultDuration>PT4M30S</defaultDuration>
8841         <durationUncertainty>PT5S</durationUncertainty>
8842         <slotActivated>true</slotActivated>
8843         <description>Freeze</description>
8844     </powerTimeSlotScheduleData>
8845 </powerTimeSlotScheduleListData>

```

5.3.19.4 powerTimeSlotValueListData

5.3.19.4.1 General

This function can be used to model miscellaneous values associated to a single slot. The most essential values are certainly the "real" power and the "real" energy. In addition it is possible to model value facets like average, skewness, minimum, etc.

Please note the units of energy and power values are modelled in the PowerSequence sub-class.

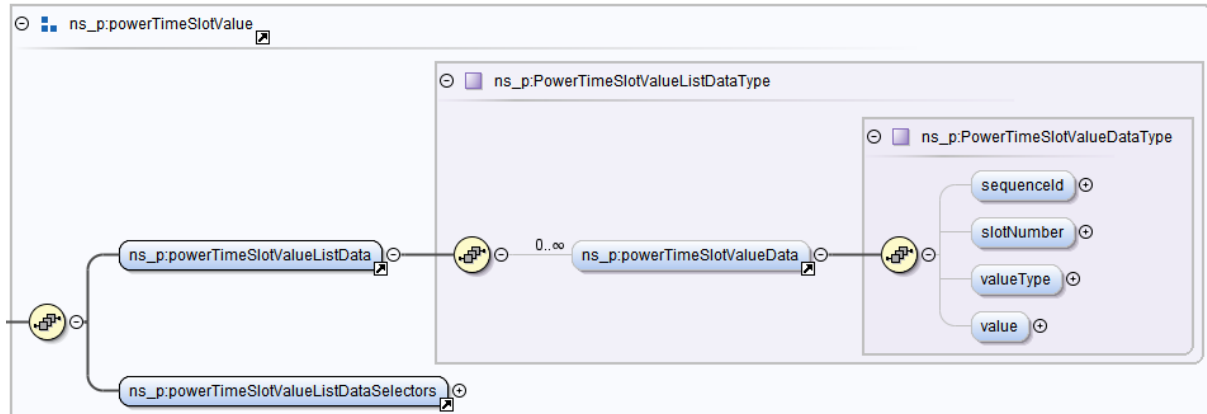


Figure 152: powerTimeSlotValueListData function overview

5.3.19.4.2 Detailed description of elements

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	The sequence identifier.
slotNumber	Identifier "PowerTimeSlotNumberType" (see Table 339).	The number of the slot.
valueType	Union "PowerTimeSlotValueTypeType": - Enum (see Table 263): PowerTimeSlotValueTypeEnumType - EnumExtendType (see section 3.10.1.5)	The kind of the value ("power", "powerMin", "energy", ...).

value	Common data type "ScaledNumberType". See section 3.10.1.8.	The magnitude of the value denoted by valueType.
-------	--	--

Table 262: powerTimeSlotValueListData function element description

Enumeration **PowerTimeSlotValueTypeEnumType**:

Value	Description
power	Power
powerMin	Lower bound of power; minimum
powerMax	Upper bound of power; maximum
powerExpectedValue	Expected value (or "expectation" in terms of probability theory) of power; may be used to express the estimation of an average value.
powerStandardDeviation	Standard deviation (in terms of probability theory) of power
powerSkewness	Skewness (in terms of probability theory) of power
energy	Energy
energyMin	Lower bound of energy; minimum
energyMax	Upper bound of energy; maximum
energyExpectedValue	Expected value (or "expectation" in terms of probability theory) of energy; may be used to express the estimation of an average value.
energyStandardDeviation	Standard deviation (in terms of probability theory) of energy
energySkewness	Skewness (in terms of probability theory) of energy

Table 263: Enumeration PowerTimeSlotValueTypeEnumType

5.3.19.4.3 Available selectors

- sequenceId
- slotNumber
- valueType

5.3.19.4.4 Examples

5.3.19.4.4.1 Read-reply

We continue with the example of section 5.3.19.3.4.1. The energy management gateway now queries for all values of sequence 2 and receives a proper reply from the fridge:

```

<powerTimeSlotValueListData>
  <powerTimeSlotValueData>
    <sequenceId>2</sequenceId>
    <slotNumber>1</slotNumber>
    <valueType>power</valueType>
    <value>
      <number>3</number>
      <scale>2</scale>
    </value>
  </powerTimeSlotValueData>
  <powerTimeSlotValueData>
    <sequenceId>2</sequenceId>
    <slotNumber>1</slotNumber>
    <valueType>powerMax</valueType>
    <value>
      <number>315</number>
    </value>
  </powerTimeSlotValueData>

```



```

8888     <powerTimeSlotValueData>
8889         <sequenceId>2</sequenceId>
8890         <slotNumber>1</slotNumber>
8891         <valueType>energyExpectedValue</valueType>
8892         <value>
8893             <number>5</number>
8894         </value>
8895     </powerTimeSlotValueData>
8896     <powerTimeSlotValueData>
8897         <sequenceId>2</sequenceId>
8898         <slotNumber>2</slotNumber>
8899         <valueType>power</valueType>
8900         <value>
8901             <number>12</number>
8902             <scale>1</scale>
8903         </value>
8904     </powerTimeSlotValueData>
8905     <powerTimeSlotValueData>
8906         <sequenceId>2</sequenceId>
8907         <slotNumber>2</slotNumber>
8908         <valueType>powerMax</valueType>
8909         <value>
8910             <number>14</number>
8911             <scale>1</scale>
8912         </value>
8913     </powerTimeSlotValueData>
8914     <powerTimeSlotValueData>
8915         <sequenceId>2</sequenceId>
8916         <slotNumber>2</slotNumber>
8917         <valueType>energyExpectedValue</valueType>
8918         <value>
8919             <number>2</number>
8920         </value>
8921     </powerTimeSlotValueData>
8922 </powerTimeSlotValueListData>

```

8923

8924 **5.3.19.5 powerTimeSlotScheduleConstraintsListData**

8925 *5.3.19.5.1 General*

8926 This function can be used to model constraints of a slot's schedule. It is useful in case a device
8927 supports the configuration or modification of a schedule. In this case the constraints can be used to
8928 determine the conditions a configuration needs to fulfil.

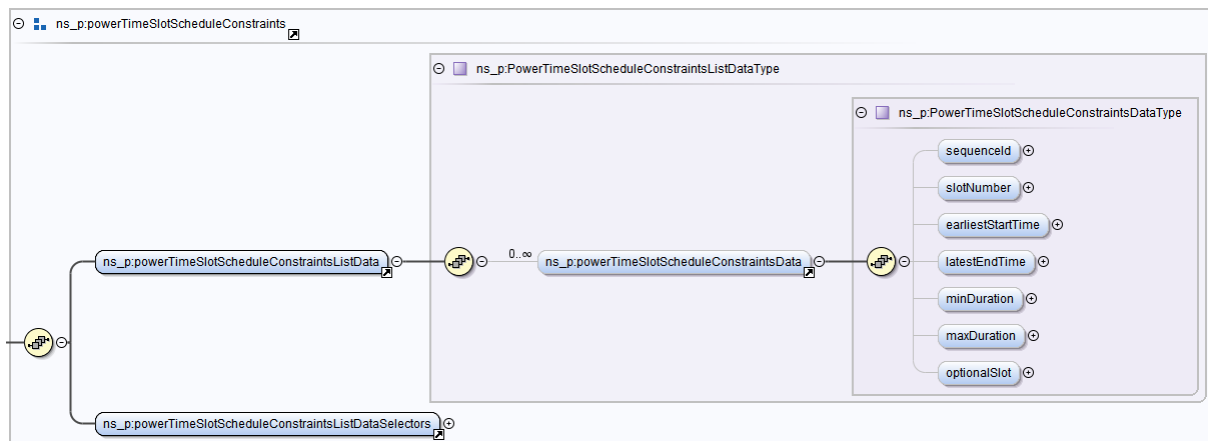


Figure 153: powerTimeSlotScheduleConstraintsListData function overview

5.3.19.5.2 Detailed description of elements

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	The sequence identifier.
slotNumber	Identifier "PowerTimeSlotNumberType" (see Table 339).	The number of the slot.
earliestStartTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The earliest start time of the slot (UTC or relative to "now").
latestEndTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The end time of the slot has to be not later than this time (UTC or relative to "now").
minDuration	xs:duration (W3C standard type)	This is the minimum duration the slot can run.
maxDuration	xs:duration (W3C standard type)	This is the maximum duration the slot has to run.
optionalSlot	xs:boolean (W3C standard type)	A slot is optional if this element is set to "true". See also element "slotActivated" of function "powerTimeSlotScheduleListData" (see Table 261) for the state of the slot (activated or deactivated).

Table 264: powerTimeSlotScheduleConstraintsListData function element description

5.3.19.5.3 Available selectors

- sequenceId
- slotNumber

8938

8939 **5.3.19.5.4 Examples**8940 **5.3.19.5.4.1 Read-reply**

8941 We continue with the examples of section 5.3.19.3.4.1 and section 5.3.19.4.4. The energy
 8942 management gateway wants to figure out whether some slots of sequence 2 can be shifted
 8943 somehow. It sends a request for powerTimeSlotScheduleConstraintsListData of sequence 2 and
 8944 receives a proper reply from the fridge:

```

8945 <powerTimeSlotScheduleConstraintsListData>
8946   <powerTimeSlotScheduleConstraintsData>
8947     <sequenceId>2</sequenceId>
8948     <slotNumber>1</slotNumber>
8949     <earliestStartTime>2006-05-04T16:15:00.0Z</earliestStartTime>
8950     <latestEndTime>2006-05-04T18:00:00.0Z</latestEndTime>
8951     <minDuration>PT1H10M</minDuration>
8952     <maxDuration>PT1H30M</maxDuration>
8953     <optionalSlot>false</optionalSlot>
8954   </powerTimeSlotScheduleConstraintsData>
8955   <powerTimeSlotScheduleConstraintsData>
8956     <sequenceId>2</sequenceId>
8957     <slotNumber>2</slotNumber>
8958     <earliestStartTime>2006-05-04T19:00:00.0Z</earliestStartTime>
8959     <latestEndTime>2006-05-04T19:45:00.0Z</latestEndTime>
8960     <minDuration>PT15M</minDuration>
8961     <maxDuration>PT20M</maxDuration>
8962     <optionalSlot>false</optionalSlot>
8963   </powerTimeSlotScheduleConstraintsData>
8964 </powerTimeSlotScheduleConstraintsListData>

```

8965

8966 **5.3.19.6 Sub-class PowerSequence**

8967 The PowerSequence sub-class delivers all information about the power sequences themselves. This
 8968 covers descriptions, states, constraints for scheduling, scheduling and price.

8969

8970 **5.3.19.7 powerSequenceAlternativesRelationListData**8971 **5.3.19.7.1 General**

8972 As already described in section 5.3.19.1.3 for some use cases it is necessary to consider power
 8973 sequences representing alternative tasks, i.e. tasks where at maximum one can finally be chosen. In
 8974 order to model such a group of alternative sequences the powerSequenceAlternativesRelation can
 8975 be used. It permits to model the relation

- 8976 • 1 alternatives group has 0..m sequences

8977 though practical use cases should have at least one sequence in an alternatives group.

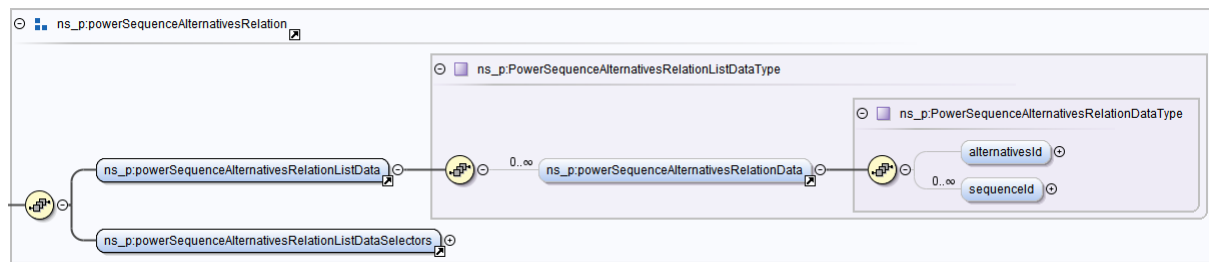


Figure 154: powerSequenceAlternativesRelationListData function overview

5.3.19.7.2 Detailed description of elements

Element	Type	Description
alternativesId	Identifier "AlternativesIdType" (see Table 339).	The identifier for the group of alternative sequences.
sequenceId (list)	Identifier "PowerSequenceIdType" (see Table 339).	A sequence identifier that belongs to the given alternatives group.

Table 265: powerSequenceAlternativesRelationListData function element description

5.3.19.7.3 Available selectors

- alternativesId
- sequenceId

5.3.19.7.4 Examples

5.3.19.7.4.1 Notify

A heating system announces some proposals for the afternoon and the evening: For the afternoon group 1 contains sequences 1 and 2 to choose from and for the evening group 2 permits to select sequence 3, 4, or 5. It sends this (complete) notification:

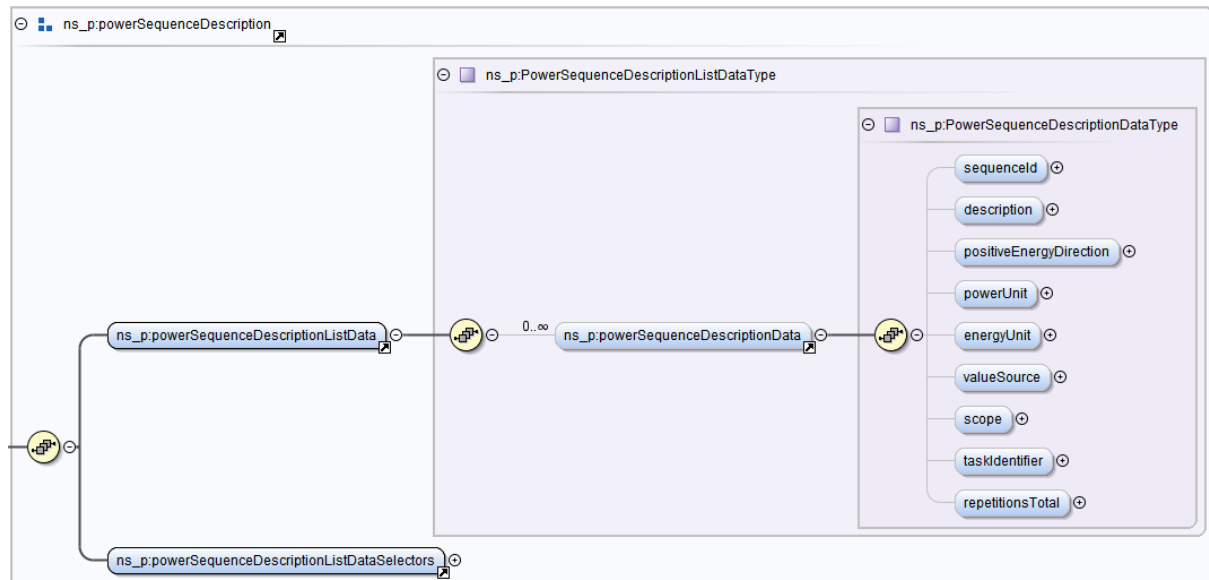
```

<powerSequenceAlternativesRelationListData>
  <powerSequenceAlternativesRelationData>
    <alternativesId>1</alternativesId>
    <sequenceId>1</sequenceId>
    <sequenceId>2</sequenceId>
  </powerSequenceAlternativesRelationData>
  <powerSequenceAlternativesRelationData>
    <alternativesId>2</alternativesId>
    <sequenceId>3</sequenceId>
    <sequenceId>4</sequenceId>
    <sequenceId>5</sequenceId>
  </powerSequenceAlternativesRelationData>
</powerSequenceAlternativesRelationListData>

```

9007 **5.3.19.8 powerSequenceDescriptionListData**9008 **5.3.19.8.1 General**

9009 The *powerSequenceDescriptionData* function models descriptive (typically non-changing) information
 9010 on sequences (one for each available sequence of a node).



9011
 9012 Figure 155: *powerSequenceDescriptionListData* function overview

9013

9014 **5.3.19.8.2 Detailed description of elements**

Element	Type	Description
sequenceld	Identifier "PowerSequenceldType" (see Table 339).	The sequence identifier.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on this sequence.
positiveEnergyDirection	Common data type "EnergyDirectionType". See section 3.10.1.13.	The <i>positiveEnergyDirection</i> states whether energy consumption or production will be counted as positive value: If the element is set to "consume": energy values are positive during consumption and negative during production of energy. If the element is set to "produce": energy values are negative during consumption and positive during production of energy.
powerUnit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	This is the unit for all values related to power.
energyUnit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	This is the unit for all values concerning energy consumption and production.
valueSource	Union "MeasurementValueSourceType":	This element enables to express whether the values of the given sequence are based upon measurement or calculation, e.g.

	<ul style="list-style-type: none"> - Enum (see Table 232): MeasurementValueSourceEnumType - EnumExtendType (see section 3.10.1.5) 	
scope	<i>Union</i> "PowerSequenceScopeType": <ul style="list-style-type: none"> - Enum (see Table 267): PowerSequenceScopeEnumType - EnumExtendType (see section 3.10.1.5) 	With this element it is possible to distinguish between sequences intended for different scopes (forecast, measurement, etc.). Note: Sequences with scope "forecast" or "measurement" typically originate from a device executing a task, whereas sequences with scope "recommendation" rather denote a configuration into a device that shall execute a task.
taskIdentifier	xs:unsignedInt (W3C standard type)	This "technical identifier" denotes which kind of task or condition applies to the power sequence. A device can provide identifiers in order to enable energy management applications recognize common (i.e. "typical" or frequent) tasks of a device. Note: This element should not be confused with "taskId" of the class TaskManagement.
repetitionsTotal	xs:unsignedInt (W3C standard type)	If the sequence applies the "repetition concept", this element denotes the total number of executions of the sequence. See also element "activeRepetitionNumber" of function "powerSequenceStateData".

9015 Table 266: powerSequenceDescriptionListData function element description

9016 Enumeration **PowerSequenceScopeEnumType**:

Value	Description
forecast	The sequence's values reflect the best guess of a device's task which typically begins in the future.
measurement	The sequence's values reflect the measurement of a device's task.
recommendation	The value "recommendation" can be used to model a "target power curve", i.e. a power profile a device SHOULD follow for a given task (if possible).

9017 Table 267: Enumeration PowerSequenceScopeEnumType

9018

9019 5.3.19.8.3 Available selectors

9020 - sequenceId

9021

5.3.19.8.4 Examples

5.3.19.8.4.1 Read-reply

One is interested in the description of all available power sequences. A query for a device's `powerSequenceDescriptionListData` may lead to the following reply:

```
<powerSequenceDescriptionListData>
  <powerSequenceDescriptionData>
    <sequenceId>1</sequenceId>
    <description>Washing programme 1</description>
    <positiveEnergyDirection>consume</positiveEnergyDirection>
    <powerUnit>W</powerUnit>
    <energyUnit>Wh</energyUnit>
    <scope>forecast</scope>
    <taskIdentifier>7</taskIdentifier>
  </powerSequenceDescriptionData>
  <powerSequenceDescriptionData>
    <sequenceId>2</sequenceId>
    <description>Washing programme 2</description>
    <positiveEnergyDirection>consume</positiveEnergyDirection>
    <powerUnit>W</powerUnit>
    <energyUnit>Wh</energyUnit>
    <scope>forecast</scope>
    <taskIdentifier>8</taskIdentifier>
  </powerSequenceDescriptionData>
</powerSequenceDescriptionListData>
```

5.3.19.9 powerSequenceStateListData

5.3.19.9.1 General

Information on the current state of a power sequence can be modelled with the `powerSequenceStateData` function.

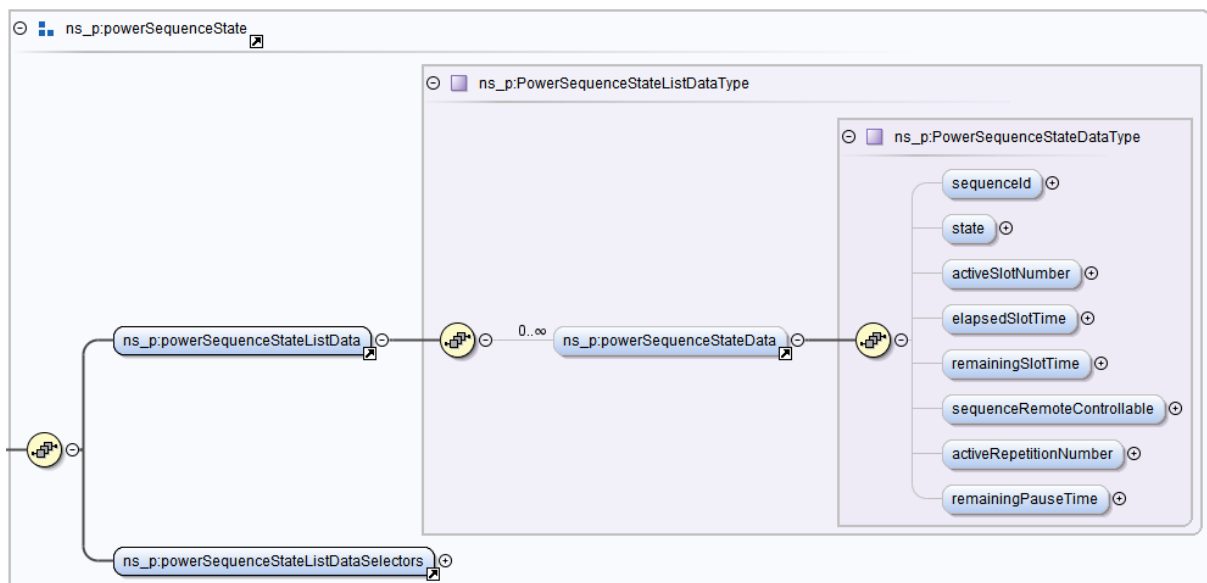


Figure 156: `powerSequenceStateListData` function overview

9054 5.3.19.9.2 Detailed description of elements

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	The sequence identifier.
state	Union "PowerSequenceStateTy pe": - Enum (see Table 269): PowerSequenceStateEnu mType - EnumExtendType (see section 3.10.1.5)	The current state of the power sequence.
activeSlotNumber	Identifier "PowerTimeSlotNumberT ype" (see Table 339).	Contains the currently active slot.
elapsedSlotTime	xs:duration (W3C standard type)	Contains the time the slot has already been in "running" state (this also means the value remains constant during a "paused" state).
remainingSlotTime	xs:duration (W3C standard type)	Contains the time the slot still needs to be in "running" state (this also means the value remains constant during a "paused" state).
sequenceRemoteCo ntrollable	xs:boolean (W3C standard type)	If this element is set to "false" the sequence is executed autonomously and cannot be modified by SPINE commands received external nodes. Note: Consider also the element "nodeRemoteControllable" of function "powerSequenceNodeScheduleInformationDat a" in Table 274!
activeRepetitionNu mber	xs:unsignedInt (W3C standard type)	If the sequence applies the "repetition concept", this element denotes the current number of the sequence execution.
remainingPauseTim e	xs:duration (W3C standard type)	Denotes the duration the current slot (element "activeSlotNumber") permits being paused.

9055 Table 268: powerSequenceStateListData function element description

9056 Enumeration **PowerSequenceStateEnumType**:

Value	Description
running	The sequence is currently executed.
paused	The sequence is currently paused.
scheduled	The sequence is not executed at the moment but will be executed (i.e. gain state "running") by the power sequences server at the scheduled time.
scheduledPaused	Similar to the value "scheduled", but at the scheduled time the sequence will be set into the state "paused".
pending	The sequence is "recommended" (in terms of "preferred") by the power sequences server but will not be executed autonomously by the server. Remark: This state is most appropriate for devices announcing that a specific task (power sequence) is going to become "valid" but still lacks something / requires configuration or advice. It is most suitable in conjunction with the function powerSequenceScheduleConfigurationRequestCall.

inactive	The sequence denotes an “optional task”. I.e. it is not invalid but it is also not running/scheduled/... so far. However, this sequence may become running or scheduled(Paused) by appropriate means.
completed	The sequence has been completed.
invalid	The sequence is invalid and shall not be considered anymore.

Table 269: Enumeration *PowerSequenceStateEnumType*

5.3.19.9.3 Available selectors

- sequenceId

5.3.19.9.4 Examples

5.3.19.9.4.1 Notify

A power sequences server executes sequence “2” and just completed slot number 1. Now, it begins with slot number 2 and notifies this change (the “filter/partial” part to announce the “restricted function exchange is simplified by “...” just to improve the readability”):

```

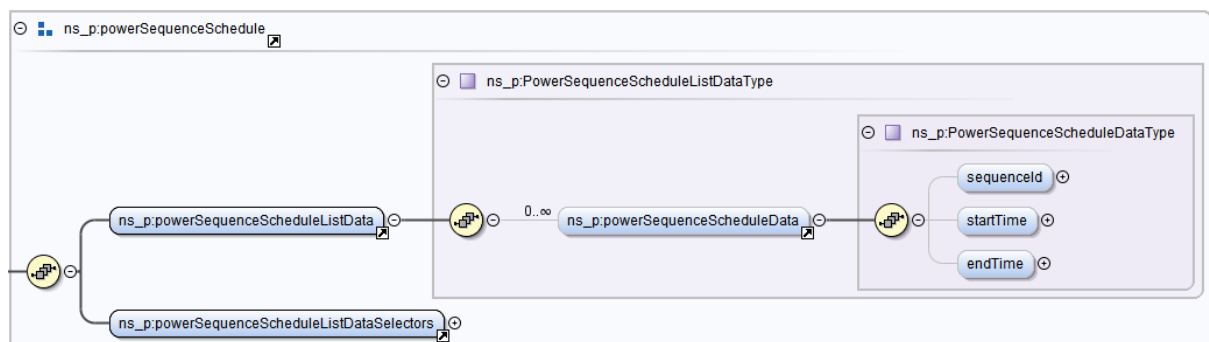
...
<powerSequenceStateListData>
  <powerSequenceStateData>
    <sequenceId>2</sequenceId>
    <activeSlotNumber>2</activeSlotNumber>
    <elapsedSlotTime>PT0M</elapsedSlotTime>
    <remainingSlotTime>PT20M</remainingSlotTime>
  </powerSequenceStateData>
</powerSequenceStateListData>

```

5.3.19.10 powerSequenceScheduleListData

5.3.19.10.1 General

The scheduling of a sequence can be modelled with this function.

Figure 157: *powerSequenceScheduleListData* function overview

5.3.19.10.2 Detailed description of elements

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	The sequence identifier.

startTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The start time of the sequence.
endTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The end time of the sequence.

Table 270: powerSequenceScheduleListData function element description

5.3.19.10.3 Available selectors

- sequenceId

5.3.19.10.4 Examples

5.3.19.10.4.1 Read-reply

An Energy Management Software wants to get information on the current schedule of a specific sequence (2). It sends a proper request and receives this response:

```
<powerSequenceScheduleListData>
  <powerSequenceScheduleData>
    <sequenceId>2</sequenceId>
    <startTime>2013-07-13T18:11:00.0Z</startTime>
    <endTime>2013-07-14T16:55:00.0Z</endTime>
  </powerSequenceScheduleData>
</powerSequenceScheduleListData>
```

5.3.19.11 powerSequenceScheduleConstraintsListData

5.3.19.11.1 General

A power sequence of an appliance may have time-related constraints that are to be considered by an energy management software during demand optimization calculations. Otherwise the appliance could reject a schedule change requested by the energy management software.

This section deals with overall schedule constraints of a sequence. Schedule constraints of slots are discussed in section 5.3.19.5. An application may make use of both kinds of constraints together. Of course, contradictory information from constraints and schedules should be avoided.

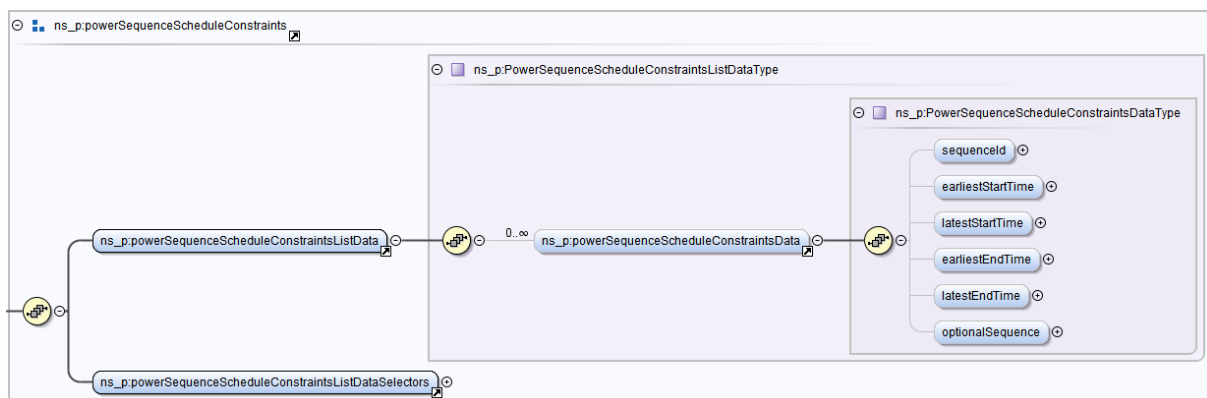


Figure 158: powerSequenceScheduleConstraintsListData function overview

9112 *5.3.19.11.2 Detailed description of elements*

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	The sequence identifier.
earliestStartTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The earliest time the sequence can start.
latestStartTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The latest time the sequence can start.
earliestEndTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The earliest time the sequence can end.
latestEndTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The latest time the sequence can end.
optionalSequence	xs:boolean (W3C standard type)	For specific use cases a sequence can be tagged explicitly as "optional task". In this case this element can be set to <i>true</i> .

9113 *Table 271: powerSequenceScheduleConstraintsListData function element description*

9114 Please note that start and end time constraints just model start and end boundaries of a power
9115 sequence. I.e. in general the duration of a sequence can be shorter than the period given by the
9116 start/end time constraints.

9117

9118 *5.3.19.11.3 Available selectors*

9119 - sequenceId

9120

9121 *5.3.19.11.4 Examples*9122 *5.3.19.11.4.1 Read-reply*

9123 Before an Energy Management Software re-schedules a specific sequence (2) of an appliance it
9124 queries the appliance for the proper constraints. The appliance sends this response:

```

9125 <powerSequenceScheduleConstraintsListData>
9126   <powerSequenceScheduleConstraintsData>
9127     <sequenceId>2</sequenceId>
9128     <earliestStartTime>2013-07-14T12:00:00.0Z</earliestStartTime>
9129     <latestStartTime>2013-07-14T12:45:00.0Z</latestStartTime>
9130     <earliestEndTime>2013-07-14T16:00:00.0Z</earliestEndTime>
9131     <latestEndTime>2013-07-14T18:00:00.0Z</latestEndTime>
9132   </powerSequenceScheduleConstraintsData>
9133 </powerSequenceScheduleConstraintsListData>

```

9134

5.3.19.12 powerSequencePriceListData

5.3.19.12.1 General

The execution of a power sequence may cause some cost or earnings due to the consumption or production of energy. This means a power sequence can be assigned a price. In fact, as the availability of energy may vary with time, the price of a power sequence could as well depend upon its time of execution (which may esp. be of interest if the scheduling of power sequences shall consider cost saving requirements). Therefore, the *powerSequencePriceData* function permits modelling a price per assumed start time of the sequence. Please note assumed start times may differ from the "final" start time.

How an appliance gains knowledge of the price associated to one of its sequences shall not be discussed in detail. Just a few possibilities shall be sketched briefly: Configuration (the sequence has a fixed price which is stored in the appliance), local calculation (the appliance obtains price information and applies it to the power consumption), or query to an external node (see *powerSequencePriceCalculationRequestCall* function in section 5.3.19.18).

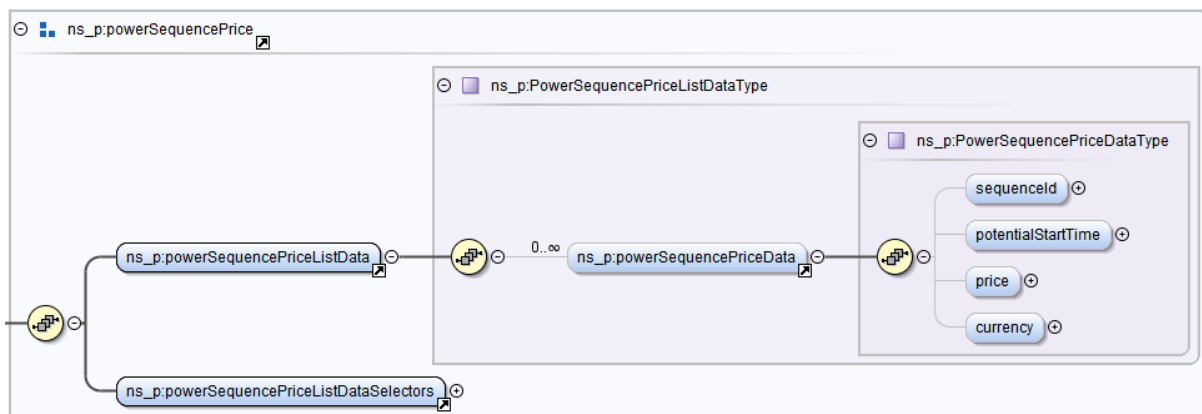


Figure 159: powerSequencePriceListData function overview

5.3.19.12.2 Detailed description of elements

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	The sequence identifier.
potentialStartTime	Common data type "AbsoluteOrRelativeTime Type". See section 3.10.2.3.	The price information belongs to the start time given with this element. Please note this can differ from the final start time of a sequence (in case the sequence has been shifted in time but the price has not been updated accordingly, e.g.). Of course, in this case the price denoted in this instance can differ from the "real" price.
price	Common data type "ScaledNumberType". See section 3.10.1.8.	The price of the sequence for the given potential start time.
currency	Common data type "CurrencyType". See section 3.10.1.19.	The currency of the price.

Table 272: powerSequencePriceListData function element description

5.3.19.12.3 Available selectors

- sequenceId
- potentialStartTimeInterval

5.3.19.12.4 Examples

5.3.19.12.4.1 Write

An energy management system got information about power sequence 2 of an appliance. The energy management system has also knowledge of the energy price and can calculate a proper cost. It submits the calculated price for a potential start time to the appliance and the appliance can then display the cost to the user:

```
<powerSequencePriceListData>
  <powerSequencePriceData>
    <sequenceId>2</sequenceId>
    <potentialStartTime>2013-07-14T12:35:00.0Z</potentialStartTime>
    <price>
      <number>95</number>
      <scale>-2</scale>
    </price>
    <currency>EUR</currency>
  </powerSequencePriceData>
</powerSequencePriceListData>
```

5.3.19.13 powerSequenceSchedulePreferenceListData

5.3.19.13.1 General

A “smart energy management” typically includes the consideration of a user’s preference for the kind of energy consumption optimisation. Some may prefer the optimisation to reduce the cost whereas others may prefer first of all the use of “green energy”. This kind of preference can be modelled using *powerSequenceSchedulePreference*, where the preference can be assigned per sequence.

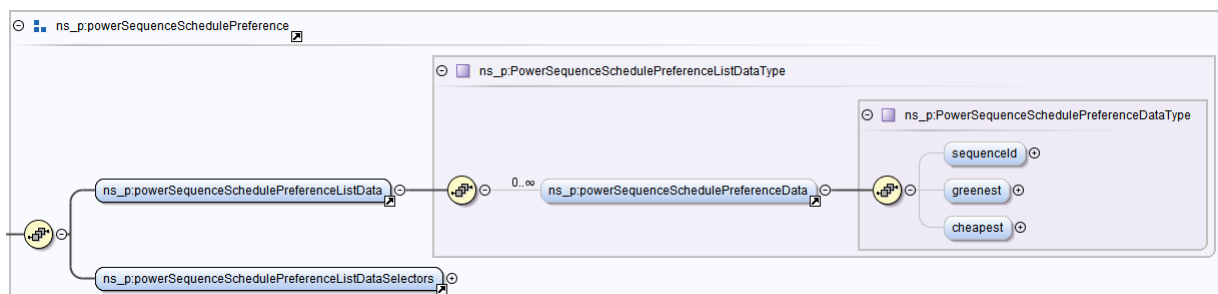


Figure 160: powerSequenceSchedulePreferenceListData function overview

5.3.19.13.2 Detailed description of elements

Element	Type	Description
---------	------	-------------

sequenceId	<i>Identifier "PowerSequenceIdType" (see Table 339).</i>	The sequence identifier.
greenest	xs:boolean (W3C standard type)	For this sequence an optimisation towards the minimum usage of NON-renewable energy is preferred.
cheapest	xs:boolean (W3C standard type)	For this sequence an optimisation that minimises the user's energy bill is preferred.

Table 273: powerSequenceSchedulePreferenceListData function element description

5.3.19.13.3 Available selectors

- sequenceId

5.3.19.13.4 Examples

5.3.19.13.4.1 Read-reply

A "smart washing machine" is connected with an energy management system. At the washing machine a user can press a button to enable the preference for renewable energy. Once the washing machine announced an upcoming task, the energy management system queries for the user's preference and receives this response:

```
<powerSequenceSchedulePreferenceListData>
  <powerSequenceSchedulePreferenceData>
    <sequenceId>3</sequenceId>
    <greenest>true</greenest>
  </powerSequenceSchedulePreferenceData>
</powerSequenceSchedulePreferenceListData>
```

5.3.19.14 Sub-class PowerSequenceNode

This sub-class includes PowerSequences information affecting the whole node rather than a specific sequence.

5.3.19.15 powerSequenceNodeScheduleInformationData

5.3.19.15.1 General

The *powerSequenceNodeScheduleInformationData* function permits modelling of some schedule information that applies to a whole node rather than to a specific sequence. This information can change during runtime (especially the *nodeRemoteControllable* element).

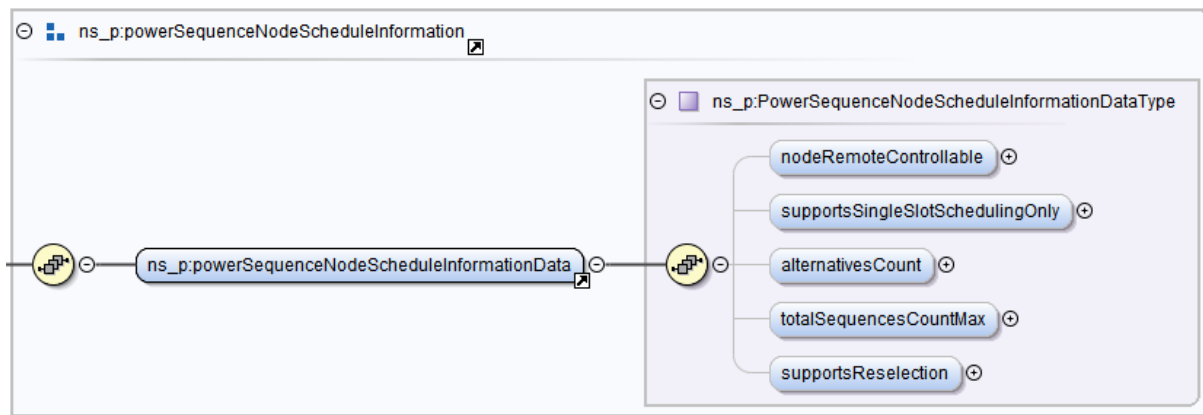


Figure 161: powerSequenceNodeScheduleInformationData function overview

5.3.19.15.2 Detailed description of elements

Element	Type	Description
nodeRemoteControl lable	xs:boolean (W3C standard type)	This element denotes whether a node permits being scheduled (value is “true”) by a remote device or not (value is “false”).
supportsSingleSlotS chedulingOnly	xs:boolean (W3C standard type)	If set to “true” the appliance does NOT permit the modification of more than one slot per configuration command: The sub-class PowerTimeSlot provides functions that can be used to configure slots of an appliance. Multiple slots may be configured with a single command (single function instance). However, some kind of appliances may support the scheduling of only a single slot per sequence as configuration request. This constraint can be modelled setting this element to <i>true</i> . In this case for each slot a separate configuration request (write command with the function properly set) for scheduling should be used.
alternativesCount	xs: unsignedInt (W3C standard type)	The total number of “alternatives groups” (see sections 5.3.19.1.3, 5.3.19.7). If powerSequenceAlternativesRelationListData is used, the element alternativesCount denotes the list size.
totalSequencesCou ntMax	xs:unsignedInt (W3C standard type)	The maximum total number of sequences the device may offer at any given moment. This information is primarily intended for energy management systems to get a notion of an appliance’s schedule complexity. If an appliance counts all of its currently known sequences (i.e. across all alternatives groups and also regardless of the sequences’ states) the result will not exceed totalSequencesCountMax.
supportsReselection	xs:boolean (W3C standard type)	As mentioned in section 5.3.19.1.3 an appliance may permit an energy management system to select one of its sequences from an alternatives group for execution. Some appliances may permit such a selection only one time per alternatives group. This

		can be expressed with “supportsReselection” setting to “false”. If an appliance grants the energy management system to alter the selection (i.e. re-select) of the alternatives group, this can be expressed with the value “true”.
--	--	---

Table 274: *powerSequenceNodeScheduleInformationData* function element description

9218	
9219	
9220	5.3.19.15.3 Available selectors
9221	None.
9222	
9223	5.3.19.15.4 Examples
9224	5.3.19.15.4.1 Notify
9225	An Energy Management Software wants to get an overview on the current state of a washing
9226	machine, esp. in order to see whether it can be scheduled. It sends a requests and gets this response:
9227	<code><powerSequenceNodeScheduleInformationData></code>
9228	<code> <nodeRemoteControllable>true</nodeRemoteControllable></code>
9229	
9230	<code><supportsSingleSlotSchedulingOnly>true</supportsSingleSlotSchedulingOnly></code>
9231	<code> <alternativesCount>1</alternativesCount></code>
9232	<code> <totalSequencesCountMax>3</totalSequencesCountMax></code>
9233	<code> <supportsReselection>false</supportsReselection></code>
9234	<code></powerSequenceNodeScheduleInformationData></code>
9235	
9236	5.3.19.16 PowerSequence calls
9237	Subsequently, the calls available in the PowerSequence Class are described.
9238	
9239	5.3.19.17 powerSequenceScheduleConfigurationRequestCall
9240	5.3.19.17.1 General
9241	Some functions of the sub-class PowerSequence can be used to model notification of new or
9242	changed sequences. Energy management gateways may listen to such notifications and take action
9243	on demand. However, the nature of notifications does not imply any mandatory action upon receipt.
9244	With the function <i>powerSequenceScheduleConfigurationRequestCall</i> the owner of a sequence (i.e. a
9245	server with regards to the PowerSequences class) can model an explicit request stating a sequence
9246	shall be configured. I.e. an Energy Management Software (i.e. a client with regards to the
9247	PowerSequences class) receiving this function would know it needs to take proper action (provided it
9248	is capable of such action, i.e. performing write operations with sequence configuration).

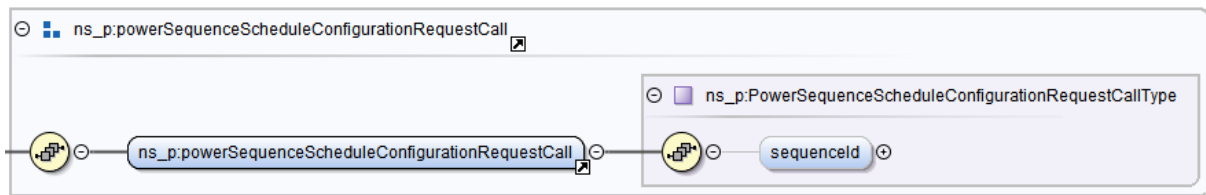


Figure 162: powerSequenceScheduleConfigurationRequestCall function overview

5.3.19.17.2 Detailed description of elements

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	The sequence identifier the owner of the power sequence wants to be configured.

Table 275: powerSequenceScheduleConfigurationRequestCall function element description

5.3.19.17.3 Available selectors

None.

5.3.19.17.4 Examples

5.3.19.17.4.1 Call

If a node wants one of its power sequences to be configured, it could send the following XML with a *call* classifier to an appropriate Energy Management Software:

```
<powerSequenceScheduleConfigurationRequestCall>
  <sequenceId>2</sequenceId>
</powerSequenceScheduleConfigurationRequestCall>
```

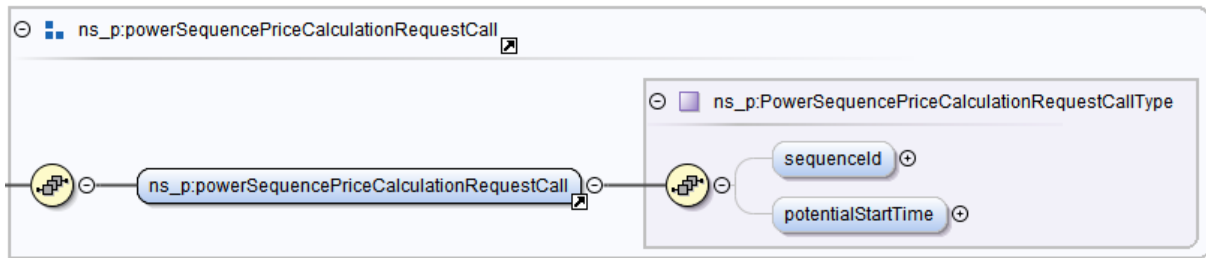
5.3.19.18 powerSequencePriceCalculationRequestCall

5.3.19.18.1 General

The owner of a power sequence can use this function in order to trigger the determination of a price for one of its sequences at an external node. This shall be explained in more detail now.

Section 5.3.19.12.1 describes how the owner of a sequence can express a price for a given sequence and start time. Before the owner is able to tell a price upon request it needs to get knowledge of the price by itself. The section also briefly mentions different possibilities to achieve this. One of these possibilities is the use of the *powerSequencePriceCalculationRequestCall* function: It requires a sequence is available that expresses its energy or power demand. It also requires the owner of the sequence knows of an external node that has the capability to determine a price for the owner's sequence. Then, the owner of the sequence can submit a *powerSequencePriceCalculationResponseCall* function to this external node in order to trigger the determination of the price. This function can take the *sequenceId* and also a potential start time.

9279 Finally, the external node may use the function *powerSequencePriceData* (see section 5.3.19.12) in
 9280 order to return the calculated price to the owner of the sequence.



9281
 9282 Figure 163: *powerSequencePriceCalculationRequestCall* function overview

9283
 9284 5.3.19.18.2 Detailed description of elements

Element	Type	Description
sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	The sequence identifier the owner of the power sequence wants to get a price for.
potentialStartTime	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The price shall be determined for a potential start time given in this element.

9285 Table 276: *powerSequencePriceCalculationRequestCall* function element description

9286
 9287 5.3.19.18.3 Available selectors
 9288 None.

9289
 9290 5.3.19.18.4 Examples

9291 5.3.19.18.4.1 Call

9292 The scenario assumes an external node (an energy management system, e.g.) that is capable of
 9293 calculating prices based upon a given energy demand and schedule and also has knowledge of the
 9294 tariff that is relevant for the required energy. A washing machine prepared a power sequence and
 9295 wants to get a price for this sequence with a potential start time. It sends this XML with a *call*
 9296 classifier to the energy management system:

```

9297 <powerSequencePriceCalculationRequestCall>
9298   <sequenceId>2</sequenceId>
9299   <potentialStartTime>2013-07-14T12:35:00.0Z</potentialStartTime>
9300 </powerSequencePriceCalculationRequestCall>
  
```

9301

5.3.20 Sensing

5.3.20.1 Introduction

The *Sensing* class is used for modelling data obtained by sensors in the meaning of more state-based sensors like contact sensors, switches and buttons, etc., in contrast to value-based sensors like temperature or humidity sensors that are treated with the *Measurement* class.

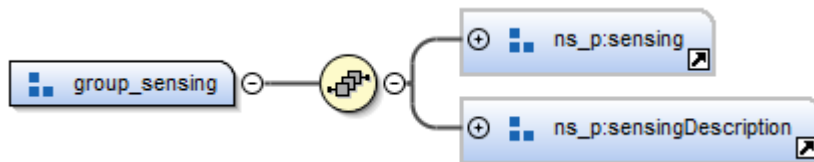


Figure 164: Sensing function-group overview

5.3.20.2 sensingListData

5.3.20.2.1 General

The list function of sensing can be used to model sensor information over time. This may be of interest for a motion detector (when someone was at the backdoor), a contact sensor (when was the window open in the last 24 hours), etc.

Some sensors may only store the latest value and therefore only support one entry in the list. To enable users and applications to query past changes, a communications technology interface may cache states/values of simple sensors that only support the single data function. That is no mandatory feature, however.

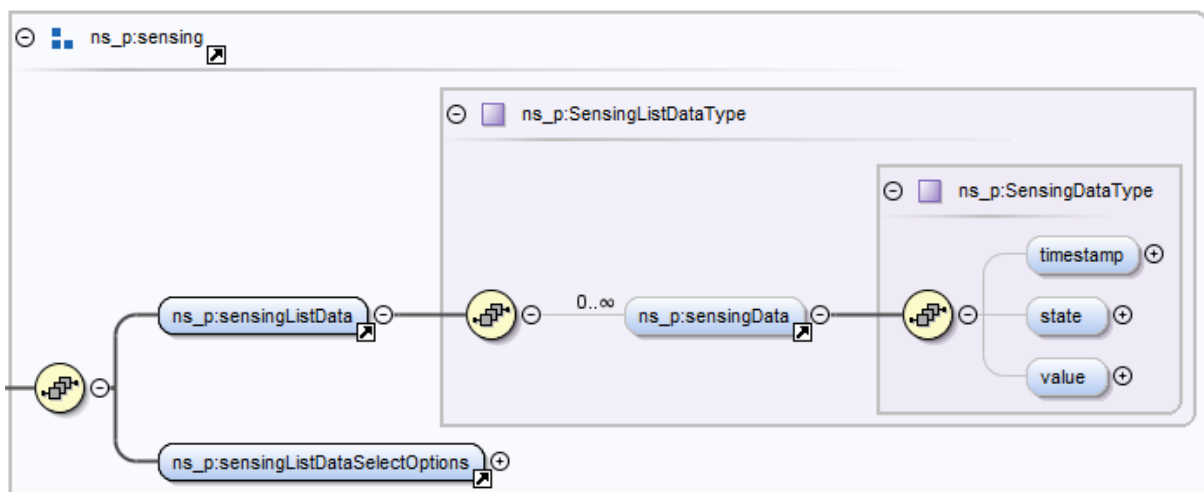


Figure 165: sensingListData function overview

5.3.20.2.2 Detailed description of elements

Element	Type	Description
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The creation time of the <i>sensingData</i> instance.

state	<i>Union "SensingStateType":</i> - Enum (see Table 278): SensingStateEnumType - EnumExtendType (see section 3.10.1.5)	The state of the sensor is denoted in this element.
value	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Some sensors deliver a value (e.g. a <i>levelSwitch</i>).

9323 Table 277: sensingListData function detailed description of elements

9324 Enumeration **SensingStateEnumType**:

Value	Description
on	"on" was detected, meaning that the according functionality was switched on.
off	"off" was detected, meaning that the according functionality was switched off.
toggle	"toggle" was detected, meaning that the according functionality was toggled (e.g. from "on" to "off" or vice versa).
level	"level" was detected, meaning that the according functionality changed its level (e.g. from 2.4 to 8.2) that can be found in the Element "value".
levelUp	"levelUp" was detected, meaning that the according functionality changed its level upwards.
levelDown	"levelDown" was detected, meaning that the according functionality changed its level downwards.
levelStart	"levelStart" was detected, meaning that the according functionality started to change its level.
levelStop	"levelStop" was detected, meaning that the according functionality stopped to change its level.
levelAbsolute	"levelAbsolute" was detected, meaning that the according functionality changed its level to an absolute value stated in the Element "value" (e.g. 1190).
levelRelative	"levelRelative" was detected, meaning that the according functionality moved its level in comparison to the old one. The relative change can be found in the Element "value" (e.g. a change from 22 to 33 would result in a content of the Element "value" of 11).
levelPercentageAbsolute	"levelPercentageAbsolute" was detected, meaning that the according functionality changed its level to an absolute percentage value stated in the Element "value" (e.g. 90%).
levelPercentageRelative	"levelPercentageRelative" was detected, meaning that the according functionality changed its percentage level relative to the old one. The relative change can be found in the Element "value" (e.g. a change from 40% to 50% would result in a content of the Element "value" of 10).
pressed	"pressed" was detected, meaning that the according functionality was pressed (e.g. a button, that was pressed).
longPressed	"longPressed" was detected, meaning that the according functionality was pressed longer (e.g. a button, that was pressed longer). The timing difference between "pressed" and "longPressed" is up to the vendor.
released	"released" was detected, meaning that the according functionality was released (e.g. a button, that was released).
changed	"changed" was detected, meaning that the according functionality changed.

started	"started" was detected, meaning that the according functionality started.
stopped	"stopped" was detected, meaning that the according functionality stopped.
paused	"paused" was detected, meaning that the according functionality paused.
middle	"middle" was detected, meaning that the according functionality (e.g. a scroll bar or a throttle) is in position "middle".
up	"up" was detected, meaning that the according functionality moves up.
down	"down" was detected, meaning that the according functionality moves down.
forward	"forward" was detected, meaning that the according functionality moves forward.
backwards	"backwards" was detected, meaning that the according functionality moves backwards.
open	"open" was detected, meaning that the according functionality is open (e.g. a door).
closed	"closed" was detected, meaning that the according functionality is closed (e.g. a door).
opening	"opening" was detected, meaning that the according functionality is opening (e.g. a door, that is changing from "closed" to "open").
closing	"closing" was detected, meaning that the according functionality is closing (e.g. a door, that is changing from "open" to "closed").
high	"high" was detected, meaning that the according functionality is at a high position or value.
low	"low" was detected, meaning that the according functionality is at a low position or value.
day	"day" was detected, meaning that the according functionality is in day-mode.
night	"night" was detected, meaning that the according functionality is in night-mode.
detected	"detected" was detected, meaning that the according functionality was detected (e.g. a motion sensor detected a motion).
notDetected	"notDetected" was detected, meaning that the according functionality was detected (e.g. a motion sensor reports that no motion is detected).
alarmed	"alarmed" was detected, meaning that the according functionality is in alarm mode (e.g. a smoke detector).
notAlarmed	"notAlarmed" was detected, meaning that the according functionality is not in alarm mode (e.g. a smoke detector).

Table 278: Enumeration SensingStateEnumType

5.3.20.2.3 Available selectors

- timestampInterval

9330 *5.3.20.2.4 Examples*9331 *5.3.20.2.4.1 Notify*

9332 If a contact sensor detects an opening (e.g. of a window), it will send a (complete) notification about
9333 that occurrence:

```
9334 <sensingListData>
9335   <sensingData>
9336     <timestamp>2013-07-15T12:01:02.0Z</timestamp>
9337     <state>opened</state>
9338   </sensingData>
9339 </sensingListData>
```

9340

9341 *5.3.20.2.4.2 Read-reply*

9342 If one is interested in the activities of a switch for the time between 12:00 and 14:00 o'clock, he
9343 could send the following read command:

```
9344 <function>sensingListData</function>
9345 <filter>
9346   <cmdControl>
9347     <partial/>
9348   </cmdControl>
9349   <sensingListDataSelectors>
9350     <timestampInterval>
9351       <startTime>2013-07-15T12:00:00.0Z</startTime>
9352       <endTime>2013-07-15T14:00:00.0Z</endTime>
9353     </timestampInterval>
9354   </sensingListDataSelectors>
9355 </filter>
9356 <sensingListData/>
```

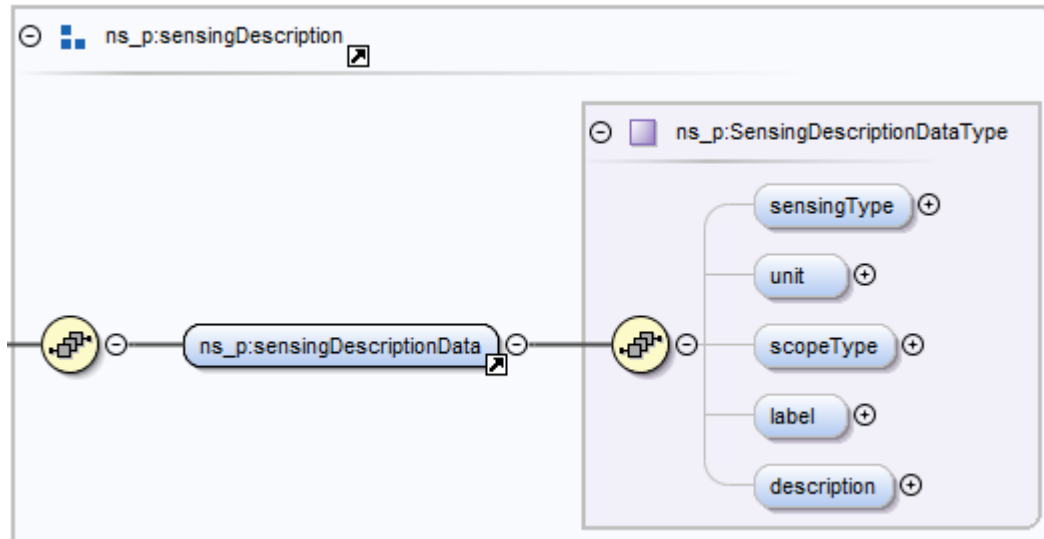
9357 As a reply, something like the following could be returned:

```
9358 <function>sensingListData</function>
9359 <filter>
9360   <cmdControl>
9361     <partial/>
9362   </cmdControl>
9363 </filter>
9364 <sensingListData>
9365   <sensingData>
9366     <timestamp>2013-07-15T12:01:00.0Z</timestamp>
9367     <state>on</state>
9368   </sensingData>
9369   <sensingData>
9370     <timestamp>2013-07-15T12:50:32.0Z</timestamp>
9371     <state>off</state>
9372   </sensingData>
9373   <sensingData>
9374     <timestamp>2013-07-15T13:15:05.0Z</timestamp>
9375     <state>on</state>
9376   </sensingData>
9377 </sensingListData>
```

9378

9379 **5.3.20.3 sensingDescriptionData**9380 **5.3.20.3.1 General**

9381 The rather static (non-changing) elements of the *Sensing* class are combined in the
 9382 *sensingDescriptionData* function. Typically, it is only requested once by a user or an application.



9383
 9384 *Figure 166: sensingDescriptionData function overview*

9385

9386 **5.3.20.3.2 Detailed description of elements**

Element	Type	Description
sensingType	Union "SensingTypeType": - Enum (see Table 280): SensingTypeEnumType - EnumExtendType (see section 3.10.1.5)	Which type of sensor is modelled is denoted with this element.
unit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	The unit in which the values of the sensor are given. Not all sensor types need the value element and therefore the unit may be omitted as well.
scopeType	Common data type "ScopeTypeType". See section 3.10.1.21.	Specifies a more detailed meaning of the sensor (e.g. shutter).
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the sensor.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the sensor.

9387 *Table 279: sensingDescriptionData function detailed description of elements*

9388 Enumeration **SensingTypeEnumType**:

Value	Description
switch	A switch that has usually two states "on" and "off".
button	A button that can be "pressed" (and eventually supports "longPressed" and "released")
level	Some level functionality supporting one or more of the level-related sensing states.
levelSwitch	Some level functionality supporting one or more of the level-related sensing states and a switch functionality ("on" / "off").

windowHandle	A window handle supporting different positions.
contactSensor	Sensor that can detect whether something is "open" or "closed".
occupancySensor	Sensor that can detect whether someone is present in a room ("detected") or not ("notDetected").
motionDetector	Sensor that can detect motion ("detected") and eventually supports the "notDetected" state, too.
fireDetector	A sensor that detects fire ("detected") and eventually the absence of fire ("notDetected").
smokeDetector	A sensor that detects smoke ("detected") and eventually the absence of smoke ("notDetected").
heatDetector	A sensor that detects heat ("detected") and eventually the absence of heat ("notDetected").
waterDetector	A sensor that detects water ("detected") and eventually the absence of water ("notDetected").
gasDetector	A sensor that detects gas ("detected") and eventually the absence of gas ("notDetected").
alarmSensor	A sensor providing alarm information ("alarmed" / "notAlarmed").
powerAlarmSensor	A sensor reporting power alarms ("alarmed" / "notAlarmed").
dayNightIndicator	A sensor detecting whether it is "day" or "night".

Table 280: Enumeration SensingTypeEnumType

5.3.20.3.3 Available selectors

None.

5.3.20.3.4 Examples

5.3.20.3.4.1 Notify

An example instance of *sensingDescriptionData* of a smoke detector, sent as notification:

```

<sensingDescriptionData>
  <sensingType>smokeDetector</sensingType>
  <label>SmokeDetector Bedroom</label>
</sensingDescriptionData>

```

5.3.21 Setpoint

5.3.21.1 Introduction

Setting setpoint values is an often chosen way to control some functionality of a device. Especially in the home automation and HVAC domain, several use cases depend on setpoints. This class handles the setpoint itself. Relations to Measurement and TimeTable class enables the control of a measurand for specific time frames.

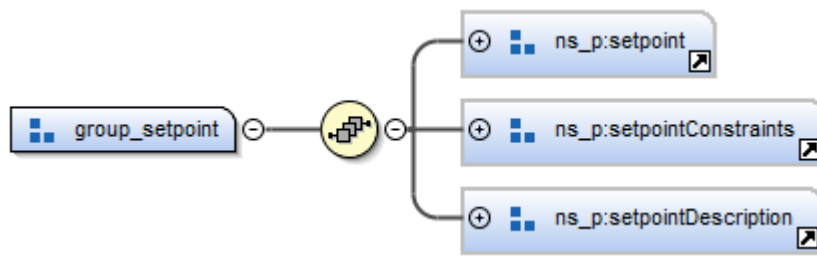


Figure 167: Setpoint function-group overview

5.3.21.2 setpointListData

5.3.21.2.1 General

The setpoint values and (if needed) the identifier of related measurands or time tables are modelled within this function. It is possible to either define an exact value, that should be reached or a range which shall not be fallen below or exceeded. Absolute or relative tolerances can be defined, too.

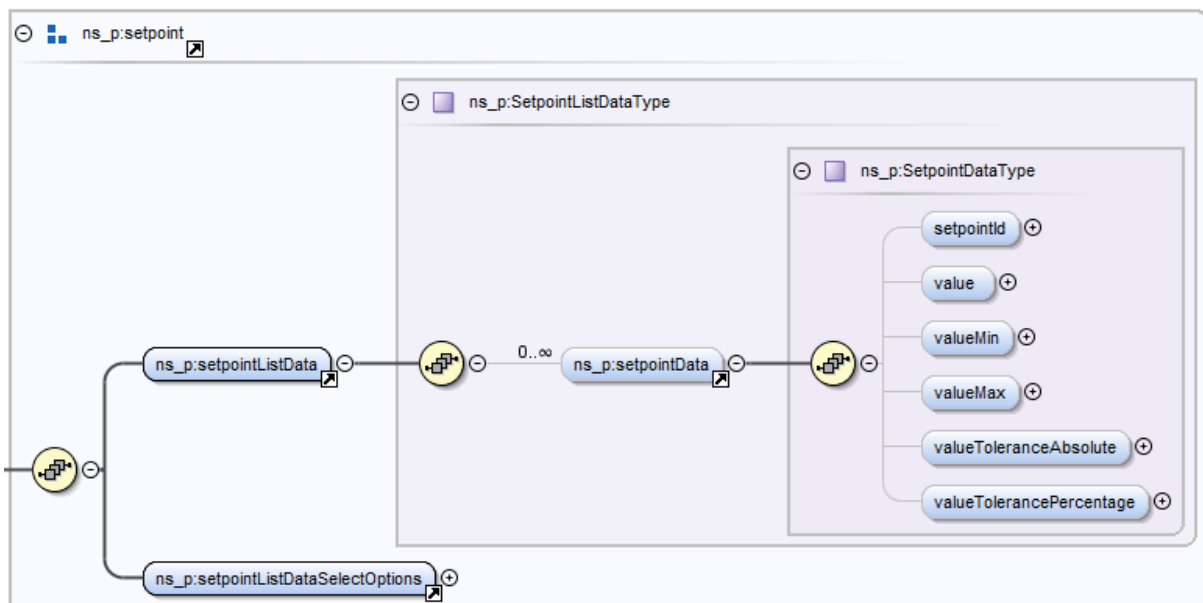


Figure 168: setpointListData function overview

5.3.21.2.2 Detailed description of elements

Element	Type	Description
setpointId	Identifier "SetpointIdType" (see Table 339).	Identifier of the setpoint.
value	Common data type "ScaledNumberType". See section 3.10.1.8.	The setpoint value itself.
valueMin	Common data type "ScaledNumberType". See section 3.10.1.8.	Instead of a fixed value (see above) a value range can be set. This would be the lower limit.

valueMax	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Instead of a fixed value (see above) a value range can be set. This would be the upper limit.
valueToleranceAbsolute	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	If the value may vary some absolute value around the desired setpoint, the maximum variation allowed (in both directions) can be stated here.
valueTolerancePercentage	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	If the value may vary some percentage around the desired setpoint, the maximum variation allowed (in both directions) can be stated here.

Table 281: setpointListData function detailed description of elements

5.3.21.2.3 Available selectors

- setpointId

5.3.21.2.4 Examples

5.3.21.2.4.1 Read-reply

If one is interested in the setpoint with the *setpointId* "3", he could send the following read command:

```

<function>setpointListData</function>
<filter>
  <cmdControl>
    <partial/>
  </cmdControl>
  <setpointListDataSelectors>
    <setpointId>3</setpointId>
  </setpointListDataSelectors>
</filter>
<setpointListData/>

```

As a reply, something like the following could be returned:

```

<function>setpointListData</function>
<filter>
  <cmdControl>
    <partial/>
  </cmdControl>
</filter>
<setpointListData>
  <setpointData>
    <setpointId>3</setpointId>
    <value>
      <number>25</number>
    </value>
    <valueToleranceAbsolute>
      <number>1</number>
    </valueToleranceAbsolute>
  </setpointData>
</setpointListData>

```

5.3.21.3 setpointConstraintsListData

5.3.21.3.1 General

The constraints regarding the possible setpoints allowed are modelled within this function.

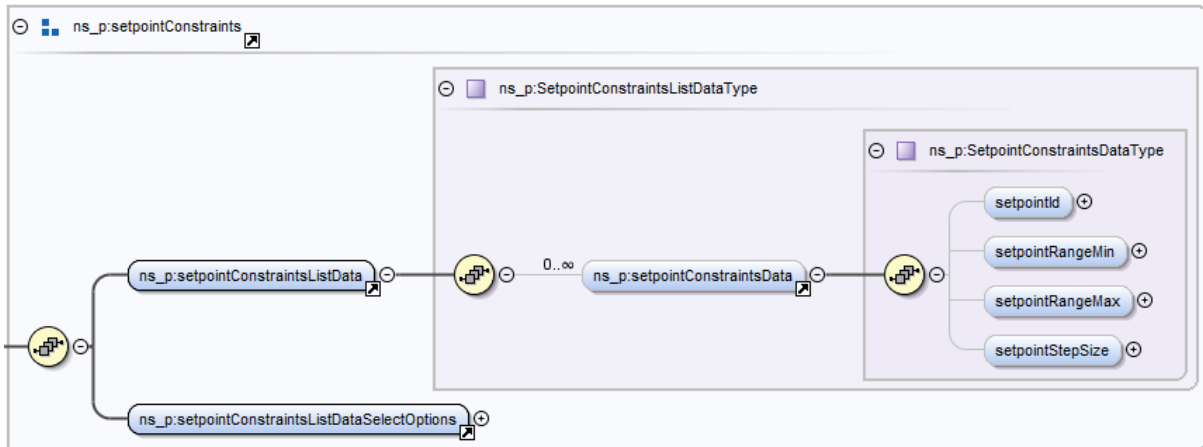


Figure 169: setpointConstraintsListData function overview

5.3.21.3.2 Detailed description of elements

Element	Type	Description
setpointId	Identifier "SetpointIdType" (see Table 339).	Identifier of the setpoint, these constraints belong to.
setpointRangeMin	Common data type "ScaledNumberType". See section 3.10.1.8.	The minimum value, the setpoint can be set to.
setpointRangeMax	Common data type "ScaledNumberType". See section 3.10.1.8.	The maximum value, the setpoint can be set to.
setpointStepSize	Common data type "ScaledNumberType". See section 3.10.1.8.	The minimum step size between two different values.

Table 282: setpointConstraintsListData function detailed description of elements

5.3.21.3.3 Available selectors

- setpointId

5.3.21.3.4 Examples

5.3.21.3.4.1 Read-reply

If one is interested in the setpoint constraints of all *setpointIds*, he could send the standard read command. A reply could be:

```
<setpointConstraintsListData>
  <setpointConstraintsData>
    <setpointId>1</setpointId>
```

```

9477         <setpointRangeMin>
9478             <number>0</number>
9479         </setpointRangeMin>
9480         <setpointRangeMax>
9481             <number>3</number>
9482             <scale>1</scale>
9483         </setpointRangeMax>
9484         <setpointStepSize>
9485             <number>5</number>
9486             <scale>-1</scale>
9487         </setpointStepSize>
9488     </setpointConstraintsData>
9489     <setpointConstraintsData>
9490         <setpointId>2</setpointId>
9491         <setpointRangeMin>
9492             <number>3</number>
9493             <scale>1</scale>
9494         </setpointRangeMin>
9495         <setpointRangeMax>
9496             <number>8</number>
9497             <scale>1</scale>
9498         </setpointRangeMax>
9499         <setpointStepSize>
9500             <number>1</number>
9501         </setpointStepSize>
9502     </setpointConstraintsData>
9503     <setpointConstraintsData>
9504         <setpointId>3</setpointId>
9505         <setpointRangeMin>
9506             <number>5</number>
9507         </setpointRangeMin>
9508         <setpointRangeMax>
9509             <number>2</number>
9510             <scale>2</scale>
9511         </setpointRangeMax>
9512         <setpointStepSize>
9513             <number>5</number>
9514         </setpointStepSize>
9515     </setpointConstraintsData>
9516 </setpointConstraintsListData>

```

9517

9518 **5.3.21.4 setpointDescriptionListData**

9519 *5.3.21.4.1 General*

9520 The rather static (non-changing) elements of the *Setpoint* class are combined in the
 9521 *setpointDescriptionListData* function. Typically, it is only requested once by a user or an application.

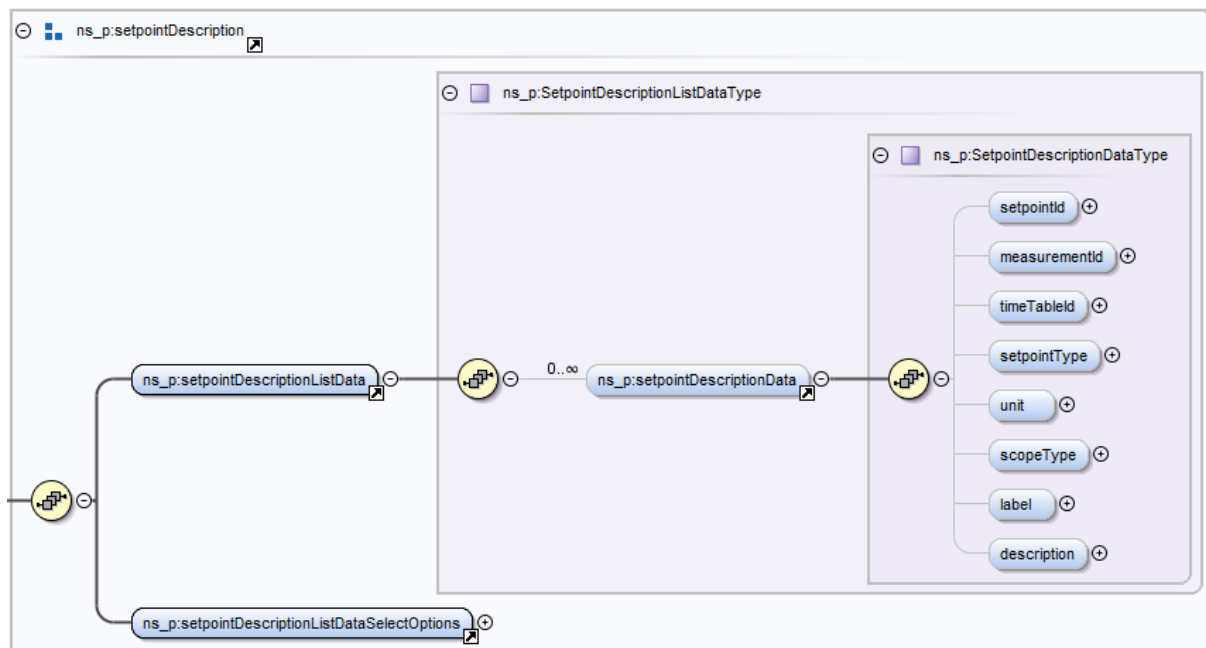


Figure 170: setpointDescriptionListData function overview

5.3.21.4.2 Detailed description of elements

Element	Type	Description
setpointId	Identifier "SetpointIdType" (see Table 339).	Identifier of the setpoint.
measurementId	Identifier "MeasurementIdType" (see Table 339).	If the setpoint relates to a measurand, the corresponding ID can be stated here.
timeTableId	Identifier "timeTableIdType" (see Table 339).	If the activation of the setpoint is time dependent, the ID of a corresponding time table can denoted in this element.
setpointType	Union "SetpointTypeType": - Enum (see Table 284): SetpointTypeEnumType - EnumExtendType (see section 3.10.1.5)	This is the type of setpoint used for this specific setpoint.
unit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	The unit used for this setpoint (see SPINE data model for all possible values).
scopeType	Common data type "ScopeTypeType". See section 3.10.1.21.	Specifies a more detailed meaning of the setpoint.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the setpoint.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the setpoint.

Table 283: setpointDescriptionListData function detailed description of elements

Enumeration **SetpointTypeEnumType**:

Value	Description
-------	-------------

valueAbsolute	An absolute value, e.g. 28°C. A unit is needed in most cases.
valueRelative	A relative value in percentage. The unit is "pct" in most cases.

Table 284: Enumeration SetpointTypeEnumType

5.3.21.4.3 Available selectors

- setpointId
- measurementId
- timeTableId
- setpointType
- scopeType

5.3.21.4.4 Examples

5.3.21.4.4.1 Read-reply

If one is interested in the setpoints related to *measurementId* "3", he could send the following read command:

```
<function>setpointDescriptionListData</function>
<filter>
  <cmdControl>
    <partial/>
  </cmdControl>
  <setpointDescriptionListDataSelectors>
    <measurementId>3</measurementId>
  </setpointDescriptionListDataSelectors>
</filter>
<setpointDescriptionListData/>
```

The reply could look like this:

```
<function>setpointDescriptionListData</function>
<filter>
  <cmdControl>
    <partial/>
  </cmdControl>
</filter>
<setpointDescriptionListData>
  <setpointDescriptionData>
    <setpointId>1</setpointId>
    <measurementId>3</measurementId>
    <timeTableId>2</timeTableId>
    <setpointType>valueAbsolute</setpointType>
    <unit>degC</unit>
    <label>Day heating</label>
  </setpointDescriptionData>
  <setpointDescriptionData>
    <setpointId>2</setpointId>
    <measurementId>3</measurementId>
    <timeTableId>4</timeTableId>
    <setpointType>valueAbsolute</setpointType>
    <unit>degC</unit>
    <label>Night heating</label>
  </setpointDescriptionData>
</setpointDescriptionListData>
```

9576

9577 **5.3.22 SubscriptionManagement**9578 **5.3.22.1 Introduction**

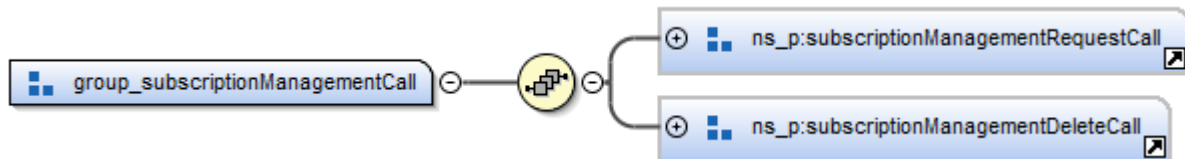
9579 Subscriptions are used to simplify communication between devices. Based on matching features,
 9580 finding suitable communication partners is a lot easier than via manual configuration only.

9581 Each subscription exists between a subscription client and a subscription server. Only clients may
 9582 initiate new subscriptions. After a successful establishment of a subscription, the subscription server
 9583 updates the client about changes of values or parameters of the corresponding feature, depending
 9584 on the subscribed functionality. E.g. a subscription server sends new temperature values of a
 9585 subscribed measurement feature to the subscription client. The exact definition of rules and
 9586 proceedings is out of scope of this section.

9587 The *SubscriptionManagement* class provides functions for establishing, reading and deleting
 9588 subscriptions.



9589

9590 *Figure 171: SubscriptionManagement function-group overview (data)*

9591

9592 *Figure 172: SubscriptionManagement function-group overview (call)*

9593

9594 **5.3.22.2 subscriptionManagementEntryListData**9595 **5.3.22.2.1 General**

9596 All existing subscriptions can be listed in the *subscriptionManagementEntryListData* function. It may
 9597 be used to store the entries in the device itself. If another device asks for the list of subscriptions, a
 9598 device may decide whether it only replies the subscriptions with the requesting device or all available
 9599 subscriptions.

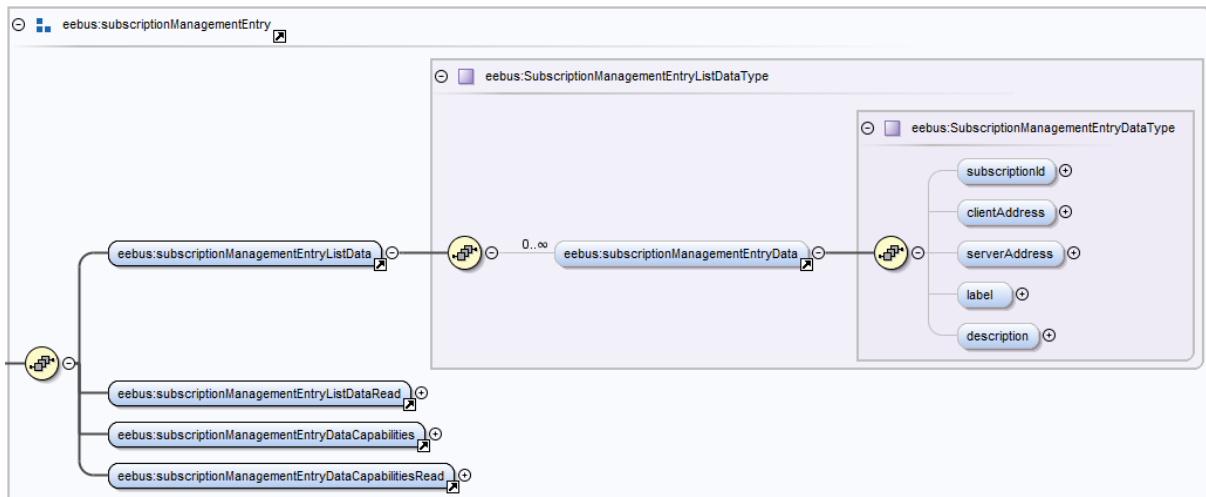


Figure 173: subscriptionManagementEntryListData function overview

5.3.22.2.2 Detailed description of elements

Element	Type	Description
subscriptionId	Identifier "SubscriptionIdType" (see Table 339).	Internal identifier for the specific subscription entry.
clientAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>client</i> in this subscription.
serverAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>server</i> in this subscription.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the subscription.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the subscription.

Table 285: subscriptionManagementEntryListData function detailed description of elements

5.3.22.2.3 Available selectors

- subscriptionId
- clientAddress
- serverAddress

5.3.22.2.4 Examples

5.3.22.2.4.1 Read-reply

If one is interested in all subscriptions of a device, he could send a standard read command to the device, which could reply something like this:

```
<subscriptionManagementEntryListData>
  <subscriptionManagementEntryData>
```



```

9617     <subscriptionId>1</subscriptionId>
9618     <clientAddress>
9619         <device>d:_i:46925_CP\_pqt1-27638162683172637812356813</device>
9620         <entity>2</entity>
9621         <feature>1</feature>
9622     </clientAddress>
9623     <serverAddress>
9624         <device>d:_i:46925_CP\_arqf-64986751356897215468975468</device>
9625         <entity>1</entity>
9626         <feature>1</feature>
9627     </serverAddress>
9628     <label>LightInfo_01</label>
9629 </subscriptionManagementEntryData>
9630 <subscriptionManagementEntryData>
9631     <subscriptionId>2</subscriptionId>
9632     <clientAddress>
9633         <device>d:_i:46925_CP\_pqt1-27638162683172637812356813</device>
9634         <entity>2</entity>
9635         <feature>4</feature>
9636     </clientAddress>
9637     <serverAddress>
9638         <device>d:_i:46925_CP\_arqf-64986751356897215468975468</device>
9639         <entity>1</entity>
9640         <feature>2</feature>
9641     </serverAddress>
9642     <label>LightInfo_02</label>
9643 </subscriptionManagementEntryData>
9644 </subscriptionManagementEntryListData>

```

5.3.22.3 *subscriptionManagementRequestCall*

5.3.22.3.1 *General*

To initiate a new subscription, an entity capable of node management on the client side, sends a *subscriptionManagementRequestCall* to an entity capable of node management on the server side, with the *clientAddress* element in the payload set to one of its features with the role *client* and the *serverAddress* element in the payload set to one of the other device's feature address with the role *server*. The receiving server device checks whether it will accept or decline the request and responds with an acknowledgement message (see [ProtocolSpecification], section "Acknowledgement message") with the *resultData* function and an according *errorNumber* set to 0 (accept subscription request) or some other value (definition of the error code is out of scope of this section; decline subscription request).

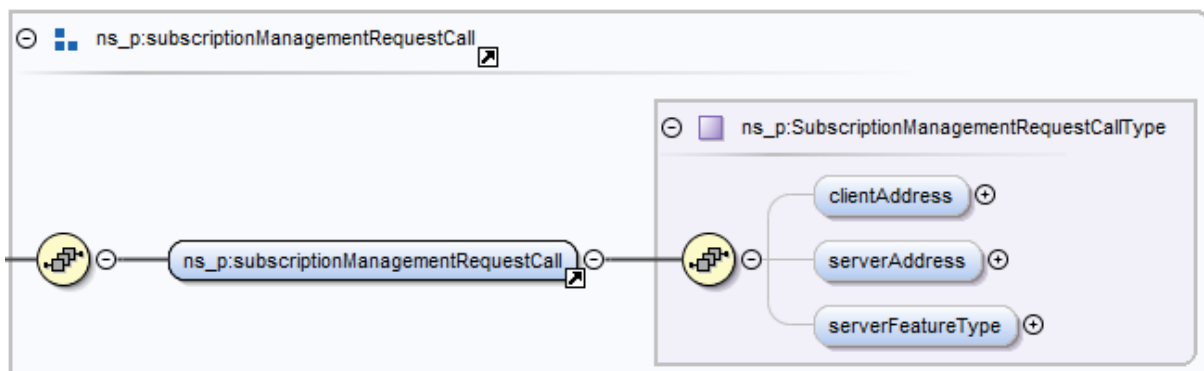


Figure 174: *subscriptionManagementRequestCall* function overview

9659

9660 **5.3.22.3.2 Detailed description of elements**

Element	Type	Description
clientAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>client</i> in the subscription request.
serverAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>server</i> in the subscription request.
serverFeatureType	Common data type "FeatureTypeType". See section 3.10.1.29.	The <i>Feature Type</i> of the server feature. Stated for verification.

9661 *Table 286: subscriptionManagementRequestCall function detailed description of elements*

9662

9663 **5.3.22.3.3 Available selectors**

9664 None.

9665

9666 **5.3.22.3.4 Examples**9667 **5.3.22.3.4.1 Call**

9668 If a new subscription shall be initiated, the client device could send a command like this to the server
 9669 device:

```

9670 <subscriptionManagementRequestCall>
9671   <clientAddress>
9672     <device>d:_i:46925_CP\_pqt1-27638162683172637812356813</device>
9673     <entity>2</entity>
9674     <feature>1</feature>
9675   </clientAddress>
9676   <serverAddress>
9677     <device>d:_i:46925_CP\_arqf-64986751356897215468975468</device>
9678     <entity>1</entity>
9679     <feature>1</feature>
9680   </serverAddress>
9681   <serverFeatureType>ActuatorSwitch</serverFeatureType>
9682 </subscriptionManagementRequestCall>

```

9683

9684 **5.3.22.4 subscriptionManagementDeleteCall**9685 **5.3.22.4.1 General**

9686 A device may delete an existing subscription with itself and another device. Therefore, this call is
 9687 used.

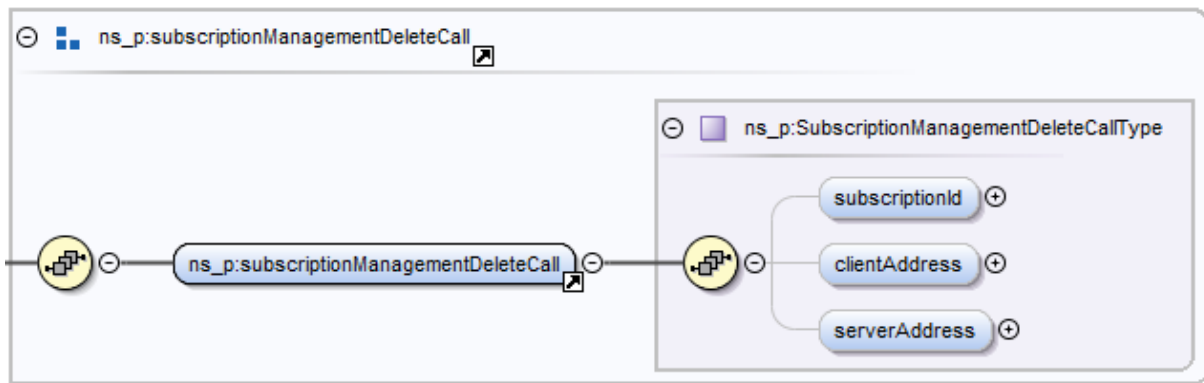


Figure 175: subscriptionManagementDeleteCall function overview

5.3.22.4.2 Detailed description of elements

Element	Type	Description
subscriptionId	Identifier "SubscriptionIdType" (see Table 339).	Internal identifier for the specific subscription entry.
clientAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>client</i> in the subscription request.
serverAddress	Common data type "FeatureAddressType". See section 3.10.3.6.	Address of the feature with the role <i>server</i> in the subscription request.

Table 287: subscriptionManagementDeleteCall function detailed description of elements

5.3.22.4.3 Available selectors

None.

5.3.22.4.4 Examples

5.3.22.4.4.1 Call

If an existing subscription shall be deleted, one device could send a command like this to the other device:

```

<subscriptionManagementDeleteCall>
  <subscriptionId>1</subscriptionId>
  <clientAddress>
    <device>d:_i:46925_CP\_pqt1-27638162683172637812356813</device>
    <entity>2</entity>
    <feature>1</feature>
  </clientAddress>
  <serverAddress>
    <device>d:_i:46925_CP\_arqf-64986751356897215468975468</device>
    <entity>1</entity>
    <feature>1</feature>
  </serverAddress>
</subscriptionManagementDeleteCall>
  
```

5.3.23 SupplyCondition

5.3.23.1 Introduction

Practice shows that the supply of electricity, water, etc. can be temporarily impaired. Some possible causes are announced maintenance work at the supply line, supply network overload, or depletion of a local energy resource (battery, e.g.). The Supply Condition concept permits indication of the supply condition by a household's SPINE data models.

The concept is less intended for any automatisms. It is rather intended for indications to users in order to prepare or apply measures in time.

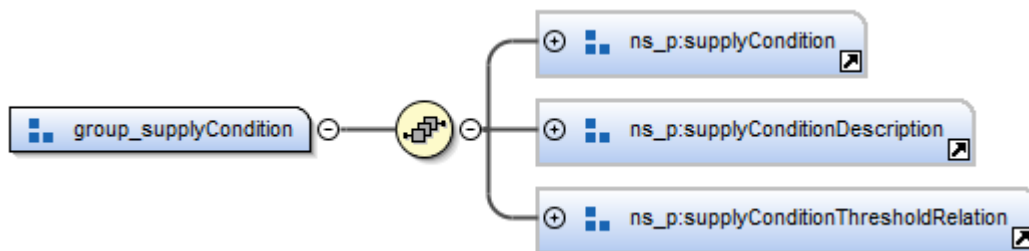


Figure 176: SupplyCondition function-group overview

5.3.23.2 supplyConditionListData

5.3.23.2.1 General

This function can be used to model the "events", i.e. the information on a specific supply condition. These events can, e.g., be sent from a DSO to a household. In this case, the household would be the data owner and the DSO would send (as a client) write commands to this resource.

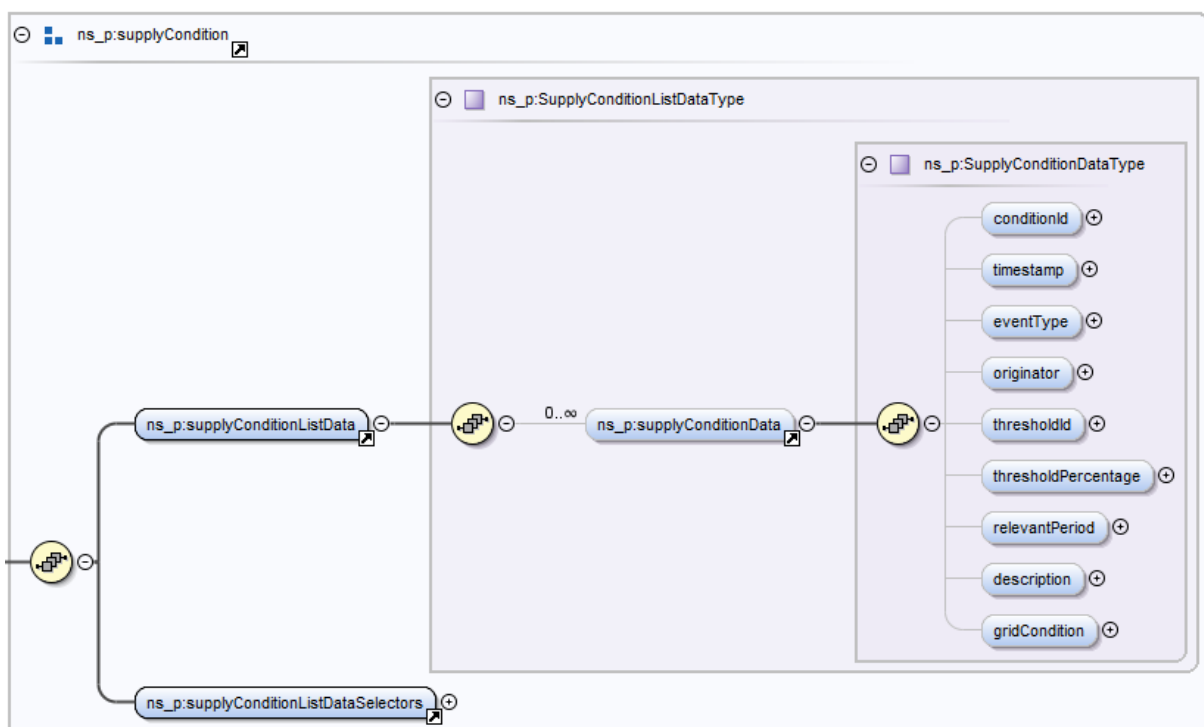


Figure 177: supplyConditionListData function overview

9733

9734 5.3.23.2.2 Detailed description of elements

Element	Type	Description
conditionId	Identifier "ConditionIdType" (see Table 339).	Identifier of the condition.
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	The timestamp of the event.
eventType	Union "SupplyConditionEventTypeType": - Enum (see Table 289): SupplyConditionEventTypeEnumType - EnumExtendType (see section 3.10.1.5)	The kind of the event.
originator	Union "SupplyConditionOriginatorType": - Enum (see Table 290): SupplyConditionOriginatorEnumType - EnumExtendType (see section 3.10.1.5)	The originator (in terms of instance, not in terms of address) of the information.
thresholdId	Identifier "ThresholdIdType" (see Table 339).	The identifier of this event's threshold (if available).
thresholdPercentage	Common data type "ScaledNumberType". See section 3.10.1.8.	Relation of current situation to (configured) threshold (if available).
relevantPeriod	Common data type "TimePeriodType". See section 3.10.2.1.	Esp. the announcement of maintenance can make use of this element in order to tell when a maintenance is planned.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the event.
gridCondition	Union "GridConditionType": - Enum (see Table 291): GridConditionEnumType - EnumExtendType (see section 3.10.1.5)	Indicates the condition of the electricity grid.

9735 Table 288: supplyConditionListData function detailed description of elements

9736 Enumeration SupplyConditionEventTypeEnumType:

Value	Description
thresholdExceeded	The related threshold was exceeded (may be good or bad, see definition of the corresponding <i>thresholdType</i> element in section 5.3.26.4.2 for details).
fallenBelowThreshold	The related threshold was fallen below (may be good or bad, see definition of the corresponding <i>thresholdType</i> element in section 5.3.26.4.2 for details).
supplyInterrupt	The supply of the corresponding commodity was interrupted.
releaseOfLimitations	"Back to normal", no limitations or threshold violations.
otherProblem	Some problem with the supply without further details about the problem.

9737 Table 289: Enumeration SupplyConditionEventTypeEnumType

9738 Enumeration **SupplyConditionOriginatorEnumType**:

Value	Description
externDSO	The originator of the current condition "event" is the DSO (distribution system operator).
externSupplier	The originator of the current condition "event" is the energy supplier.
internalLimit	Some internal limit at the premises generated the current condition "event".
internalService	Due to an internal service at the premises, the current condition "event" was generated.
internalUser	An user interaction is the source of the current condition "event".

9739 *Table 290: Enumeration SupplyConditionOriginatorEnumType*

9740 Enumeration **GridConditionEnumType**:

Value	Description
consumptionRed	The electricity grid condition for consumption from the grid is "red".
consumptionYellow	The electricity grid condition for consumption from the grid is "yellow".
good	The electricity grid condition is "good".
productionYellow	The electricity grid condition for feeding into the grid is "yellow".
productionRed	The electricity grid condition for feeding into the grid is "red".

9741 *Table 291: Enumeration GridConditionEnumType*

9742

9743 *5.3.23.2.3 Available selectors*

9744 - conditionId

9745 - timestampInterval

9746

9747 *5.3.23.2.4 Examples*

9748 *5.3.23.2.4.1 Notify*

9749 A user ordered a plumber and broadcasts this information to all displays (the “filter/partial” part to
9750 announce the “restricted function exchange is simplified by “...” just to improve the readability”):

9751 ...

9752 <supplyConditionListData>

9753 <supplyConditionData>

9754 <conditionId>3</conditionId>

9755 <timestamp>2006-05-04T10:13:51.0Z</timestamp>

9756 <eventType>supplyInterrupt</eventType>

9757 <originator>internalService</originator>

9758 <relevantPeriod>

9759 <startTime>2006-05-04T14:00:00.0Z</startTime>

9760 <endTime>2006-05-04T15:00:00.0Z</endTime>

9761 </relevantPeriod>

9762 <description>Attention: No water supply from 2PM to

9763 3PM</description>

9764 <gridCondition>good</gridCondition>

9765 </supplyConditionData>

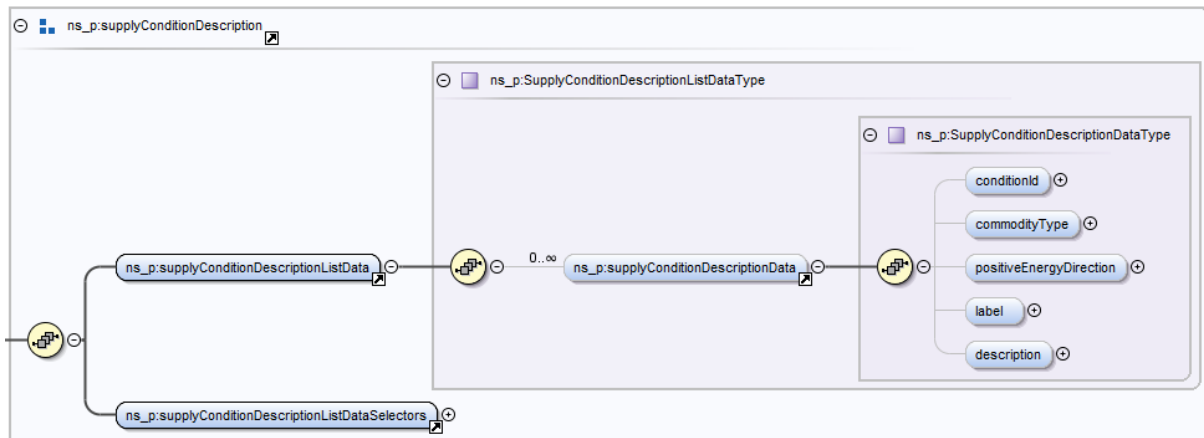
9766 </supplyConditionListData>

9767

9768 5.3.23.3 *supplyConditionDescriptionListData*

9769 5.3.23.3.1 *General*

9770 The more static information about a supply condition is modelled within this function.



9771
9772 Figure 178: *supplyConditionDescriptionListData* function overview

9773

9774 5.3.23.3.2 *Detailed description of elements*

Element	Type	Description
conditionId	Identifier "ConditionIdType" (see Table 339).	Identifier of the condition.
commodityType	Common data type "CommodityTypeType". See section 3.10.1.9.	The type of the commodity handled by this condition is stated here.
positiveEnergyDirection	Common data type "EnergyDirectionType". See section 3.10.1.13.	The <i>positiveEnergyDirection</i> states whether energy consumption or production will be counted as positive value.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the condition.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the condition.

9775 Table 292: *supplyConditionDescriptionListData* function detailed description of elements

9776

9777 5.3.23.3.3 *Available selectors*

9778 - conditionId

9779

9780 5.3.23.3.4 *Examples*

9781 5.3.23.3.4.1 *Read-Reply*

9782 If one is interested in all supply conditions, he could send a standard read command to the server.

9783 The reply could look like this:

```

9784 <supplyConditionDescriptionListData>
9785   <supplyConditionDescriptionData>
9786     <conditionId>1</conditionId>
9787     <commodityType>electricity</commodityType>
9788     <positiveEnergyDirection>consume</positiveEnergyDirection>
9789     <label>Electricity supply</label>
9790     <description>Supply for electricity (3-phase
9791 connection).</description>
9792   </supplyConditionDescriptionData>
9793   <supplyConditionDescriptionData>
9794     <conditionId>2</conditionId>
9795     <commodityType>water</commodityType>
9796     <positiveEnergyDirection>consume</positiveEnergyDirection>
9797     <label>Water supply</label>
9798     <description>Supply for domestic water.</description>
9799   </supplyConditionDescriptionData>
9800 </supplyConditionDescriptionListData>

```

9801

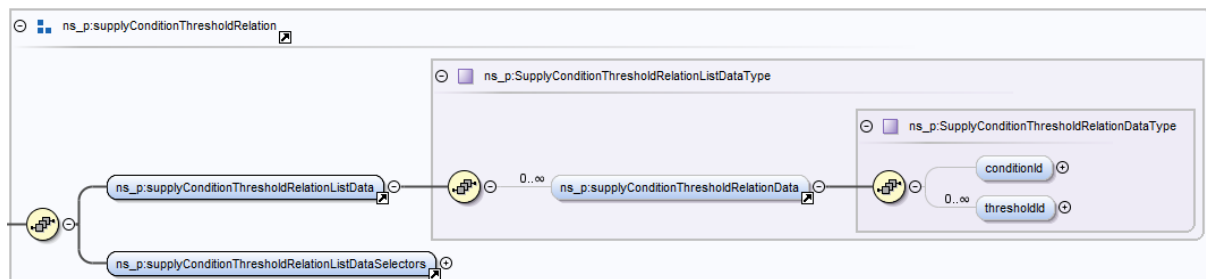
9802 5.3.23.4 supplyConditionThresholdRelationListData

9803 5.3.23.4.1 General

9804 Multiple thresholds can be associated to a condition. This leads to the relation

- 9805 • 1 conditionId -> 0..m thresholdId

9806 The supplyConditionThresholdRelationListData function can be used to model concrete relations.



9807

9808 Figure 179: supplyConditionThresholdRelationListData function overview

9809

9810 5.3.23.4.2 Detailed description of elements

Element	Type	Description
conditionId	Identifier "ConditionIdType" (see Table 339).	Identifier of the condition.
thresholdId (list)	Identifier "ThresholdIdType" (see Table 339).	Related threshold identifier(s).

9811 Table 293: supplyConditionThresholdRelationListData function detailed description of elements

9812

9813 5.3.23.4.3 Available selectors

- 9814 - conditionId
- 9815 - thresholdId

9816

9817 **5.3.23.4.4 Examples**

9818 **5.3.23.4.4.1 Read-reply**

9819 An energy management gateway is able to keep relation information between condition and (supply)
9820 threshold and provide the list upon request.

9821 A user selects the energy management gateway on an external display and queries for information
9822 on conditionId 5. The display sends a read classifier together with the read command:

```
9823 <function>supplyConditionThresholdRelationListData</function>
9824 <filter>
9825     <cmdControl>
9826         <partial/>
9827     </cmdControl>
9828     <supplyConditionThresholdRelationListDataSelectors>
9829         <conditionId>5</conditionId>
9830     </supplyConditionThresholdRelationListDataSelectors>
9831 </filter>
9832 <messagingListData/>
```

9833 The energy management gateway replies with a proper supplyConditionThresholdRelationListData:

```
9834 <function>supplyConditionThresholdRelationListData</function>
9835 <filter>
9836     <cmdControl>
9837         <partial/>
9838     </cmdControl>
9839 </filter>
9840 <supplyConditionThresholdRelationListData>
9841     <supplyConditionThresholdRelationData>
9842         <conditionId>5</conditionId>
9843         <thresholdId>3</thresholdId>
9844         <thresholdId>4</thresholdId>
9845     </supplyConditionThresholdRelationData>
9846 </supplyConditionThresholdRelationListData>
```

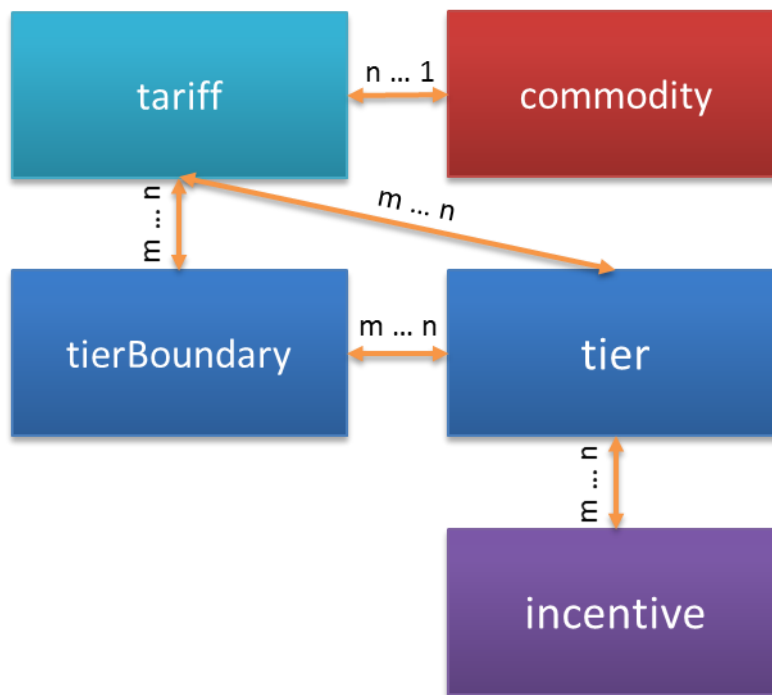
9847

9848 **5.3.24 TariffInformation**

9849 **5.3.24.1 Introduction**

9850 The TariffInformation Class provides functions for modelling simple or complex data related to tariffs.

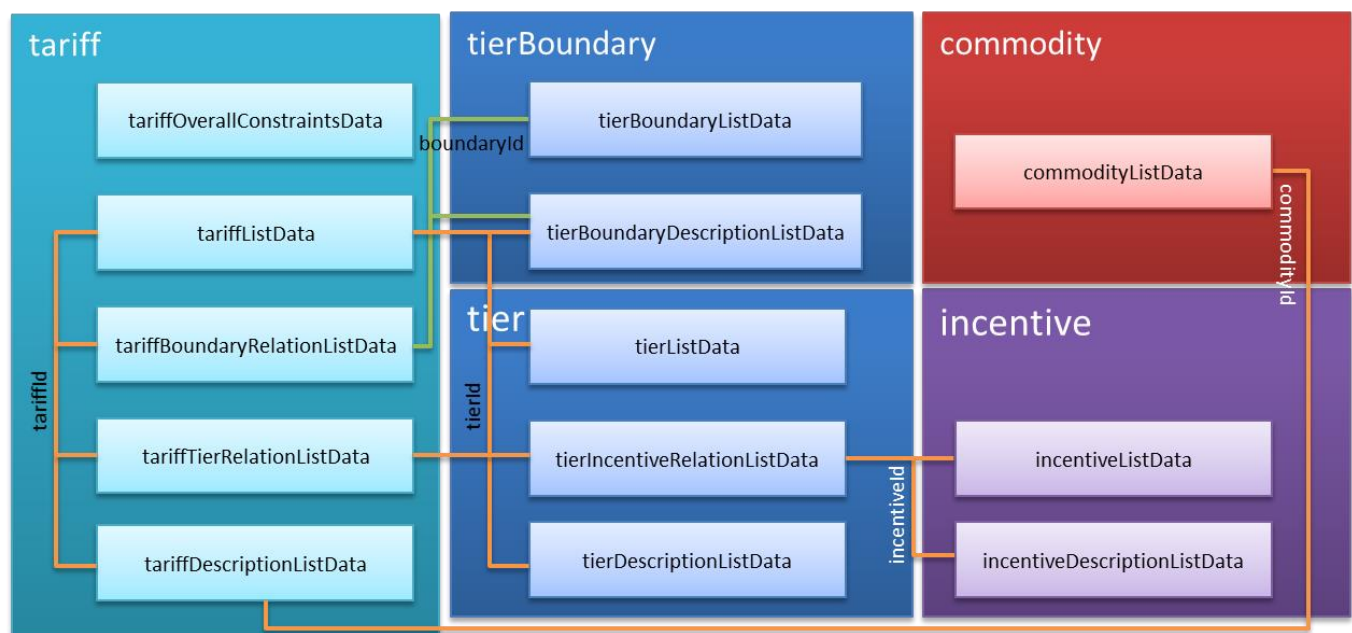
9851 The following figures visualize the dependencies of the different categories available in this class:



9852

9853 *Figure 180: TariffInformation relations, sub-class level*

9854 Basis is the *tariff* that has one *commodity* and some *tiers* with *tierBoundaries*. Each *tier* itself has
 9855 *incentives*. Identifiers are used for referencing the different aspects of the tariff model. They are
 9856 shown in the figure below:



9857

9858 *Figure 181: TariffInformation relations, function level*

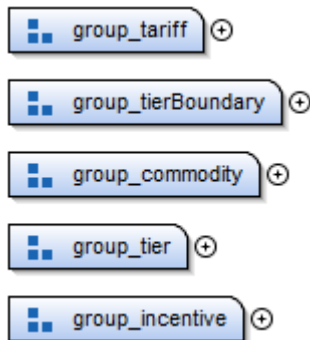
9859 As it can be seen, the **tariff** consists of some overall constraints (*tariffOverallConstraintsData*), a
 9860 dynamic data part (*tariffListData*) and the rather static part (*tariffDescriptionListData*). Additionally,
 9861 relations are given to the related boundaries (*tariffBoundaryRelationListData*) and tiers
 9862 (*tariffTierRelationListData*).

9863 The **tierBoundary** has a dynamic (*tierBoundaryListData*) and a rather static part
9864 (*tierBoundaryDescriptionListData*).

9865 The **tier** consists of the dynamic (*tierListData*), the rather static part (*tierDescriptionListData*) and a
9866 relation to related incentives (*tierIncentiveRelationListData*).

9867 The **incentive** has two functions, the dynamic part (*incentiveListData*) and the rather static part
9868 (*incentiveDescriptionListData*).

9869 The **commodity** only has one information (*commodityListData*).



9870
9871 *Figure 182: TariffInformation Sub-class overview*

9872

9873 **5.3.24.2 Sub-class Tariff**

9874 The information related to the tariff itself are modelled within the functions of this sub-class.

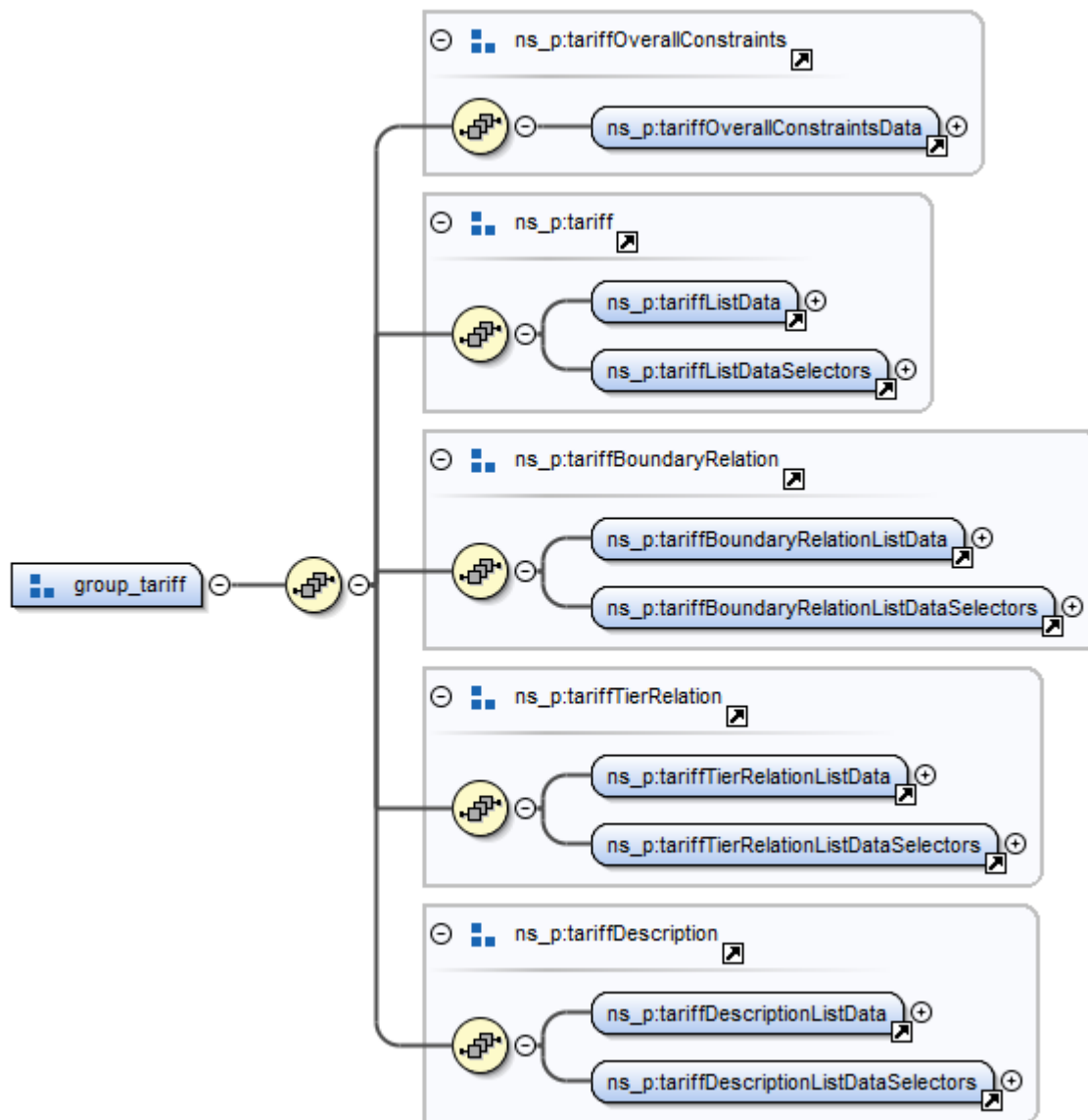


Figure 183: TariffInformation, sub-class Tariff, function-group overview

5.3.24.3 tariffOverallConstraintsData

5.3.24.3.1 General

Constraints that are valid for all tariffs on this particular feature are included in this function.

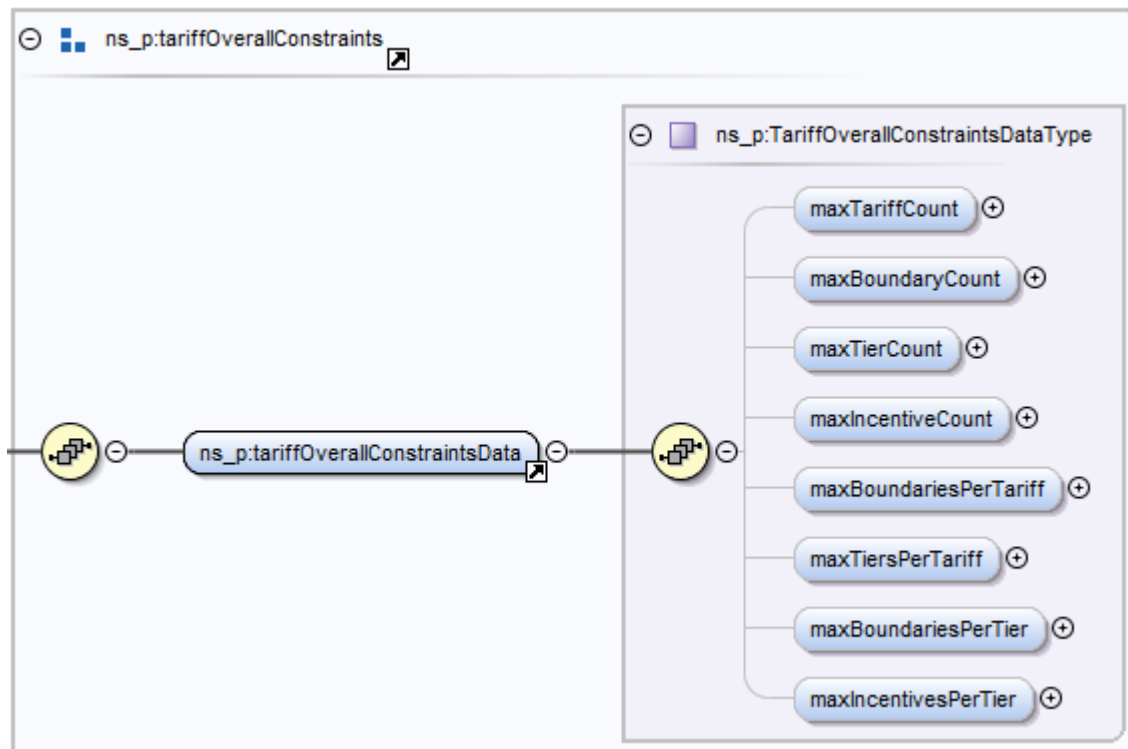


Figure 184: tariffOverallConstraintsData function overview

5.3.24.3.2 Detailed description of elements

Element	Type	Description
maxTariffCount	Simple type "TariffCountType" (restriction of "TariffIdType" (see Table 339)).	Maximum amount of tariffs supported by the server.
maxBoundaryCount	Simple type "TierBoundaryCountType" (restriction of "TierBoundaryIdType" (see Table 339)).	Maximum amount of tier boundaries supported by the server.
maxTierCount	Simple type "TierCountType" (restriction of "TierIdType" (see Table 339)).	Maximum amount of tiers supported by the server.
maxIncentiveCount	Simple type "IncentiveCountType" (restriction of "IncentiveIdType" (see Table 339)).	Maximum amount of incentives supported by the server.
maxBoundariesPerTariff	Simple type "TierBoundaryCountType" (restriction of "TierBoundaryIdType" (see Table 339)).	Maximum amount of (tier) boundaries per tariff supported by the server.
maxTiersPerTariff	Simple type "TierCountType" (restriction of "TierIdType" (see Table 339)).	Maximum amount of tiers per tariff supported by the server.
maxBoundariesPerTier	Simple type "TierBoundaryCountType" (restriction of "TierBoundaryIdType" (see Table 339)).	Maximum amount of boundaries per tier supported by the server.

maxIncentivesPerTier	Simple type "IncentiveCountType" (restriction of "IncentiveIdType" (see Table 339)).	Maximum amount of incentives per tier supported by the server.
----------------------	--	--

Table 294: tariffOverallConstraintsData function detailed description of elements

5.3.24.3.3 Available selectors

None.

5.3.24.3.4 Examples

5.3.24.3.4.1 Read-reply

If one is interested in the tariff overall constraints, he could send a standard read command to the server. The reply could look like this:

```
<tariffOverallConstraintsData>
  <maxTariffCount>5</maxTariffCount>
  <maxBoundaryCount>40</maxBoundaryCount>
  <maxTierCount>20</maxTierCount>
  <maxIncentiveCount>60</maxIncentiveCount>
  <maxBoundariesPerTariff>8</maxBoundariesPerTariff>
  <maxTiersPerTariff>4</maxTiersPerTariff>
  <maxBoundariesPerTier>2</maxBoundariesPerTier>
  <maxIncentivesPerTier>3</maxIncentivesPerTier>
</tariffOverallConstraintsData>
```

5.3.24.4 tariffListData

5.3.24.4.1 General

The dynamic part of the tariff sub-class delivers the information which tier(s) is/are currently active for this tariff.

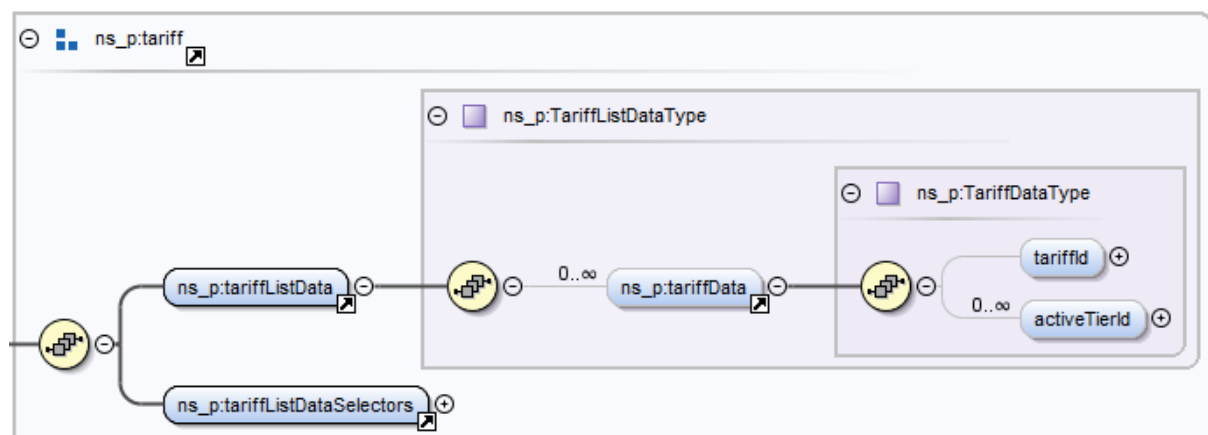


Figure 185: tariffListData function overview

5.3.24.4.2 Detailed description of elements

Element	Type	Description
tariffId	Identifier "TariffIdType" (see Table 339).	Primary identifier for the tariff.
activeTierId (list)	Identifier "TierIdType" (see Table 339).	Foreign key(s) of the active tier(s) for this tariff.

Table 295: tariffListData function detailed description of elements

5.3.24.4.3 Available selectors

- tariffId
- activeTierId

5.3.24.4.4 Examples

5.3.24.4.4.1 Read-reply

If one is interested in all dynamic tariff information, he could send a standard read command to the server. The reply could look like this:

```
<tariffListData>
  <tariffData>
    <tariffId>1</tariffId>
    <activeTierId>1</activeTierId>
    <activeTierId>2</activeTierId>
  </tariffData>
  <tariffData>
    <tariffId>2</tariffId>
    <activeTierId>3</activeTierId>
  </tariffData>
</tariffListData>
```

5.3.24.5 tariffBoundaryRelationListData

5.3.24.5.1 General

The boundaries related to a tariff are modelled within this function.

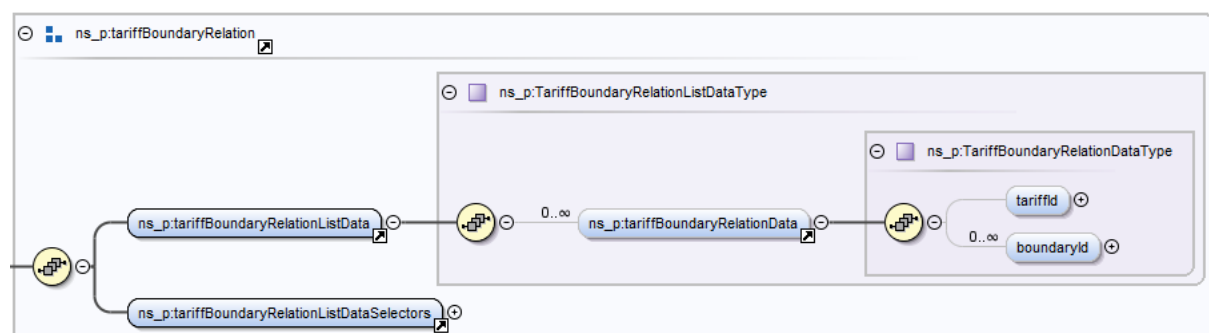


Figure 186: tariffBoundaryRelationListData function overview

5.3.24.5.2 Detailed description of elements

Element	Type	Description
tariffId	Identifier "TariffIdType" (see Table 339).	Primary identifier for the tariff.
boundaryId (list)	Identifier "TierBoundaryIdType" (see Table 339).	Foreign key(s) of the related boundary(/ies) for this tariff.

Table 296: tariffBoundaryRelationListData function detailed description of elements

5.3.24.5.3 Available selectors

- tariffId
- boundaryId

5.3.24.5.4 Examples

5.3.24.5.4.1 Read-reply

If one is interested in the possible boundaries of all tariffs, he could send a standard read command to the server. The reply could look like this:

```

<tariffBoundaryRelationListData>
  <tariffBoundaryRelationData>
    <tariffId>1</tariffId>
    <boundaryId>1</boundaryId>
    <boundaryId>2</boundaryId>
    <boundaryId>3</boundaryId>
  </tariffBoundaryRelationData>
  <tariffBoundaryRelationData>
    <tariffId>2</tariffId>
    <boundaryId>1</boundaryId>
    <boundaryId>3</boundaryId>
  </tariffBoundaryRelationData>
</tariffBoundaryRelationListData>

```

5.3.24.6 tariffTierRelationListData

5.3.24.6.1 General

The tiers related to a tariff are modelled within this function.

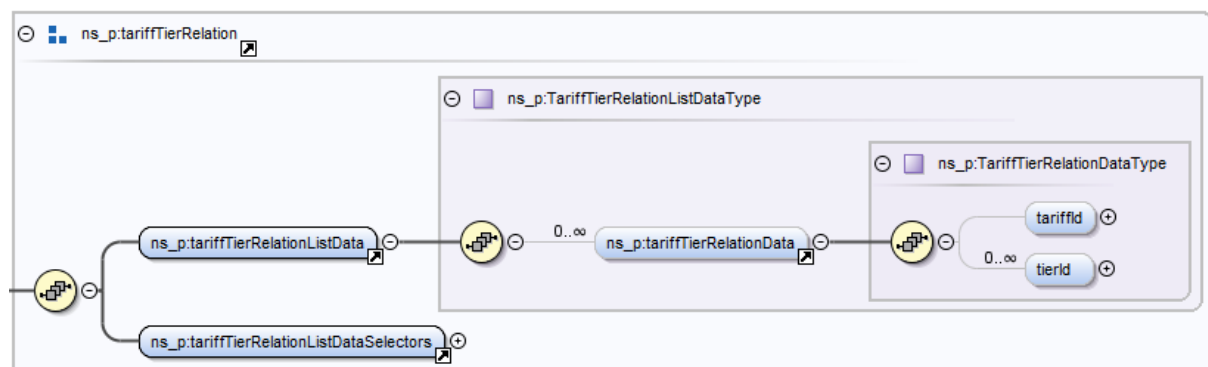


Figure 187: tariffTierRelationListData function overview

9972 **5.3.24.6.2 Detailed description of elements**

Element	Type	Description
tariffId	Identifier "TariffIdType" (see Table 339).	Primary identifier for the tariff.
tierId (list)	Identifier "TierIdType" (see Table 339).	Foreign key(s) of the related tier(s) for this tariff.

9973 *Table 297: tariffTierRelationListData function detailed description of elements*

9974

9975 **5.3.24.6.3 Available selectors**

9976 - tariffId

9977 - tierId

9978

9979 **5.3.24.6.4 Examples**9980 **5.3.24.6.4.1 Read-reply**

9981 If one is interested in the possible tiers of all tariffs, he could send a standard read command to the
 9982 server. The reply could look like this:

```

9983 <tariffTierRelationListData>
9984   <tariffTierRelationData>
9985     <tariffId>1</tariffId>
9986     <tierId>1</tierId>
9987     <tierId>2</tierId>
9988     <tierId>3</tierId>
9989     <tierId>4</tierId>
9990   </tariffTierRelationData>
9991   <tariffTierRelationData>
9992     <tariffId>2</tariffId>
9993     <tierId>1</tierId>
9994     <tierId>2</tierId>
9995   </tariffTierRelationData>
9996 </tariffTierRelationListData>

```

9997

9998 **5.3.24.7 tariffDescriptionListData**9999 **5.3.24.7.1 General**

10000 The rather static information about a tariff (e.g. the references to a commodity or a measurand, the
 10001 scopeType, etc.) are modelled within this function.

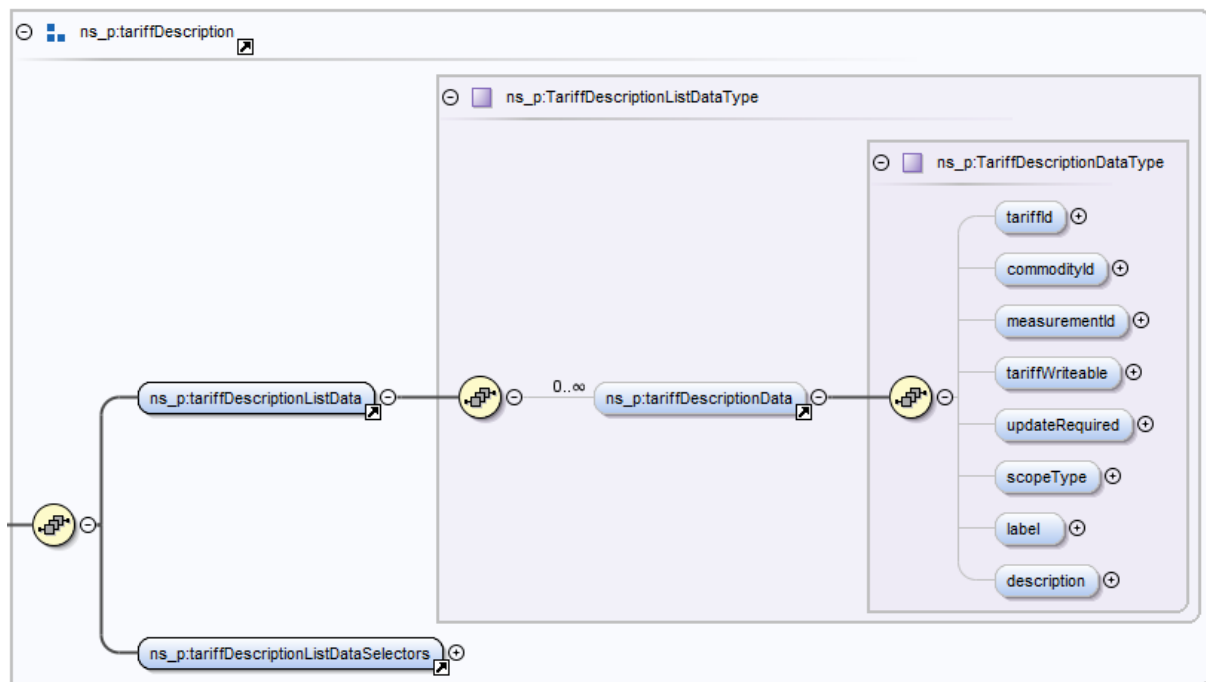


Figure 188: tariffDescriptionListData function overview

5.3.24.7.2 Detailed description of elements

Element	Type	Description
tariffId	Identifier "TariffIdType" (see Table 339).	Primary identifier for the tariff.
commodityId	Identifier "CommodityIdType" (see Table 339).	If the tariff is linked to a commodity, the corresponding identifier can be stated here.
measurementId	Identifier "MeasurementIdType" (see Table 339).	If the tariff is linked to a measurand, the corresponding identifier can be stated here.
tariffWriteable	xs:boolean (W3C standard type)	Whether the tariff is writeable or not can be denoted in this element.
updateRequired	xs:boolean (W3C standard type)	If an update for this tariff (by some bound client) is required, this element will be set to "true".
scopeType	Common data type "ScopeTypeType". See section 3.10.1.21.	The scope type for this tariff.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the tariff.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the tariff.

Table 298: tariffDescriptionListData function detailed description of elements

5.3.24.7.3 Available selectors

- tariffId
- commodityId
- measurementId
- scopeType

10013

10014 **5.3.24.7.4 Examples**10015 **5.3.24.7.4.1 Read-reply**

10016 If one is interested in all tariff descriptions, he could send a standard read command to the server.

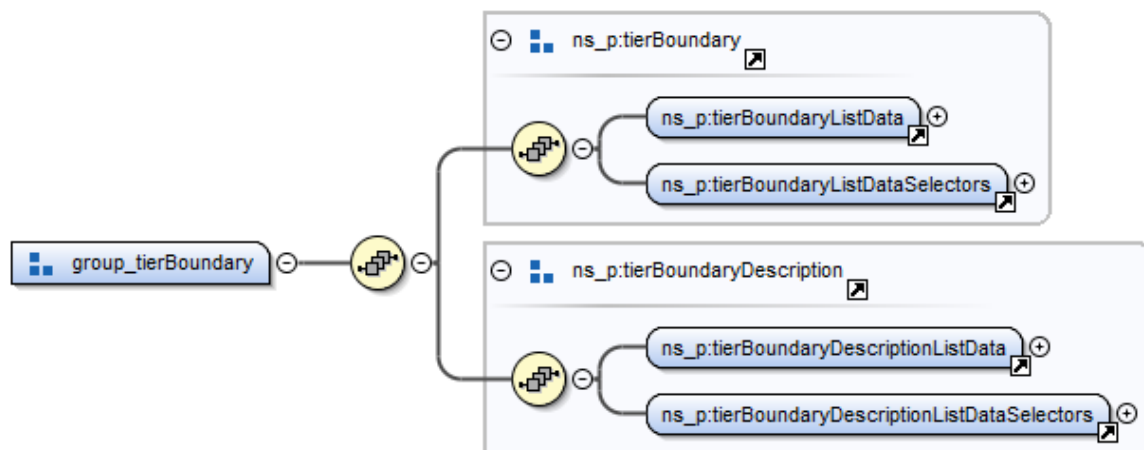
10017 The reply could look like this:

```

10018 <tariffDescriptionListData>
10019   <tariffDescriptionData>
10020     <tariffId>1</tariffId>
10021     <commodityId>1</commodityId>
10022     <measurementId>1</measurementId>
10023     <tariffWriteable>false</tariffWriteable>
10024     <updateRequired>true</updateRequired>
10025     <label>Tariff 1 - Basic</label>
10026   </tariffDescriptionData>
10027   <tariffDescriptionData>
10028     <tariffId>2</tariffId>
10029     <commodityId>1</commodityId>
10030     <measurementId>4</measurementId>
10031     <tariffWriteable>true</tariffWriteable>
10032     <updateRequired>false</updateRequired>
10033     <label>Tariff 2 - Eco Green</label>
10034   </tariffDescriptionData>
10035 </tariffDescriptionListData>

```

10036

10037 **5.3.24.8 Sub-class TierBoundary**10038 All functions related to the boundaries of tiers are clustered in this *TierBoundary* sub-class.

10039

10040 *Figure 189: TariffInformation, sub-class TierBoundary, function-group overview*

10041

10042 **5.3.24.9 tierBoundaryListData**10043 **5.3.24.9.1 General**

10044 Boundaries (for tiers) are needed to identify in which value range the tier is valid. The validity can
 10045 also depend on a time interval / time table.

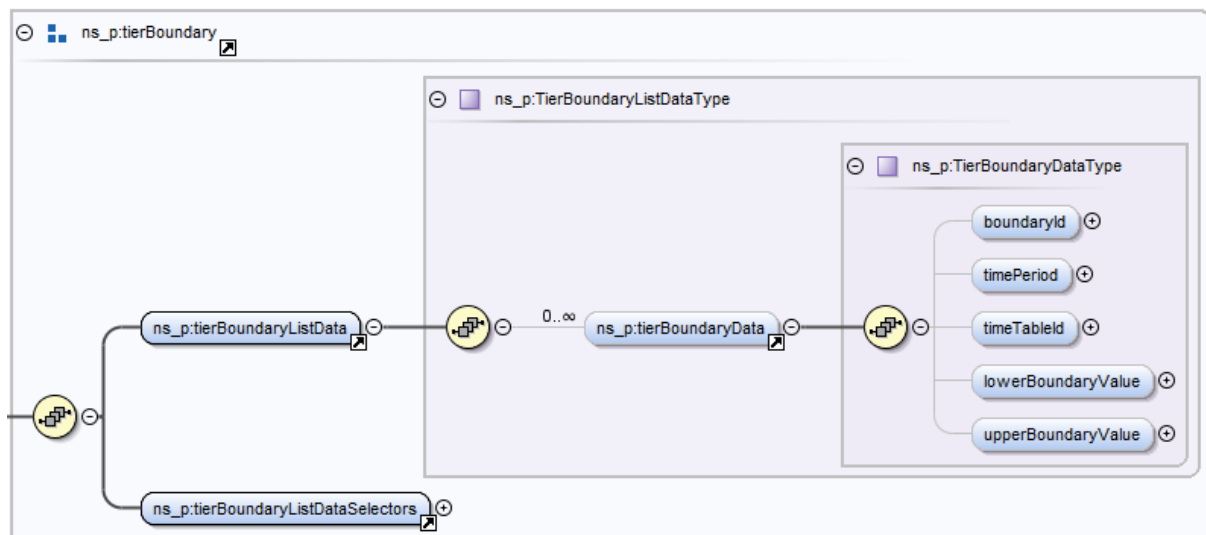


Figure 190: tierBoundaryListData function overview

5.3.24.9.2 Detailed description of elements

Element	Type	Description
boundaryId	Identifier "TierBoundaryIdType" (see Table 339).	Primary identifier for the boundary.
timePeriod	Common data type "TimePeriodType". See section 3.10.2.1.	If the boundary depends on a time period, it can be denoted here.
timeTableId	Identifier "TimeTableIdType" (see Table 339).	The identifier of a time table can be set here if the boundary depends on the given start and end times there.
lowerBoundaryValue	Common data type "ScaledNumberType". See section 3.10.1.8.	This represents to lower end value range boundary.
upperBoundaryValue	Common data type "ScaledNumberType". See section 3.10.1.8.	This represents to upper end value range boundary.

Table 299: tierBoundaryListData function detailed description of elements

5.3.24.9.3 Available selectors

- boundaryId

5.3.24.9.4 Examples

5.3.24.9.4.1 Read-reply

If one is interested in the dynamic parts of all tier boundaries, he could send a standard read command to the server. The reply could look like this:

```
<tierBoundaryListData>
  <tierBoundaryData>
```

```

10061      <boundaryId>1</boundaryId>
10062      <timePeriod>
10063          <startTime>2018-05-06T10:00:00.0Z</startTime>
10064          <endTime>2018-05-06T20:00:00.0Z</endTime>
10065      </timePeriod>
10066      <lowerBoundaryValue>
10067          <number>0</number>
10068      </lowerBoundaryValue>
10069      <upperBoundaryValue>
10070          <number>20</number>
10071          <scale>3</scale>
10072      </upperBoundaryValue>
10073  </tierBoundaryData>
10074  <tierBoundaryData>
10075      <boundaryId>2</boundaryId>
10076      <timePeriod>
10077          <startTime>2018-05-06T20:00:00.0Z</startTime>
10078          <endTime>2018-05-07T10:00:00.0Z</endTime>
10079      </timePeriod>
10080      <lowerBoundaryValue>
10081          <number>0</number>
10082      </lowerBoundaryValue>
10083      <upperBoundaryValue>
10084          <number>20</number>
10085          <scale>3</scale>
10086      </upperBoundaryValue>
10087  </tierBoundaryData>
10088  <tierBoundaryData>
10089      <boundaryId>3</boundaryId>
10090      <timePeriod>
10091          <startTime>2018-05-06T10:00:00.0Z</startTime>
10092          <endTime>2018-05-06T20:00:00.0Z</endTime>
10093      </timePeriod>
10094      <lowerBoundaryValue>
10095          <number>20</number>
10096          <scale>3</scale>
10097      </lowerBoundaryValue>
10098  </tierBoundaryData>
10099  <tierBoundaryData>
10100      <boundaryId>4</boundaryId>
10101      <timePeriod>
10102          <startTime>2018-05-06T20:00:00.0Z</startTime>
10103          <endTime>2018-05-07T10:00:00.0Z</endTime>
10104      </timePeriod>
10105      <lowerBoundaryValue>
10106          <number>20</number>
10107          <scale>3</scale>
10108      </lowerBoundaryValue>
10109  </tierBoundaryData>
10110 </tierBoundaryListData>

```

10111

10112 **5.3.24.10tierBoundaryDescriptionListData**

10113 *5.3.24.10.1 General*

10114 The rather static part of the (tier) boundary is modelled in this function.

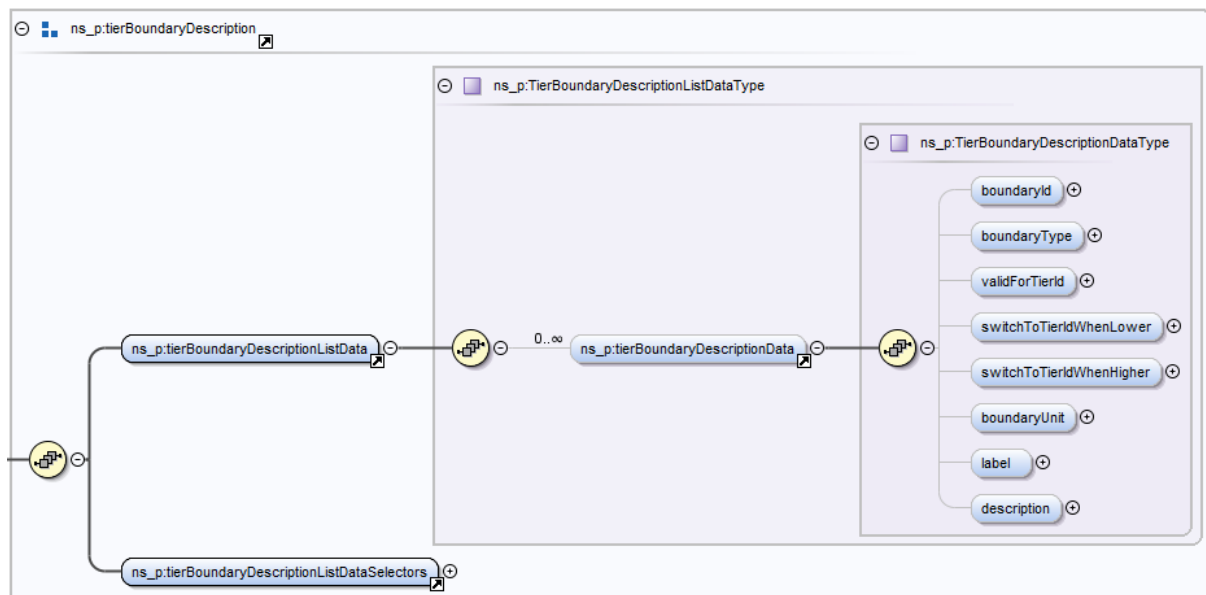


Figure 191: tierBoundaryDescriptionListData function overview

5.3.24.10.2 Detailed description of elements

Element	Type	Description
boundaryId	Identifier "TierBoundaryIdType" (see Table 339).	Primary identifier for the boundary.
boundaryType	Union "TierBoundaryTypeType": - Enum (see Table 301): TierBoundaryTypeEnumType - EnumExtendType (see section 3.10.1.5)	The type of the boundary.
validForTierId	Identifier "TierIdType" (see Table 339).	Identifier of the tier this boundary is valid for.
switchToTierIdWhenLower	Identifier "TierIdType" (see Table 339).	Identifier of the tier that will become valid if the according value falls below the element lowerBoundaryValue in function tierBoundaryListData.
switchToTierIdWhenHigher	Identifier "TierIdType" (see Table 339).	Identifier of the tier that will become valid if the according value rises above the element upperBoundaryValue in function tierBoundaryListData.
boundaryUnit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	A unit for the boundary.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the tier boundary.

description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Descriptive information on the tier boundary.
-------------	--	---

10119 *Table 300: tierBoundaryDescriptionListData function detailed description of elements*

10120 Enumeration **TierBoundaryTypeEnumType**:

Value	Description
powerBoundary	Provides boundaries for power.
energyBoundary	Provides boundaries for energy.
countBoundary	Provides boundaries for counts.

10121 *Table 301: Enumeration TierBoundaryTypeEnumType*

10122

10123 *5.3.24.10.3 Available selectors*

10124 - boundaryId

10125 - boundaryType

10126

10127 *5.3.24.10.4 Examples*

10128 *5.3.24.10.4.1 Read-reply*

10129 If one is interested in the rather static parts of all tier boundaries, he could send a standard read
10130 command to the server. The reply could look like this:

```

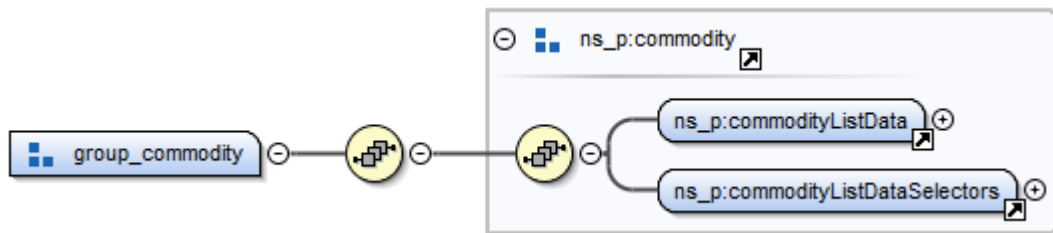
10131 <tierBoundaryDescriptionListData>
10132   <tierBoundaryDescriptionData>
10133     <boundaryId>1</boundaryId>
10134     <boundaryType>powerBoundary</boundaryType>
10135     <validForTierId>1</validForTierId>
10136     <switchToTierIdWhenHigher>3</switchToTierIdWhenHigher>
10137     <boundaryUnit>W</boundaryUnit>
10138   </tierBoundaryDescriptionData>
10139   <tierBoundaryDescriptionData>
10140     <boundaryId>2</boundaryId>
10141     <boundaryType>powerBoundary</boundaryType>
10142     <validForTierId>2</validForTierId>
10143     <switchToTierIdWhenHigher>4</switchToTierIdWhenHigher>
10144     <boundaryUnit>W</boundaryUnit>
10145   </tierBoundaryDescriptionData>
10146   <tierBoundaryDescriptionData>
10147     <boundaryId>3</boundaryId>
10148     <boundaryType>powerBoundary</boundaryType>
10149     <validForTierId>3</validForTierId>
10150     <switchToTierIdWhenLower>1</switchToTierIdWhenLower>
10151     <boundaryUnit>W</boundaryUnit>
10152   </tierBoundaryDescriptionData>
10153   <tierBoundaryDescriptionData>
10154     <boundaryId>4</boundaryId>
10155     <boundaryType>powerBoundary</boundaryType>
10156     <validForTierId>4</validForTierId>
10157     <switchToTierIdWhenLower>2</switchToTierIdWhenLower>
10158     <boundaryUnit>W</boundaryUnit>
10159   </tierBoundaryDescriptionData>
10160 </tierBoundaryDescriptionListData>

```

10161

10162 **5.3.24.11 Sub-class Commodity**

10163 Commodities (like "electricity", "water", etc.) are described within this sub-class.



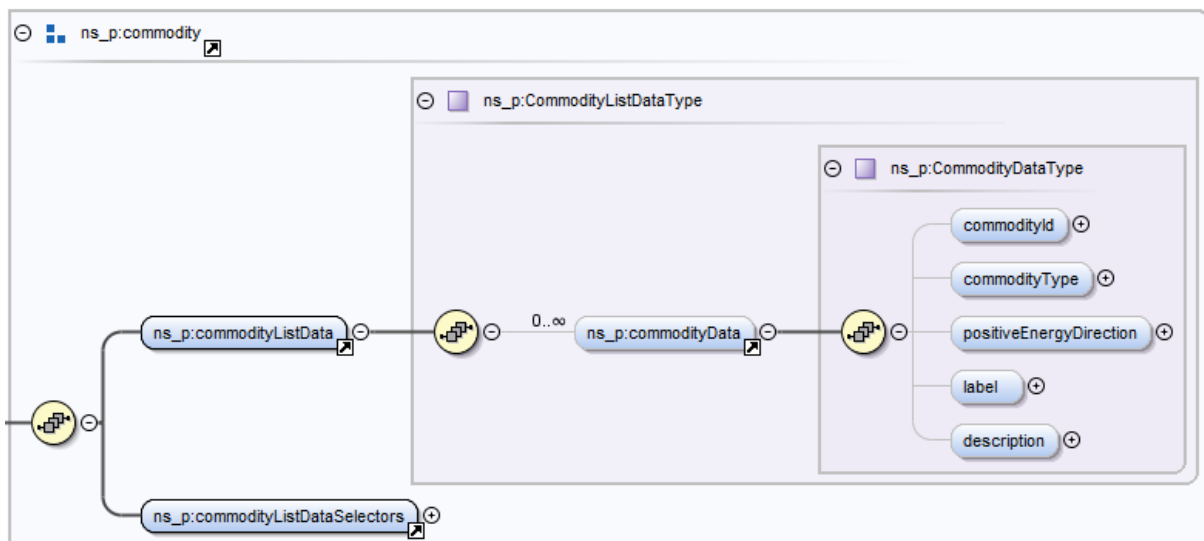
10164

10165 *Figure 192: TariffInformation, sub-class Commodity, function-group overview*

10166

10167 **5.3.24.12 commodityListData**10168 **5.3.24.12.1 General**

10169 A tariff may be linked to a commodity. The commodity information is modelled with this function.



10170

10171 *Figure 193: commodityListData function overview*

10172

10173 **5.3.24.12.2 Detailed description of elements**

Element	Type	Description
commodityId	Identifier "CommodityIdType" (see Table 339).	Primary identifier for the commodity.
commodityType	Common data type "CommodityTypeType". See section 3.10.1.9.	The type of the commodity
positiveEnergyDirection	Common data type "EnergyDirectionType". See section 3.10.1.13.	States which direction counts positives values for this commodity.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the commodity.

description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the commodity.
-------------	---	--

Table 302: commodityListData function detailed description of elements

5.3.24.12.3 Available selectors

- commodityId
- commodityType

5.3.24.12.4 Examples

5.3.24.12.4.1 Read-reply

If one is interested in the information about all commodities, he could send a standard read command to the server. The reply could look like this:

```

<commodityListData>
  <commodityData>
    <commodityId>1</commodityId>
    <commodityType>electricity</commodityType>
    <positiveEnergyDirection>consume</positiveEnergyDirection>
  </commodityData>
</commodityListData>

```

5.3.24.13 Sub-class Tier

Tiers are relevant for tariffs as they allow to different levels within the tariff (e.g. time based ("day" and "night" prices), boundary based ("the more you consume, the more you pay")).

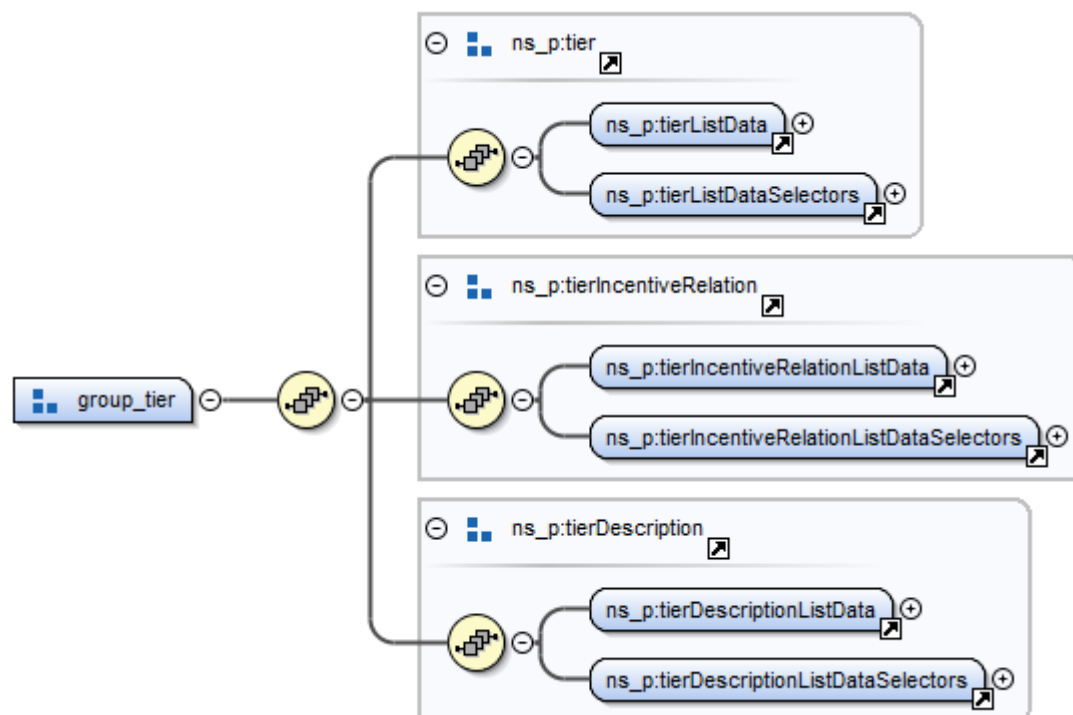
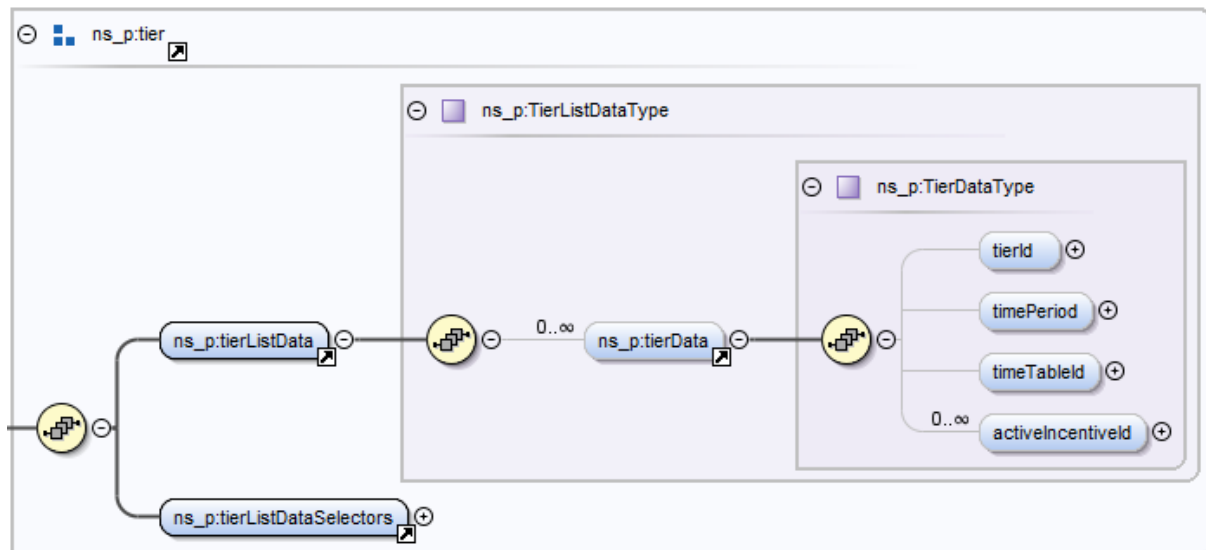


Figure 194: TariffInformation, sub-class Tier, function-group overview

10197

10198 **5.3.24.14 tierListData**10199 **5.3.24.14.1 General**

10200 The dynamic parts (time dependency and active incentives) of the tiers is described within this
 10201 function.



10202

10203 *Figure 195: tierListData function overview*

10204

10205 **5.3.24.14.2 Detailed description of elements**

Element	Type	Description
tierId	Identifier "TierIdType" (see Table 339).	Primary identifier for the tier.
timePeriod	Common data type "TimePeriodType". See section 3.10.2.1.	If the tier is only valid for some time period, this can be denoted here.
timeTableId	Identifier "TimeTableIdType" (see Table 339).	The foreign key of a time table can be set here if the tier is only valid during the given start and end times there.
activeIncentiveId (list)	Identifier "IncentiveIdType" (see Table 339).	Foreign key(s) of the currently active incentive(s) for this tier.

10206 *Table 303: tierListData function detailed description of elements*

10207

10208 **5.3.24.14.3 Available selectors**

- 10209 - tierId
- 10210 - activeIncentiveId

10211

5.3.24.14.4 Examples

5.3.24.14.4.1 Read-reply

If one is interested in the dynamic parts of all tiers, he could send a standard read command to the server. The reply could look like this:

```
<tierListData>
  <tierData>
    <tierId>1</tierId>
    <timeTableId>1</timeTableId>
    <activeIncentiveId>1</activeIncentiveId>
  </tierData>
  <tierData>
    <tierId>2</tierId>
    <timeTableId>2</timeTableId>
    <activeIncentiveId>1</activeIncentiveId>
  </tierData>
  <tierData>
    <tierId>3</tierId>
    <timeTableId>1</timeTableId>
    <activeIncentiveId>1</activeIncentiveId>
  </tierData>
  <tierData>
    <tierId>4</tierId>
    <timeTableId>2</timeTableId>
    <activeIncentiveId>1</activeIncentiveId>
  </tierData>
</tierListData>
```

5.3.24.15 tierIncentiveRelationListData

5.3.24.15.1 General

Which incentives can be used for specific tiers is modelled in the tierIncentiveRelationListData function.

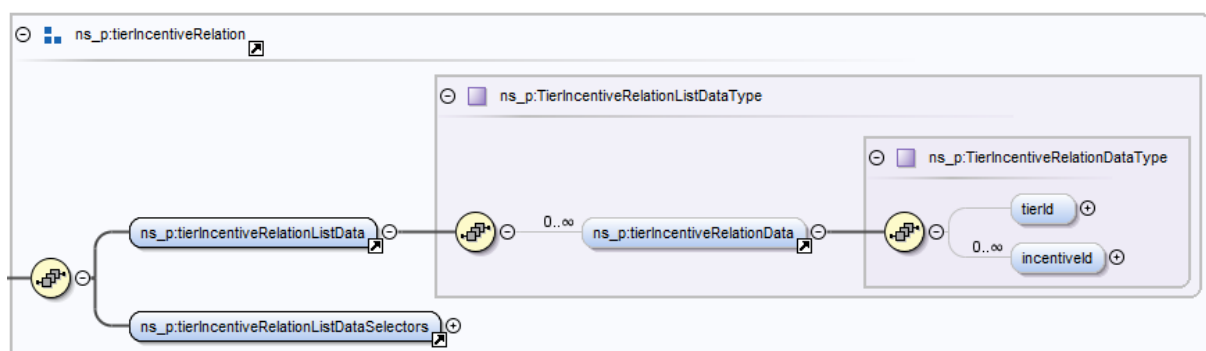


Figure 196: tierIncentiveRelationListData function overview

5.3.24.15.2 Detailed description of elements

Element	Type	Description
tierId	Identifier "TierIdType" (see Table 339).	Primary identifier for the tier.
incentiveId (list)	Identifier "IncentiveIdType" (see Table 339).	Foreign key(s) of the related incentive(s) for this tier.

10247 *Table 304: tierIncentiveRelationListData function detailed description of elements*

10248

10249 *5.3.24.15.3 Available selectors*

10250 - tierId

10251 - incentiveId

10252

10253 *5.3.24.15.4 Examples*

10254 *5.3.24.15.4.1 Read-reply*

10255 If one is interested in the possible incentives of all tiers, he could send a standard read command to
10256 the server. The reply could look like this:

```
10257 <tierIncentiveRelationListData>
10258   <tierIncentiveRelationData>
10259     <tierId>1</tierId>
10260     <incentiveId>1</incentiveId>
10261   </tierIncentiveRelationData>
10262   <tierIncentiveRelationData>
10263     <tierId>2</tierId>
10264     <incentiveId>1</incentiveId>
10265   </tierIncentiveRelationData>
10266   <tierIncentiveRelationData>
10267     <tierId>3</tierId>
10268     <incentiveId>1</incentiveId>
10269   </tierIncentiveRelationData>
10270   <tierIncentiveRelationData>
10271     <tierId>4</tierId>
10272     <incentiveId>1</incentiveId>
10273   </tierIncentiveRelationData>
10274 </tierIncentiveRelationListData>
```

10275

10276 **5.3.24.16 tierDescriptionListData**

10277 *5.3.24.16.1 General*

10278 The rather static part of the tiers is modelled with this function.

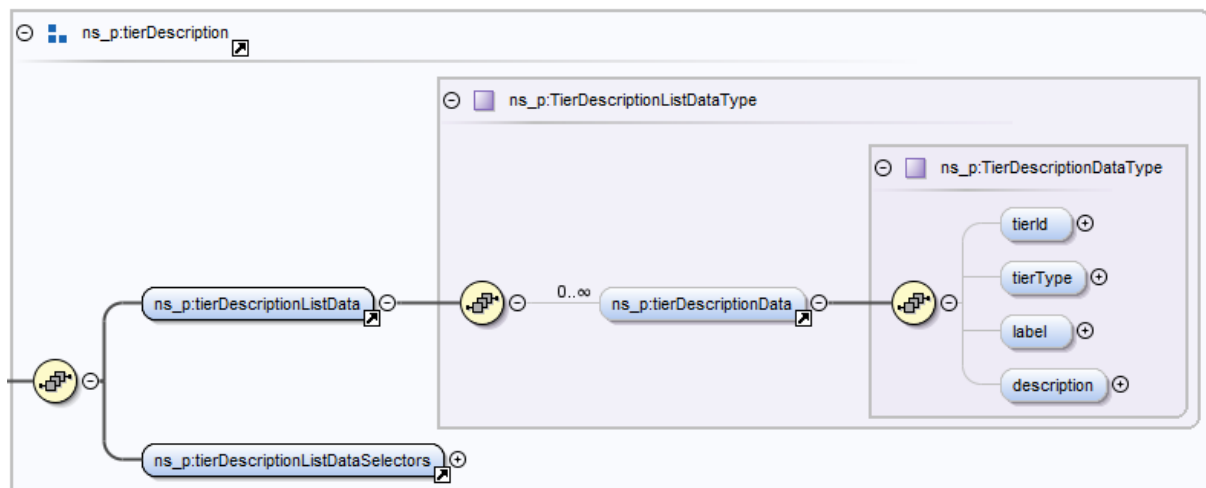


Figure 197: tierDescriptionListData function overview

5.3.24.16.2 Detailed description of elements

Element	Type	Description
tierId	Identifier "TierIdType " (see Table 339).	Primary identifier for the tier.
tierType	Union "TierTypeType": - Enum (see Table 306): TierTypeEnumType - EnumExtendType (see section 3.10.1.5)	The type of the tier.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the tier.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the tier.

Table 305: tierDescriptionListData function detailed description of elements

Enumeration TierTypeEnumType:

Value	Description
fixedCost	The fixed cost have to be payed, independent from the consumed energy.
dynamicCost	The dynamic cost has a (linear) dependency to the consumed energy.

Table 306: Enumeration TierTypeEnumType

5.3.24.16.3 Available selectors

- tierId
- tierType

5.3.24.16.4 Examples

5.3.24.16.4.1 Read-reply

If one is interested in the rather static parts of all tiers, he could send a standard read command to the server. The reply could look like this:

```
<tierDescriptionListData>
```

```

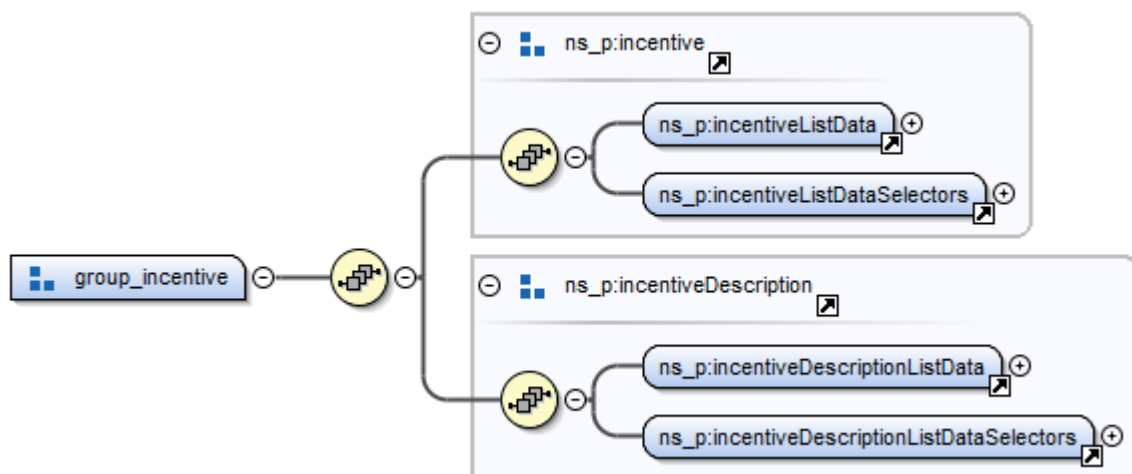
10296     <tierDescriptionData>
10297         <tierId>1</tierId>
10298         <tierType>dynamicCost</tierType>
10299     </tierDescriptionData>
10300     <tierDescriptionData>
10301         <tierId>2</tierId>
10302         <tierType>dynamicCost</tierType>
10303     </tierDescriptionData>
10304     <tierDescriptionData>
10305         <tierId>3</tierId>
10306         <tierType>dynamicCost</tierType>
10307     </tierDescriptionData>
10308     <tierDescriptionData>
10309         <tierId>4</tierId>
10310         <tierType>dynamicCost</tierType>
10311     </tierDescriptionData>
10312 </tierDescriptionListData>

```

10313

10314 **5.3.24.17 Sub-class Incentive**

10315 The actual costs are modelled as so-called incentives (e.g. real monetary costs, "co2Emission", etc.).



10316

10317 *Figure 198: TariffInformation, sub-class Incentive, function-group overview*

10318

10319 **5.3.24.18 incentiveListData**

10320 **5.3.24.18.1 General**

10321 The value of the costs, together with a time dependency (if needed), is part of this dynamic incentive
 10322 function.

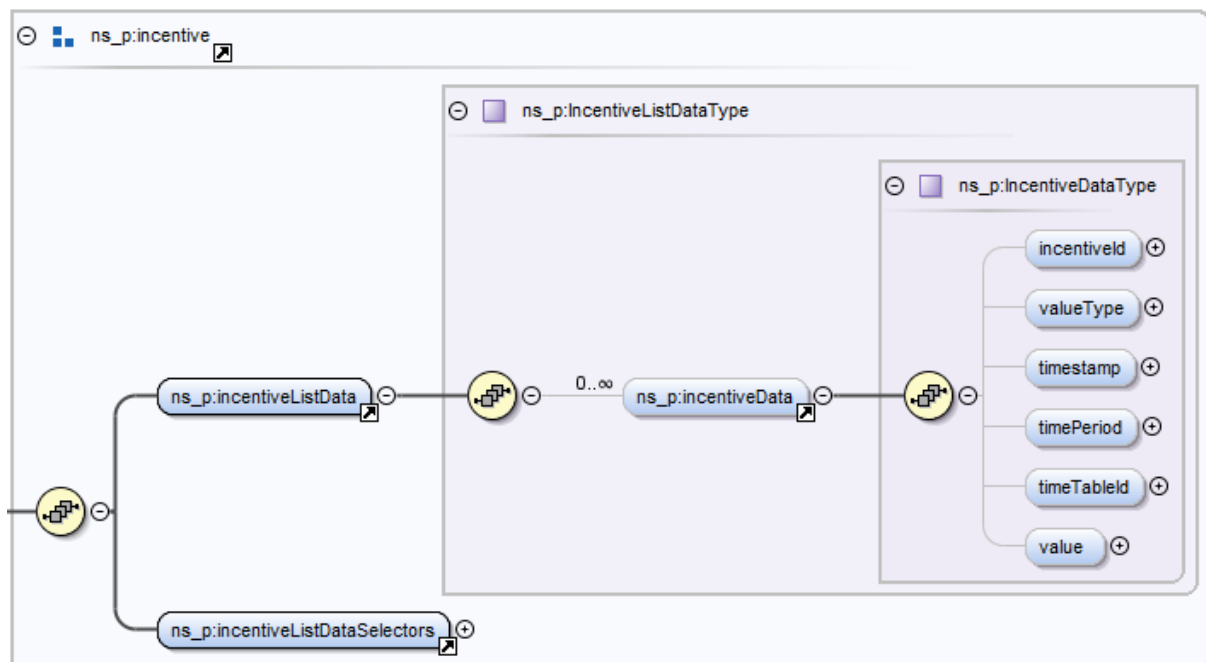


Figure 199: incentiveListData function overview

5.3.24.18.2 Detailed description of elements

Element	Type	Description
incentiveld	Identifier "IncentiveldType" (see Table 339).	Primary identifier for the incentive.
valueType	Union "IncentiveValueTypeType": - Enum (see Table 308): IncentiveValueTypeEnumType - EnumExtendType (see section 3.10.1.5)	A type for the value (e.g. "value", "minValue", etc.).
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	A timestamp for the value information.
timePeriod	Common data type "TimePeriodType". See section 3.10.2.1.	If there exists a time dependency for which the incentive value is valid, this can be denoted here.
timeTableId	Identifier "TimeTableIdType" (see Table 339).	The foreign key of a time table can be set here if the incentive value depends on the given start and end times there.
value	Common data type "ScaledNumberType". See section 3.10.1.8.	The actual value for the incentive costs.

Table 307: incentiveListData function detailed description of elements

Enumeration IncentiveValueTypeEnumType:

Value	Description
value	Used for normal cost values for an incentive.
averageValue	Used if only an average cost value can be stated for an incentive.
minValue	Used for the minimum cost value of an incentive.
maxValue	Used for the maximum cost value of an incentive.

10329 *Table 308: Enumeration IncentiveValueTypeEnumType*

10330

10331 *5.3.24.18.3 Available selectors*

10332 - incentiveId

10333 - timestamp

10334 - timestampInterval

10335

10336 *5.3.24.18.4 Examples*

10337 *5.3.24.18.4.1 Read-reply*

10338 If one is interested in the dynamic parts of all incentives, he could send a standard read command to
10339 the server. The reply could look like this:

```
10340 <incentiveListData>
10341   <incentiveData>
10342     <incentiveId>1</incentiveId>
10343     <valueType>value</valueType>
10344     <timestamp>2018-05-06T09:00:00.0Z</timestamp>
10345     <timePeriod>
10346       <startTime>2018-05-06T10:00:00.0Z</startTime>
10347       <endTime>2018-05-07T10:00:00.0Z</endTime>
10348     </timePeriod>
10349     <value>
10350       <number>28</number>
10351       <scale>-2</scale>
10352     </value>
10353   </incentiveData>
10354 </incentiveListData>
```

10355

10356 ***5.3.24.19 incentiveDescriptionListData***

10357 *5.3.24.19.1 General*

10358 The rather static information about the incentives is modelled with this function.

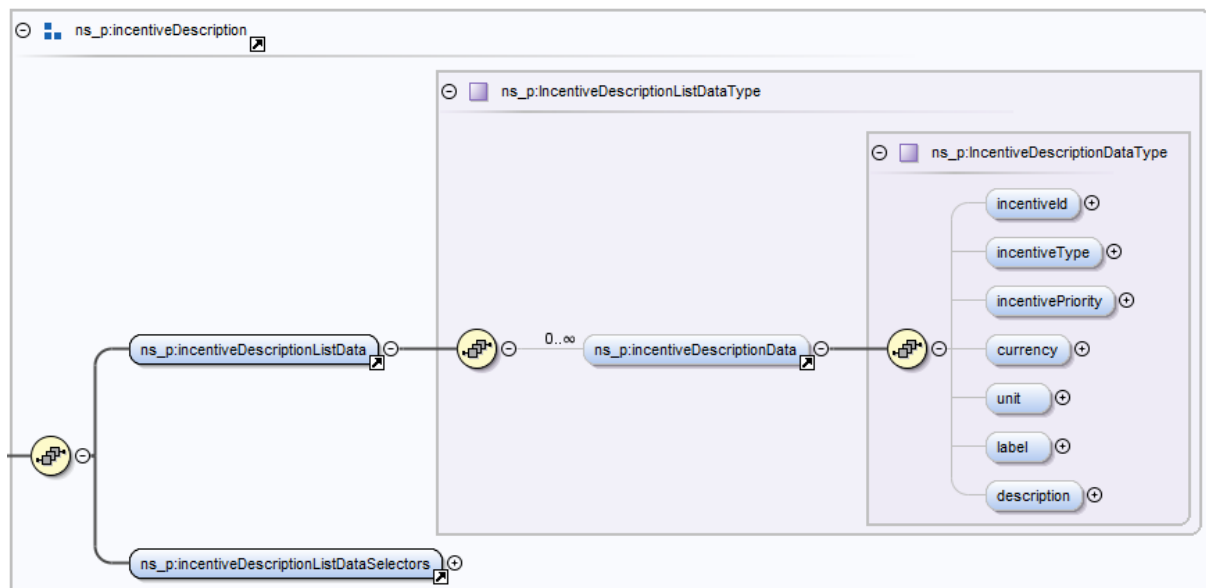


Figure 200: incentiveDescriptionListData function overview

5.3.24.19.2 Detailed description of elements

Element	Type	Description
incentiveld	Identifier "IncentiveldType" (see Table 339).	Primary identifier for the incentive.
incentiveType	Union "IncentiveTypeType": - Enum (see Table 310): IncentiveTypeEnumType - EnumExtendType (see section 3.10.1.5)	The type of the incentive.
incentivePriority	Simple type "IncentivePriorityType" (restriction of xs:unsignedInt)	A priority can be stated for an incentive.
currency	Common data type "CurrencyType". See section 3.10.1.19.	The currency for the incentive costs.
unit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	A unit for the measured commodity this incentives models costs for.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the incentive.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the incentive.

Table 309: incentiveDescriptionListData function detailed description of elements

Enumeration **IncentiveTypeEnumType**:

Value	Description
absoluteCost	Absolute costs with a currency and a unit. The total costs can be calculated.
relativeCost	Relative costs that cannot be interpreted as some monetary costs.
renewableEnergyPercentage	Percentage of renewable energy of the total obtained energy. This is an ecological incentive that allows optimizing consumption of renewable energy instead of costs.
co2Emission	CO2 emission related to the obtained energy. This is an ecological incentive that allows optimizing the CO2 emission instead of costs.

10365 *Table 310: Enumeration IncentiveTypeEnumType*

10366

10367 *5.3.24.19.3 Available selectors*

10368 - incentiveId

10369 - incentiveType

10370

10371 *5.3.24.19.4 Examples*

10372 *5.3.24.19.4.1 Read-reply*

10373 If one is interested in the rather static parts of all incentives, he could send a standard read
10374 command to the server. The reply could look like this:

```
10375 <incentiveDescriptionListData>
10376   <incentiveDescriptionData>
10377     <incentiveId>1</incentiveId>
10378     <incentiveType>absoluteCost</incentiveType>
10379     <incentivePriority>1</incentivePriority>
10380     <currency>EUR</currency>
10381     <unit>Wh</unit>
10382   </incentiveDescriptionData>
10383 </incentiveDescriptionListData>
```

10384

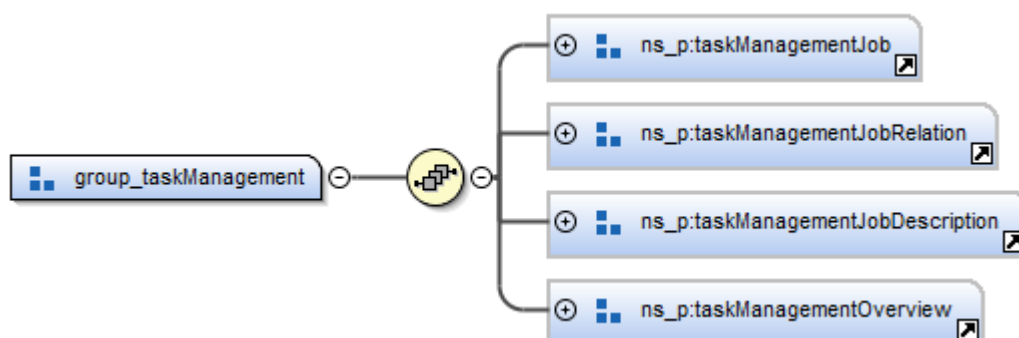
10385 5.3.25 TaskManagement

10386 5.3.25.1 Introduction

10387 The TaskManagement Class lists all "tasks" that are currently relevant on an entity. This enables one
10388 to have a quick look on the tasks of related classes, but not for their modification. Further
10389 information has to be gathered with the corresponding classes supporting the specific functionality.

10390 In this version of the specification, tasks can be "jobs" like in DirectControl, PowerSequences /
10391 SmartEnergyManagementPs, etc. (see the related function groups below).

10392 Remark: Future versions of this specification may add other kinds of tasks (i.e. tasks that are not
10393 covered by "jobs").



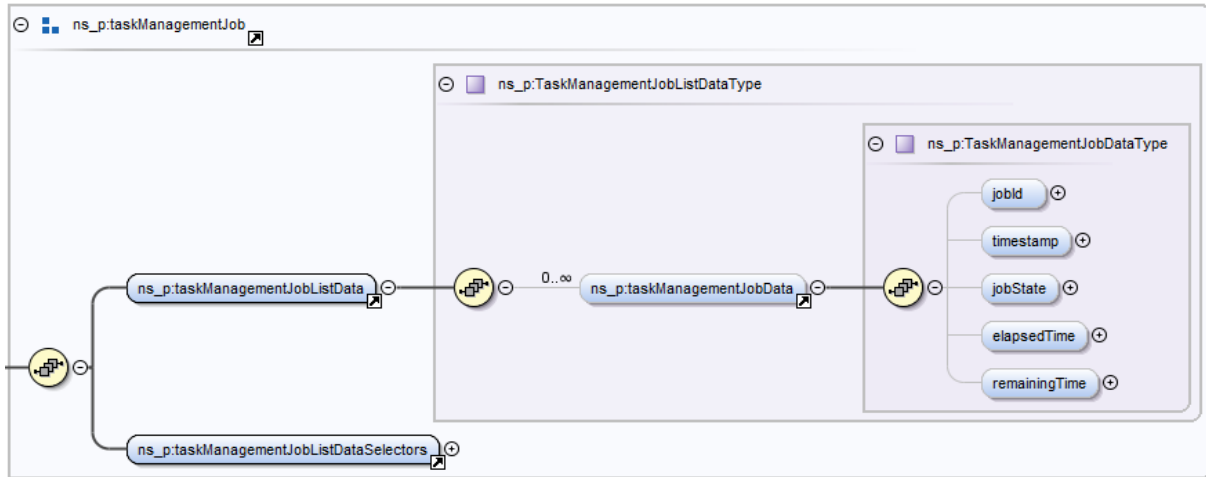
10394

10395 *Figure 201: TaskManagement function-group overview*

10396

10397 **5.3.25.2 taskManagementJobListData**10398 **5.3.25.2.1 General**

10399 All jobs that are available on the entity are listed here, together with their state and (if available) the
 10400 elapsed and remaining time.



10401
 10402 *Figure 202: taskManagementJobListData function overview*

10403

10404 **5.3.25.2.2 Detailed description of elements**

Element	Type	Description
jobId	Identifier "TaskManagementJobIdType" (see Table 339).	The <i>jobId</i> is used for indentifying different job information collected by this class.
timestamp	Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.	Point in time, this information was generated.
jobState	Union "TaskManagementJobStateType": - Enum (see Table 196): DirectControlActivityStateEnumType - Enum (see Table 215): HvacOverrunStatusEnumType - Enum (see Table 224): LoadControlEventStateEnumType - Enum (see Table 269): PowerSequenceStateEnumType - EnumExtendType (see section 3.10.1.5)	Current state of the job.
elapsedTime	xs:duration (W3C standard type)	The duration, this job was already active (normally, pause times are not counted).
remainingTime	xs:duration (W3C standard type)	The duration, this job will most likely be active before it ends.

10405 *Table 311: taskManagementJobListData function detailed description of elements*

10406

10407 **5.3.25.2.3 Available selectors**

10408 - jobId

10409 - jobState

10410

10411 5.3.25.2.4 Examples

10412 5.3.25.2.4.1 Read-reply

10413 If one wants to request information about a specific job (jobId = "2"), he could sent the following
10414 read command:

```
10415 <function>taskManagementJobListData</function>
10416 <filter>
10417     <cmdControl>
10418         <partial/>
10419     </cmdControl>
10420     <taskManagementJobListDataSelectors>
10421         <jobId>2</jobId>
10422     </taskManagementJobListDataSelectors>
10423 </filter>
10424 <taskManagementJobListData/>
```

10425 A response could be:

```
10426 <function>taskManagementJobListData</function>
10427 <filter>
10428     <cmdControl>
10429         <partial/>
10430     </cmdControl>
10431 </filter>
10432 <taskManagementJobListData>
10433     <taskManagementJobData>
10434         <jobId>2</jobId>
10435         <timestamp>2015-11-18T14:59:23.3Z</timestamp>
10436         <jobState>running</jobState>
10437         <elapsedTime>PT2H8M32S</elapsedTime>
10438         <remainingTime>PT21M28S</remainingTime>
10439     </taskManagementJobData>
10440 </taskManagementJobListData>
```

10441

10442 5.3.25.3 taskManagementJobRelationListData

10443 5.3.25.3.1 General

10444 To identify where a job originates from, the *taskManagementJobRelationListData* function is used. If
10445 the related class allows more than one job the corresponding identifier(s) are denoted.

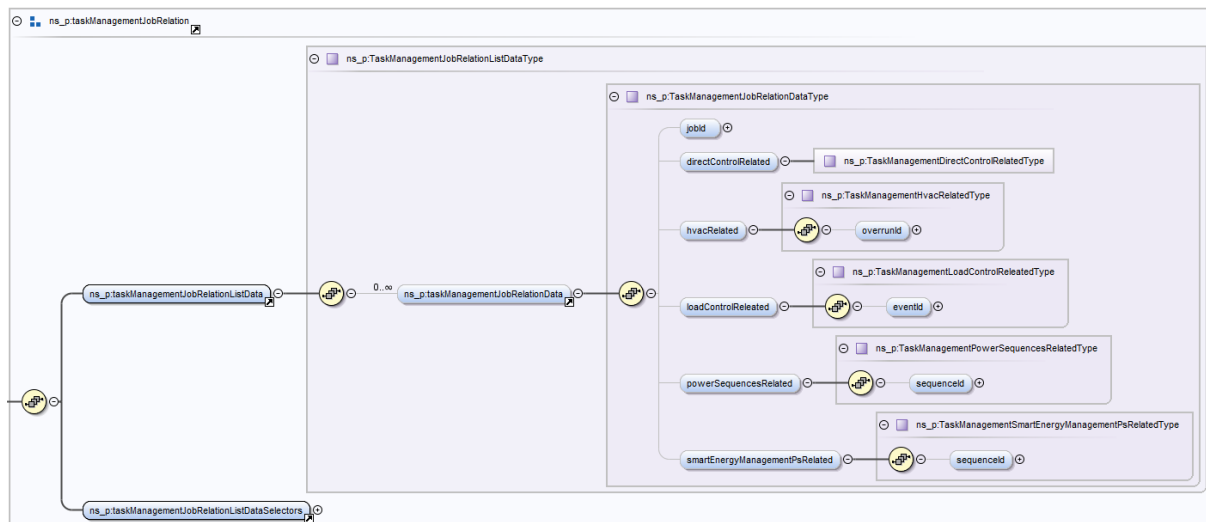


Figure 203: taskManagementJobRelationListData function overview

5.3.25.3.2 Detailed description of elements

Element	Type	Description
jobId	Identifier "TaskManagementJobIdType" (see Table 339).	The <i>jobId</i> is used for indentifying different job information collected by this class.
directControlRelated	ComplexType "TaskManagementDirectControlRelatedType"	Flag that indicates that the job is related to DirectControl.
hvacRelated	ComplexType "TaskManagementHvacRelatedType"	Flag that indicates that the job is related to HVAC.
hvacRelated. overrunId	Identifier "HvacOverrunIdType" (see Table 339).	Identifier for the related overrun.
loadControlRelated	ComplexType "TaskManagementLoadControlRelatedType"	Flag that indicates that the job is related to LoadControl.
loadControlRelated. eventId	Identifier "LoadControlEventIdType" (see Table 339).	Identifier for the related load control event.
powerSequencesRelated	ComplexType "TaskManagementPowerSequencesRelatedType"	Flag that indicates that the job is related to PowerSequences.
powerSequencesRelated. sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	Identifier for the related power sequence.
smartEnergyManagementPs Related	ComplexType "TaskManagementSmartEnergyManagementPsRelatedType"	Flag that indicates that the job is related to SmartEnergyManagementPs.
smartEnergyManagementPs Related. sequenceId	Identifier "PowerSequenceIdType" (see Table 339).	Identifier for the related power sequence.

Table 312: taskManagementJobRelationListData function detailed description of elements

5.3.25.3.3 Available selectors

- jobId

5.3.25.3.4 Examples

5.3.25.3.4.1 Read-reply

If one requests all job relations, the reply could look like the following:

```
<taskManagementJobRelationListData
  <taskManagementJobRelationData>
    <jobId>1</jobId>
    <hvacRelated>
      <overrunId>3</overrunId>
    </hvacRelated>
  </taskManagementJobRelationData>
  <taskManagementJobRelationData>
    <jobId>2</jobId>
    <directControlRelated/>
  </taskManagementJobRelationData>
</taskManagementJobRelationListData>
```

5.3.25.4 taskManagementJobDescriptionListData

5.3.25.4.1 General

The more static information about the jobs are communicated with this function.

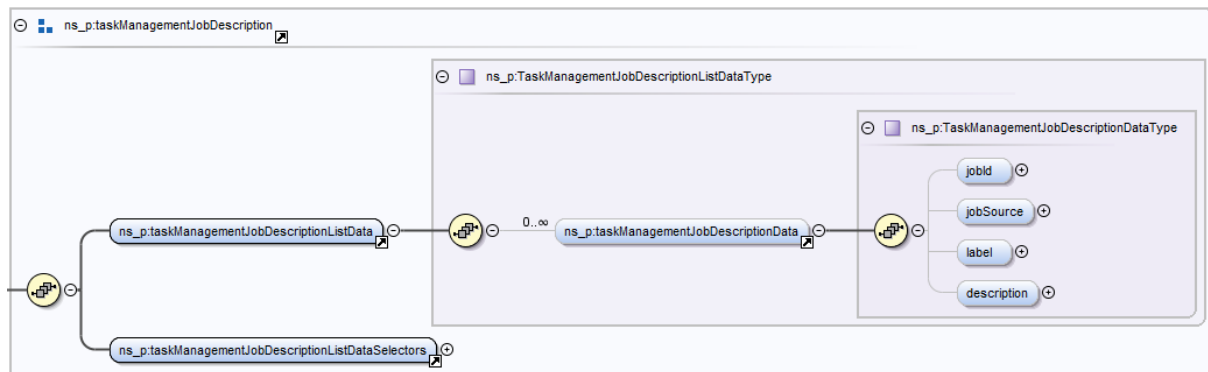


Figure 204: taskManagementJobDescriptionListData function overview

5.3.25.4.2 Detailed description of elements

Element	Type	Description
jobId	Identifier "TaskManagementJobIdType" (see Table 339).	The <i>jobId</i> is used for indentifying different job information collected by this class.
jobSource	Union "TaskManagementJobSourceType": - Enum (see Table 314): TaskManagementJobSourceEnumType - EnumExtendType (see section 3.10.1.5)	Specifies what kind of reason caused this job.

label	<i>Common data type "LabelType". See section 3.10.1.2.</i>	A short label of the job.
description	<i>Common data type "DescriptionType". See section 3.10.1.3.</i>	Descriptive information on the job.

Table 313: *taskManagementJobDescriptionListData* function detailed description of elements

Enumeration **TaskManagementJobSourceEnumType**:

Value	Description
internalMechanism	The job was started due to some device internal mechanism (e.g. a regular cleaning cycle).
userInteraction	The user activated some functionality directly via the UI of the device.
externalConfiguration	The job started, because it was triggered from some external device (e.g. an energy manager).

Table 314: Enumeration *TaskManagementJobSourceEnumType*

5.3.25.4.3 Available selectors

- jobId
- jobSource

5.3.25.4.4 Examples

5.3.25.4.4.1 Read-reply

If one requests all job descriptions, the reply could look like the following:

```
<taskManagementJobDescriptionListData>
  <taskManagementJobDescriptionData>
    <jobId>1</jobId>
    <jobSource>userInteraction</jobSource>
  </taskManagementJobDescriptionData>
  <taskManagementJobDescriptionData>
    <jobId>2</jobId>
    <jobSource>externalConfiguration</jobSource>
  </taskManagementJobDescriptionData>
</taskManagementJobDescriptionListData>
```

5.3.25.5 taskManagementOverviewData

5.3.25.5.1 General

A general overview is given with this function. It specifies whether a remote controlling is permitted at that moment and if currently jobs are active.

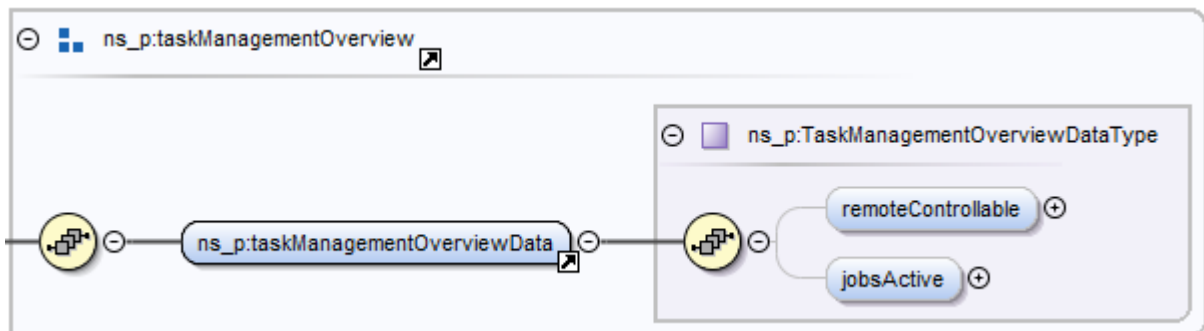


Figure 205: taskManagementOverviewData function overview

5.3.25.5.2 Detailed description of elements

Element	Type	Description
remoteControllable	xs:boolean (W3C standard type)	States if remote controlling of the device or entity is permitted.
jobsActive	xs:boolean (W3C standard type)	Denotes whether jobs are active on the device or entity or not.

Table 315: taskManagementOverviewData function detailed description of elements

5.3.25.5.3 Available selectors

None.

5.3.25.5.4 Examples

5.3.25.5.4.1 Read-reply

If one requests the task management overview, a response could be:

```
<taskManagementOverviewData>
  <remoteControllable>true</remoteControllable>
  <jobsActive>true</jobsActive>
</taskManagementOverviewData>
```

5.3.26 Threshold

5.3.26.1 Introduction

Thresholds are helpful in several kind of use cases. E.g. a device or application can be configured to do some specific behaviour if a measured value is above or below a configured threshold. Some measurands need to stay in a certain value range, to ensure proper and safe working of devices. Automatic alarms after reaching a threshold help reacting on the occurrence immediately.

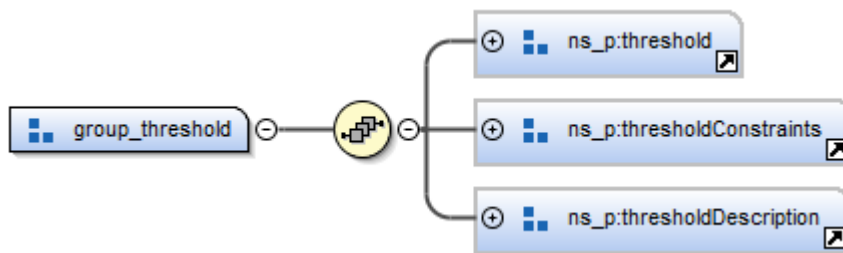


Figure 206: Threshold function-group overview

5.3.26.2 thresholdListData

5.3.26.2.1 General

Each threshold (identified by its *thresholdId*) has a configured value. This function is used to read or write the actual value.

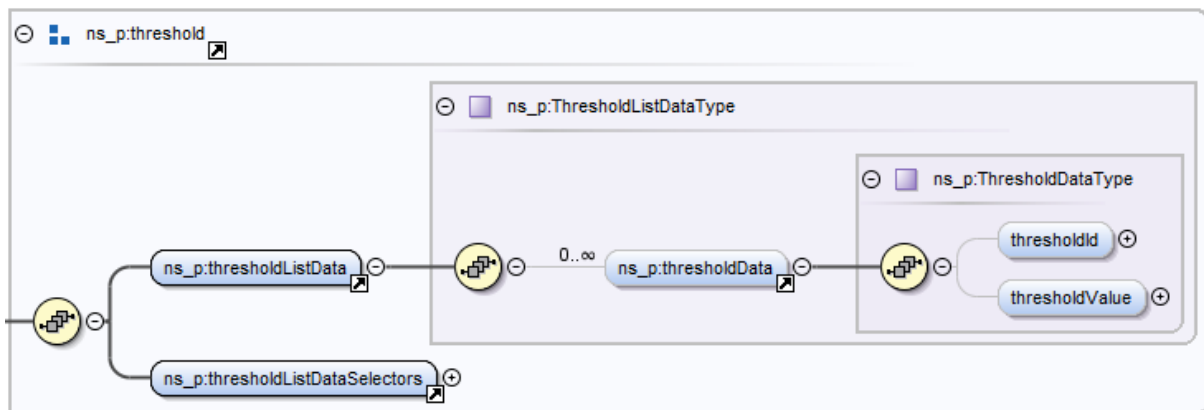


Figure 207: thresholdListData function overview

5.3.26.2.2 Detailed description of elements

Element	Type	Description
thresholdId	Identifier "ThresholdIdType" (see Table 339).	Identifier for the threshold.
thresholdValue	Common data type "ScaledNumberType". See section 3.10.1.8.	Value that should not be fallen below or exceeded (depends on the <i>thresholdType</i> , see section 5.3.26.4.2).

Table 316: thresholdListData function detailed description of elements

5.3.26.2.3 Available selectors

- thresholdId

5.3.26.2.4 Examples

5.3.26.2.4.1 Read-reply

If one is interested in all thresholds, he could send the a standard read command to the device. The device could respond with something like this:

```
<thresholdListData>
  <thresholdData>
    <thresholdId>1</thresholdId>
    <thresholdValue>
      <number>155</number>
      <scale>-1</scale>
    </thresholdValue>
  </thresholdData>
  <thresholdData>
    <thresholdId>2</thresholdId>
    <thresholdValue>
      <number>23</number>
    </thresholdValue>
  </thresholdData>
</thresholdListData>
```

5.3.26.3 thresholdConstraintsListData

5.3.26.3.1 General

The value range from which a threshold value configuration may be chosen from may be restricted by a minimum and a maximum value. Between these, not every value, but only ones with a specific gap, may be used, specified with the step size.

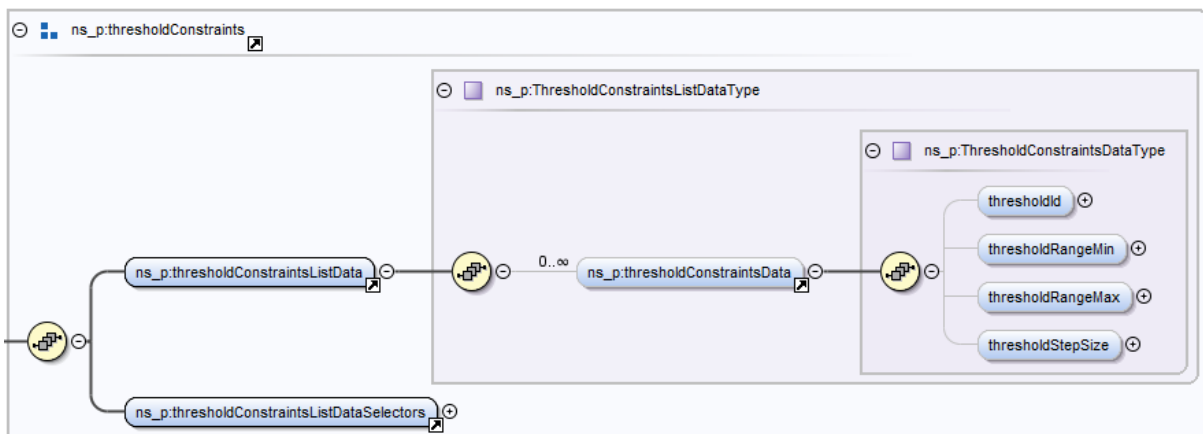


Figure 208: thresholdConstraintsListData function overview

5.3.26.3.2 Detailed description of elements

Element	Type	Description
thresholdId	Identifier "ThresholdIdType" (see Table 339).	Identifier for the threshold.
thresholdRangeMin	Common data type "ScaledNumberType". See section 3.10.1.8.	Minumum value that may be used to configure the element <i>thresholdValue</i> (see section 5.3.26.2.2).

thresholdRangeMax	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Maximum value that may be used to configure the element <i>thresholdValue</i> (see section 5.3.26.2.2).
thresholdStepSize	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Step size for the configuration of the element <i>thresholdValue</i> (see section 5.3.26.2.2).

Table 317: *thresholdConstraintsListData* function detailed description of elements

5.3.26.3.3 Available selectors

- thresholdId

5.3.26.3.4 Examples

5.3.26.3.4.1 Read-reply

If one is interested in the constraints of all thresholds, he could send the a standard read command to the device. The device could respond with something like this:

```
<thresholdConstraintsListData>
  <thresholdConstraintsData>
    <thresholdId>1</thresholdId>
    <thresholdRangeMin>
      <number>5</number>
    </thresholdRangeMin>
    <thresholdRangeMax>
      <number>35</number>
    </thresholdRangeMax>
    <thresholdStepSize>
      <number>5</number>
      <scale>-1</scale>
    </thresholdStepSize>
  </thresholdConstraintsData>
  <thresholdConstraintsData>
    <thresholdId>2</thresholdId>
    <thresholdRangeMin>
      <number>5</number>
    </thresholdRangeMin>
    <thresholdRangeMax>
      <number>35</number>
    </thresholdRangeMax>
    <thresholdStepSize>
      <number>1</number>
    </thresholdStepSize>
  </thresholdConstraintsData>
</thresholdConstraintsListData>
```

5.3.26.4 thresholdDescriptionListData

5.3.26.4.1 General

The rather static information about thresholds is defined within this function.

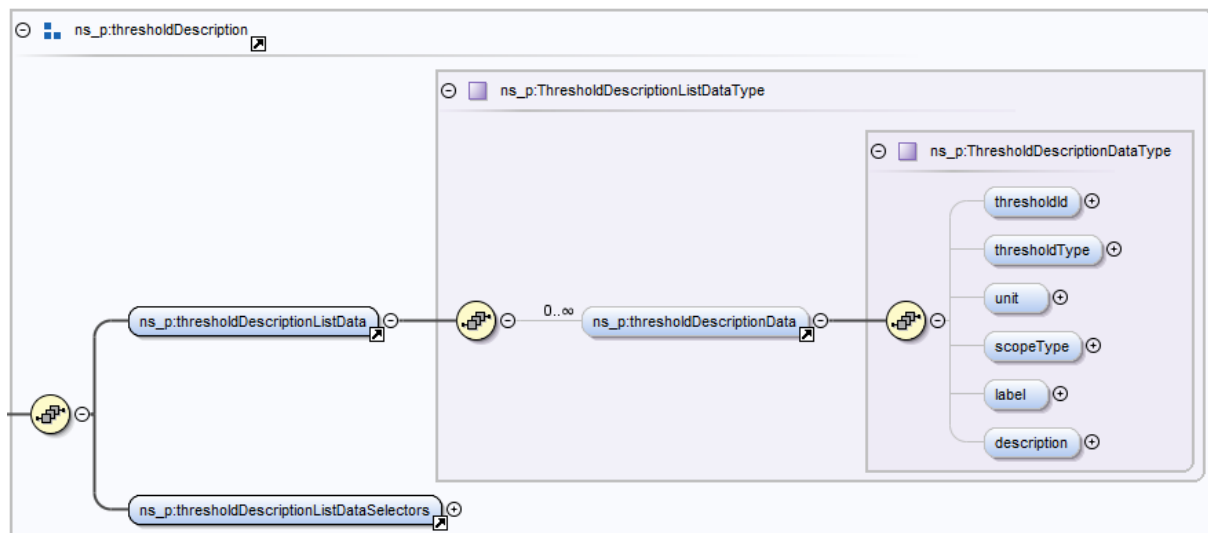


Figure 209: thresholdDescriptionListData function overview

5.3.26.4.2 Detailed description of elements

Element	Type	Description
thresholdId	Identifier "ThresholdIdType" (see Table 339).	Identifier for the threshold.
thresholdType	Union "ThresholdTypeType": - Enum (see Table 319): ThresholdTypeEnumType - EnumExtendType (see section 3.10.1.5)	The type of the treshold.
unit	Common data type "UnitOfMeasurementType". See section 3.10.1.17.	The unit in which the value of the threshold is denoted.
scopeType	Common data type "ScopeTypeType". See section 3.10.1.21.	Specifies a more detailed meaning of the threshold.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the threshold.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the threshold.

Table 318: thresholdDescriptionListData function detailed description of elements

Enumeration ThresholdTypeEnumType:

Value	Description
goodAbove	A positive reaction will follow after the value exceeds the threshold.
badAbove	A negative reaction will follow after the value exceeds the threshold.
goodBelow	A positive reaction will follow after the value falls below the threshold.
badBelow	A negative reaction will follow after the value falls below the threshold.

minValueThreshold	If this minimum threshold is underrun, some warning will be activated.
maxValueThreshold	If this maximum threshold is exceeded, some warning will be activated.
minValueThresholdExtreme	If this extreme minimum threshold is underrun, some important warning will be activated.
maxValueThresholdExtreme	If this extreme maximum threshold is exceeded, some important warning will be activated.
sagThreshold	If this sag threshold is underrun, some warning will be activated. Commonly used for electricity voltage sags.
swellThreshold	If this swell threshold is exceeded, some warning will be activated. Commonly used for electricity voltage swells.

Table 319: Enumeration ThresholdTypeEnumType

5.3.26.4.3 Available selectors

- thresholdId
- scopeType

5.3.26.4.4 Examples

5.3.26.4.4.1 Read-reply

If one is interested in the description of all thresholds, he could send the a standard read command to the device. The device could respond with something like this:

```
<thresholdDescriptionListData>
  <thresholdDescriptionData>
    <thresholdId>1</thresholdId>
    <thresholdType>badBelow</thresholdType>
    <unit>degC</unit>
    <label>Minimum temperature</label>
  </thresholdDescriptionData>
  <thresholdDescriptionData>
    <thresholdId>2</thresholdId>
    <thresholdType>badAbove</thresholdType>
    <unit>degC</unit>
    <label>Maximum temperature</label>
  </thresholdDescriptionData>
</thresholdDescriptionListData>
```

5.3.27 TimeInformation

5.3.27.1 Introduction

The exchange of time information is important for most smart grid and home automation systems. Please note the underlying revision of the SPINE data model does not specify a sophisticated synchronisation mechanism that includes a given precision. This is due to the complex of problems appearing with communication delays in the different technologies and hardware platforms. However, an application could measure all relevant delays and take care of a real synchronisation in the network using the *TimeInformation* class.

Independent of any synchronisation aspect the "owner concept" applies here as well: The "owner" of any given data (a measured value of a sensor, e.g.) applies its own SPINE time to its own data. This is relevant for the case of a substantial difference between the SPINE time of two nodes.

In order to avoid problems from such differences a typical sequence could be that a new device sends a *timeDistributorEnquiryCall* (with the *isTimeDistributor* element set to true) to all devices in the SPINE network. All time distributors would send a *timeDistributorData* instance as notification to the device. The new device would pick the time distributor with the highest priority and send a read command to the *timeInformationData* function of this distributor. The time distributor would reply with a *timeInformationData* containing its current time.

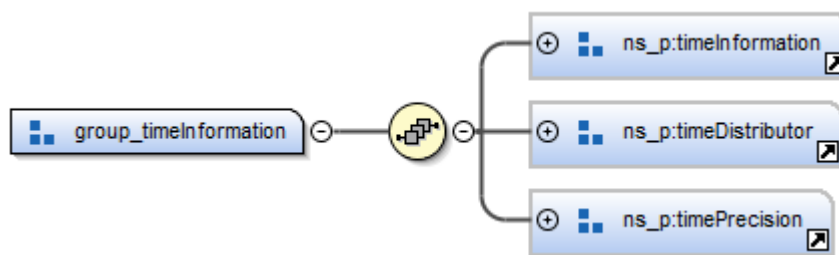


Figure 210: TimeInformation function-group overview (data)

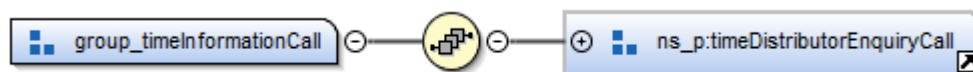


Figure 211: TimeInformation function-group overview (call)

5.3.27.2 timeInformationData

5.3.27.2.1 General

The *timeInformationData* command is used for exchanging the SPINE time.

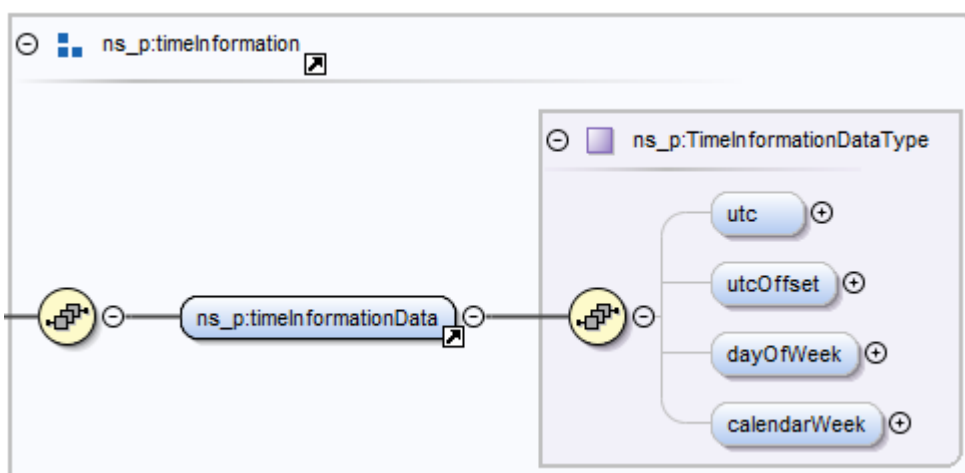


Figure 212: timeInformationData function overview

5.3.27.2.2 Detailed description of elements

Element	Type	Description
---------	------	-------------

utc	xs:dateTime (W3C standard type)	The current time in UTC.
utcOffset	xs:duration (W3C standard type)	The configured offset to UTC (e.g. PT2H [= +2 hours] for Central European Summer Time (CEST)).
dayOfWeek	<i>Common data type "DayOfWeekType". See section 3.10.2.9.</i>	To model the actual day of week (Monday to Sunday) this element is used. It refers to the local time. If there is no utcOffset given, it is assumed, that the local time is equivalent to UTC-time.
calendarWeek	<i>Common data type "CalendarWeekType". See section 3.10.2.8.</i>	A value between 1 and 53 stating the current calendar week number. This element refers to the local time. If there is no utcOffset given, it is assumed, that the local time is equivalent to UTC-time.

Table 320: *timeInformationData* function detailed description of elements

5.3.27.2.3 Available selectors

None.

5.3.27.2.4 Examples

5.3.27.2.4.1 Read-reply

A *timeInformationData* command sent as reply to a read command on *timeInformationData*:

```
<timeInformationData>
  <utc>2013-07-15T12:00:00.0Z</utc>
  <utcOffset>PT2H</utcOffset>
  <dayOfWeek>tuesday</dayOfWeek>
  <calendarWeek>17</calendarWeek>
</timeInformationData>
```

5.3.27.3 timeDistributorData

5.3.27.3.1 General

One or some devices in a SPINE network should act as time distributor. The information thereof and the dedicated priority can be modelled with the *timeDistributorData* command.

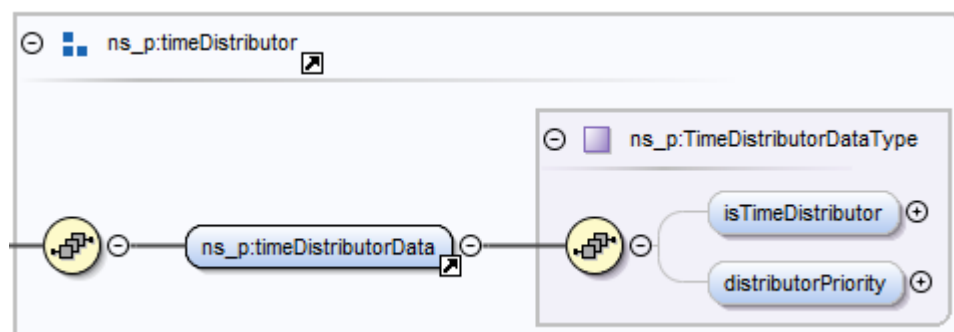


Figure 213: *timeDistributorData* function overview

10693

10694 **5.3.27.3.2 Detailed description of elements**

Element	Type	Description
isTimeDistributor	xs:boolean (W3C standard type)	Denotes whether a device can act as a time distributor in the SPINE network or not.
distributorPriority	xs:unsignedInt (W3C standard type)	A (configured) priority of the time distributor.

10695 *Table 321: timeDistributorData function detailed description of elements*

10696

10697 **5.3.27.3.3 Available selectors**

10698 None.

10699

10700 **5.3.27.3.4 Examples**10701 **5.3.27.3.4.1 Notify**

10702 Example of a notification from a device, stating that it can act as a time distributor with a priority of
 10703 "2":

```

10704 <timeDistributorData>
10705   <isTimeDistributor>true</isTimeDistributor>
10706   <distributorPriority>2</distributorPriority>
10707 </timeDistributorData>

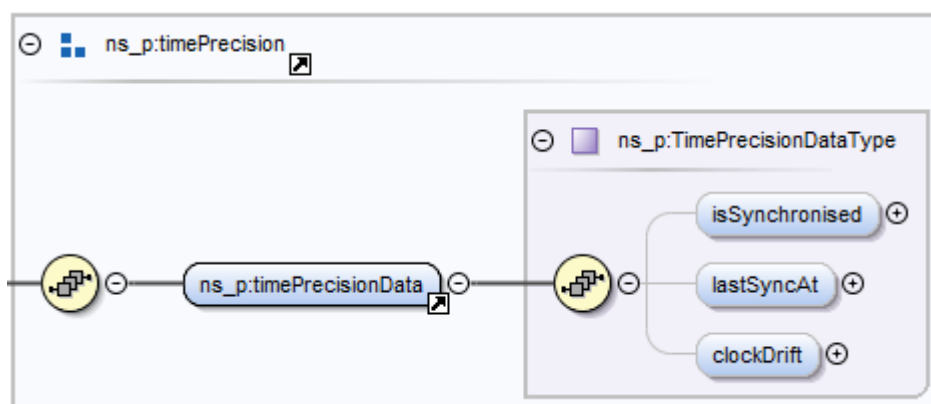
```

10708 Such notification could typically be send from a device that previously received a
 10709 *timeDistributorEnquiryCall*.

10710

10711 **5.3.27.4 timePrecisionData**10712 **5.3.27.4.1 General**

10713 Information regarding the precision of the time of a device can be modelled using *timePrecisionData*.



10714

10715 *Figure 214: timePrecisionData function overview*

10716

10717 5.3.27.4.2 Detailed description of elements

Element	Type	Description
isSynchronised	xs:boolean (W3C standard type)	This is a Boolean value, stating whether the device's SPINE time information is synchronized with the SPINE network or not, meaning that it could receive the SPINE time from a time distributor and has still a "sufficiently precise time".
lastSyncAt	xs:dateTime (W3C standard type)	This is the timestamp of the last synchronisation with a time distributor, or, in case of a time distributor itself, the timestamp of the last official date and time information (e.g. from an NTP server).
clockDrift	xs:integer (W3C standard type)	The drift of the internal clock (in seconds per day) is indicated by this element.

10718 Table 322: timePrecisionData function detailed description of elements

10719

10720 5.3.27.4.3 Available selectors

10721 None.

10722

10723 5.3.27.4.4 Examples

10724 5.3.27.4.4.1 Read-reply

10725 We assume a device had a synchronization with a time distributor more than two days ago. We also
 10726 assume the device has a clock drift of 5 seconds per day and only 10 seconds deviation are
 10727 acceptable to fulfil the network synchronization criteria. Then, if the device is requested for its
 10728 current *timePrecisionData* information it could send the following reply:

```

10729 <timePrecisionData>
10730   <isSynchronised>false</isSynchronised>
10731   <lastSyncAt>2013-07-14T15:00:00.0Z</lastSyncAt>
10732   <clockDrift>5</clockDrift>
10733 </timePrecisionData>

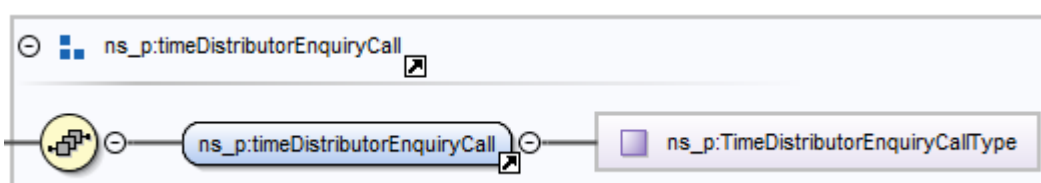
```

10734

10735 5.3.27.5 timeDistributorEnquiryCall

10736 5.3.27.5.1 General

10737 This function can be used for finding all time distributors in the network. Therefore, it can be sent to
 10738 all devices in the network that have their *isTimeDistributor* element in the function *timeDistributor*
 10739 (see section 5.3.27.3) set to true.



10740

10741 Figure 215: timeDistributorEnquiryCall function overview

10742

10743 *5.3.27.5.2 Detailed description of elements*

10744 In this version of the specification the function `timeDistributorEnquiryCall` does not contain child
 10745 elements.

10746

10747 *5.3.27.5.3 Available selectors*

10748 None.

10749

10750 *5.3.27.5.4 Examples*10751 *5.3.27.5.4.1 Call*

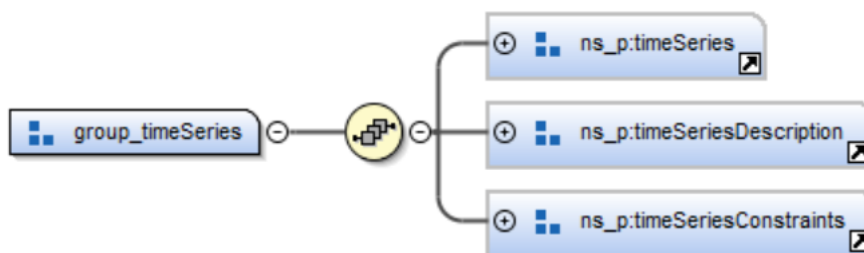
10752 A `timeDistributorEnquiryCall` instance, sent as call (the only classifier allowed for this function):

10753 `<timeDistributorEnquiryCall/>`

10754

10755 **5.3.28 TimeSeries**10756 **5.3.28.1 Introduction**

10757 TimeSeries is a versatile class that allows to express any kind of time dependent values as a time
 10758 series of values. It can be used to express electricity related values like power constraints or power
 10759 consumption over time, but also other values that are not electricity related like temperature or
 10760 humidity over time.



10761

10762 *Figure 216: TimeSeries function-group overview*

10763

10764 **5.3.28.2 timeSeriesListData**10765 *5.3.28.2.1 General*

10766 The `timeSeriesListData` function is used to describe the value of a `timeSeries` over time.

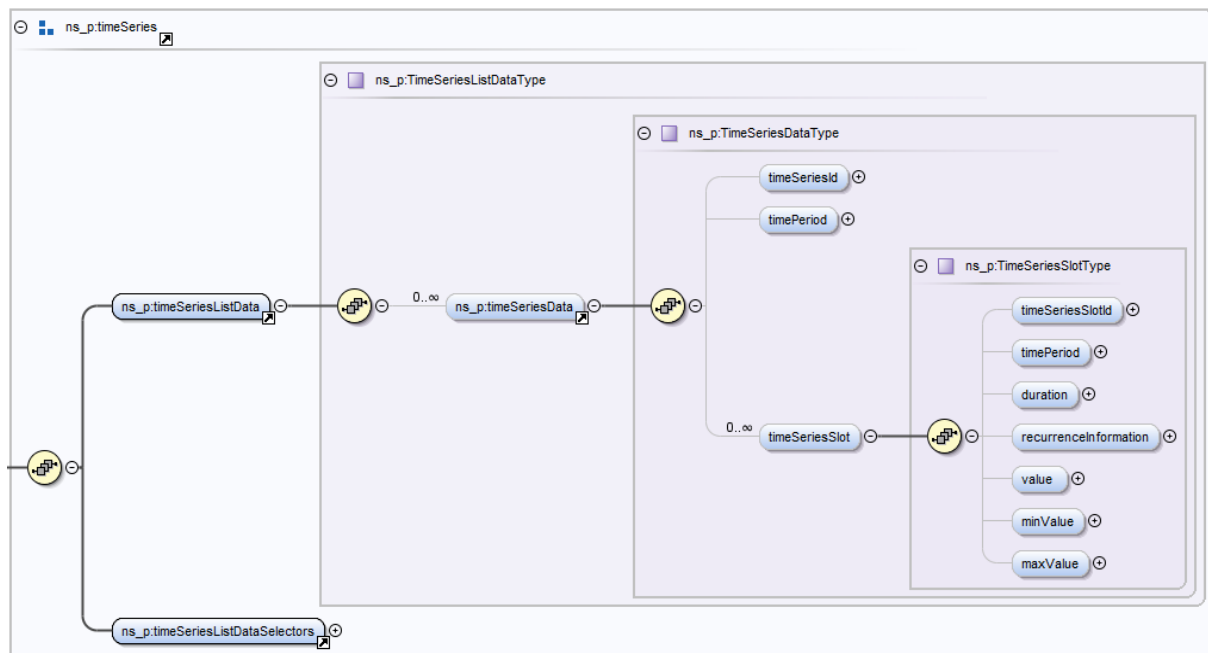


Figure 217: timeSeriesListData function overview

5.3.28.2.2 Detailed description of elements

Element	Type	Description
timeSeriesId	Identifier "TimeSeriesIdType" (see Table 339).	Allows the identification of a timeSeries. Allows linking of the different functions to the same timeSeries. Allows linking of the timeSeries within other Features that are placed in the same entity.
timePeriod	Common data type "TimePeriodType". See section 3.10.2.1.	Time period of the complete timeSeries.
timeSeriesSlot	Complex type "TimeSeriesSlotType". See Table 324.	Allows to define slots of a timeSeries.

Table 323: timeSeriesListData function detailed description of elements

Complex type TimeSeriesSlotType:

Element	Type	Description
timeSeriesSlotId	Identifier "TimeSeriesSlotIdType" (see Table 339).	Allows identification of a timeSeriesSlot within a timeSeries.
timePeriod	Common data type "TimePeriodType". See section 3.10.2.1.	Allows to define a timePeriod of a timeSeriesSlot and to model time gaps inbetween slots.
duration	xs:duration (W3C standard type)	Allows to define a duration of a timeSeriesSlot.

recurrenceInformation	<i>Common data type "RecurrenceInformationType". See section 3.10.2.14.</i>	If the time series slot is a recurring one, the corresponding information can be found within this element and its sub-elements.
value	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Defines the expected value during a timeSeriesSlot.
minValue	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Defines a lower value limit
maxValue	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	Defines a upper value limit

Table 324: Complex type TimeSeriesSlotType

5.3.28.2.3 Available selectors

- timeSeriesId
- timeSeriesSlotId

5.3.28.2.4 Examples

5.3.28.2.4.1 Read-reply

To read the time series entry with the *timeSeriesId* "3", the following read command can be send:

```

<function>timeSeriesListData</function>
<filter>
  <cmdControl>
    <partial/>
  </cmdControl>
  <timeSeriesListDataSelectors>
    <timeSeriesId>3</timeSeriesId>
  </timeSeriesListDataSelectors>
</filter>
<timeSeriesListData/>

```

A reply could look like this:

```

<timeSeriesListData>
  <timeSeriesData>
    <timeSeriesId>3</timeSeriesId>
    <timePeriod>
      <startTime>2018-05-30T10:00:00.0Z</startTime>
      <endTime>2018-05-30T12:00:00.0Z</endTime>
    </timePeriod>
    <timeSeriesSlot>
      <timeSeriesSlotId>1</timeSeriesSlotId>
      <timePeriod>
        <startTime>2018-05-30T10:00:00.0Z</startTime>
      </timePeriod>
      <duration>PT30M</duration>
      <recurrenceInformation>
        <daysOfWeek>
          <saturday/>

```

```

10809         <sunday/>
10810     </daysOfWeek>
10811 </recurrenceInformation>
10812 <value>
10813     <number>21</number>
10814     <scale>2</scale>
10815 </value>
10816 </timeSeriesSlot>
10817 <timeSeriesSlot>
10818     <timeSeriesSlotId>2</timeSeriesSlotId>
10819     <timePeriod>
10820         <startTime>2018-05-30T11:00:00.0Z</startTime>
10821     </timePeriod>
10822     <duration>PT1H</duration>
10823     <recurrenceInformation>
10824         <daysOfWeek>
10825             <saturday/>
10826             <sunday/>
10827         </daysOfWeek>
10828     </recurrenceInformation>
10829     <value>
10830         <number>35</number>
10831         <scale>2</scale>
10832     </value>
10833 </timeSeriesSlot>
10834 </timeSeriesData>
10835 </timeSeriesListData>

```

5.3.28.3 timeSeriesDescriptionListData

5.3.28.3.1 General

The timeSeriesDescriptionListData function is used to describe a timeSeries.

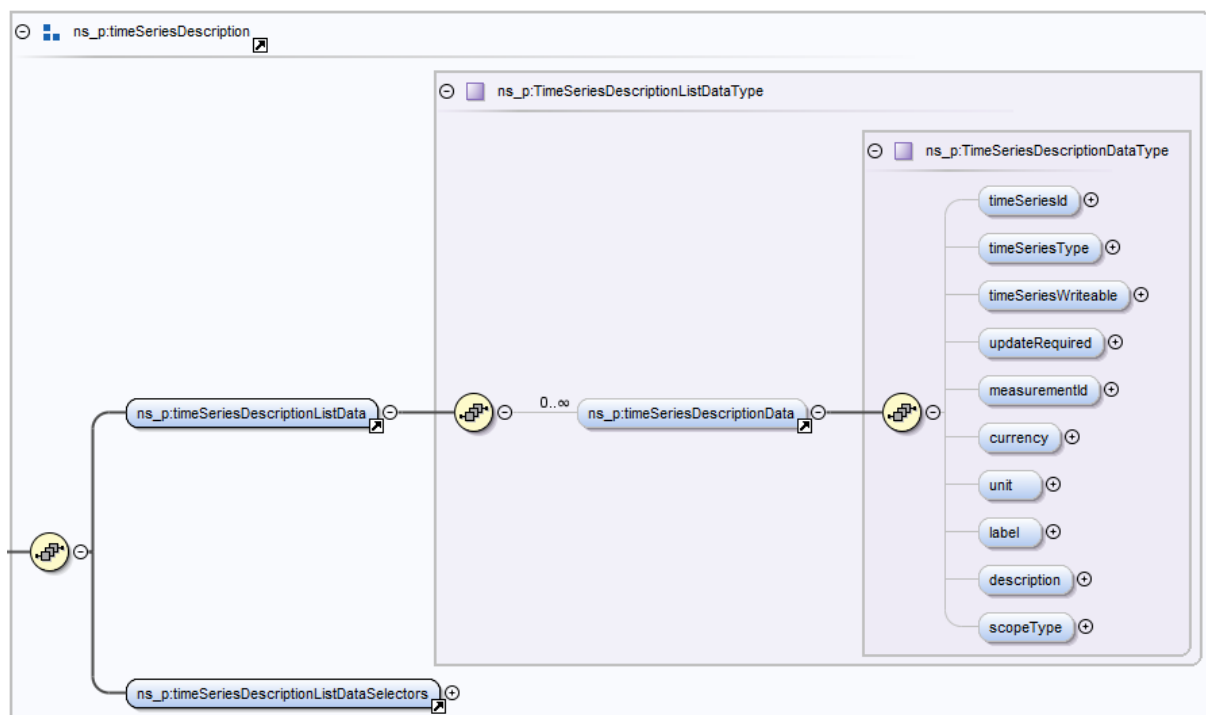


Figure 218: timeSeriesDescriptionListData function overview

10842

10843 5.3.28.3.2 Detailed description of elements

Element	Type	Description
timeSeriesId	<i>Identifier</i> "TimeSeriesIdType" (see Table 339).	Allows the identification of a timeSeries. Allows linking of the different functions to the same timeSeries. Allows linking of the timeSeries within other Features that are placed in the same entity.
timeSeriesType	<i>Union</i> "TimeSeriesTypeType": - Enum (see Table 326): TimeSeriesTypeEnumType - EnumExtendType (see section 3.10.1.5)	Allows to define which timeSeries type is supported.
timeSeriesWriteable	xs:boolean (W3C standard type)	Allows to define if the timeSeries is writeable.
updateRequired	xs:boolean (W3C standard type)	Request a write to timeSeriesData from the bound client.
measurementId	<i>Identifier</i> "MeasurementIdType" (see Table 339).	Allows to link data of other Features to a timeSeries, e.g. Measurement or ElectricalConnection data.
currency	<i>Common data type</i> "CurrencyType". See section 3.10.1.19.	Allows to describe the currency of the value, if the value is a price.
unit	<i>Common data type</i> "UnitOfMeasurementType". See section 3.10.1.17.	Allows to describe the unit of the value, if the value has a unit.
label	<i>Common data type</i> "LabelType". See section 3.10.1.2.	Allows to define user friendly name for the timeSeries.
description	<i>Common data type</i> "DescriptionType". See section 3.10.1.3.	Allows to define a user friendly description for the timeSeries.
scopeType	<i>Common data type</i> "ScopeTypeType". See section 3.10.1.21.	Specifies a more detailed meaning of the time series.

10844 Table 325: timeSeriesDescriptionListData function detailed description of elements

10845 Enumeration TimeSeriesTypeEnumType:

Value	Description
plan	"plan" is used to model how a certain value changes over time..
singleDemand	"singleDemand" is used to model a single demand with a simple timeSeries (e.g. this can be used to model a simple energy demand).
constraints	"constraints" is used to model certain value constraints over time with a min- and/or max value curve.

10846 Table 326: Enumeration TimeSeriesTypeEnumType

10847

10848 5.3.28.3.3 Available selectors

10849 - timeSeriesId

- 10850 - timeSeriesType
- 10851 - measurementId
- 10852 - scopeType

10853

10854 5.3.28.3.4 Examples

10855 5.3.28.3.4.1 Read-reply

10856 To read the time series description entry with the *timeSeriesId* "3", the following read command can
10857 be send:

```
10858 <function>timeSeriesDescriptionListData</function>
10859 <filter>
10860   <cmdControl>
10861     <partial/>
10862   </cmdControl>
10863   <timeSeriesDescriptionListDataSelectors>
10864     <timeSeriesId>3</timeSeriesId>
10865   </timeSeriesDescriptionListDataSelectors>
10866 </filter>
```

10867 A reply could look like this:

```
10868 <timeSeriesDescriptionListData>
10869   <timeSeriesDescriptionData>
10870     <timeSeriesId>3</timeSeriesId>
10871     <timeSeriesType>plan</timeSeriesType>
10872     <timeSeriesWriteable>true</timeSeriesWriteable>
10873     <updateRequired>false</updateRequired>
10874     <measurementId>1</measurementId>
10875     <currency>EUR</currency>
10876     <unit>Wh</unit>
10877   </timeSeriesDescriptionData>
10878 </timeSeriesDescriptionListData>
```

10879

10880 5.3.28.4 timeSeriesConstraintsListData

10881 5.3.28.4.1 General

10882 Allows to define constraints for the timeSeries, e.g. what is the maximum amount of slots allowed or
10883 similar. This function can be especially valuable if timeSeriesData is writeable but the server has
10884 certain constraints. The client can than match its own constraints with the server constraints, to form
10885 a conforming write.

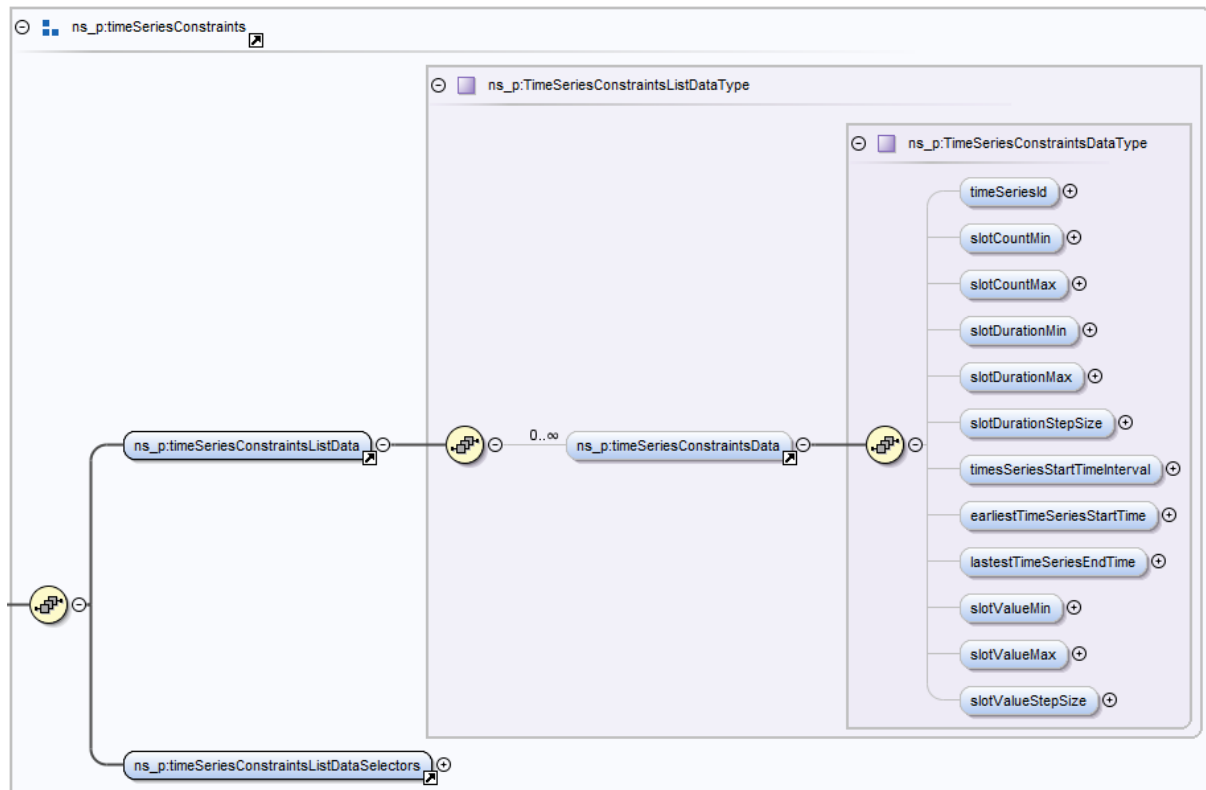


Figure 219: timeSeriesConstraintsListData function overview

5.3.28.4.2 Detailed description of elements

Element	Type	Description
timeSeriesId	Identifier "TimeSeriesIdType" (see Table 339).	Allows the identification of a timeSeries. Allows linking of the different functions to the same timeSeries. Allows linking of the timeSeries within other Features that are placed in the same entity.
slotCountMin	Simple type "TimeSeriesSlotCountType" (restriction of "TimeSeriesSlotIdType", see Table 324).	The minimum amount of slots for a specific time series is stated in this element. Recurring slots only count as one.
slotCountMax	Simple type "TimeSeriesSlotCountType" (restriction of "TimeSeriesSlotIdType", see Table 324).	The maximum amount of slots for a specific time series is stated in this element. Recurring slots only count as one.
slotDurationMin	xs:duration (W3C standard type)	If the slots of this time series have a minimum duration, it is denoted here.
slotDurationMax	xs:duration (W3C standard type)	If the slots of this time series have a maximum duration, it is denoted here.

slotDurationStepSize	xs:duration (W3C standard type)	Slots can have a specific duration step size. This element holds the corresponding value.
earliestTimeSeriesStartTime	<i>Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.</i>	This is the earliest start time the corresponding time series is allowed to start.
latestTimeSeriesEndTime	<i>Common data type "AbsoluteOrRelativeTimeType". See section 3.10.2.3.</i>	This is the latest end time the corresponding times series is allowed to end.
slotValueMin	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	The values within the slots of the corresponding time series are not allowed to be lower than this value.
slotValueMax	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	The values within the slots of the corresponding time series are not allowed to be higher than this value.
slotValueStepSize	<i>Common data type "ScaledNumberType". See section 3.10.1.8.</i>	The minimum step size between two different values within the slots of the corresponding time series.

Table 327: timeSeriesConstraintsListData function detailed description of elements

5.3.28.4.3 Available selectors

- timeSeriesId

5.3.28.4.4 Examples

5.3.28.4.4.1 Read-reply

To read timeSeriesId "3", the following read command can be send:

```
<function>timeSeriesConstraintsListData</function>
<filter>
  <cmdControl>
    <partial/>
  </cmdControl>
  <timeSeriesConstraintsListDataSelectors>
    <timeSeriesId>3</timeSeriesId>
  </timeSeriesConstraintsListDataSelectors>
</filter>
```

A reply could look like this:

```
<timeSeriesConstraintsListData>
  <timeSeriesConstraintsData>
    <timeSeriesId>3</timeSeriesId>
    <slotCountMin>1</slotCountMin>
    <slotCountMax>50</slotCountMax>
    <slotDurationMin>PT1M</slotDurationMin>
    <slotDurationMax>PT24H</slotDurationMax>
    <slotDurationStepSize>PT1S</slotDurationStepSize>
    <earliestTimeSeriesStartTime>2018-10-
15T10:00:00.0Z</earliestTimeSeriesStartTime>
```

```

10918         <latestTimeSeriesEndTime>2018-10-
10919 15T18:00:00.0Z</latestTimeSeriesEndTime>
10920         <slotValueMin>
10921             <number>0</number>
10922         </slotValueMin>
10923         <slotValueMax>
10924             <number>99999</number>
10925         </slotValueMax>
10926         <slotValueStepSize>
10927             <number>1</number>
10928             <scale>-1</scale>
10929         </slotValueStepSize>
10930     </timeSeriesConstraintsData>
10931 </timeSeriesConstraintsListData>

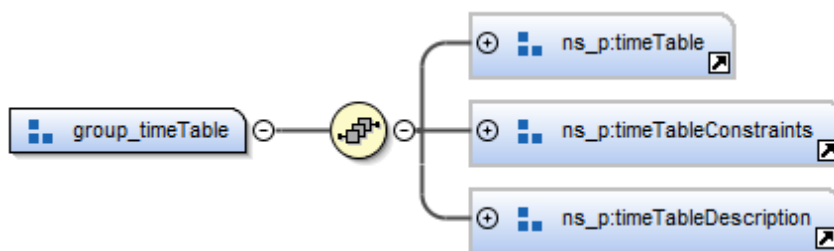
```

10932

10933 5.3.29 TimeTable

10934 5.3.29.1 Introduction

10935 All time dependent functionality like setpoint-configuration, etc. makes use of this elementary class.
 10936 It provides a list of time tables, each containing some number of time slots. Each slot has a start and
 10937 end time. The times can either be absolute (*dateTime*) or recurring (*daysOfWeek* plus *time* and, if
 10938 required, the recurring interval step elements). The count of slots and the start and end times can be
 10939 configurable or static (device specific). Other classes that make use of time tables only reference to
 10940 the *timeTableId*, not to specific time slots.



10941

10942 Figure 220: TimeTable function-group overview

10943

10944 5.3.29.2 timeTableListData

10945 5.3.29.2.1 General

10946 The list of time tables and time slots together with the start and end times are modelled in this
 10947 function.

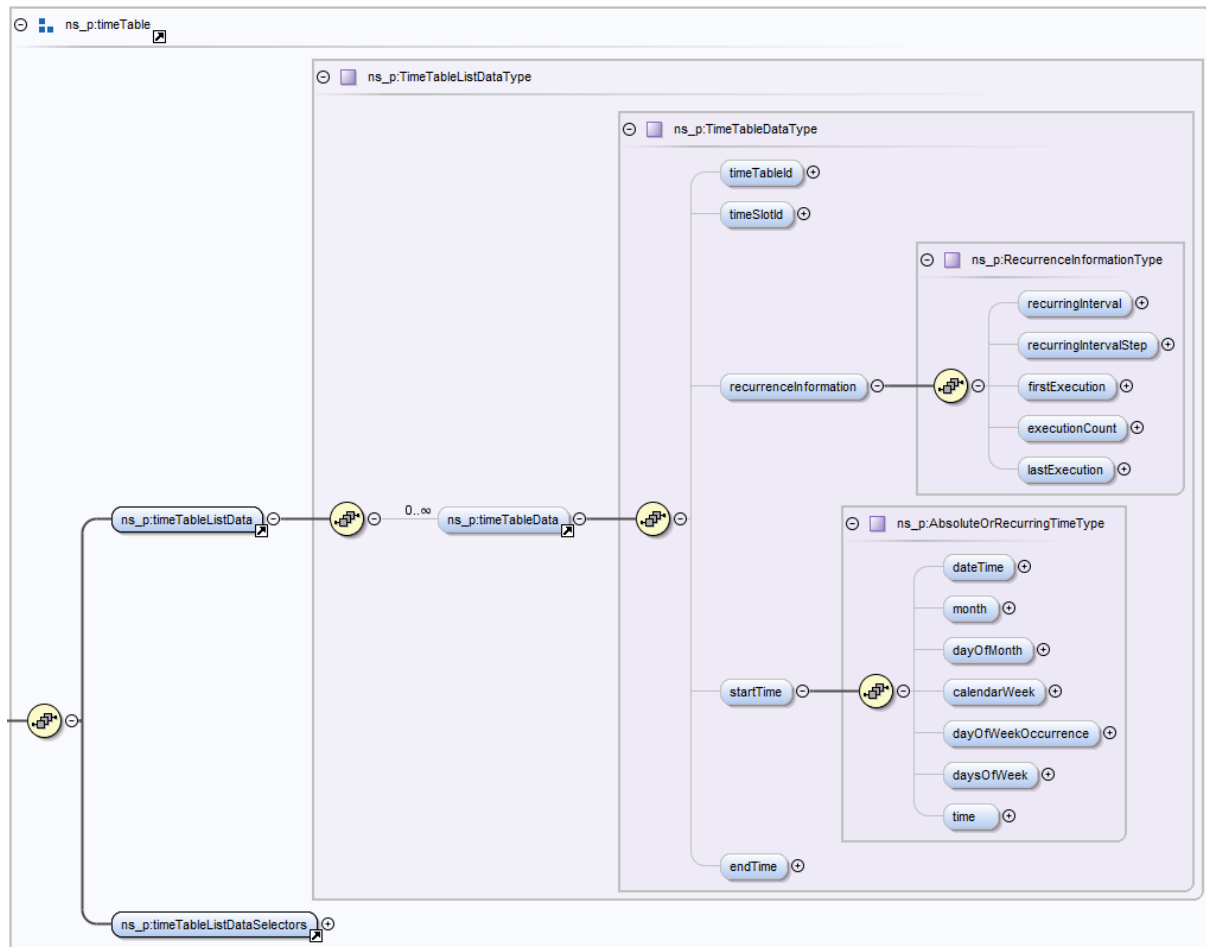


Figure 221: timeTableListData function overview

5.3.29.2.2 Detailed description of elements

Element	Type	Description
timeTableId	Identifier "TimeTableIdType" (see Table 339).	The <i>timeTableId</i> is used for identifying specific table information in the different functions and for relations in other classes like <i>Setpoint</i> .
timeSlotId	Identifier "TimeSlotIdType" (see Table 339).	Each slot of a time table has its own slot ID.
recurrenceInformation	Common data type "RecurrenceInformationType". See section 3.10.2.14.	Information about the recurrence of a slot.
startTime	Common data type "AbsoluteOrRecurringTimeType". See section 3.10.2.13.	Start time of one slot.
endTime	Common data type "AbsoluteOrRecurringTimeType". See section 3.10.2.13.	End time of one slot.

Table 328: timeTableListData function detailed description of elements

10954 *5.3.29.2.3 Available selectors*

- 10955 - timeTableId
- 10956 - timeSlotId

10957

10958 *5.3.29.2.4 Examples*10959 *5.3.29.2.4.1 Read-reply*

10960 If one wants to request a specific time table ("2"), he could sent the following read command:

```

10961 <function>timeTableListData</function>
10962 <filter>
10963     <cmdControl>
10964         <partial/>
10965     </cmdControl>
10966     <timeTableListDataSelectors>
10967         <timeTableId>2</timeTableId>
10968     </timeTableListDataSelectors>
10969 </filter>
10970 <timeTableListData/>

```

10971 A response could be:

```

10972 <function>timeTableListData</function>
10973 <filter>
10974     <cmdControl>
10975         <partial/>
10976     </cmdControl>
10977 </filter>
10978 <timeTableListData>
10979     <timeTableData>
10980         <timeTableId>2</timeTableId>
10981         <timeSlotId>0</timeSlotId>
10982         <recurrenceInformation>
10983             <recurringInterval>weekly</recurringInterval>
10984         </recurrenceInformation>
10985         <startTime>
10986             <daysOfWeek>
10987                 <monday/>
10988             </daysOfWeek>
10989             <time>12:00:00Z</time>
10990         </startTime>
10991         <endTime>
10992             <daysOfWeek>
10993                 <monday/>
10994             </daysOfWeek>
10995             <time>14:00:00Z</time>
10996         </endTime>
10997     </timeTableData>
10998     <timeTableData>
10999         <timeTableId>2</timeTableId>
11000         <timeSlotId>1</timeSlotId>
11001         <recurrenceInformation>
11002             <recurringInterval>weekly</recurringInterval>
11003         </recurrenceInformation>
11004         <startTime>
11005             <daysOfWeek>
11006                 <tuesday/>
11007             </daysOfWeek>
11008             <time>12:00:00Z</time>

```

```

11009         </startTime>
11010         <endTime>
11011             <daysOfWeek>
11012                 <tuesday/>
11013             </daysOfWeek>
11014             <time>14:00:00Z</time>
11015         </endTime>
11016     </timeTableData>
11017     <timeTableData>
11018         <timeTableId>2</timeTableId>
11019         <timeSlotId>2</timeSlotId>
11020         <recurrenceInformation>
11021             <recurringInterval>weekly</recurringInterval>
11022         </recurrenceInformation>
11023         <startTime>
11024             <daysOfWeek>
11025                 <friday/>
11026             </daysOfWeek>
11027             <time>11:00:00Z</time>
11028         </startTime>
11029         <endTime>
11030             <daysOfWeek>
11031                 <friday/>
11032             </daysOfWeek>
11033             <time>17:00:00Z</time>
11034         </endTime>
11035     </timeTableData>
11036 </timeTableListData>

```

11037

11038 5.3.29.2.4.2 Write (partial)

11039 To write new times (if allowed) into slots, one could send a message with classifier *write* like this:

```

11040 <function>timeTableListData</function>
11041 <filter>
11042     <cmdControl>
11043         <partial/>
11044     </cmdControl>
11045 </filter>
11046 <timeTableListData>
11047     <timeTableData>
11048         <timeTableId>1</timeTableId>
11049         <timeSlotId>0</timeSlotId>
11050         <startTime>
11051             <daysOfWeek>
11052                 <monday/>
11053                 <tuesday/>
11054                 <wednesday/>
11055                 <thursday/>
11056                 <friday/>
11057                 <saturday/>
11058                 <sunday/>
11059             </daysOfWeek>
11060             <time>10:00:00Z</time>
11061         </startTime>
11062         <endTime>
11063             <daysOfWeek>
11064                 <monday/>
11065                 <tuesday/>
11066                 <wednesday/>
11067                 <thursday/>

```

```

11068         <friday/>
11069         <saturday/>
11070         <sunday/>
11071     </daysOfWeek>
11072     <time>16:00:00Z</time>
11073 </endTime>
11074 </timeTableData>
11075 </timeTableListData>

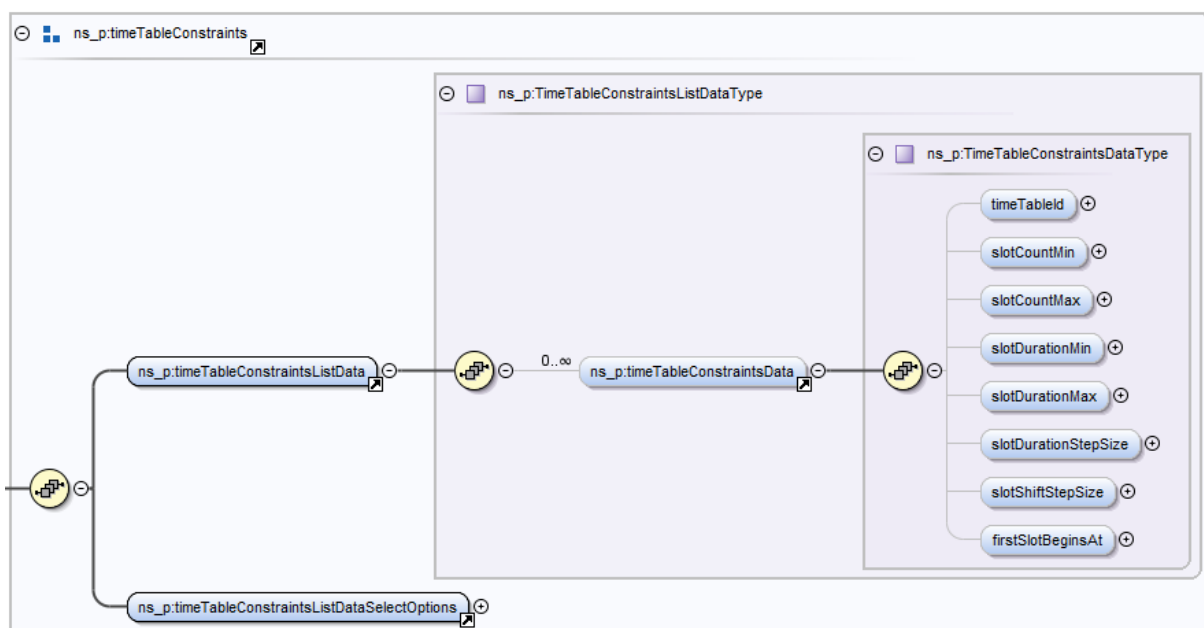
```

11076

11077 5.3.29.3 timeTableConstraintsListData

11078 5.3.29.3.1 General

11079 If a time table has some constraints, they are denoted with the help of this function (e.g. the
 11080 minimum and maximum count of slots (may differ between the time tables)).



11081

11082 Figure 222: timeTableConstraintsListData function overview

11083

11084 5.3.29.3.2 Detailed description of elements

Element	Type	Description
timeTableId	Identifier "TimeTableIdType" (see Table 339).	The <i>timeTableId</i> is used for indentifying specific table information in the different functions and for relations in other classes like <i>Setpoint</i> .
slotCountMin	Simple type "TimeSlotCountType" (restriction of "TimeSlotIdType", see Table 328).	The minimum amount of slots for a specific time table is stated in this element. Recurring slots only count as one.
slotCountMax	Simple type "TimeSlotCountType" (restriction of "TimeSlotIdType", see Table 328).	The maximum amount of slots for a specific time table is stated in this element. Recurring slots only count as one.

slotDurationMin	xs:duration (W3C standard type)	If the slots of this time table have a minimum duration, it is denoted here.
slotDurationMax	xs:duration (W3C standard type)	If the slots of this time table have a maximum duration, it is denoted here.
slotDurationStepSize	xs:duration (W3C standard type)	Slots can have a specific duration step size. This element holds the corresponding value.
slotShiftStepSize	xs:duration (W3C standard type)	Slots can have a specific shift step size. This element holds the corresponding value.
firstSlotBeginsAt	xs:time (W3C standard type)	Especially time tables with fixed slot durations probably need this information to specify, where in time the slots are located (i.e. fixed 15 minute slots can start either at 00:00:00 (default) or e.g. at 00:10:00). This value only defines, when the time pattern (defined by the element <i>slotDurationStepSize</i>) begins on a day.

Table 329: *timeTableConstraintsListData* function detailed description of elements

5.3.29.3.3 Available selectors

- `timeTableId`

5.3.29.3.4 Examples

5.3.29.3.4.1 Notify

If some constraints data has changed, a device could send an update of its information with a *notify* classifier to all bound devices on the corresponding feature (the “filter/partial” part to announce the “restricted function exchange is simplified by “...” just to improve the readability”):

```
...
<timeTableConstraintsListData>
  <timeTableConstraintsData>
    <timeTableId>3</timeTableId>
    <slotCountMin>1</slotCountMin>
    <slotCountMax>10</slotCountMax>
    <slotDurationMin>PT10M</slotDurationMin>
    <slotDurationMax>PT20M</slotDurationMax>
    <slotDurationStepSize>PT5M</slotDurationStepSize>
    <slotShiftStepSize>PT5M</slotShiftStepSize>
    <firstSlotBeginsAt>00:00:00</firstSlotBeginsAt>
  </timeTableConstraintsData>
</timeTableConstraintsListData>
```

5.3.29.4 timeTableDescriptionListData

5.3.29.4.1 General

The more static information about the time tables are described in the *timeTableDescriptionData* function.

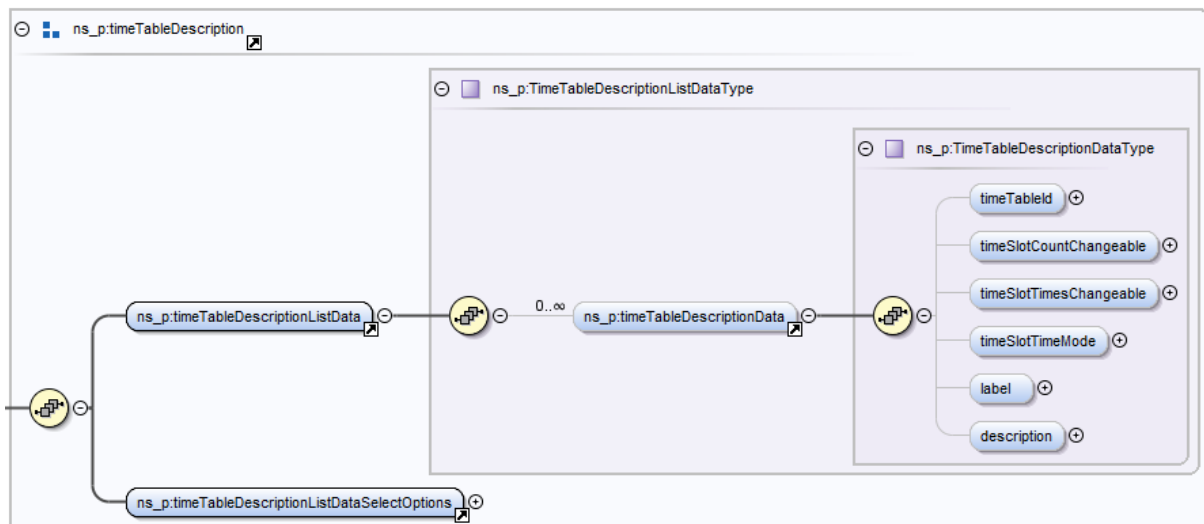


Figure 223: timeTableDescriptionListData function overview

5.3.29.4.2 Detailed description of elements

Element	Type	Description
timeTableId	Identifier "TimeTableIdType" (see Table 339).	The <i>timeTableId</i> is used for indentifying specific table information in the different functions and for relations in other classes like <i>Setpoint</i> .
timeSlotCountChangeable	xs:boolean (W3C standard type)	States whether the amount of slots is changeable by a client for a specific time table or not.
timeSlotTimesChangeable	xs:boolean (W3C standard type)	States whether the times of the slots of a specific time table are changeable by a client or not.
timeSlotTimeMode	Union "TimeSlotTimeModeType": - Enum (see Table 331): TimeSlotTimeModeEnumType - EnumExtendType (see section 3.10.1.5)	If either <i>absolute</i> or <i>recurring</i> time slots or <i>both</i> modes are supported is stated in this element.
label	Common data type "LabelType". See section 3.10.1.2.	A short label of the time table.
description	Common data type "DescriptionType". See section 3.10.1.3.	Descriptive information on the time table.

Table 330: timeTableDescriptionListData function detailed description of elements

Enumeration TimeSlotTimeModeEnumType:

Value	Description
absolute	Only absolute times are allowed for the time table.
recurring	Only recurring times are allowed for the time table.
both	Absolute and recurring times are allowed for the time table.

Table 331: Enumeration TimeSlotTimeModeEnumType

11120

11121 **5.3.29.4.3 Available selectors**

11122 - timeTableId

11123

11124 **5.3.29.4.4 Examples**11125 **5.3.29.4.4.1 Read-reply**

11126 If one is interested in the time table description of all available time tables, he would send the
 11127 standard read command on the *timeTableDescriptionListData* resource. A response could look like
 11128 this:

```

11129 <timeTableDescriptionListData>
11130   <timeTableDescriptionData>
11131     <timeTableId>1</timeTableId>
11132     <timeSlotCountChangeable>false</timeSlotCountChangeable>
11133     <timeSlotTimesChangeable>true</timeSlotTimesChangeable>
11134     <timeSlotTimeMode>absolute</timeSlotTimeMode>
11135   </timeTableDescriptionData>
11136   <timeTableDescriptionData>
11137     <timeTableId>2</timeTableId>
11138     <timeSlotCountChangeable>false</timeSlotCountChangeable>
11139     <timeSlotTimesChangeable>true</timeSlotTimesChangeable>
11140     <timeSlotTimeMode>both</timeSlotTimeMode>
11141   </timeTableDescriptionData>
11142 </timeTableDescriptionListData>

```

11143

11144 **5.3.30 UseCaseInformation**11145 **5.3.30.1 Introduction**

11146 To model which Use Cases are supported by a node the UseCaseInformation Class can be used.



11147

11148 *Figure 224: UseCaseInformation function-group overview*

11149

11150 **5.3.30.2 useCaseInformationListData**11151 **5.3.30.2.1 General**

11152 This function provides information about the supported Use Cases together with a version
 11153 information and further relevant details.

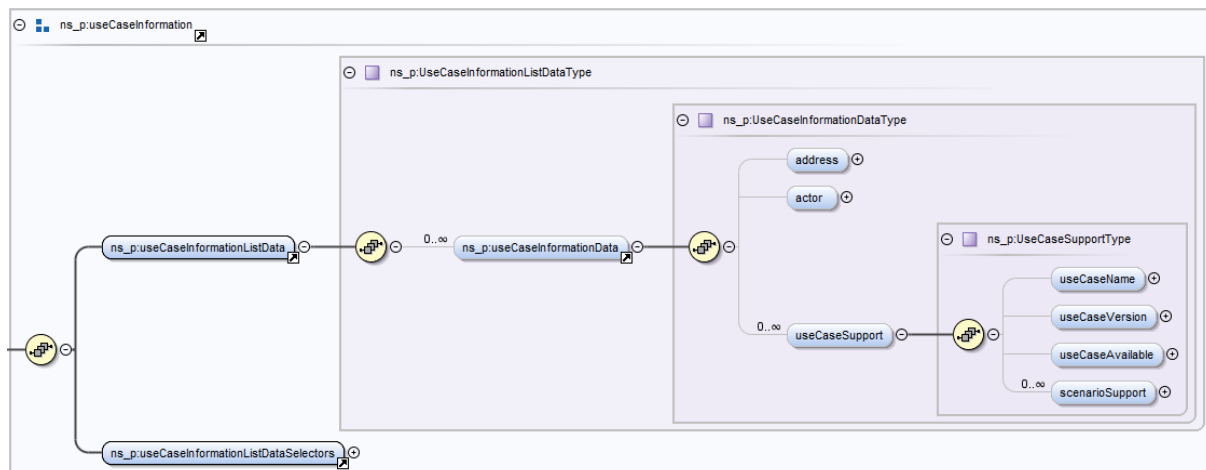


Figure 225: useCaseInformationListData function overview

5.3.30.2.2 Detailed description of elements

Element	Type	Description
address	Common data type "FeatureAddressType". See section 3.10.3.6.	The address where the supported Use Case is located. Note: not all sub-elements always need to be set here!
actor	Union "UseCaseActorType": - Simple type "UseCaseActorEnumType" (restriction of xs:string) - EnumExtendType (see section 3.10.1.5)	The actor of the supported Use Case located on the above stated address.
useCaseSupport (list)		List of supported Use Cases.
useCaseSupport. useCaseName	Union "UseCaseNameType": - Simple type "UseCaseNameEnumType" (restriction of xs:string) - EnumExtendType (see section 3.10.1.5)	The name of the supported Use Case.
useCaseSupport. useCaseVersion	Common data type "SpecificationVersionType". See section 3.10.1.4.	Version of the supported Use Case.
useCaseSupport. useCaseAvailable	xs:boolean (W3C standard type)	Stated whether the Use Case is available at the moment.
useCaseSupport. scenarioSupport (list)	Simple type "UseCaseScenarioSupportType" (restriction of "xs:unsignedInt")	List of supported scenarios of the Use Case.

Table 332: useCaseInformationListData function detailed description of elements

5.3.30.2.3 Available selectors

- address (with sub-elements, see above)
- actor
- useCaseSupport (with sub-elements, see above)

11164

11165 **5.3.30.2.4 Examples**11166 **5.3.30.2.4.1 Read-reply**11167 A *useCaseInformationListData* instance sent as reply to a standard read command:

```

11168 <useCaseInformationListData>
11169   <useCaseInformationData>
11170     <address>
11171       <device>d:_i:46925_TestDevice_1</device>
11172       <entity>1</entity>
11173       <entity>1</entity>
11174     </address>
11175     <actor>Compressor</actor>
11176     <useCaseSupport>
11177
11178 <useCaseName>visualizationOfPowerConsumptionOfHeatPumpCompressor</useCaseName>
11179
11180       <useCaseVersion>1.0.0</useCaseVersion>
11181       <useCaseAvailable>true</useCaseAvailable>
11182       <scenarioSupport>1</scenarioSupport>
11183     </useCaseSupport>
11184   </useCaseInformationData>
11185   <useCaseInformationData>
11186     <address>
11187       <device>d:_i:46925_TestDevice_1</device>
11188       <entity>2</entity>
11189       <entity>1</entity>
11190     </address>
11191     <actor>GridConnectionPointOfPremises</actor>
11192     <useCaseSupport>
11193
11194 <useCaseName>optimizationForOptionalHeatPumpPowerConsumption</useCaseName>
11195
11196       <useCaseVersion>1.0.0</useCaseVersion>
11197       <useCaseAvailable>false</useCaseAvailable>
11198       <scenarioSupport>1</scenarioSupport>
11199     </useCaseSupport>
11200   </useCaseInformationData>
11201   <useCaseInformationData>
11202     <address>
11203       <device>d:_i:46925_TestDevice_1</device>
11204       <entity>2</entity>
11205       <feature>1</feature>
11206     </address>
11207     <actor>GridConnectionPointOfPremises</actor>
11208     <useCaseSupport>
11209
11210 <useCaseName>remoteMonitoringOfElectricityGridConnectionPoint</useCaseName>
11211
11212       <useCaseVersion>1.0.0</useCaseVersion>
11213       <useCaseAvailable>true</useCaseAvailable>
11214       <scenarioSupport>1</scenarioSupport>
11215       <scenarioSupport>2</scenarioSupport>
11216       <scenarioSupport>3</scenarioSupport>
11217       <scenarioSupport>4</scenarioSupport>
11218     </useCaseSupport>
11219   </useCaseInformationData>
11220 </useCaseInformationListData>

```

11211

11212 **5.3.31 Version**11213 **5.3.31.1 Introduction**

11214 This class can be used to model the version information of the SPINE specification. The version string
 11215 is divided into *major*, *minor* and *revision*, separated by a dot (e.g. "1.0.0"), see section 5.3.31.2.1.

11216 The version information is also used within the “header” of “datagram” (see document

11217 [ProtocolSpecification]). This section, however, provides the definition and proper usage of “Version”
 11218 as SPINE class.



Figure 226: Version function-group overview

5.3.31.2 *specificationVersionListData*

5.3.31.2.1 *General*

The knowledge of the supported SPINE data model versions of a communication partner is essential for the proper interworking of devices that implemented different versions. Considering the compatibility rules defined in [ProtocolSpecification] is only possible with this kind of knowledge. Therefore, the *specificationVersionListData* function is used.

The three parts of the version have the following meaning:

major

Typically, a specification's major part of the version number is increased by the proper authority in case of “fundamental changes”. Among others, changes that break mandatory compatibility requirements belong to such kind of changes.

minor

Typically, a specification's minor part of the version number is increased by the proper authority in case of optional additions to the predecessor of the specification version.

revision

Typically, a specification's revision part of the version number is increased by the proper authority in case of “minor changes” where major and minor part can remain unchanged.

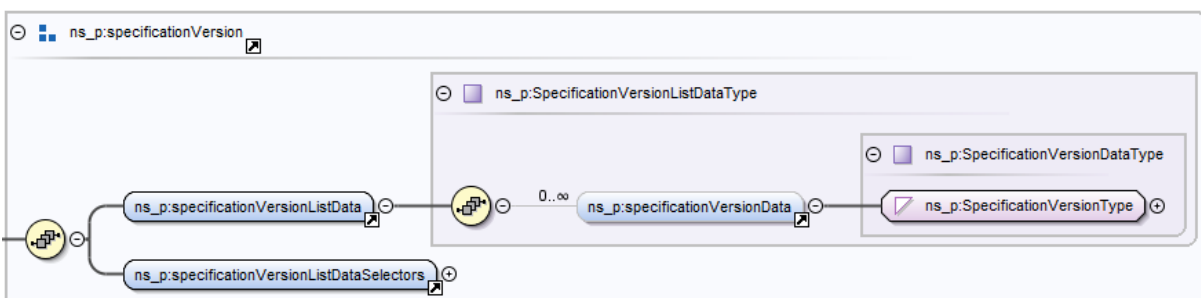


Figure 227: *specificationVersionListData* function overview

5.3.31.2.2 *Detailed description of elements*

Element	Type	Description
specificationVersionData	Common data type "SpecificationVersionType". See section 3.10.1.4.	One of the supported SPINE specification versions.

Table 333: *specificationVersionListData* function detailed description of elements

11244

11245 5.3.31.2.3 *Available selectors*

11246 None.

11247

11248 5.3.31.2.4 *Examples*

11249 5.3.31.2.4.1 *Notify*

11250 A device sends the *specificationVersionListData* function (as (complete) notification) to another
11251 device, including all supported versions of the SPINE specification:

11252 <specificationVersionListData>

11253 <specificationVersionData>1.0.0</specificationVersionData>

11254 </specificationVersionListData>

11255

Annex A - SPINE XSDs

All necessary XSDs defined for SPINE are included in this annex. The so-called "overview" XSDs are left out as they only reference to the functions defined within the "normal" XSDs. An example for an overview XSD is given in section A.3.6 Overview example.

The EEBus Initiative e.V. also provides the XSD files for download (please consider the corresponding website [EEBus]). These XSD files should be preferred as the the printed XSDs in this annex contain additional line breaks due to page layout reasons.

A.1 Complex Classes

A.1.1 IncentiveTable

File name: EEBus_SPINE_TS_IncentiveTable.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    Smart Premises Interoperable Neutral-Message Exchange (SPINE)
    Version 1.1.1
    2018-12-21
    Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
    Source: https://www.eebus.org/en/specifications/
-->
<xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
  blockDefault="#all" elementFormDefault="qualified">
  <xs:annotation>
    <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
    e.V. All rights reserved.</xs:documentation>
  </xs:annotation>
  <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
  <xs:include schemaLocation="EEBus_SPINE_TS_TariffInformation.xsd"/>
  <xs:complexType name="IncentiveTableType">
    <xs:sequence>
      <xs:element minOccurs="0" name="tariff">
        <xs:complexType>
          <xs:complexContent>
            <xs:restriction base="ns_p:TariffDataType">
              <xs:sequence>
                <xs:element minOccurs="0" name="tariffId"
type="ns_p:TariffIdType"/>
              </xs:sequence>
            </xs:restriction>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element maxOccurs="unbounded" minOccurs="0" name="incentiveSlot"
type="ns_p:IncentiveTableIncentiveSlotType"> </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="IncentiveTableIncentiveSlotType">
    <xs:sequence>
      <xs:element minOccurs="0" name="timeInterval">
        <xs:complexType>
          <xs:complexContent>
            <xs:restriction base="ns_p:TimeTableDataType">
              <xs:sequence>
                <xs:element minOccurs="0" name="timeSlotId"
type="ns_p:TimeSlotIdType"/>
                <xs:element minOccurs="0" name="startTime">
                  <xs:complexType>
                    <xs:complexContent>
                      <xs:restriction
base="ns_p:AbsoluteOrRecurringTimeType">
                    </xs:restriction>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:restriction>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

```

11319                                     <xs:element minOccurs="0" name="dateTime"
11320 type="xs:dateTime"/>
11321                                     <xs:element minOccurs="0" name="relative"
11322 type="xs:duration"/>
11323                                     </xs:sequence>
11324                                     </xs:restriction>
11325                                     </xs:complexContent>
11326                                     </xs:complexType>
11327                                 </xs:element>
11328                                 <xs:element minOccurs="0" name="endTime">
11329                                     <xs:complexType>
11330                                         <xs:complexContent>
11331                                             <xs:restriction
11332 base="ns_p:AbsoluteOrRecurringTimeType">
11333                                             <xs:sequence>
11334                                                 <xs:element minOccurs="0" name="dateTime"
11335 type="xs:dateTime"/>
11336                                                 <xs:element minOccurs="0" name="relative"
11337 type="xs:duration"/>
11338                                                 </xs:sequence>
11339                                                 </xs:restriction>
11340                                                 </xs:complexContent>
11341                                                 </xs:complexType>
11342                                             </xs:element>
11343                                         </xs:sequence>
11344                                     </xs:restriction>
11345                                     </xs:complexContent>
11346                                     </xs:complexType>
11347                                 </xs:element>
11348                                 <xs:element maxOccurs="unbounded" minOccurs="0" name="tier"
11349 type="ns_p:IncentiveTableTierType"> </xs:element>
11350                             </xs:sequence>
11351                         </xs:complexType>
11352                         <xs:complexType name="IncentiveTableTierType">
11353                             <xs:sequence>
11354                                 <xs:element minOccurs="0" name="tier">
11355                                     <xs:complexType>
11356                                         <xs:complexContent>
11357                                             <xs:restriction base="ns_p:TierDataType">
11358                                                 <xs:sequence>
11359                                                     <xs:element minOccurs="0" name="tierId"
11360 type="ns_p:TierIdType"/>
11361                                                     </xs:sequence>
11362                                                     </xs:restriction>
11363                                                 </xs:complexContent>
11364                                                 </xs:complexType>
11365                                             </xs:element>
11366                                             <xs:element maxOccurs="unbounded" minOccurs="0" name="boundary">
11367                                                 <xs:complexType>
11368                                                     <xs:complexContent>
11369                                                         <xs:restriction base="ns_p:TierBoundaryDataType">
11370                                                             <xs:sequence>
11371                                                                 <xs:element minOccurs="0" name="boundaryId"
11372 type="ns_p:TierBoundaryIdType"/>
11373                                                                 <xs:element minOccurs="0" name="lowerBoundaryValue"
11374 type="ns_p:ScaledNumberType"/>
11375                                                                 <xs:element minOccurs="0" name="upperBoundaryValue"
11376 type="ns_p:ScaledNumberType"/>
11377                                                             </xs:sequence>
11378                                                         </xs:restriction>
11379                                                         </xs:complexContent>
11380                                                         </xs:complexType>
11381                                                     </xs:element>
11382                                                 <xs:element minOccurs="0" name="incentive" maxOccurs="unbounded">
11383                                                     <xs:complexType>
11384                                                         <xs:complexContent>
11385                                                             <xs:restriction base="ns_p:IncentiveDataType">
11386                                                                 <xs:sequence>
11387                                                                     <xs:element minOccurs="0" name="incentiveId"
11388 type="ns_p:IncentiveIdType"/>
11389                                                                     <xs:element minOccurs="0" name="value"
11390 type="ns_p:ScaledNumberType"/>
11391                                                                 </xs:sequence>
11392                                                             </xs:restriction>
11393                                                         </xs:complexContent>
11394                                                         </xs:complexType>
11395                                                     </xs:element>
11396                                                 </xs:sequence>

```

```

11397     </xs:complexType>
11398     <xs:complexType name="IncentiveTableDataType">
11399         <xs:sequence>
11400             <xs:element maxOccurs="unbounded" minOccurs="0" name="incentiveTable"
11401 type="ns_p:IncentiveTableType"> </xs:element>
11402         </xs:sequence>
11403     </xs:complexType>
11404     <xs:element name="incentiveTableData" type="ns_p:IncentiveTableDataType"/>
11405     <xs:complexType name="IncentiveTableElementsType">
11406         <xs:sequence>
11407             <xs:element minOccurs="0" name="tariff">
11408                 <xs:complexType>
11409                     <xs:complexContent>
11410                         <xs:restriction base="ns_p:TariffDataElementsType">
11411                             <xs:sequence>
11412                                 <xs:element minOccurs="0" name="tariffId"
11413 type="ns_p:ElementTagType"/>
11414                             </xs:sequence>
11415                         </xs:restriction>
11416                     </xs:complexContent>
11417                 </xs:complexType>
11418             </xs:element>
11419             <xs:element minOccurs="0" name="incentiveSlot"
11420 type="ns_p:IncentiveTableIncentiveSlotElementsType"> </xs:element>
11421         </xs:sequence>
11422     </xs:complexType>
11423     <xs:complexType name="IncentiveTableIncentiveSlotElementsType">
11424         <xs:sequence>
11425             <xs:element minOccurs="0" name="timeInterval">
11426                 <xs:complexType>
11427                     <xs:complexContent>
11428                         <xs:restriction base="ns_p:TimeTableDataElementsType">
11429                             <xs:sequence>
11430                                 <xs:element minOccurs="0" name="timeSlotId"
11431 type="ns_p:ElementTagType"/>
11432                                 <xs:element minOccurs="0" name="startTime">
11433                                     <xs:complexType>
11434                                         <xs:complexContent>
11435                                             <xs:restriction
11436 base="ns_p:AbsoluteOrRecurringTimeElementsType">
11437                                                 <xs:sequence>
11438                                                     <xs:element minOccurs="0" name="dateTime"
11439 type="ns_p:ElementTagType"/>
11440                                                     <xs:element minOccurs="0" name="relative"
11441 type="ns_p:ElementTagType"/>
11442                                                 </xs:sequence>
11443                                             </xs:restriction>
11444                                         </xs:complexContent>
11445                                     </xs:complexType>
11446                                 </xs:element>
11447                                 <xs:element minOccurs="0" name="endTime">
11448                                     <xs:complexType>
11449                                         <xs:complexContent>
11450                                             <xs:restriction
11451 base="ns_p:AbsoluteOrRecurringTimeElementsType">
11452                                                 <xs:sequence>
11453                                                     <xs:element minOccurs="0" name="dateTime"
11454 type="ns_p:ElementTagType"/>
11455                                                     <xs:element minOccurs="0" name="relative"
11456 type="ns_p:ElementTagType"/>
11457                                                 </xs:sequence>
11458                                             </xs:restriction>
11459                                         </xs:complexContent>
11460                                     </xs:complexType>
11461                                 </xs:element>
11462                             </xs:sequence>
11463                         </xs:restriction>
11464                     </xs:complexContent>
11465                 </xs:complexType>
11466             </xs:element>
11467             <xs:element minOccurs="0" name="tier" type="ns_p:IncentiveTableTierElementsType">
11468 </xs:element>
11469         </xs:sequence>
11470     </xs:complexType>
11471     <xs:complexType name="IncentiveTableTierElementsType">
11472         <xs:sequence>
11473             <xs:element minOccurs="0" name="tier">
11474                 <xs:complexType>

```



```
11475         <xs:complexContent>
11476           <xs:restriction base="ns_p:TierDataElementsType">
11477             <xs:sequence>
11478               <xs:element minOccurs="0" name="tierId"
11479 type="ns_p:ElementTagType"/>
11480             </xs:sequence>
11481           </xs:restriction>
11482         </xs:complexContent>
11483       </xs:complexType>
11484     </xs:element>
11485     <xs:element minOccurs="0" name="boundary">
11486       <xs:complexType>
11487         <xs:complexContent>
11488           <xs:restriction base="ns_p:TierBoundaryDataElementsType">
11489             <xs:sequence>
11490               <xs:element minOccurs="0" name="boundaryId"
11491 type="ns_p:ElementTagType"/>
11492               <xs:element minOccurs="0" name="lowerBoundaryValue"
11493 type="ns_p:ScaledNumberElementsType"/>
11494               <xs:element minOccurs="0" name="upperBoundaryValue"
11495 type="ns_p:ScaledNumberElementsType"/>
11496             </xs:sequence>
11497           </xs:restriction>
11498         </xs:complexContent>
11499       </xs:complexType>
11500     </xs:element>
11501     <xs:element minOccurs="0" name="incentive">
11502       <xs:complexType>
11503         <xs:complexContent>
11504           <xs:restriction base="ns_p:IncentiveDataElementsType">
11505             <xs:sequence>
11506               <xs:element minOccurs="0" name="incentiveId"
11507 type="ns_p:ElementTagType"/>
11508               <xs:element minOccurs="0" name="value"
11509 type="ns_p:ElementTagType"/>
11510             </xs:sequence>
11511           </xs:restriction>
11512         </xs:complexContent>
11513       </xs:complexType>
11514     </xs:element>
11515   </xs:sequence>
11516 </xs:complexType>
11517 <xs:complexType name="IncentiveTableDataElementsType">
11518   <xs:sequence>
11519     <xs:element minOccurs="0" name="incentiveTable"
11520 type="ns_p:IncentiveTableElementsType"/> </xs:element>
11521   </xs:sequence>
11522 </xs:complexType>
11523 <xs:element name="incentiveTableDataElements" type="ns_p:IncentiveTableDataElementsType"/>
11524 <xs:complexType name="IncentiveTableDataSelectorsType">
11525   <xs:sequence>
11526     <xs:element minOccurs="0" name="tariff">
11527       <xs:complexType>
11528         <xs:complexContent>
11529           <xs:restriction base="ns_p:TariffListDataSelectorsType">
11530             <xs:sequence>
11531               <xs:element minOccurs="0" name="tariffId"
11532 type="ns_p:TariffIdType"/>
11533             </xs:sequence>
11534           </xs:restriction>
11535         </xs:complexContent>
11536       </xs:complexType>
11537     </xs:element>
11538   </xs:sequence>
11539 </xs:complexType>
11540 <xs:element name="incentiveTableDataSelectors"
11541 type="ns_p:IncentiveTableDataSelectorsType"/>
11542 <xs:complexType name="IncentiveTableDescriptionType">
11543   <xs:sequence>
11544     <xs:element minOccurs="0" name="tariffDescription">
11545       <xs:complexType>
11546         <xs:complexContent>
11547           <xs:restriction base="ns_p:TariffDescriptionDataType">
11548             <xs:sequence>
11549               <xs:element minOccurs="0" name="tariffId"
11550 type="ns_p:TariffIdType"/>
11551               <xs:element minOccurs="0" name="measurementId"
11552 type="ns_p:MeasurementIdType"/>

```

```

11553                                     <xs:element minOccurs="0" name="tariffWriteable"
11554 type="xs:boolean"/>
11555                                     <xs:element minOccurs="0" name="updateRequired"
11556 type="xs:boolean"/>
11557                                     <xs:element minOccurs="0" name="scopeType"
11558 type="ns_p:ScopeTypeType"/>
11559                                     <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
11560                                     <xs:element minOccurs="0" name="description"
11561 type="ns_p:DescriptionType"/>
11562                                     <xs:element minOccurs="0" name="slotIdSupport"
11563 type="xs:boolean"/>
11564                                     </xs:sequence>
11565                                     </xs:restriction>
11566                                     </xs:complexContent>
11567                                     </xs:complexType>
11568                                 </xs:element>
11569                                 <xs:element maxOccurs="unbounded" minOccurs="0" name="tier"
11570 type="ns_p:IncentiveTableDescriptionTierType"> </xs:element>
11571                             </xs:sequence>
11572                         </xs:complexType>
11573                     <xs:complexType name="IncentiveTableDescriptionTierType">
11574                         <xs:sequence>
11575                             <xs:element minOccurs="0" name="tierDescription">
11576                                 <xs:complexType>
11577                                     <xs:complexContent>
11578                                         <xs:restriction base="ns_p:TierDescriptionDataType">
11579                                             <xs:sequence>
11580                                                 <xs:element minOccurs="0" name="tierId"
11581 type="ns_p:TierIdType"/>
11582                                                 <xs:element minOccurs="0" name="tierType"
11583 type="ns_p:TierTypeType"/>
11584                                                 <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
11585                                                 <xs:element minOccurs="0" name="description"
11586 type="ns_p:DescriptionType"/>
11587                                             </xs:sequence>
11588                                         </xs:restriction>
11589                                     </xs:complexContent>
11590                                 </xs:complexType>
11591                             </xs:element>
11592                             <xs:element maxOccurs="unbounded" minOccurs="0" name="boundaryDescription">
11593                                 <xs:complexType>
11594                                     <xs:complexContent>
11595                                         <xs:restriction base="ns_p:TierBoundaryDescriptionDataType">
11596                                             <xs:sequence>
11597                                                 <xs:element minOccurs="0" name="boundaryId"
11598 type="ns_p:TierBoundaryIdType"/>
11599                                                 <xs:element minOccurs="0" name="boundaryType"
11600 type="ns_p:TierBoundaryTypeType"/>
11601                                                 <xs:element minOccurs="0" name="switchToTierIdWhenLower"
11602 type="ns_p:TierIdType"/>
11603                                                 <xs:element minOccurs="0" name="switchToTierIdWhenHigher"
11604 type="ns_p:TierIdType"/>
11605                                                 <xs:element minOccurs="0" name="boundaryUnit"
11606 type="ns_p:UnitOfMeasurementType"/>
11607                                             </xs:sequence>
11608                                         </xs:restriction>
11609                                     </xs:complexContent>
11610                                 </xs:complexType>
11611                             </xs:element>
11612                             <xs:element minOccurs="0" name="incentiveDescription" maxOccurs="unbounded">
11613                                 <xs:complexType>
11614                                     <xs:complexContent>
11615                                         <xs:restriction base="ns_p:IncentiveDescriptionDataType">
11616                                             <xs:sequence>
11617                                                 <xs:element minOccurs="0" name="incentiveId"
11618 type="ns_p:IncentiveIdType"/>
11619                                                 <xs:element minOccurs="0" name="incentiveType"
11620 type="ns_p:IncentiveTypeType"/>
11621                                                 <xs:element minOccurs="0" name="incentivePriority"
11622 type="ns_p:IncentivePriorityType"/>
11623                                                 <xs:element minOccurs="0" name="currency"
11624 type="ns_p:CurrencyType"/>
11625                                                 <xs:element minOccurs="0" name="unit"
11626 type="ns_p:UnitOfMeasurementType"/>
11627                                                 <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
11628                                                 <xs:element minOccurs="0" name="description"
11629 type="ns_p:DescriptionType"/>
11630                                             </xs:sequence>

```

```

11631         </xs:restriction>
11632     </xs:complexContent>
11633 </xs:complexType>
11634 </xs:element>
11635 </xs:sequence>
11636 </xs:complexType>
11637 <xs:complexType name="IncentiveTableDescriptionDataType">
11638     <xs:sequence>
11639         <xs:element maxOccurs="unbounded" minOccurs="0" name="incentiveTableDescription"
11640 type="ns_p:IncentiveTableDescriptionType"> </xs:element>
11641     </xs:sequence>
11642 </xs:complexType>
11643 <xs:element name="incentiveTableDescriptionData"
11644 type="ns_p:IncentiveTableDescriptionDataType"/>
11645 <xs:complexType name="IncentiveTableDescriptionElementsType">
11646     <xs:sequence>
11647         <xs:element minOccurs="0" name="tariffDescription">
11648             <xs:complexType>
11649                 <xs:complexContent>
11650                     <xs:restriction base="ns_p:TariffDescriptionDataElementsType">
11651                         <xs:sequence>
11652                             <xs:element minOccurs="0" name="tariffId"
11653 type="ns_p:ElementTagType"/>
11654                             <xs:element minOccurs="0" name="measurementId"
11655 type="ns_p:ElementTagType"/>
11656                             <xs:element minOccurs="0" name="tariffWriteable"
11657 type="ns_p:ElementTagType"/>
11658                             <xs:element minOccurs="0" name="updateRequired"
11659 type="ns_p:ElementTagType"/>
11660                             <xs:element minOccurs="0" name="scopeType"
11661 type="ns_p:ElementTagType"/>
11662                             <xs:element minOccurs="0" name="label"
11663 type="ns_p:ElementTagType"/>
11664                             <xs:element minOccurs="0" name="description"
11665 type="ns_p:ElementTagType"/>
11666                             <xs:element minOccurs="0" name="slotIdSupport"
11667 type="ns_p:ElementTagType"/>
11668                         </xs:sequence>
11669                     </xs:restriction>
11670                 </xs:complexContent>
11671             </xs:complexType>
11672             <xs:element minOccurs="0" name="tier"
11673 type="ns_p:IncentiveTableDescriptionTierElementsType"> </xs:element>
11674         </xs:sequence>
11675     </xs:complexType>
11676 <xs:complexType name="IncentiveTableDescriptionTierElementsType">
11677     <xs:sequence>
11678         <xs:element minOccurs="0" name="tierDescription">
11679             <xs:complexType>
11680                 <xs:complexContent>
11681                     <xs:restriction base="ns_p:TierDescriptionDataElementsType">
11682                         <xs:sequence>
11683                             <xs:element minOccurs="0" name="tierId"
11684 type="ns_p:ElementTagType"/>
11685                             <xs:element minOccurs="0" name="tierType"
11686 type="ns_p:ElementTagType"/>
11687                             <xs:element minOccurs="0" name="label"
11688 type="ns_p:ElementTagType"/>
11689                             <xs:element minOccurs="0" name="description"
11690 type="ns_p:ElementTagType"/>
11691                         </xs:sequence>
11692                     </xs:restriction>
11693                 </xs:complexContent>
11694             </xs:complexType>
11695             <xs:element minOccurs="0" name="boundaryDescription">
11696                 <xs:complexType>
11697                     <xs:complexContent>
11698                         <xs:restriction base="ns_p:TierBoundaryDescriptionDataElementsType">
11699                             <xs:sequence>
11700                                 <xs:element minOccurs="0" name="boundaryId"
11701 type="ns_p:ElementTagType"/>
11702                                 <xs:element minOccurs="0" name="boundaryType"
11703 type="ns_p:ElementTagType"/>
11704                                 <xs:element minOccurs="0" name="switchToTierIdWhenLower"
11705 type="ns_p:ElementTagType"/>
11706                                 <xs:element minOccurs="0" name="switchToTierIdWhenLower"
11707 type="ns_p:ElementTagType"/>

```

```

11708         <xs:element minOccurs="0" name="switchToTierIdWhenHigher"
11709 type="ns_p:ElementTagType"/>
11710         <xs:element minOccurs="0" name="boundaryUnit"
11711 type="ns_p:ElementTagType"/>
11712     </xs:sequence>
11713 </xs:restriction>
11714 </xs:complexContent>
11715 </xs:complexType>
11716 </xs:element>
11717 <xs:element minOccurs="0" name="incentiveDescription">
11718     <xs:complexType>
11719         <xs:complexContent>
11720             <xs:restriction base="ns_p:IncentiveDescriptionDataElementsType">
11721                 <xs:sequence>
11722                     <xs:element minOccurs="0" name="incentiveId"
11723 type="ns_p:ElementTagType"/>
11724                     <xs:element minOccurs="0" name="incentiveType"
11725 type="ns_p:ElementTagType"/>
11726                     <xs:element minOccurs="0" name="incentivePriority"
11727 type="ns_p:ElementTagType"/>
11728                     <xs:element minOccurs="0" name="currency"
11729 type="ns_p:ElementTagType"/>
11730                     <xs:element minOccurs="0" name="unit"
11731 type="ns_p:ElementTagType"/>
11732                     <xs:element minOccurs="0" name="label"
11733 type="ns_p:ElementTagType"/>
11734                     <xs:element minOccurs="0" name="description"
11735 type="ns_p:ElementTagType"/>
11736                 </xs:sequence>
11737             </xs:restriction>
11738         </xs:complexContent>
11739     </xs:complexType>
11740 </xs:element>
11741 </xs:sequence>
11742 </xs:complexType>
11743 <xs:complexType name="IncentiveTableDescriptionDataElementsType">
11744     <xs:sequence>
11745         <xs:element minOccurs="0" name="incentiveTableDescription"
11746 type="ns_p:IncentiveTableDescriptionElementsType"> </xs:element>
11747     </xs:sequence>
11748 </xs:complexType>
11749 <xs:element name="incentiveTableDescriptionDataElements"
11750 type="ns_p:IncentiveTableDescriptionDataElementsType"/>
11751 <xs:complexType name="IncentiveTableDescriptionDataSelectorsType">
11752     <xs:sequence>
11753         <xs:element minOccurs="0" name="tariffDescription">
11754             <xs:complexType>
11755                 <xs:complexContent>
11756                     <xs:restriction base="ns_p:TariffDescriptionListDataSelectorsType">
11757                         <xs:sequence>
11758                             <xs:element minOccurs="0" name="tariffId"
11759 type="ns_p:TariffIdType"/>
11760                             <xs:element minOccurs="0" name="scopeType"
11761 type="ns_p:ScopeTypeType"/>
11762                         </xs:sequence>
11763                     </xs:restriction>
11764                 </xs:complexContent>
11765             </xs:complexType>
11766         </xs:element>
11767     </xs:sequence>
11768 </xs:complexType>
11769 <xs:element name="incentiveTableDescriptionDataSelectors"
11770 type="ns_p:IncentiveTableDescriptionDataSelectorsType"/>
11771 <xs:complexType name="IncentiveTableConstraintsType">
11772     <xs:sequence>
11773         <xs:element minOccurs="0" name="tariff">
11774             <xs:complexType>
11775                 <xs:complexContent>
11776                     <xs:restriction base="ns_p:TariffDataType">
11777                         <xs:sequence>
11778                             <xs:element minOccurs="0" name="tariffId"
11779 type="ns_p:TariffIdType"/>
11780                         </xs:sequence>
11781                     </xs:restriction>
11782                 </xs:complexContent>
11783             </xs:complexType>
11784         </xs:element>
11785         <xs:element name="tariffConstraints" minOccurs="0">

```

```

11786         <xs:complexType>
11787             <xs:complexContent>
11788                 <xs:restriction base="ns_p:TariffOverallConstraintsDataType">
11789                     <xs:sequence>
11790                         <xs:element minOccurs="0" name="maxTiersPerTariff"
type="ns_p:TierCountType"/>
11791                         <xs:element minOccurs="0" name="maxBoundariesPerTier"
type="ns_p:TierBoundaryCountType"/>
11792                         <xs:element minOccurs="0" name="maxIncentivesPerTier"
type="ns_p:IncentiveCountType"/>
11793                     </xs:sequence>
11794                 </xs:restriction>
11795             </xs:complexContent>
11796         </xs:complexType>
11797     </xs:element>
11798     <xs:element name="incentiveSlotConstraints" minOccurs="0">
11799         <xs:complexType>
11800             <xs:complexContent>
11801                 <xs:restriction base="ns_p:TimeTableConstraintsDataType">
11802                     <xs:sequence>
11803                         <xs:element minOccurs="0" name="slotCountMax"
type="ns_p:TimeSlotCountType"/>
11804                     </xs:sequence>
11805                 </xs:restriction>
11806             </xs:complexContent>
11807         </xs:complexType>
11808     </xs:element>
11809     </xs:sequence>
11810 </xs:complexType>
11811 </xs:complexType>
11812 <xs:complexType name="IncentiveTableConstraintsDataType">
11813     <xs:sequence>
11814         <xs:element maxOccurs="unbounded" minOccurs="0" name="incentiveTableConstraints"
type="ns_p:IncentiveTableConstraintsType"/>
11815     </xs:sequence>
11816 </xs:complexType>
11817 <xs:element name="incentiveTableConstraintsData"
type="ns_p:IncentiveTableConstraintsDataType"/>
11818 <xs:complexType name="IncentiveTableConstraintsElementsType">
11819     <xs:sequence>
11820         <xs:element minOccurs="0" name="tariff">
11821             <xs:complexType>
11822                 <xs:complexContent>
11823                     <xs:restriction base="ns_p:TariffDataElementsType">
11824                         <xs:sequence>
11825                             <xs:element minOccurs="0" name="tariffId"
type="ns_p:ElementTagType"/>
11826                             </xs:sequence>
11827                     </xs:restriction>
11828                 </xs:complexContent>
11829             </xs:complexType>
11830         </xs:element>
11831         <xs:element name="tariffConstraints" minOccurs="0">
11832             <xs:complexType>
11833                 <xs:complexContent>
11834                     <xs:restriction base="ns_p:TariffOverallConstraintsDataElementsType">
11835                         <xs:sequence>
11836                             <xs:element minOccurs="0" name="maxTiersPerTariff"
type="ns_p:ElementTagType"/>
11837                             <xs:element minOccurs="0" name="maxBoundariesPerTier"
type="ns_p:ElementTagType"/>
11838                             <xs:element minOccurs="0" name="maxIncentivesPerTier"
type="ns_p:ElementTagType"/>
11839                         </xs:sequence>
11840                     </xs:restriction>
11841                 </xs:complexContent>
11842             </xs:complexType>
11843         </xs:element>
11844         <xs:element name="incentiveSlotConstraints" minOccurs="0">
11845             <xs:complexType>
11846                 <xs:complexContent>
11847                     <xs:restriction base="ns_p:TimeTableConstraintsDataElementsType">
11848                         <xs:sequence>
11849                             <xs:element minOccurs="0" name="slotCountMax"
type="ns_p:ElementTagType"/>
11850                         </xs:sequence>
11851                     </xs:restriction>
11852                 </xs:complexContent>
11853             </xs:complexType>
11854         </xs:element>
11855     </xs:sequence>
11856 </xs:complexType>
11857 </xs:complexType>
11858 </xs:complexType>
11859 </xs:complexType>
11860 </xs:complexType>
11861 </xs:complexType>
11862 </xs:complexType>
11863 </xs:complexType>

```

```

11864         </xs:element>
11865     </xs:sequence>
11866 </xs:complexType>
11867 <xs:complexType name="IncentiveTableConstraintsDataElementsType">
11868     <xs:sequence>
11869         <xs:element minOccurs="0" name="incentiveTableConstraints"
11870 type="ns_p:IncentiveTableConstraintsDataElementsType"/>
11871     </xs:sequence>
11872 </xs:complexType>
11873 <xs:element name="incentiveTableConstraintsDataElements"
11874 type="ns_p:IncentiveTableConstraintsDataElementsType"/>
11875 <xs:complexType name="IncentiveTableConstraintsDataSelectorsType">
11876     <xs:sequence>
11877         <xs:element minOccurs="0" name="tariff">
11878             <xs:complexType>
11879                 <xs:complexContent>
11880                     <xs:restriction base="ns_p:TariffListDataSelectorsType">
11881                         <xs:sequence>
11882                             <xs:element minOccurs="0" name="tariffId"
11883 type="ns_p:TariffIdType"/>
11884                         </xs:sequence>
11885                     </xs:restriction>
11886                 </xs:complexContent>
11887             </xs:complexType>
11888         </xs:element>
11889     </xs:sequence>
11890 </xs:complexType>
11891 <xs:element name="incentiveTableConstraintsDataSelectors"
11892 type="ns_p:IncentiveTableConstraintsDataSelectorsType"/>
11893 </xs:schema>
11894
11895

```

11896

11897 A.1.2 NodeManagement

11898 File name: EEBus_SPINE_TS_NodeManagement.xsd

```

11899 <?xml version="1.0" encoding="UTF-8"?>
11900 <!--
11901     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
11902     Version 1.1.1
11903     2018-12-21
11904     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
11905     Source: https://www.eebus.org/en/specifications/
11906 -->
11907 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
11908 xmlns:xs="http://www.w3.org/2001/XMLSchema"
11909 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1" blockDefault="#all"
11910 elementFormDefault="qualified">
11911     <xs:annotation>
11912         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
11913 e.V. All rights reserved.</xs:documentation>
11914     </xs:annotation>
11915     <xs:include schemaLocation="EEBus_SPINE_TS_BindingManagement.xsd"/>
11916     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
11917     <xs:include schemaLocation="EEBus_SPINE_TS_NetworkManagement.xsd"/>
11918     <xs:include schemaLocation="EEBus_SPINE_TS_SubscriptionManagement.xsd"/>
11919     <xs:include schemaLocation="EEBus_SPINE_TS_UseCaseInformation.xsd"/>
11920     <xs:include schemaLocation="EEBus_SPINE_TS_Version.xsd"/>
11921     <xs:complexType name="NodeManagementSpecificationVersionListType">
11922         <xs:sequence>
11923             <xs:element maxOccurs="unbounded" minOccurs="0" name="specificationVersion"
11924 type="ns_p:SpecificationVersionDataType"/>
11925         </xs:sequence>
11926     </xs:complexType>
11927     <xs:complexType name="NodeManagementDetailedDiscoveryDeviceInformationType">
11928         <xs:sequence>
11929             <xs:element name="description" minOccurs="0">
11930                 <xs:complexType>
11931                     <xs:complexContent>
11932                         <xs:restriction
11933 base="ns_p:NetworkManagementDeviceDescriptionDataType">
11934                             <xs:sequence>
11935                                 <xs:element name="deviceAddress" minOccurs="0">

```

```

11937         <xs:complexType>
11938             <xs:complexContent>
11939                 <xs:restriction base="ns_p:DeviceAddressType">
11940                     <xs:sequence>
11941                         <xs:element ref="ns_p:device"
11942 minOccurs="0"/>
11943                     </xs:sequence>
11944                 </xs:restriction>
11945             </xs:complexContent>
11946         </xs:complexType>
11947     </xs:element>
11948     <xs:element name="deviceType" type="ns_p:DeviceTypeType"
11949 minOccurs="0"/>
11950     <xs:element minOccurs="0"
11951 name="networkManagementResponsibleAddress" type="ns_p:FeatureAddressType"/>
11952     <xs:element minOccurs="0" name="nativeSetup"
11953 type="ns_p:NetworkManagementNativeSetupType"/>
11954     <xs:element minOccurs="0" name="technologyAddress"
11955 type="ns_p:NetworkManagementTechnologyAddressType"/>
11956     <xs:element minOccurs="0"
11957 name="communicationsTechnologyInformation"
11958 type="ns_p:NetworkManagementCommunicationsTechnologyInformationType"/>
11959     <xs:element minOccurs="0" name="networkFeatureSet"
11960 type="ns_p:NetworkManagementFeatureSetType"/>
11961     <xs:element minOccurs="0" name="lastStateChange"
11962 type="ns_p:NetworkManagementStateChangeType"/>
11963     <xs:element minOccurs="0" name="minimumTrustLevel"
11964 type="ns_p:NetworkManagementMinimumTrustLevelType"/>
11965     <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
11966     <xs:element minOccurs="0" name="description"
11967 type="ns_p:DescriptionType"/>
11968     </xs:sequence>
11969 </xs:restriction>
11970 </xs:complexContent>
11971 </xs:complexType>
11972 </xs:element>
11973 </xs:sequence>
11974 </xs:complexType>
11975 <xs:complexType name="NodeManagementDetailedDiscoveryEntityInformationType">
11976     <xs:sequence>
11977         <xs:element name="description" minOccurs="0">
11978             <xs:complexType>
11979                 <xs:complexContent>
11980                     <xs:restriction
11981 base="ns_p:NetworkManagementEntityDescriptionDataType">
11982                         <xs:sequence>
11983                             <xs:element name="entityAddress" minOccurs="0">
11984                                 <xs:complexType>
11985                                     <xs:complexContent>
11986                                         <xs:restriction base="ns_p:EntityAddressType">
11987                                             <xs:sequence>
11988                                                 <xs:sequence>
11989                                                     <xs:element maxOccurs="unbounded"
11990 ref="ns_p:entity" minOccurs="0"/>
11991                                                 </xs:sequence>
11992                                             </xs:sequence>
11993                                         </xs:restriction>
11994                                     </xs:complexContent>
11995                                 </xs:complexType>
11996                             </xs:element>
11997                             <xs:element name="entityType" type="ns_p:EntityTypeType"
11998 minOccurs="0"/>
11999                             <xs:element minOccurs="0" name="lastStateChange"
12000 type="ns_p:NetworkManagementStateChangeType"/>
12001                             <xs:element minOccurs="0" name="minimumTrustLevel"
12002 type="ns_p:NetworkManagementMinimumTrustLevelType"/>
12003                             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
12004                             <xs:element minOccurs="0" name="description"
12005 type="ns_p:DescriptionType"/>
12006                         </xs:sequence>
12007                     </xs:restriction>
12008                 </xs:complexContent>
12009             </xs:complexType>
12010         </xs:element>
12011     </xs:sequence>
12012 </xs:complexType>
12013 <xs:complexType name="NodeManagementDetailedDiscoveryFeatureInformationType">
12014     <xs:sequence>

```

```

12015         <xs:element name="description" minOccurs="0">
12016             <xs:complexType>
12017                 <xs:complexContent>
12018                     <xs:restriction
12019 base="ns_p:NetworkManagementFeatureDescriptionDataType">
12020                         <xs:sequence>
12021                             <xs:element name="featureAddress" minOccurs="0">
12022                                 <xs:complexType>
12023                                     <xs:complexContent>
12024                                         <xs:restriction base="ns_p:FeatureAddressType">
12025                                             <xs:sequence>
12026                                                 <xs:sequence>
12027                                                     <xs:element maxOccurs="unbounded"
12028 ref="ns_p:entity" minOccurs="0"/>
12029                                                     </xs:sequence>
12030                                                 <xs:sequence>
12031                                                     <xs:element ref="ns_p:feature"
12032 minOccurs="0"/>
12033                                                     </xs:sequence>
12034                                                 </xs:sequence>
12035                                             </xs:restriction>
12036                                         </xs:complexContent>
12037                                     </xs:complexType>
12038                                 </xs:element>
12039                                 <xs:element minOccurs="0" name="featureType"
12040 type="ns_p:FeatureTypeType"/>
12041                                 <xs:element maxOccurs="unbounded" minOccurs="0"
12042 name="specificUsage" type="ns_p:FeatureSpecificUsageType"/>
12043                                 <xs:element minOccurs="0" name="featureGroup"
12044 type="ns_p:FeatureGroupType"/>
12045                                 <xs:element name="role" type="ns_p:RoleType" minOccurs="0"/>
12046                                 <xs:element minOccurs="0" name="supportedFunction"
12047 type="ns_p:FunctionPropertyType" maxOccurs="unbounded"/>
12048                                 <xs:element minOccurs="0" name="lastStateChange"
12049 type="ns_p:NetworkManagementStateChangeType"/>
12050                                 <xs:element minOccurs="0" name="minimumTrustLevel"
12051 type="ns_p:NetworkManagementMinimumTrustLevelType"/>
12052                                 <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
12053                                 <xs:element minOccurs="0" name="description"
12054 type="ns_p:DescriptionType"/>
12055                                 <xs:element minOccurs="0" name="maxResponseDelay"
12056 type="ns_p:MaxResponseDelayType"/>
12057                             </xs:sequence>
12058                         </xs:restriction>
12059                     </xs:complexContent>
12060                 </xs:complexType>
12061             </xs:element>
12062         </xs:sequence>
12063     </xs:complexType>
12064     <xs:complexType name="NodeManagementDetailedDiscoveryDataType">
12065         <xs:sequence>
12066             <xs:element maxOccurs="1" name="specificationVersionList" minOccurs="0"
12067 type="ns_p:NodeManagementSpecificationVersionListType"/>
12068             <xs:element name="deviceInformation"
12069 type="ns_p:NodeManagementDetailedDiscoveryDeviceInformationType" minOccurs="0"/>
12070             <xs:element name="entityInformation" maxOccurs="unbounded" minOccurs="0"
12071 type="ns_p:NodeManagementDetailedDiscoveryEntityInformationType"/>
12072             <xs:element name="featureInformation" maxOccurs="unbounded" minOccurs="0"
12073 type="ns_p:NodeManagementDetailedDiscoveryFeatureInformationType"/>
12074         </xs:sequence>
12075     </xs:complexType>
12076     <xs:element name="nodeManagementDetailedDiscoveryData"
12077 type="ns_p:NodeManagementDetailedDiscoveryDataType"/>
12078     <xs:complexType name="NodeManagementSpecificationVersionListElementsType">
12079         <xs:sequence>
12080             <xs:element maxOccurs="1" name="specificationVersion" minOccurs="0">
12081                 <xs:complexType>
12082                     <xs:complexContent>
12083                         <xs:restriction base="ns_p:SpecificationVersionDataElementsType"/>
12084                     </xs:complexContent>
12085                 </xs:complexType>
12086             </xs:element>
12087         </xs:sequence>
12088     </xs:complexType>
12089     <xs:complexType name="NodeManagementDetailedDiscoveryDeviceInformationElementsType">
12090         <xs:sequence>
12091             <xs:element name="description" minOccurs="0">
12092                 <xs:complexType>

```



```

12093         <xs:complexContent>
12094             <xs:restriction
12095 base="ns_p:NetworkManagementDeviceDescriptionDataElementsType">
12096                 <xs:sequence>
12097                     <xs:element name="deviceAddress" minOccurs="0">
12098                         <xs:complexType>
12099                             <xs:complexContent>
12100                                 <xs:restriction
12101 base="ns_p:DeviceAddressElementsType">
12102                                     <xs:sequence>
12103                                         <xs:element minOccurs="0" name="device"
12104 type="ns_p:ElementTagType"/>
12105                                     </xs:sequence>
12106                                 </xs:restriction>
12107                             </xs:complexContent>
12108                         </xs:complexType>
12109                     </xs:element>
12110                     <xs:element name="deviceType" minOccurs="0"
12111 type="ns_p:ElementTagType"/>
12112                     <xs:element minOccurs="0"
12113 name="networkManagementResponsibleAddress" type="ns_p:ElementTagType"/>
12114                     <xs:element minOccurs="0" name="nativeSetup"
12115 type="ns_p:ElementTagType"/>
12116                     <xs:element minOccurs="0" name="technologyAddress"
12117 type="ns_p:ElementTagType"/>
12118                     <xs:element minOccurs="0"
12119 name="communicationsTechnologyInformation" type="ns_p:ElementTagType"/>
12120                     <xs:element minOccurs="0" name="networkFeatureSet"
12121 type="ns_p:ElementTagType"/>
12122                     <xs:element minOccurs="0" name="lastStateChange"
12123 type="ns_p:ElementTagType"/>
12124                     <xs:element minOccurs="0" name="minimumTrustLevel"
12125 type="ns_p:ElementTagType"/>
12126                     <xs:element minOccurs="0" name="label"
12127 type="ns_p:ElementTagType"/>
12128                     <xs:element minOccurs="0" name="description"
12129 type="ns_p:ElementTagType"/>
12130                 </xs:sequence>
12131             </xs:restriction>
12132         </xs:complexContent>
12133     </xs:complexType>
12134 </xs:element>
12135 </xs:sequence>
12136 </xs:complexType>
12137 <xs:complexType name="NodeManagementDetailedDiscoveryEntityInformationElementsType">
12138     <xs:sequence>
12139         <xs:element name="description" minOccurs="0">
12140             <xs:complexType>
12141                 <xs:complexContent>
12142                     <xs:restriction
12143 base="ns_p:NetworkManagementEntityDescriptionDataElementsType">
12144                         <xs:sequence>
12145                             <xs:element name="entityAddress" minOccurs="0">
12146                                 <xs:complexType>
12147                                     <xs:complexContent>
12148                                         <xs:restriction
12149 base="ns_p:EntityAddressElementsType">
12150                                             <xs:sequence>
12151                                                 <xs:element minOccurs="0" name="entity"
12152 type="ns_p:ElementTagType"/>
12153                                             </xs:sequence>
12154                                         </xs:restriction>
12155                                     </xs:complexContent>
12156                                 </xs:complexType>
12157                             </xs:element>
12158                             <xs:element name="entityType" minOccurs="0"
12159 type="ns_p:ElementTagType"/>
12160                             <xs:element minOccurs="0" name="lastStateChange"
12161 type="ns_p:ElementTagType"/>
12162                             <xs:element minOccurs="0" name="minimumTrustLevel"
12163 type="ns_p:ElementTagType"/>
12164                             <xs:element minOccurs="0" name="label"
12165 type="ns_p:ElementTagType"/>
12166                             <xs:element minOccurs="0" name="description"
12167 type="ns_p:ElementTagType"/>
12168                         </xs:sequence>
12169                     </xs:restriction>
12170                 </xs:complexContent>

```

```
12171         </xs:complexType>
12172     </xs:element>
12173 </xs:sequence>
12174 </xs:complexType>
12175 <xs:complexType name="NodeManagementDetailedDiscoveryFeatureInformationElementsType">
12176     <xs:sequence>
12177         <xs:element name="description" minOccurs="0">
12178             <xs:complexType>
12179                 <xs:complexContent>
12180                     <xs:restriction
12181 base="ns_p:NetworkManagementFeatureDescriptionDataElementsType">
12182                         <xs:sequence>
12183                             <xs:element name="featureAddress" minOccurs="0">
12184                                 <xs:complexType>
12185                                     <xs:complexContent>
12186                                         <xs:restriction
12187 base="ns_p:FeatureAddressElementsType">
12188                                             <xs:sequence>
12189                                                 <xs:element minOccurs="0" name="entity"
12190 type="ns_p:ElementTagType"/>
12191                                                 <xs:element minOccurs="0" name="feature"
12192 type="ns_p:ElementTagType"/>
12193                                             </xs:sequence>
12194                                         </xs:restriction>
12195                                     </xs:complexContent>
12196                                 </xs:complexType>
12197                             </xs:element>
12198                             <xs:element minOccurs="0" name="featureType"
12199 type="ns_p:ElementTagType"/>
12200                             <xs:element minOccurs="0" name="specificUsage"
12201 type="ns_p:ElementTagType"/>
12202                             <xs:element minOccurs="0" name="featureGroup"
12203 type="ns_p:ElementTagType"/>
12204                             <xs:element minOccurs="0" name="role"
12205 type="ns_p:ElementTagType"/>
12206                             <xs:element minOccurs="0" name="supportedFunction"
12207 type="ns_p:FunctionPropertyElementsType"/>
12208                             <xs:element minOccurs="0" name="lastStateChange"
12209 type="ns_p:ElementTagType"/>
12210                             <xs:element minOccurs="0" name="minimumTrustLevel"
12211 type="ns_p:ElementTagType"/>
12212                             <xs:element minOccurs="0" name="label"
12213 type="ns_p:ElementTagType"/>
12214                             <xs:element minOccurs="0" name="description"
12215 type="ns_p:ElementTagType"/>
12216                             <xs:element minOccurs="0" name="maxResponseDelay"
12217 type="ns_p:ElementTagType"/>
12218                         </xs:sequence>
12219                     </xs:restriction>
12220                 </xs:complexContent>
12221             </xs:complexType>
12222         </xs:element>
12223     </xs:sequence>
12224 </xs:complexType>
12225 <xs:complexType name="NodeManagementDetailedDiscoveryDataElementsType">
12226     <xs:sequence>
12227         <xs:element maxOccurs="1" name="specificationVersionList" minOccurs="0"
12228 type="ns_p:NodeManagementSpecificationVersionListElementsType"/>
12229         <xs:element name="deviceInformation"
12230 type="ns_p:NodeManagementDetailedDiscoveryDeviceInformationElementsType" minOccurs="0"/>
12231         <xs:element name="entityInformation" minOccurs="0"
12232 type="ns_p:NodeManagementDetailedDiscoveryEntityInformationElementsType"/>
12233         <xs:element name="featureInformation" minOccurs="0"
12234 type="ns_p:NodeManagementDetailedDiscoveryFeatureInformationElementsType"/>
12235     </xs:sequence>
12236 </xs:complexType>
12237 <xs:element name="nodeManagementDetailedDiscoveryDataElements"
12238 type="ns_p:NodeManagementDetailedDiscoveryDataElementsType"/>
12239 <xs:complexType name="NodeManagementDetailedDiscoveryDataSelectorsType">
12240     <xs:sequence>
12241         <xs:element minOccurs="0" name="deviceInformation">
12242             <xs:complexType>
12243                 <xs:complexContent>
12244                     <xs:restriction
12245 base="ns_p:NetworkManagementDeviceDescriptionListDataSelectorsType">
12246                         <xs:sequence>
12247                             <xs:element minOccurs="0" name="deviceAddress"
12248 type="ns_p:DeviceAddressType"/>
```

```

12249         <xs:element minOccurs="0" name="deviceType"
12250 type="ns_p:DeviceTypeType"/>
12251     </xs:sequence>
12252 </xs:restriction>
12253 </xs:complexContent>
12254 </xs:complexType>
12255 </xs:element>
12256 <xs:element minOccurs="0" name="entityInformation">
12257 <xs:complexType>
12258 <xs:complexContent>
12259 <xs:restriction
12260 base="ns_p:NetworkManagementEntityDescriptionListDataSelectorsType">
12261 <xs:sequence>
12262 <xs:element minOccurs="0" name="entityAddress"
12263 type="ns_p:EntityAddressType"/>
12264 <xs:element minOccurs="0" name="entityType"
12265 type="ns_p:EntityTypeType"/>
12266 </xs:sequence>
12267 </xs:restriction>
12268 </xs:complexContent>
12269 </xs:complexType>
12270 </xs:element>
12271 <xs:element minOccurs="0" name="featureInformation">
12272 <xs:complexType>
12273 <xs:complexContent>
12274 <xs:restriction
12275 base="ns_p:NetworkManagementFeatureDescriptionListDataSelectorsType">
12276 <xs:sequence>
12277 <xs:element minOccurs="0" name="featureAddress"
12278 type="ns_p:FeatureAddressType"/>
12279 <xs:element minOccurs="0" name="featureType"
12280 type="ns_p:FeatureTypeType"/>
12281 </xs:sequence>
12282 </xs:restriction>
12283 </xs:complexContent>
12284 </xs:complexType>
12285 </xs:element>
12286 </xs:sequence>
12287 </xs:complexType>
12288 <xs:element name="nodeManagementDetailedDiscoveryDataSelectors"
12289 type="ns_p:NodeManagementDetailedDiscoveryDataSelectorsType"/>
12290 <xs:complexType name="NodeManagementBindingDataType">
12291 <xs:sequence>
12292 <xs:element maxOccurs="unbounded" minOccurs="0" name="bindingEntry">
12293 <xs:complexType>
12294 <xs:complexContent>
12295 <xs:restriction base="ns_p:BindingManagementEntryDataType">
12296 <xs:sequence>
12297 <xs:element name="bindingId" type="ns_p:BindingIdType"
12298 minOccurs="0"/>
12299 <xs:element name="clientAddress" minOccurs="0"
12300 type="ns_p:FeatureAddressType"/>
12301 <xs:element name="serverAddress" minOccurs="0"
12302 type="ns_p:FeatureAddressType"/>
12303 <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
12304 <xs:element minOccurs="0" name="description"
12305 type="ns_p:DescriptionType"/>
12306 </xs:sequence>
12307 </xs:restriction>
12308 </xs:complexContent>
12309 </xs:complexType>
12310 </xs:element>
12311 </xs:sequence>
12312 </xs:complexType>
12313 <xs:element name="nodeManagementBindingData" type="ns_p:NodeManagementBindingDataType"/>
12314 <xs:complexType name="NodeManagementBindingDataElementsType">
12315 <xs:sequence>
12316 <xs:element minOccurs="0" name="bindingEntry">
12317 <xs:complexType>
12318 <xs:complexContent>
12319 <xs:restriction base="ns_p:BindingManagementEntryDataElementsType">
12320 <xs:sequence>
12321 <xs:element minOccurs="0" name="bindingId"
12322 type="ns_p:ElementTagType"/>
12323 <xs:element minOccurs="0" name="clientAddress"
12324 type="ns_p:FeatureAddressElementsType"/>
12325 <xs:element minOccurs="0" name="serverAddress"
12326 type="ns_p:FeatureAddressElementsType"/>

```

```

12327         <xs:element minOccurs="0" name="label"
12328 type="ns_p:ElementTagType"/>
12329         <xs:element minOccurs="0" name="description"
12330 type="ns_p:ElementTagType"/>
12331     </xs:sequence>
12332 </xs:restriction>
12333 </xs:complexContent>
12334 </xs:complexType>
12335 </xs:element>
12336 </xs:sequence>
12337 </xs:complexType>
12338 <xs:element name="nodeManagementBindingDataElements"
12339 type="ns_p:NodeManagementBindingDataElementsType"/>
12340 <xs:complexType name="NodeManagementBindingDataSelectorsType">
12341     <xs:sequence>
12342         <xs:element minOccurs="0" name="bindingEntry">
12343             <xs:complexType>
12344                 <xs:complexContent>
12345                     <xs:restriction
12346 base="ns_p:BindingManagementEntryListDataSelectorsType"/>
12347                 </xs:complexContent>
12348             </xs:complexType>
12349         </xs:element>
12350     </xs:sequence>
12351 </xs:complexType>
12352 <xs:element name="nodeManagementBindingDataSelectors"
12353 type="ns_p:NodeManagementBindingDataSelectorsType"/>
12354 <xs:complexType name="NodeManagementBindingRequestCallType">
12355     <xs:sequence>
12356         <xs:element minOccurs="0" name="bindingRequest">
12357             <xs:complexType>
12358                 <xs:complexContent>
12359                     <xs:restriction base="ns_p:BindingManagementRequestCallType">
12360                         <xs:sequence>
12361                             <xs:element minOccurs="0" name="clientAddress"
12362 type="ns_p:FeatureAddressType"/>
12363                             <xs:element minOccurs="0" name="serverAddress"
12364 type="ns_p:FeatureAddressType"/>
12365                             <xs:element minOccurs="0" name="serverFeatureType"
12366 type="ns_p:FeatureTypeType"/>
12367                         </xs:sequence>
12368                     </xs:restriction>
12369                 </xs:complexContent>
12370             </xs:complexType>
12371         </xs:element>
12372     </xs:sequence>
12373 </xs:complexType>
12374 <xs:element name="nodeManagementBindingRequestCall"
12375 type="ns_p:NodeManagementBindingRequestCallType"/>
12376 <xs:complexType name="NodeManagementBindingRequestCallElementsType">
12377     <xs:sequence>
12378         <xs:element minOccurs="0" name="bindingRequest">
12379             <xs:complexType>
12380                 <xs:complexContent>
12381                     <xs:restriction base="ns_p:BindingManagementRequestCallElementsType">
12382                         <xs:sequence>
12383                             <xs:element minOccurs="0" name="clientAddress"
12384 type="ns_p:FeatureAddressElementsType"/>
12385                             <xs:element minOccurs="0" name="serverAddress"
12386 type="ns_p:FeatureAddressElementsType"/>
12387                             <xs:element minOccurs="0" name="serverFeatureType"
12388 type="ns_p:ElementTagType"/>
12389                         </xs:sequence>
12390                     </xs:restriction>
12391                 </xs:complexContent>
12392             </xs:complexType>
12393         </xs:element>
12394     </xs:sequence>
12395 </xs:complexType>
12396 <xs:element name="nodeManagementBindingRequestCallElements"
12397 type="ns_p:NodeManagementBindingRequestCallElementsType"/>
12398 <xs:complexType name="NodeManagementBindingDeleteCallType">
12399     <xs:sequence>
12400         <xs:element minOccurs="0" name="bindingDelete">
12401             <xs:complexType>
12402                 <xs:complexContent>
12403                     <xs:restriction base="ns_p:BindingManagementDeleteCallType">
12404                         <xs:sequence>

```

```

12405         <xs:element minOccurs="0" name="bindingId"
12406 type="ns_p:BindingIdType"/>
12407         <xs:element minOccurs="0" name="clientAddress"
12408 type="ns_p:FeatureAddressType"/>
12409         <xs:element minOccurs="0" name="serverAddress"
12410 type="ns_p:FeatureAddressType"/>
12411     </xs:sequence>
12412 </xs:restriction>
12413 </xs:complexContent>
12414 </xs:complexType>
12415 </xs:element>
12416 </xs:sequence>
12417 </xs:complexType>
12418 <xs:element name="nodeManagementBindingDeleteCall"
12419 type="ns_p:NodeManagementBindingDeleteCallType"/>
12420 <xs:complexType name="NodeManagementBindingDeleteCallElementsType">
12421     <xs:sequence>
12422         <xs:element minOccurs="0" name="bindingDelete">
12423             <xs:complexType>
12424                 <xs:complexContent>
12425                     <xs:restriction base="ns_p:BindingManagementDeleteCallElementsType">
12426                         <xs:sequence>
12427                             <xs:element minOccurs="0" name="bindingId"
12428 type="ns_p:ElementTagType"/>
12429                             <xs:element minOccurs="0" name="clientAddress"
12430 type="ns_p:FeatureAddressElementsType"/>
12431                             <xs:element minOccurs="0" name="serverAddress"
12432 type="ns_p:FeatureAddressElementsType"/>
12433                         </xs:sequence>
12434                     </xs:restriction>
12435                 </xs:complexContent>
12436             </xs:complexType>
12437         </xs:element>
12438     </xs:sequence>
12439 </xs:complexType>
12440 <xs:element name="nodeManagementBindingDeleteCallElements"
12441 type="ns_p:NodeManagementBindingDeleteCallElementsType"/>
12442 <xs:complexType name="NodeManagementSubscriptionDataType">
12443     <xs:sequence>
12444         <xs:element maxOccurs="unbounded" minOccurs="0" name="subscriptionEntry">
12445             <xs:complexType>
12446                 <xs:complexContent>
12447                     <xs:restriction base="ns_p:SubscriptionManagementEntryDataType">
12448                         <xs:sequence>
12449                             <xs:element name="subscriptionId"
12450 type="ns_p:SubscriptionIdType" minOccurs="0"/>
12451                             <xs:element name="clientAddress" minOccurs="0"
12452 type="ns_p:FeatureAddressType"/>
12453                             <xs:element name="serverAddress" minOccurs="0"
12454 type="ns_p:FeatureAddressType"/>
12455                             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
12456                             <xs:element minOccurs="0" name="description"
12457 type="ns_p:DescriptionType"/>
12458                         </xs:sequence>
12459                     </xs:restriction>
12460                 </xs:complexContent>
12461             </xs:complexType>
12462         </xs:element>
12463     </xs:sequence>
12464 </xs:complexType>
12465 <xs:element name="nodeManagementSubscriptionData"
12466 type="ns_p:NodeManagementSubscriptionDataType"/>
12467 <xs:complexType name="NodeManagementSubscriptionDataElementsType">
12468     <xs:sequence>
12469         <xs:element minOccurs="0" name="subscriptionEntry">
12470             <xs:complexType>
12471                 <xs:complexContent>
12472                     <xs:restriction
12473 base="ns_p:SubscriptionManagementEntryDataElementsType">
12474                         <xs:sequence>
12475                             <xs:element minOccurs="0" name="subscriptionId"
12476 type="ns_p:ElementTagType"/>
12477                             <xs:element minOccurs="0" name="clientAddress"
12478 type="ns_p:FeatureAddressElementsType"/>
12479                             <xs:element minOccurs="0" name="serverAddress"
12480 type="ns_p:FeatureAddressElementsType"/>
12481                             <xs:element minOccurs="0" name="label"
12482 type="ns_p:ElementTagType"/>

```

```

12483         <xs:element minOccurs="0" name="description"
12484 type="ns_p:ElementTagType"/>
12485     </xs:sequence>
12486 </xs:restriction>
12487 </xs:complexContent>
12488 </xs:complexType>
12489 </xs:element>
12490 </xs:sequence>
12491 </xs:complexType>
12492 <xs:element name="nodeManagementSubscriptionDataElements"
12493 type="ns_p:NodeManagementSubscriptionDataElementsType"/>
12494 <xs:complexType name="NodeManagementSubscriptionDataSelectorsType">
12495     <xs:sequence>
12496         <xs:element minOccurs="0" name="subscriptionEntry">
12497             <xs:complexType>
12498                 <xs:complexContent>
12499                     <xs:restriction
12500 base="ns_p:SubscriptionManagementEntryListDataSelectorsType">
12501                         <xs:sequence>
12502                             <xs:element minOccurs="0" name="subscriptionId"
12503 type="ns_p:SubscriptionIdType"/>
12504                             <xs:element minOccurs="0" name="clientAddress"
12505 type="ns_p:FeatureAddressType"/>
12506                             <xs:element minOccurs="0" name="serverAddress"
12507 type="ns_p:FeatureAddressType"/>
12508                         </xs:sequence>
12509                     </xs:restriction>
12510                 </xs:complexContent>
12511             </xs:complexType>
12512         </xs:element>
12513     </xs:sequence>
12514 </xs:complexType>
12515 <xs:element name="nodeManagementSubscriptionDataSelectors"
12516 type="ns_p:NodeManagementSubscriptionDataSelectorsType"/>
12517 <xs:complexType name="NodeManagementSubscriptionRequestCallType">
12518     <xs:sequence>
12519         <xs:element minOccurs="0" name="subscriptionRequest">
12520             <xs:complexType>
12521                 <xs:complexContent>
12522                     <xs:restriction base="ns_p:SubscriptionManagementRequestCallType">
12523                         <xs:sequence>
12524                             <xs:element minOccurs="0" name="clientAddress"
12525 type="ns_p:FeatureAddressType"/>
12526                             <xs:element minOccurs="0" name="serverAddress"
12527 type="ns_p:FeatureAddressType"/>
12528                             <xs:element minOccurs="0" name="serverFeatureType"
12529 type="ns_p:FeatureTypeType"/>
12530                         </xs:sequence>
12531                     </xs:restriction>
12532                 </xs:complexContent>
12533             </xs:complexType>
12534         </xs:element>
12535     </xs:sequence>
12536 </xs:complexType>
12537 <xs:element name="nodeManagementSubscriptionRequestCall"
12538 type="ns_p:NodeManagementSubscriptionRequestCallType"/>
12539 <xs:complexType name="NodeManagementSubscriptionRequestCallElementsType">
12540     <xs:sequence>
12541         <xs:element minOccurs="0" name="subscriptionRequest">
12542             <xs:complexType>
12543                 <xs:complexContent>
12544                     <xs:restriction
12545 base="ns_p:SubscriptionManagementRequestCallElementsType">
12546                         <xs:sequence>
12547                             <xs:element minOccurs="0" name="clientAddress"
12548 type="ns_p:FeatureAddressElementsType"/>
12549                             <xs:element minOccurs="0" name="serverAddress"
12550 type="ns_p:FeatureAddressElementsType"/>
12551                             <xs:element minOccurs="0" name="serverFeatureType"
12552 type="ns_p:ElementTagType"/>
12553                         </xs:sequence>
12554                     </xs:restriction>
12555                 </xs:complexContent>
12556             </xs:complexType>
12557         </xs:element>
12558     </xs:sequence>
12559 </xs:complexType>

```

```

12560     <xs:element name="nodeManagementSubscriptionRequestCallElements"
12561 type="ns_p:NodeManagementSubscriptionRequestCallElementsType"/>
12562     <xs:complexType name="NodeManagementSubscriptionDeleteCallType">
12563       <xs:sequence>
12564         <xs:element minOccurs="0" name="subscriptionDelete">
12565           <xs:complexType>
12566             <xs:complexContent>
12567               <xs:restriction base="ns_p:SubscriptionManagementDeleteCallType">
12568                 <xs:sequence>
12569                   <xs:element minOccurs="0" name="subscriptionId"
12570 type="ns_p:SubscriptionIdType"/>
12571                   <xs:element minOccurs="0" name="clientAddress"
12572 type="ns_p:FeatureAddressType"/>
12573                   <xs:element minOccurs="0" name="serverAddress"
12574 type="ns_p:FeatureAddressType"/>
12575                 </xs:sequence>
12576               </xs:restriction>
12577             </xs:complexContent>
12578           </xs:complexType>
12579         </xs:element>
12580       </xs:sequence>
12581     </xs:complexType>
12582     <xs:element name="nodeManagementSubscriptionDeleteCall"
12583 type="ns_p:NodeManagementSubscriptionDeleteCallType"/>
12584     <xs:complexType name="NodeManagementSubscriptionDeleteCallElementsType">
12585       <xs:sequence>
12586         <xs:element minOccurs="0" name="subscriptionDelete">
12587           <xs:complexType>
12588             <xs:complexContent>
12589               <xs:restriction
12590 base="ns_p:SubscriptionManagementDeleteCallElementsType">
12591                 <xs:sequence>
12592                   <xs:element minOccurs="0" name="subscriptionId"
12593 type="ns_p:ElementTagType"/>
12594                   <xs:element minOccurs="0" name="clientAddress"
12595 type="ns_p:FeatureAddressElementsType"/>
12596                   <xs:element minOccurs="0" name="serverAddress"
12597 type="ns_p:FeatureAddressElementsType"/>
12598                 </xs:sequence>
12599               </xs:restriction>
12600             </xs:complexContent>
12601           </xs:complexType>
12602         </xs:element>
12603       </xs:sequence>
12604     </xs:complexType>
12605     <xs:element name="nodeManagementSubscriptionDeleteCallElements"
12606 type="ns_p:NodeManagementSubscriptionDeleteCallElementsType"/>
12607     <xs:complexType name="NodeManagementDestinationDataType">
12608       <xs:sequence>
12609         <xs:element minOccurs="0" name="deviceDescription">
12610           <xs:complexType>
12611             <xs:complexContent>
12612               <xs:restriction
12613 base="ns_p:NetworkManagementDeviceDescriptionDataType">
12614                 <xs:sequence>
12615                   <xs:element minOccurs="0" name="deviceAddress"
12616 type="ns_p:DeviceAddressType"/>
12617                   <xs:element minOccurs="0"
12618 name="communicationsTechnologyInformation"
12619 type="ns_p:NetworkManagementCommunicationsTechnologyInformationType"/>
12620                   <xs:element minOccurs="0" name="networkFeatureSet"
12621 type="ns_p:NetworkManagementFeatureSetType"/>
12622                   <xs:element minOccurs="0" name="lastStateChange"
12623 type="ns_p:NetworkManagementStateChangeType"/>
12624                   <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
12625                 </xs:sequence>
12626               </xs:restriction>
12627             </xs:complexContent>
12628           </xs:complexType>
12629         </xs:element>
12630       </xs:sequence>
12631     </xs:complexType>
12632     <xs:element name="nodeManagementDestinationData"
12633 type="ns_p:NodeManagementDestinationDataType"/>
12634     <xs:complexType name="NodeManagementDestinationDataElementsType">
12635       <xs:sequence>
12636         <xs:element minOccurs="0" name="deviceDescription">
12637           <xs:complexType>

```

```

12638         <xs:complexContent>
12639             <xs:restriction
12640 base="ns_p:NetworkManagementDeviceDescriptionDataElementsType">
12641                 <xs:sequence>
12642                     <xs:element minOccurs="0" name="deviceAddress"
12643 type="ns_p:DeviceAddressElementsType"/>
12644                     <xs:element minOccurs="0"
12645 name="communicationsTechnologyInformation" type="ns_p:ElementTagType"/>
12646                     <xs:element minOccurs="0" name="networkFeatureSet"
12647 type="ns_p:ElementTagType"/>
12648                     <xs:element minOccurs="0" name="lastStateChange"
12649 type="ns_p:ElementTagType"/>
12650                     <xs:element minOccurs="0" name="label"
12651 type="ns_p:ElementTagType"/>
12652                 </xs:sequence>
12653             </xs:restriction>
12654         </xs:complexContent>
12655     </xs:complexType>
12656 </xs:element>
12657 </xs:sequence>
12658 </xs:complexType>
12659 <xs:element name="nodeManagementDestinationDataElements"
12660 type="ns_p:NodeManagementDestinationDataElementsType"/>
12661 <xs:complexType name="NodeManagementDestinationListDataType">
12662     <xs:sequence>
12663         <xs:element maxOccurs="unbounded" minOccurs="0"
12664 ref="ns_p:nodeManagementDestinationData"/>
12665     </xs:sequence>
12666 </xs:complexType>
12667 <xs:element name="nodeManagementDestinationListData"
12668 type="ns_p:NodeManagementDestinationListDataType"/>
12669 <xs:complexType name="NodeManagementDestinationListDataSelectorsType">
12670     <xs:sequence>
12671         <xs:element minOccurs="0" name="deviceDescription">
12672             <xs:complexType>
12673                 <xs:complexContent>
12674                     <xs:restriction
12675 base="ns_p:NetworkManagementDeviceDescriptionListDataSelectorsType">
12676                         <xs:sequence>
12677                             <xs:element minOccurs="0" name="deviceAddress"
12678 type="ns_p:DeviceAddressType"/>
12679                         </xs:sequence>
12680                     </xs:restriction>
12681                 </xs:complexContent>
12682             </xs:complexType>
12683         </xs:element>
12684     </xs:sequence>
12685 </xs:complexType>
12686 <xs:element name="nodeManagementDestinationListDataSelectors"
12687 type="ns_p:NodeManagementDestinationListDataSelectorsType"/>
12688 <xs:complexType name="NodeManagementUseCaseDataType">
12689     <xs:sequence>
12690         <xs:element maxOccurs="unbounded" minOccurs="0" name="useCaseInformation">
12691             <xs:complexType>
12692                 <xs:complexContent>
12693                     <xs:restriction base="ns_p:UseCaseInformationDataType">
12694                         <xs:sequence>
12695                             <xs:element minOccurs="0" name="address"
12696 type="ns_p:FeatureAddressType"/>
12697                             <xs:element minOccurs="0" name="actor"
12698 type="ns_p:UseCaseActorType"/>
12699                             <xs:element maxOccurs="unbounded" minOccurs="0"
12700 name="useCaseSupport">
12701                                 <xs:complexType>
12702                                     <xs:complexContent>
12703                                         <xs:restriction base="ns_p:UseCaseSupportType">
12704                                             <xs:sequence>
12705                                                 <xs:element minOccurs="0"
12706 name="useCaseName" type="ns_p:UseCaseNameType"/>
12707                                                 <xs:element minOccurs="0"
12708 name="useCaseVersion" type="ns_p:SpecificationVersionType"/>
12709                                                 <xs:element minOccurs="0"
12710 name="useCaseAvailable" type="xs:boolean"/>
12711                                                 <xs:element maxOccurs="unbounded"
12712 minOccurs="0" name="scenarioSupport" type="ns_p:UseCaseScenarioSupportType"/>
12713                                             </xs:sequence>
12714                                         </xs:restriction>
12715                                     </xs:complexContent>

```



```

12716         </xs:complexType>
12717     </xs:element>
12718 </xs:sequence>
12719 </xs:restriction>
12720 </xs:complexContent>
12721 </xs:complexType>
12722 </xs:element>
12723 </xs:sequence>
12724 </xs:complexType>
12725 <xs:element name="nodeManagementUseCaseData" type="ns_p:NodeManagementUseCaseDataType"/>
12726 <xs:complexType name="NodeManagementUseCaseDataElementsType">
12727     <xs:sequence>
12728         <xs:element maxOccurs="1" minOccurs="0" name="useCaseInformation">
12729             <xs:complexType>
12730                 <xs:complexContent>
12731                     <xs:restriction base="ns_p:UseCaseInformationDataElementsType">
12732                         <xs:sequence>
12733                             <xs:element minOccurs="0" name="address"
12734 type="ns_p:FeatureAddressElementsType"/>
12735                             <xs:element minOccurs="0" name="actor"
12736 type="ns_p:ElementTagType"/>
12737                             <xs:element minOccurs="0" name="useCaseSupport">
12738                                 <xs:complexType>
12739                                     <xs:complexContent>
12740                                         <xs:restriction
12741 base="ns_p:UseCaseSupportElementsType">
12742                                             <xs:sequence>
12743                                                 <xs:element minOccurs="0"
12744 name="useCaseName" type="ns_p:ElementTagType"/>
12745                                                 <xs:element minOccurs="0"
12746 name="useCaseVersion" type="ns_p:ElementTagType"/>
12747                                                 <xs:element minOccurs="0"
12748 name="useCaseAvailable" type="ns_p:ElementTagType"/>
12749                                                 <xs:element minOccurs="0"
12750 name="scenarioSupport" type="ns_p:ElementTagType"/>
12751                                             </xs:sequence>
12752                                         </xs:restriction>
12753                                     </xs:complexContent>
12754                                 </xs:complexType>
12755                             </xs:element>
12756                         </xs:sequence>
12757                     </xs:restriction>
12758                 </xs:complexContent>
12759             </xs:complexType>
12760         </xs:element>
12761     </xs:sequence>
12762 </xs:complexType>
12763 <xs:element name="nodeManagementUseCaseDataElements"
12764 type="ns_p:NodeManagementUseCaseDataElementsType"/>
12765 <xs:complexType name="NodeManagementUseCaseDataSelectorsType">
12766     <xs:sequence>
12767         <xs:element minOccurs="0" name="useCaseInformation">
12768             <xs:complexType>
12769                 <xs:complexContent>
12770                     <xs:restriction base="ns_p:UseCaseInformationListDataSelectorsType">
12771                         <xs:sequence>
12772                             <xs:element minOccurs="0" name="address"
12773 type="ns_p:FeatureAddressType"/>
12774                             <xs:element minOccurs="0" name="actor"
12775 type="ns_p:UseCaseActorType"/>
12776                             <xs:element minOccurs="0" name="useCaseSupport">
12777                                 <xs:complexType>
12778                                     <xs:complexContent>
12779                                         <xs:restriction
12780 base="ns_p:UseCaseSupportSelectorsType">
12781                                             <xs:sequence>
12782                                                 <xs:element minOccurs="0"
12783 name="useCaseName" type="ns_p:UseCaseNameType"/>
12784                                                 <xs:element minOccurs="0"
12785 name="useCaseVersion" type="ns_p:SpecificationVersionType"/>
12786                                                 <xs:element minOccurs="0"
12787 name="scenarioSupport" type="ns_p:UseCaseScenarioSupportType"/>
12788                                             </xs:sequence>
12789                                         </xs:restriction>
12790                                     </xs:complexContent>
12791                                 </xs:complexType>
12792                             </xs:element>
12793                         </xs:sequence>

```

```

12794         </xs:restriction>
12795     </xs:complexContent>
12796 </xs:complexType>
12797 </xs:element>
12798 </xs:sequence>
12799 </xs:complexType>
12800 <xs:element name="nodeManagementUseCaseDataSelectors"
12801 type="ns_p:NodeManagementUseCaseDataSelectorsType"/>
12802 </xs:schema>
12803
12804

```

12806 A.1.3 SmartEnergyManagementPs

12807 File name: EEBus_SPINE_TS_SmartEnergyManagementPs.xsd

```

12808
12809 <?xml version="1.0" encoding="UTF-8"?>
12810 <!--
12811     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
12812     Version 1.1.1
12813     2018-12-21
12814     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
12815     Source: https://www.eebus.org/en/specifications/
12816 -->
12817 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
12818 xmlns:xs="http://www.w3.org/2001/XMLSchema"
12819 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1" blockDefault="#all"
12820 elementFormDefault="qualified">
12821     <xs:annotation>
12822         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
12823 e.V. All rights reserved.</xs:documentation>
12824     </xs:annotation>
12825     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
12826     <xs:include schemaLocation="EEBus_SPINE_TS_OperatingConstraints.xsd"/>
12827     <xs:include schemaLocation="EEBus_SPINE_TS_PowerSequences.xsd"/>
12828     <xs:complexType name="SmartEnergyManagementPsAlternativesRelationType">
12829         <xs:complexContent>
12830             <xs:restriction base="ns_p:PowerSequenceAlternativesRelationDataType">
12831                 <xs:sequence>
12832                     <xs:element minOccurs="0" name="alternativesId"
12833 type="ns_p:AlternativesIdType"/>
12834                 </xs:sequence>
12835             </xs:restriction>
12836         </xs:complexContent>
12837     </xs:complexType>
12838     <xs:complexType name="SmartEnergyManagementPsAlternativesType">
12839         <xs:sequence>
12840             <xs:element minOccurs="0" name="relation"
12841 type="ns_p:SmartEnergyManagementPsAlternativesRelationType"/>
12842             <xs:element name="powerSequence" maxOccurs="unbounded" minOccurs="0"
12843 type="ns_p:SmartEnergyManagementPsPowerSequenceType"/>
12844         </xs:sequence>
12845     </xs:complexType>
12846     <xs:complexType name="SmartEnergyManagementPsPowerSequenceType">
12847         <xs:sequence>
12848             <xs:element name="description" minOccurs="0">
12849                 <xs:complexType>
12850                     <xs:complexContent>
12851                         <xs:restriction base="ns_p:PowerSequenceDescriptionDataType">
12852                             <xs:sequence>
12853                                 <xs:element name="sequenceId" type="ns_p:PowerSequenceIdType"
12854 minOccurs="0"/>
12855                                 <xs:element minOccurs="0" name="description">
12856                                     <xs:simpleType>
12857                                         <xs:restriction base="ns_p:DescriptionType">
12858                                             <xs:minLength value="1"/>
12859                                             <xs:maxLength value="60"/>
12860                                         </xs:restriction>
12861                                     </xs:simpleType>
12862                                 </xs:element>
12863                                 <xs:element minOccurs="0" name="powerUnit"
12864 type="ns_p:UnitOfMeasurementType"/>
12865                                 <xs:element minOccurs="0" name="energyUnit"
12866 type="ns_p:UnitOfMeasurementType"/>

```

```

12867         <xs:element minOccurs="0" name="valueSource"
12868 type="ns_p:MeasurementValueSourceType"/>
12869         <xs:element minOccurs="0" name="taskIdentifier"
12870 type="xs:unsignedInt"/>
12871         <xs:element minOccurs="0" name="repetitionsTotal"
12872 type="xs:unsignedInt"/>
12873     </xs:sequence>
12874 </xs:restriction>
12875 </xs:complexContent>
12876 </xs:complexType>
12877 </xs:element>
12878 <xs:element name="state" minOccurs="0">
12879     <xs:complexType>
12880         <xs:complexContent>
12881             <xs:restriction base="ns_p:PowerSequenceStateDataType">
12882                 <xs:sequence>
12883                     <xs:element name="state" type="ns_p:PowerSequenceStateType"
12884 minOccurs="0"/>
12885                     <xs:element minOccurs="0" name="activeSlotNumber"
12886 type="ns_p:PowerTimeSlotNumberType"/>
12887                     <xs:element minOccurs="0" name="elapsedSlotTime"
12888 type="xs:duration"/>
12889                     <xs:element minOccurs="0" name="remainingSlotTime"
12890 type="xs:duration"/>
12891                     <xs:element name="sequenceRemoteControllable"
12892 type="xs:boolean" minOccurs="0"/>
12893                     <xs:element minOccurs="0" name="activeRepetitionNumber"
12894 type="xs:unsignedInt"/>
12895                     <xs:element minOccurs="0" name="remainingPauseTime"
12896 type="xs:duration"/>
12897                 </xs:sequence>
12898             </xs:restriction>
12899         </xs:complexContent>
12900     </xs:complexType>
12901 </xs:element>
12902 <xs:element minOccurs="0" name="schedule">
12903     <xs:complexType>
12904         <xs:complexContent>
12905             <xs:restriction base="ns_p:PowerSequenceScheduleDataType">
12906                 <xs:sequence>
12907                     <xs:element name="startTime" type="xs:duration"
12908 minOccurs="0"/>
12909                     <xs:element name="endTime" type="xs:duration" minOccurs="0"/>
12910                 </xs:sequence>
12911             </xs:restriction>
12912         </xs:complexContent>
12913     </xs:complexType>
12914 </xs:element>
12915 <xs:element name="scheduleConstraints" minOccurs="0">
12916     <xs:complexType>
12917         <xs:complexContent>
12918             <xs:restriction base="ns_p:PowerSequenceScheduleConstraintsDataType">
12919                 <xs:sequence>
12920                     <xs:element name="earliestStartTime" type="xs:duration"
12921 minOccurs="0"/>
12922                     <xs:element name="latestEndTime" type="xs:duration"
12923 minOccurs="0"/>
12924                 </xs:sequence>
12925             </xs:restriction>
12926         </xs:complexContent>
12927     </xs:complexType>
12928 </xs:element>
12929 <xs:element minOccurs="0" name="schedulePreference">
12930     <xs:complexType>
12931         <xs:complexContent>
12932             <xs:restriction base="ns_p:PowerSequenceSchedulePreferenceDataType">
12933                 <xs:sequence>
12934                     <xs:element minOccurs="0" name="greenest" type="xs:boolean"/>
12935                     <xs:element minOccurs="0" name="cheapest" type="xs:boolean"/>
12936                 </xs:sequence>
12937             </xs:restriction>
12938         </xs:complexContent>
12939     </xs:complexType>
12940 </xs:element>
12941 <xs:element name="operatingConstraintsInterrupt" minOccurs="0">
12942     <xs:complexType>
12943         <xs:complexContent>
12944             <xs:restriction base="ns_p:OperatingConstraintsInterruptDataType">

```

```

12945         <xs:sequence>
12946             <xs:element minOccurs="0" name="isPausable"
12947 type="xs:boolean"/>
12948             <xs:element minOccurs="0" name="isStoppable"
12949 type="xs:boolean"/>
12950             <xs:element minOccurs="0" name="maxCyclesPerDay"
12951 type="xs:unsignedInt"/>
12952         </xs:sequence>
12953     </xs:restriction>
12954 </xs:complexContent>
12955 </xs:complexType>
12956 </xs:element>
12957 <xs:element name="operatingConstraintsDuration" minOccurs="0">
12958     <xs:complexType>
12959         <xs:complexContent>
12960             <xs:restriction base="ns_p:OperatingConstraintsDurationDataType">
12961                 <xs:sequence>
12962                     <xs:element minOccurs="0" name="activeDurationMin"
12963 type="xs:duration"/>
12964                     <xs:element minOccurs="0" name="activeDurationMax"
12965 type="xs:duration"/>
12966                     <xs:element minOccurs="0" name="pauseDurationMin"
12967 type="xs:duration"/>
12968                     <xs:element minOccurs="0" name="pauseDurationMax"
12969 type="xs:duration"/>
12970                     <xs:element minOccurs="0" name="activeDurationSumMin"
12971 type="xs:duration"/>
12972                     <xs:element minOccurs="0" name="activeDurationSumMax"
12973 type="xs:duration"/>
12974                 </xs:sequence>
12975             </xs:restriction>
12976         </xs:complexContent>
12977     </xs:complexType>
12978 </xs:element>
12979 <xs:element name="operatingConstraintsResumeImplication" minOccurs="0">
12980     <xs:complexType>
12981         <xs:complexContent>
12982             <xs:restriction
12983 base="ns_p:OperatingConstraintsResumeImplicationDataType">
12984                 <xs:sequence>
12985                     <xs:element minOccurs="0" name="resumeEnergyEstimated"
12986 type="ns_p:ScaledNumberType"/>
12987                     <xs:element minOccurs="0" name="energyUnit"
12988 type="ns_p:UnitOfMeasurementType"/>
12989                     <xs:element minOccurs="0" name="resumeCostEstimated"
12990 type="ns_p:ScaledNumberType"/>
12991                     <xs:element minOccurs="0" name="currency"
12992 type="ns_p:CurrencyType"/>
12993                 </xs:sequence>
12994             </xs:restriction>
12995         </xs:complexContent>
12996     </xs:complexType>
12997 </xs:element>
12998     <xs:element maxOccurs="unbounded" minOccurs="0" name="powerTimeSlot"
12999 type="ns_p:SmartEnergyManagementPsPowerTimeSlotType"/>
13000 </xs:sequence>
13001 </xs:complexType>
13002 <xs:complexType name="SmartEnergyManagementPsPowerTimeSlotType">
13003     <xs:sequence>
13004         <xs:element name="schedule" minOccurs="0">
13005             <xs:complexType>
13006                 <xs:complexContent>
13007                     <xs:restriction base="ns_p:PowerTimeSlotScheduleDataType">
13008                         <xs:sequence>
13009                             <xs:element name="slotNumber"
13010 type="ns_p:PowerTimeSlotNumberType" minOccurs="0"/>
13011                             <xs:element name="timePeriod" minOccurs="0">
13012                                 <xs:complexType>
13013                                     <xs:complexContent>
13014                                         <xs:restriction base="ns_p:TimePeriodType">
13015                                             <xs:sequence>
13016                                                 <xs:element name="startTime"
13017 type="xs:duration" minOccurs="0"/>
13018                                                 <xs:element name="endTime"
13019 type="xs:duration" minOccurs="0"/>
13020                                             </xs:sequence>
13021                                         </xs:restriction>
13022                                     </xs:complexContent>

```

```

13023         </xs:complexType>
13024     </xs:element>
13025     <xs:element name="defaultDuration" type="xs:duration"
13026 minOccurs="0"/>
13027         <xs:element minOccurs="0" name="durationUncertainty"
13028 type="xs:duration"/>
13029         <xs:element name="slotActivated" type="xs:boolean"
13030 minOccurs="0"/>
13031         <xs:element minOccurs="0" name="description">
13032             <xs:simpleType>
13033                 <xs:restriction base="ns_p:DescriptionType">
13034                     <xs:minLength value="1"/>
13035                     <xs:maxLength value="60"/>
13036                 </xs:restriction>
13037             </xs:simpleType>
13038         </xs:element>
13039     </xs:sequence>
13040 </xs:restriction>
13041 </xs:complexContent>
13042 </xs:complexType>
13043 </xs:element>
13044     <xs:element minOccurs="0" name="valueList"
13045 type="ns_p:SmartEnergyManagementPsPowerTimeSlotValueListType"/>
13046     <xs:element name="scheduleConstraints" minOccurs="0">
13047         <xs:complexType>
13048             <xs:complexContent>
13049                 <xs:restriction base="ns_p:PowerTimeSlotScheduleConstraintsDataType">
13050                     <xs:sequence>
13051                         <xs:element name="earliestStartTime" type="xs:duration"
13052 minOccurs="0"/>
13053                         <xs:element name="latestEndTime" type="xs:duration"
13054 minOccurs="0"/>
13055                         <xs:element minOccurs="0" name="minDuration"
13056 type="xs:duration"/>
13057                         <xs:element minOccurs="0" name="maxDuration"
13058 type="xs:duration"/>
13059                         <xs:element minOccurs="0" name="optionalSlot"
13060 type="xs:boolean"/>
13061                     </xs:sequence>
13062                 </xs:restriction>
13063             </xs:complexContent>
13064         </xs:complexType>
13065     </xs:element>
13066 </xs:sequence>
13067 </xs:complexType>
13068 <xs:complexType name="SmartEnergyManagementPsPowerTimeSlotValueListType">
13069     <xs:sequence>
13070         <xs:element maxOccurs="unbounded" minOccurs="0" name="value">
13071             <xs:complexType>
13072                 <xs:complexContent>
13073                     <xs:restriction base="ns_p:PowerTimeSlotValueDataType">
13074                         <xs:sequence>
13075                             <xs:element minOccurs="0" name="valueType"
13076 type="ns_p:PowerTimeSlotValueTypeType"/>
13077                             <xs:element minOccurs="0" name="value"
13078 type="ns_p:ScaledNumberType"/>
13079                         </xs:sequence>
13080                     </xs:restriction>
13081                 </xs:complexContent>
13082             </xs:complexType>
13083         </xs:element>
13084     </xs:sequence>
13085 </xs:complexType>
13086 <xs:complexType name="SmartEnergyManagementPsDataType">
13087     <xs:sequence>
13088         <xs:element name="nodeScheduleInformation" minOccurs="0">
13089             <xs:complexType>
13090                 <xs:complexContent>
13091                     <xs:restriction
13092 base="ns_p:PowerSequenceNodeScheduleInformationDataType">
13093                         <xs:sequence>
13094                             <xs:element name="nodeRemoteControllable" type="xs:boolean"
13095 minOccurs="0"/>
13096                             <xs:element name="supportsSingleSlotSchedulingOnly"
13097 type="xs:boolean" minOccurs="0"/>
13098                             <xs:element name="alternativesCount" type="xs:unsignedInt"
13099 minOccurs="0"/>

```

```

13100         <xs:element name="totalSequencesCountMax"
13101 type="xs:unsignedInt" minOccurs="0"/>
13102         <xs:element name="supportsReselection" type="xs:boolean"
13103 minOccurs="0"/>
13104     </xs:sequence>
13105 </xs:restriction>
13106 </xs:complexContent>
13107 </xs:complexType>
13108 </xs:element>
13109 <xs:element name="alternatives" maxOccurs="unbounded" minOccurs="0"
13110 type="ns_p:SmartEnergyManagementPsAlternativesType"/>
13111 </xs:sequence>
13112 </xs:complexType>
13113 <xs:element name="smartEnergyManagementPsData"
13114 type="ns_p:SmartEnergyManagementPsDataType"/>
13115 <xs:complexType name="SmartEnergyManagementPsAlternativesRelationElementsType">
13116 <xs:complexContent>
13117 <xs:restriction base="ns_p:PowerSequenceAlternativesRelationDataElementsType">
13118 <xs:sequence>
13119 <xs:element minOccurs="0" name="alternativesId"
13120 type="ns_p:ElementTagType"/>
13121 </xs:sequence>
13122 </xs:restriction>
13123 </xs:complexContent>
13124 </xs:complexType>
13125 <xs:complexType name="SmartEnergyManagementPsAlternativesElementsType">
13126 <xs:sequence>
13127 <xs:element minOccurs="0" name="relation"
13128 type="ns_p:SmartEnergyManagementPsAlternativesRelationElementsType"/>
13129 <xs:element name="powerSequence" minOccurs="0"
13130 type="ns_p:SmartEnergyManagementPsPowerSequenceElementsType"/>
13131 </xs:sequence>
13132 </xs:complexType>
13133 <xs:complexType name="SmartEnergyManagementPsPowerSequenceElementsType">
13134 <xs:sequence>
13135 <xs:element name="description" minOccurs="0">
13136 <xs:complexType>
13137 <xs:complexContent>
13138 <xs:restriction base="ns_p:PowerSequenceDescriptionDataElementsType">
13139 <xs:sequence>
13140 <xs:element name="sequenceId" minOccurs="0"
13141 type="ns_p:ElementTagType"/>
13142 <xs:element minOccurs="0" name="description"
13143 type="ns_p:ElementTagType"/>
13144 <xs:element minOccurs="0" name="powerUnit"
13145 type="ns_p:ElementTagType"/>
13146 <xs:element minOccurs="0" name="energyUnit"
13147 type="ns_p:ElementTagType"/>
13148 <xs:element minOccurs="0" name="valueSource"
13149 type="ns_p:ElementTagType"/>
13150 <xs:element minOccurs="0" name="taskIdIdentifier"
13151 type="ns_p:ElementTagType"/>
13152 <xs:element minOccurs="0" name="repetitionsTotal"
13153 type="ns_p:ElementTagType"/>
13154 </xs:sequence>
13155 </xs:restriction>
13156 </xs:complexContent>
13157 </xs:complexType>
13158 </xs:element>
13159 <xs:element name="state" minOccurs="0">
13160 <xs:complexType>
13161 <xs:complexContent>
13162 <xs:restriction base="ns_p:PowerSequenceStateDataElementsType">
13163 <xs:sequence>
13164 <xs:element name="state" minOccurs="0"
13165 type="ns_p:ElementTagType"/>
13166 <xs:element minOccurs="0" name="activeSlotNumber"
13167 type="ns_p:ElementTagType"/>
13168 <xs:element minOccurs="0" name="elapsedSlotTime"
13169 type="ns_p:ElementTagType"/>
13170 <xs:element minOccurs="0" name="remainingSlotTime"
13171 type="ns_p:ElementTagType"/>
13172 <xs:element name="sequenceRemoteControllable" minOccurs="0"
13173 type="ns_p:ElementTagType"/>
13174 <xs:element minOccurs="0" name="activeRepetitionNumber"
13175 type="ns_p:ElementTagType"/>
13176 <xs:element minOccurs="0" name="remainingPauseTime"
13177 type="ns_p:ElementTagType"/>

```

```
13178         </xs:sequence>
13179     </xs:restriction>
13180 </xs:complexContent>
13181 </xs:complexType>
13182 </xs:element>
13183 <xs:element minOccurs="0" name="schedule">
13184     <xs:complexType>
13185         <xs:complexContent>
13186             <xs:restriction base="ns_p:PowerSequenceScheduleDataElementsType">
13187                 <xs:sequence>
13188                     <xs:element name="startTime" minOccurs="0"
13189 type="ns_p:ElementTagType"/>
13190                     <xs:element name="endTime" minOccurs="0"
13191 type="ns_p:ElementTagType"/>
13192                 </xs:sequence>
13193             </xs:restriction>
13194 </xs:complexContent>
13195 </xs:complexType>
13196 </xs:element>
13197 <xs:element name="scheduleConstraints" minOccurs="0">
13198     <xs:complexType>
13199         <xs:complexContent>
13200             <xs:restriction
13201 base="ns_p:PowerSequenceScheduleConstraintsDataElementsType">
13202                 <xs:sequence>
13203                     <xs:element name="earliestStartTime" minOccurs="0"
13204 type="ns_p:ElementTagType"/>
13205                     <xs:element name="latestEndTime" minOccurs="0"
13206 type="ns_p:ElementTagType"/>
13207                 </xs:sequence>
13208             </xs:restriction>
13209 </xs:complexContent>
13210 </xs:complexType>
13211 </xs:element>
13212 <xs:element minOccurs="0" name="schedulePreference">
13213     <xs:complexType>
13214         <xs:complexContent>
13215             <xs:restriction
13216 base="ns_p:PowerSequenceSchedulePreferenceDataElementsType">
13217                 <xs:sequence>
13218                     <xs:element minOccurs="0" name="greenest"
13219 type="ns_p:ElementTagType"/>
13220                     <xs:element minOccurs="0" name="cheapest"
13221 type="ns_p:ElementTagType"/>
13222                 </xs:sequence>
13223             </xs:restriction>
13224 </xs:complexContent>
13225 </xs:complexType>
13226 </xs:element>
13227 <xs:element name="operatingConstraintsInterrupt" minOccurs="0">
13228     <xs:complexType>
13229         <xs:complexContent>
13230             <xs:restriction
13231 base="ns_p:OperatingConstraintsInterruptDataElementsType">
13232                 <xs:sequence>
13233                     <xs:element minOccurs="0" name="isPausable"
13234 type="ns_p:ElementTagType"/>
13235                     <xs:element minOccurs="0" name="isStoppable"
13236 type="ns_p:ElementTagType"/>
13237                     <xs:element minOccurs="0" name="maxCyclesPerDay"
13238 type="ns_p:ElementTagType"/>
13239                 </xs:sequence>
13240             </xs:restriction>
13241 </xs:complexContent>
13242 </xs:complexType>
13243 </xs:element>
13244 <xs:element name="operatingConstraintsDuration" minOccurs="0">
13245     <xs:complexType>
13246         <xs:complexContent>
13247             <xs:restriction
13248 base="ns_p:OperatingConstraintsDurationDataElementsType">
13249                 <xs:sequence>
13250                     <xs:element minOccurs="0" name="activeDurationMin"
13251 type="ns_p:ElementTagType"/>
13252                     <xs:element minOccurs="0" name="activeDurationMax"
13253 type="ns_p:ElementTagType"/>
13254                     <xs:element minOccurs="0" name="pauseDurationMin"
13255 type="ns_p:ElementTagType"/>
```

```

13256         <xs:element minOccurs="0" name="pauseDurationMax"
13257 type="ns_p:ElementTagType"/>
13258         <xs:element minOccurs="0" name="activeDurationSumMin"
13259 type="ns_p:ElementTagType"/>
13260         <xs:element minOccurs="0" name="activeDurationSumMax"
13261 type="ns_p:ElementTagType"/>
13262     </xs:sequence>
13263 </xs:restriction>
13264 </xs:complexContent>
13265 </xs:complexType>
13266 </xs:element>
13267 <xs:element name="operatingConstraintsResumeImplication" minOccurs="0">
13268     <xs:complexType>
13269         <xs:complexContent>
13270             <xs:restriction
13271 base="ns_p:OperatingConstraintsResumeImplicationDataElementsType">
13272                 <xs:sequence>
13273                     <xs:element minOccurs="0" name="resumeEnergyEstimated"
13274 type="ns_p:ScaledNumberElementsType"/>
13275                     <xs:element minOccurs="0" name="energyUnit"
13276 type="ns_p:ElementTagType"/>
13277                     <xs:element minOccurs="0" name="resumeCostEstimated"
13278 type="ns_p:ScaledNumberElementsType"/>
13279                     <xs:element minOccurs="0" name="currency"
13280 type="ns_p:ElementTagType"/>
13281                 </xs:sequence>
13282             </xs:restriction>
13283         </xs:complexContent>
13284     </xs:complexType>
13285 </xs:element>
13286 <xs:element minOccurs="0" name="powerTimeSlot"
13287 type="ns_p:SmartEnergyManagementPsPowerTimeSlotElementsType"/>
13288 </xs:sequence>
13289 </xs:complexType>
13290 <xs:complexType name="SmartEnergyManagementPsPowerTimeSlotElementsType">
13291     <xs:sequence>
13292         <xs:element name="schedule" minOccurs="0">
13293             <xs:complexType>
13294                 <xs:complexContent>
13295                     <xs:restriction base="ns_p:PowerTimeSlotScheduleDataElementsType">
13296                         <xs:sequence>
13297                             <xs:element name="slotNumber" minOccurs="0"
13298 type="ns_p:ElementTagType"/>
13299                             <xs:element name="timePeriod" minOccurs="0">
13300                                 <xs:complexType>
13301                                     <xs:complexContent>
13302                                         <xs:restriction
13303 base="ns_p:TimePeriodElementsType">
13304                                             <xs:sequence>
13305                                                 <xs:element name="startTime" minOccurs="0"
13306 type="ns_p:ElementTagType"/>
13307                                                 <xs:element name="endTime" minOccurs="0"
13308 type="ns_p:ElementTagType"/>
13309                                             </xs:sequence>
13310                                         </xs:restriction>
13311                                     </xs:complexContent>
13312                                 </xs:complexType>
13313                             </xs:element>
13314                             <xs:element name="defaultDuration" minOccurs="0"
13315 type="ns_p:ElementTagType"/>
13316                             <xs:element minOccurs="0" name="durationUncertainty"
13317 type="ns_p:ElementTagType"/>
13318                             <xs:element name="slotActivated" minOccurs="0"
13319 type="ns_p:ElementTagType"/>
13320                             <xs:element minOccurs="0" name="description"
13321 type="ns_p:ElementTagType"/>
13322                         </xs:sequence>
13323                     </xs:restriction>
13324                 </xs:complexContent>
13325             </xs:complexType>
13326 </xs:element>
13327 <xs:element name="valueList" minOccurs="0"
13328 type="ns_p:SmartEnergyManagementPsPowerTimeSlotValueListElementsType"/>
13329 <xs:element name="scheduleConstraints" minOccurs="0">
13330     <xs:complexType>
13331         <xs:complexContent>
13332             <xs:restriction
13333 base="ns_p:PowerTimeSlotScheduleConstraintsDataElementsType">

```



```

13334         <xs:sequence>
13335             <xs:element name="earliestStartTime" minOccurs="0"
13336 type="ns_p:ElementTagType"/>
13337             <xs:element name="latestEndTime" minOccurs="0"
13338 type="ns_p:ElementTagType"/>
13339             <xs:element minOccurs="0" name="minDuration"
13340 type="ns_p:ElementTagType"/>
13341             <xs:element minOccurs="0" name="maxDuration"
13342 type="ns_p:ElementTagType"/>
13343             <xs:element minOccurs="0" name="optionalSlot"
13344 type="ns_p:ElementTagType"/>
13345         </xs:sequence>
13346     </xs:restriction>
13347 </xs:complexContent>
13348 </xs:complexType>
13349 </xs:element>
13350 </xs:sequence>
13351 </xs:complexType>
13352 <xs:complexType name="SmartEnergyManagementPsPowerTimeSlotValueListElementsType">
13353     <xs:sequence>
13354         <xs:element name="value" minOccurs="0">
13355             <xs:complexType>
13356                 <xs:complexContent>
13357                     <xs:restriction base="ns_p:PowerTimeSlotValueDataElementsType">
13358                         <xs:sequence>
13359                             <xs:element name="valueType" minOccurs="0"
13360 type="ns_p:ElementTagType"/>
13361                             <xs:element name="value" minOccurs="0"
13362 type="ns_p:ScaledNumberElementsType"/>
13363                         </xs:sequence>
13364                     </xs:restriction>
13365                 </xs:complexContent>
13366             </xs:complexType>
13367         </xs:element>
13368     </xs:sequence>
13369 </xs:complexType>
13370 <xs:complexType name="SmartEnergyManagementPsDataElementsType">
13371     <xs:sequence>
13372         <xs:element name="nodeScheduleInformation" minOccurs="0">
13373             <xs:complexType>
13374                 <xs:complexContent>
13375                     <xs:restriction
13376 base="ns_p:PowerSequenceNodeScheduleInformationDataElementsType">
13377                         <xs:sequence>
13378                             <xs:element name="nodeRemoteControllable" minOccurs="0"
13379 type="ns_p:ElementTagType"/>
13380                             <xs:element name="supportsSingleSlotSchedulingOnly"
13381 minOccurs="0" type="ns_p:ElementTagType"/>
13382                             <xs:element name="alternativesCount" minOccurs="0"
13383 type="ns_p:ElementTagType"/>
13384                             <xs:element name="totalSequencesCountMax" minOccurs="0"
13385 type="ns_p:ElementTagType"/>
13386                             <xs:element name="supportsReselection" minOccurs="0"
13387 type="ns_p:ElementTagType"/>
13388                         </xs:sequence>
13389                     </xs:restriction>
13390                 </xs:complexContent>
13391             </xs:complexType>
13392         </xs:element>
13393         <xs:element name="alternatives" minOccurs="0"
13394 type="ns_p:SmartEnergyManagementPsAlternativesElementsType"/>
13395     </xs:sequence>
13396 </xs:complexType>
13397 <xs:element name="smartEnergyManagementPsDataElements"
13398 type="ns_p:SmartEnergyManagementPsDataElementsType"/>
13399 <xs:complexType name="SmartEnergyManagementPsDataSelectorsType">
13400     <xs:sequence>
13401         <xs:element minOccurs="0" name="alternativesRelation">
13402             <xs:complexType>
13403                 <xs:complexContent>
13404                     <xs:restriction
13405 base="ns_p:PowerSequenceAlternativesRelationListDataSelectorsType">
13406                         <xs:sequence>
13407                             <xs:element minOccurs="0" name="alternativesId"
13408 type="ns_p:AlternativesIdType"/>
13409                         </xs:sequence>
13410                     </xs:restriction>
13411                 </xs:complexContent>

```

```

13412         </xs:complexType>
13413     </xs:element>
13414     <xs:element minOccurs="0" name="powerSequenceDescription">
13415         <xs:complexType>
13416             <xs:complexContent>
13417                 <xs:restriction
13418 base="ns_p:PowerSequenceDescriptionListDataSelectorsType">
13419                     <xs:sequence>
13420                         <xs:element minOccurs="0" name="sequenceId"
13421 type="ns_p:PowerSequenceIdType"/>
13422                     </xs:sequence>
13423                 </xs:restriction>
13424             </xs:complexContent>
13425         </xs:complexType>
13426     </xs:element>
13427     <xs:element minOccurs="0" name="powerTimeSlotSchedule">
13428         <xs:complexType>
13429             <xs:complexContent>
13430                 <xs:restriction
13431 base="ns_p:PowerTimeSlotScheduleListDataSelectorsType">
13432                     <xs:sequence>
13433                         <xs:element minOccurs="0" name="slotNumber"
13434 type="ns_p:PowerTimeSlotNumberType"/>
13435                     </xs:sequence>
13436                 </xs:restriction>
13437             </xs:complexContent>
13438         </xs:complexType>
13439     </xs:element>
13440     <xs:element minOccurs="0" name="powerTimeSlotValue">
13441         <xs:complexType>
13442             <xs:complexContent>
13443                 <xs:restriction base="ns_p:PowerTimeSlotValueListDataSelectorsType">
13444                     <xs:sequence>
13445                         <xs:element minOccurs="0" name="valueType"
13446 type="ns_p:PowerTimeSlotValueTypeType"/>
13447                     </xs:sequence>
13448                 </xs:restriction>
13449             </xs:complexContent>
13450         </xs:complexType>
13451     </xs:element>
13452 </xs:sequence>
13453 </xs:complexType>
13454 <xs:element name="smartEnergyManagementPsDataSelectors"
13455 type="ns_p:SmartEnergyManagementPsDataSelectorsType"/>
13456 <xs:complexType name="SmartEnergyManagementPsPriceDataType">
13457     <xs:sequence>
13458         <xs:element maxOccurs="unbounded" minOccurs="0" name="price">
13459             <xs:complexType>
13460                 <xs:complexContent>
13461                     <xs:restriction base="ns_p:PowerSequencePriceDataType">
13462                         <xs:sequence>
13463                             <xs:element minOccurs="0" name="sequenceId"
13464 type="ns_p:PowerSequenceIdType"/>
13465                             <xs:element minOccurs="0" name="price"
13466 type="ns_p:ScaledNumberType"/>
13467                             <xs:element minOccurs="0" name="currency"
13468 type="ns_p:CurrencyType"/>
13469                         </xs:sequence>
13470                     </xs:restriction>
13471                 </xs:complexContent>
13472             </xs:complexType>
13473         </xs:element>
13474     </xs:sequence>
13475 </xs:complexType>
13476 <xs:element name="smartEnergyManagementPsPriceData"
13477 type="ns_p:SmartEnergyManagementPsPriceDataType"/>
13478 <xs:complexType name="SmartEnergyManagementPsPriceDataElementsType">
13479     <xs:sequence>
13480         <xs:element minOccurs="0" name="price">
13481             <xs:complexType>
13482                 <xs:complexContent>
13483                     <xs:restriction base="ns_p:PowerSequencePriceDataElementsType">
13484                         <xs:sequence>
13485                             <xs:element minOccurs="0" name="sequenceId"
13486 type="ns_p:ElementTagType"/>
13487                             <xs:element minOccurs="0" name="price"
13488 type="ns_p:ScaledNumberElementsType"/>

```

```

13489         <xs:element minOccurs="0" name="currency"
13490 type="ns_p:ElementTagType"/>
13491     </xs:sequence>
13492 </xs:restriction>
13493 </xs:complexContent>
13494 </xs:complexType>
13495 </xs:element>
13496 </xs:sequence>
13497 </xs:complexType>
13498 <xs:element name="smartEnergyManagementPsPriceDataElements"
13499 type="ns_p:SmartEnergyManagementPsPriceDataElementsType"/>
13500 <xs:complexType name="SmartEnergyManagementPsPriceDataSelectorsType">
13501     <xs:sequence>
13502         <xs:element minOccurs="0" name="price">
13503             <xs:complexType>
13504                 <xs:complexContent>
13505                     <xs:restriction base="ns_p:PowerSequencePriceListDataSelectorsType">
13506                         <xs:sequence>
13507                             <xs:element minOccurs="0" name="sequenceId"
13508 type="ns_p:PowerSequenceIdType"/>
13509                         </xs:sequence>
13510                     </xs:restriction>
13511                 </xs:complexContent>
13512             </xs:complexType>
13513         </xs:element>
13514     </xs:sequence>
13515 </xs:complexType>
13516 <xs:element name="smartEnergyManagementPsPriceDataSelectors"
13517 type="ns_p:SmartEnergyManagementPsPriceDataSelectorsType"/>
13518 <xs:complexType name="SmartEnergyManagementPsConfigurationRequestCallType">
13519     <xs:sequence>
13520         <xs:element minOccurs="0" name="scheduleConfigurationRequest">
13521             <xs:complexType>
13522                 <xs:complexContent>
13523                     <xs:restriction
13524 base="ns_p:PowerSequenceScheduleConfigurationRequestCallType">
13525                         <xs:sequence>
13526                             <xs:element minOccurs="0" name="sequenceId"
13527 type="ns_p:PowerSequenceIdType"/>
13528                         </xs:sequence>
13529                     </xs:restriction>
13530                 </xs:complexContent>
13531             </xs:complexType>
13532         </xs:element>
13533     </xs:sequence>
13534 </xs:complexType>
13535 <xs:element name="smartEnergyManagementPsConfigurationRequestCall"
13536 type="ns_p:SmartEnergyManagementPsConfigurationRequestCallType"/>
13537 <xs:complexType name="SmartEnergyManagementPsConfigurationRequestCallElementsType">
13538     <xs:sequence>
13539         <xs:element minOccurs="0" name="scheduleConfigurationRequest">
13540             <xs:complexType>
13541                 <xs:complexContent>
13542                     <xs:restriction
13543 base="ns_p:PowerSequenceScheduleConfigurationRequestCallElementsType">
13544                         <xs:sequence>
13545                             <xs:element minOccurs="0" name="sequenceId"
13546 type="ns_p:ElementTagType"/>
13547                         </xs:sequence>
13548                     </xs:restriction>
13549                 </xs:complexContent>
13550             </xs:complexType>
13551         </xs:element>
13552     </xs:sequence>
13553 </xs:complexType>
13554 <xs:element name="smartEnergyManagementPsConfigurationRequestCallElements"
13555 type="ns_p:SmartEnergyManagementPsConfigurationRequestCallElementsType"/>
13556 <xs:complexType name="SmartEnergyManagementPsPriceCalculationRequestCallType">
13557     <xs:sequence>
13558         <xs:element minOccurs="0" name="priceCalculationRequest">
13559             <xs:complexType>
13560                 <xs:complexContent>
13561                     <xs:restriction
13562 base="ns_p:PowerSequencePriceCalculationRequestCallType">
13563                         <xs:sequence>
13564                             <xs:element minOccurs="0" name="sequenceId"
13565 type="ns_p:PowerSequenceIdType"/>
13566                         </xs:sequence>

```

```

13567         </xs:restriction>
13568     </xs:complexType>
13569 </xs:complexType>
13570 </xs:element>
13571 </xs:sequence>
13572 </xs:complexType>
13573 <xs:element name="smartEnergyManagementPsPriceCalculationRequestCall"
13574 type="ns_p:SmartEnergyManagementPsPriceCalculationRequestCallType"/>
13575 <xs:complexType name="SmartEnergyManagementPsPriceCalculationRequestCallElementsType">
13576 <xs:sequence>
13577     <xs:element minOccurs="0" name="priceCalculationRequest">
13578         <xs:complexType>
13579             <xs:complexContent>
13580                 <xs:restriction
13581 base="ns_p:PowerSequencePriceCalculationRequestCallElementsType">
13582                     <xs:sequence>
13583                         <xs:element minOccurs="0" name="sequenceId"
13584 type="ns_p:ElementTagType"/>
13585                     </xs:sequence>
13586                 </xs:restriction>
13587             </xs:complexContent>
13588         </xs:complexType>
13589     </xs:element>
13590 </xs:sequence>
13591 </xs:complexType>
13592 <xs:element name="smartEnergyManagementPsPriceCalculationRequestCallElements"
13593 type="ns_p:SmartEnergyManagementPsPriceCalculationRequestCallElementsType"/>
13594 </xs:schema>
13595
13596

```

13597

13598 A.2 Standard Classes

13599 A.2.1 ActuatorLevel

13600 File name: EEBus_SPINE_TS_ActuatorLevel.xsd

```

13601 <?xml version="1.0" encoding="UTF-8"?>
13602 <!--
13603     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
13604     Version 1.1.1
13605     2018-12-21
13606     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
13607     Source: https://www.eebus.org/en/specifications/
13608 -->
13609 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
13610 xmlns:xs="http://www.w3.org/2001/XMLSchema"
13611 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
13612 blockDefault="#all" elementFormDefault="qualified">
13613     <xs:annotation>
13614         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
13615 e.V. All rights reserved.</xs:documentation>
13616     </xs:annotation>
13617     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
13618     <xs:simpleType name="ActuatorLevelFctType">
13619         <xs:union memberTypes="ns_p:ActuatorLevelFctEnumType ns_p:EnumExtendType"/>
13620     </xs:simpleType>
13621     <xs:simpleType name="ActuatorLevelFctEnumType">
13622         <xs:restriction base="xs:string">
13623             <xs:enumeration value="start"/>
13624             <xs:enumeration value="up"/>
13625             <xs:enumeration value="down"/>
13626             <xs:enumeration value="stop"/>
13627             <xs:enumeration value="percentageAbsolute"/>
13628             <xs:enumeration value="percentageRelative"/>
13629             <xs:enumeration value="absolute"/>
13630             <xs:enumeration value="relative"/>
13631         </xs:restriction>
13632     </xs:simpleType>
13633     <xs:complexType name="ActuatorLevelDataType">
13634         <xs:sequence>
13635             <xs:element name="function" type="ns_p:ActuatorLevelFctType" minOccurs="0"/>
13636             <xs:element name="value" type="ns_p:ScaledNumberType" minOccurs="0"/>
13637

```

```

13638         </xs:sequence>
13639     </xs:complexType>
13640     <xs:element name="actuatorLevelData" type="ns_p:ActuatorLevelDataType"/>
13641     <xs:complexType name="ActuatorLevelDataElementsType">
13642         <xs:sequence>
13643             <xs:element name="function" minOccurs="0" type="ns_p:ElementTagType"/>
13644             <xs:element name="value" type="ns_p:ScaledNumberElementsType" minOccurs="0"/>
13645         </xs:sequence>
13646     </xs:complexType>
13647     <xs:element name="actuatorLevelDataElements" type="ns_p:ActuatorLevelDataElementsType"/>
13648     <xs:complexType name="ActuatorLevelDescriptionDataType">
13649         <xs:sequence>
13650             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
13651             <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
13652             <xs:element minOccurs="0" name="levelDefaultUnit"
13653 type="ns_p:UnitOfMeasurementType"/>
13654         </xs:sequence>
13655     </xs:complexType>
13656     <xs:element name="actuatorLevelDescriptionData"
13657 type="ns_p:ActuatorLevelDescriptionDataType"/>
13658     <xs:complexType name="ActuatorLevelDescriptionDataElementsType">
13659         <xs:sequence>
13660             <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
13661             <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
13662             <xs:element minOccurs="0" name="levelDefaultUnit" type="ns_p:ElementTagType"/>
13663         </xs:sequence>
13664     </xs:complexType>
13665     <xs:element name="actuatorLevelDescriptionDataElements"
13666 type="ns_p:ActuatorLevelDescriptionDataElementsType"/>
13667 </xs:schema>
13668
13669

```

13670

13671 A.2.2 ActuatorSwitch

13672 File name: EEBus_SPINE_TS_ActuatorSwitch.xsd

```

13673 <?xml version="1.0" encoding="UTF-8"?>
13674 <!--
13675     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
13676     Version 1.1.1
13677     2018-12-21
13678     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
13679     Source: https://www.eebus.org/en/specifications/
13680 -->
13681 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
13682 xmlns:xs="http://www.w3.org/2001/XMLSchema"
13683 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
13684 blockDefault="#all" elementFormDefault="qualified">
13685     <xs:annotation>
13686         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
13687 e.V. All rights reserved.</xs:documentation>
13688     </xs:annotation>
13689     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
13690     <xs:simpleType name="ActuatorSwitchFctType">
13691         <xs:union memberTypes="ns_p:ActuatorSwitchFctEnumType ns_p:EnumExtendType"/>
13692     </xs:simpleType>
13693     <xs:simpleType name="ActuatorSwitchFctEnumType">
13694         <xs:restriction base="xs:string">
13695             <xs:enumeration value="on"/>
13696             <xs:enumeration value="off"/>
13697             <xs:enumeration value="toggle"/>
13698         </xs:restriction>
13699     </xs:simpleType>
13700     <xs:complexType name="ActuatorSwitchDataType">
13701         <xs:sequence>
13702             <xs:element name="function" type="ns_p:ActuatorSwitchFctType" minOccurs="0"/>
13703         </xs:sequence>
13704     </xs:complexType>
13705     <xs:element name="actuatorSwitchData" type="ns_p:ActuatorSwitchDataType"/>
13706     <xs:complexType name="ActuatorSwitchDataElementsType">
13707         <xs:sequence>
13708             <xs:element name="function" minOccurs="0" type="ns_p:ElementTagType"/>
13709         </xs:sequence>
13710

```

```

13711     </xs:complexType>
13712     <xs:element name="actuatorSwitchDataElements" type="ns_p:ActuatorSwitchDataElementsType"/>
13713     <xs:complexType name="ActuatorSwitchDescriptionDataType">
13714         <xs:sequence>
13715             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
13716             <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
13717         </xs:sequence>
13718     </xs:complexType>
13719     <xs:element name="actuatorSwitchDescriptionData"
13720 type="ns_p:ActuatorSwitchDescriptionDataType"/>
13721     <xs:complexType name="ActuatorSwitchDescriptionDataElementsType">
13722         <xs:sequence>
13723             <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
13724             <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
13725         </xs:sequence>
13726     </xs:complexType>
13727     <xs:element name="actuatorSwitchDescriptionDataElements"
13728 type="ns_p:ActuatorSwitchDescriptionDataElementsType"/>
13729 </xs:schema>
13730
13731

```

13732

13733 A.2.3 Alarm

13734 File name: EEBus_SPINE_TS_Alarm.xsd

```

13735
13736 <?xml version="1.0" encoding="UTF-8"?>
13737 <!--
13738     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
13739     Version 1.1.1
13740     2018-12-21
13741     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
13742     Source: https://www.eebus.org/en/specifications/
13743 -->
13744 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
13745 xmlns:xs="http://www.w3.org/2001/XMLSchema"
13746 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
13747 blockDefault="#all" elementFormDefault="qualified">
13748     <xs:annotation>
13749         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
13750 e.V. All rights reserved.</xs:documentation>
13751     </xs:annotation>
13752     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
13753     <xs:include schemaLocation="EEBus_SPINE_TS_Threshold.xsd"/>
13754     <xs:simpleType name="AlarmIdType">
13755         <xs:restriction base="xs:unsignedInt"/>
13756     </xs:simpleType>
13757     <xs:simpleType name="AlarmTypeType">
13758         <xs:union memberTypes="ns_p:AlarmTypeEnumType ns_p:EnumExtendType"/>
13759     </xs:simpleType>
13760     <xs:simpleType name="AlarmTypeEnumType">
13761         <xs:restriction base="xs:string">
13762             <xs:enumeration value="alarmCancelled"/>
13763             <xs:enumeration value="underThreshold"/>
13764             <xs:enumeration value="overThreshold"/>
13765         </xs:restriction>
13766     </xs:simpleType>
13767     <xs:complexType name="AlarmDataType">
13768         <xs:sequence>
13769             <xs:element name="alarmId" minOccurs="0" type="ns_p:AlarmIdType"/>
13770             <xs:element minOccurs="0" name="thresholdId" type="ns_p:ThresholdIdType"/>
13771             <xs:element minOccurs="0" name="timestamp"
13772 type="ns_p:AbsoluteOrRelativeTimeType"/>
13773             <xs:element name="alarmType" minOccurs="0" type="ns_p:AlarmTypeType"/>
13774             <xs:element minOccurs="0" name="measuredValue" type="ns_p:ScaledNumberType"/>
13775             <xs:element minOccurs="0" name="evaluationPeriod" type="ns_p:TimePeriodType"/>
13776             <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
13777             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
13778             <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
13779         </xs:sequence>
13780     </xs:complexType>
13781     <xs:element name="alarmData" type="ns_p:AlarmDataType"/>
13782     <xs:complexType name="AlarmDataElementsType">
13783         <xs:sequence>

```

```

13784     <xs:element name="alarmId" minOccurs="0" type="ns_p:ElementTagType"/>
13785     <xs:element minOccurs="0" name="thresholdId" type="ns_p:ElementTagType"/>
13786     <xs:element minOccurs="0" name="timestamp" type="ns_p:ElementTagType"/>
13787     <xs:element name="alarmType" minOccurs="0" type="ns_p:ElementTagType"/>
13788     <xs:element minOccurs="0" name="measuredValue"
13789 type="ns_p:ScaledNumberElementsType"/>
13790     <xs:element minOccurs="0" name="evaluationPeriod"
13791 type="ns_p:TimePeriodElementsType"/>
13792     <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
13793     <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
13794     <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
13795   </xs:sequence>
13796 </xs:complexType>
13797 <xs:element name="alarmDataElements" type="ns_p:AlarmDataElementsType"/>
13798 <xs:complexType name="AlarmListDataType">
13799   <xs:sequence>
13800     <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:alarmData"/>
13801   </xs:sequence>
13802 </xs:complexType>
13803 <xs:element name="alarmListData" type="ns_p:AlarmListDataType"/>
13804 <xs:complexType name="AlarmListDataSelectorsType">
13805   <xs:sequence>
13806     <xs:element name="alarmId" minOccurs="0" type="ns_p:AlarmIdType"/>
13807     <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
13808   </xs:sequence>
13809 </xs:complexType>
13810 <xs:element name="alarmListDataSelectors" type="ns_p:AlarmListDataSelectorsType"/>
13811 </xs:schema>
13812
13813

```

13814

13815 A.2.4 Bill

13816 File name: EEBus_SPINE_TS_Bill.xsd

```

13817 <?xml version="1.0" encoding="UTF-8"?>
13818 <!--
13819   Smart Premises Interoperable Neutral-Message Exchange (SPINE)
13820   Version 1.1.1
13821   2018-12-21
13822   Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
13823   Source: https://www.eebus.org/en/specifications/
13824 -->
13825 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
13826 xmlns:xs="http://www.w3.org/2001/XMLSchema"
13827 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
13828 blockDefault="#all" elementFormDefault="qualified">
13829   <xs:annotation>
13830     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
13831 e.V. All rights reserved.</xs:documentation>
13832   </xs:annotation>
13833   <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
13834   <xs:simpleType name="BillIdType">
13835     <xs:restriction base="xs:unsignedInt"/>
13836   </xs:simpleType>
13837   <xs:simpleType name="BillTypeType">
13838     <xs:union memberTypes="ns_p:BillTypeEnumType ns_p:EnumExtendType"/>
13839   </xs:simpleType>
13840   <xs:simpleType name="BillTypeEnumType">
13841     <xs:restriction base="xs:string">
13842       <xs:enumeration value="chargingSummary"/>
13843     </xs:restriction>
13844   </xs:simpleType>
13845   <xs:simpleType name="BillPositionIdType">
13846     <xs:restriction base="xs:unsignedInt"/>
13847   </xs:simpleType>
13848   <xs:simpleType name="BillPositionCountType">
13849     <xs:restriction base="ns_p:BillPositionIdType"/>
13850   </xs:simpleType>
13851   <xs:simpleType name="BillPositionTypeType">
13852     <xs:union memberTypes="ns_p:BillPositionTypeEnumType ns_p:EnumExtendType"/>
13853   </xs:simpleType>
13854   <xs:simpleType name="BillPositionTypeEnumType">
13855     <xs:restriction base="xs:string">

```

```
13857         <xs:enumeration value="gridElectricEnergy"/>
13858         <xs:enumeration value="selfProducedElectricEnergy"/>
13859     </xs:restriction>
13860 </xs:simpleType>
13861 <xs:simpleType name="BillValueIdType">
13862     <xs:restriction base="xs:unsignedInt"/>
13863 </xs:simpleType>
13864 <xs:simpleType name="BillCostIdType">
13865     <xs:restriction base="xs:unsignedInt"/>
13866 </xs:simpleType>
13867 <xs:simpleType name="BillCostTypeType">
13868     <xs:union memberTypes="ns_p:BillCostTypeEnumType ns_p:EnumExtendType"/>
13869 </xs:simpleType>
13870 <xs:simpleType name="BillCostTypeEnumType">
13871     <xs:restriction base="xs:string">
13872         <xs:enumeration value="absolutePrice"/>
13873         <xs:enumeration value="relativePrice"/>
13874         <xs:enumeration value="co2Emission"/>
13875         <xs:enumeration value="renewableEnergy"/>
13876         <xs:enumeration value="radioactiveWaste"/>
13877     </xs:restriction>
13878 </xs:simpleType>
13879 <xs:complexType name="BillValueType">
13880     <xs:sequence>
13881         <xs:element minOccurs="0" name="valueId" type="ns_p:BillValueIdType"/>
13882         <xs:element minOccurs="0" name="unit" type="ns_p:UnitOfMeasurementType"/>
13883         <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberType"/>
13884         <xs:element minOccurs="0" name="valuePercentage" type="ns_p:ScaledNumberType"/>
13885     </xs:sequence>
13886 </xs:complexType>
13887 <xs:complexType name="BillValueElementsType">
13888     <xs:sequence>
13889         <xs:element minOccurs="0" name="valueId" type="ns_p:ElementTagType"/>
13890         <xs:element minOccurs="0" name="unit" type="ns_p:ElementTagType"/>
13891         <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberElementsType"/>
13892         <xs:element minOccurs="0" name="valuePercentage"
13893 type="ns_p:ScaledNumberElementsType"/>
13894     </xs:sequence>
13895 </xs:complexType>
13896 <xs:complexType name="BillCostType">
13897     <xs:sequence>
13898         <xs:element minOccurs="0" name="costId" type="ns_p:BillCostIdType"/>
13899         <xs:element minOccurs="0" name="costType" type="ns_p:BillCostTypeType"/>
13900         <xs:element minOccurs="0" name="valueId" type="ns_p:BillValueIdType"/>
13901         <xs:element minOccurs="0" name="unit" type="ns_p:UnitOfMeasurementType"/>
13902         <xs:element minOccurs="0" name="currency" type="ns_p:CurrencyType"/>
13903         <xs:element minOccurs="0" name="cost" type="ns_p:ScaledNumberType"/>
13904         <xs:element minOccurs="0" name="costPercentage" type="ns_p:ScaledNumberType"/>
13905     </xs:sequence>
13906 </xs:complexType>
13907 <xs:complexType name="BillCostElementsType">
13908     <xs:sequence>
13909         <xs:element minOccurs="0" name="costId" type="ns_p:ElementTagType"/>
13910         <xs:element minOccurs="0" name="costType" type="ns_p:ElementTagType"/>
13911         <xs:element minOccurs="0" name="valueId" type="ns_p:ElementTagType"/>
13912         <xs:element minOccurs="0" name="unit" type="ns_p:ElementTagType"/>
13913         <xs:element minOccurs="0" name="currency" type="ns_p:ElementTagType"/>
13914         <xs:element minOccurs="0" name="cost" type="ns_p:ScaledNumberElementsType"/>
13915         <xs:element minOccurs="0" name="costPercentage"
13916 type="ns_p:ScaledNumberElementsType"/>
13917     </xs:sequence>
13918 </xs:complexType>
13919 <xs:complexType name="BillPositionType">
13920     <xs:sequence>
13921         <xs:element minOccurs="0" name="positionId" type="ns_p:BillPositionIdType"/>
13922         <xs:element minOccurs="0" name="positionType" type="ns_p:BillPositionTypeType"/>
13923         <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodType"/>
13924         <xs:element maxOccurs="unbounded" minOccurs="0" name="value"
13925 type="ns_p:BillValueType"/>
13926         <xs:element minOccurs="0" name="cost" maxOccurs="unbounded"
13927 type="ns_p:BillCostType"/>
13928         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
13929         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
13930     </xs:sequence>
13931 </xs:complexType>
13932 <xs:complexType name="BillPositionElementsType">
13933     <xs:sequence>
13934         <xs:element minOccurs="0" name="positionId" type="ns_p:ElementTagType"/>
```



```

13935         <xs:element minOccurs="0" name="positionType" type="ns_p:ElementTagType"/>
13936         <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodElementsType"/>
13937         <xs:element minOccurs="0" name="value" type="ns_p:BillValueElementsType"/>
13938         <xs:element minOccurs="0" name="cost" type="ns_p:BillCostElementsType"/>
13939         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
13940         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
13941     </xs:sequence>
13942 </xs:complexType>
13943 <xs:complexType name="BillDataType">
13944     <xs:sequence>
13945         <xs:element minOccurs="0" name="billId" type="ns_p:BillIdType"/>
13946         <xs:element minOccurs="0" name="billType" type="ns_p:BillTypeType"/>
13947         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
13948         <xs:element minOccurs="0" name="total">
13949             <xs:complexType>
13950                 <xs:complexContent>
13951                     <xs:restriction base="ns_p:BillPositionType">
13952                         <xs:sequence>
13953                             <xs:element minOccurs="0" name="timePeriod"
13954 type="ns_p:TimePeriodType"/>
13955                             <xs:element maxOccurs="unbounded" minOccurs="0" name="value">
13956                                 <xs:complexType>
13957                                     <xs:complexContent>
13958                                         <xs:restriction base="ns_p:BillValueType">
13959                                             <xs:sequence>
13960                                                 <xs:element minOccurs="0" name="valueId"
13961 type="ns_p:BillValueIdType"/>
13962                                                 <xs:element minOccurs="0" name="unit"
13963 type="ns_p:UnitOfMeasurementType"/>
13964                                                 <xs:element minOccurs="0" name="value"
13965 type="ns_p:ScaledNumberType"/>
13966                                             </xs:sequence>
13967                                         </xs:restriction>
13968                                     </xs:complexContent>
13969                                 </xs:complexType>
13970                             </xs:element>
13971                             <xs:element minOccurs="0" name="cost" maxOccurs="unbounded">
13972                                 <xs:complexType>
13973                                     <xs:complexContent>
13974                                         <xs:restriction base="ns_p:BillCostType">
13975                                             <xs:sequence>
13976                                                 <xs:element minOccurs="0" name="costId"
13977 type="ns_p:BillCostIdType"/>
13978                                                 <xs:element minOccurs="0" name="costType"
13979 type="ns_p:BillCostTypeType"/>
13980                                                 <xs:element minOccurs="0" name="valueId"
13981 type="ns_p:BillValueIdType"/>
13982                                                 <xs:element minOccurs="0" name="unit"
13983 type="ns_p:UnitOfMeasurementType"/>
13984                                                 <xs:element minOccurs="0" name="currency"
13985 type="ns_p:CurrencyType"/>
13986                                                 <xs:element minOccurs="0" name="cost"
13987 type="ns_p:ScaledNumberType"/>
13988                                             </xs:sequence>
13989                                         </xs:restriction>
13990                                     </xs:complexContent>
13991                                 </xs:complexType>
13992                             </xs:element>
13993                             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
13994                             <xs:element minOccurs="0" name="description"
13995 type="ns_p:DescriptionType"/>
13996                         </xs:sequence>
13997                     </xs:restriction>
13998                 </xs:complexContent>
13999             </xs:complexType>
14000         </xs:element>
14001         <xs:element maxOccurs="unbounded" minOccurs="0" name="position">
14002             <xs:complexType>
14003                 <xs:complexContent>
14004                     <xs:restriction base="ns_p:BillPositionType">
14005                         <xs:sequence>
14006                             <xs:element minOccurs="0" name="positionId"
14007 type="ns_p:BillPositionIdType"/>
14008                             <xs:element minOccurs="0" name="positionType"
14009 type="ns_p:BillPositionTypeType"/>
14010                             <xs:element minOccurs="0" name="timePeriod"
14011 type="ns_p:TimePeriodType"/>
14012                             <xs:element maxOccurs="unbounded" minOccurs="0" name="value">

```

```

14013         <xs:complexType>
14014             <xs:complexContent>
14015                 <xs:restriction base="ns_p:BillValueType">
14016                     <xs:sequence>
14017                         <xs:element minOccurs="0" name="valueId"
14018 type="ns_p:BillValueIdType"/>
14019                         <xs:element minOccurs="0" name="value"
14020 type="ns_p:ScaledNumberType"/>
14021                         <xs:element minOccurs="0"
14022 name="valuePercentage" type="ns_p:ScaledNumberType"/>
14023                     </xs:sequence>
14024                 </xs:restriction>
14025             </xs:complexContent>
14026         </xs:complexType>
14027     </xs:element>
14028     <xs:element minOccurs="0" name="cost" maxOccurs="unbounded">
14029         <xs:complexType>
14030             <xs:complexContent>
14031                 <xs:restriction base="ns_p:BillCostType">
14032                     <xs:sequence>
14033                         <xs:element minOccurs="0" name="costId"
14034 type="ns_p:BillCostIdType"/>
14035                         <xs:element minOccurs="0" name="cost"
14036 type="ns_p:ScaledNumberType"/>
14037                         <xs:element minOccurs="0"
14038 name="costPercentage" type="ns_p:ScaledNumberType"/>
14039                     </xs:sequence>
14040                 </xs:restriction>
14041             </xs:complexContent>
14042         </xs:complexType>
14043     </xs:element>
14044     <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
14045     <xs:element minOccurs="0" name="description"
14046 type="ns_p:DescriptionType"/>
14047     </xs:sequence>
14048 </xs:restriction>
14049 </xs:complexContent>
14050 </xs:complexType>
14051 </xs:element>
14052 </xs:sequence>
14053 </xs:complexType>
14054 <xs:element name="billData" type="ns_p:BillDataType"/>
14055 <xs:complexType name="BillDataElementsType">
14056     <xs:sequence>
14057         <xs:element minOccurs="0" name="billId" type="ns_p:ElementTagType"/>
14058         <xs:element minOccurs="0" name="billType" type="ns_p:ElementTagType"/>
14059         <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
14060         <xs:element minOccurs="0" name="total">
14061             <xs:complexType>
14062                 <xs:complexContent>
14063                     <xs:restriction base="ns_p:BillPositionElementsType">
14064                         <xs:sequence>
14065                             <xs:element minOccurs="0" name="timePeriod"
14066 type="ns_p:TimePeriodElementsType"/>
14067                             <xs:element minOccurs="0" name="value">
14068                                 <xs:complexType>
14069                                     <xs:complexContent>
14070                                         <xs:restriction base="ns_p:BillValueElementsType">
14071                                             <xs:sequence>
14072                                                 <xs:element minOccurs="0" name="valueId"
14073 type="ns_p:ElementTagType"/>
14074                                                 <xs:element minOccurs="0" name="unit"
14075 type="ns_p:ElementTagType"/>
14076                                                 <xs:element minOccurs="0" name="value"
14077 type="ns_p:ScaledNumberElementsType"/>
14078                                             </xs:sequence>
14079                                         </xs:restriction>
14080                                     </xs:complexContent>
14081                                 </xs:complexType>
14082                             </xs:element>
14083                             <xs:element minOccurs="0" name="cost">
14084                                 <xs:complexType>
14085                                     <xs:complexContent>
14086                                         <xs:restriction base="ns_p:BillCostElementsType">
14087                                             <xs:sequence>
14088                                                 <xs:element minOccurs="0" name="costId"
14089 type="ns_p:ElementTagType"/>

```

```

14090                                     <xs:element minOccurs="0" name="costType"
14091 type="ns_p:ElementTagType"/>
14092                                     <xs:element minOccurs="0" name="valueId"
14093 type="ns_p:ElementTagType"/>
14094                                     <xs:element minOccurs="0" name="unit"
14095 type="ns_p:ElementTagType"/>
14096                                     <xs:element minOccurs="0" name="currency"
14097 type="ns_p:ElementTagType"/>
14098                                     <xs:element minOccurs="0" name="cost"
14099 type="ns_p:ScaledNumberElementsType"/>
14100                                     </xs:sequence>
14101                                     </xs:restriction>
14102                                     </xs:complexContent>
14103                                     </xs:complexType>
14104                                   </xs:element>
14105                                   <xs:element minOccurs="0" name="label"
14106 type="ns_p:ElementTagType"/>
14107                                   <xs:element minOccurs="0" name="description"
14108 type="ns_p:ElementTagType"/>
14109                                   </xs:sequence>
14110                                   </xs:restriction>
14111                                   </xs:complexContent>
14112                                   </xs:complexType>
14113                                 </xs:element>
14114                                 <xs:element minOccurs="0" name="position">
14115                                   <xs:complexType>
14116                                     <xs:complexContent>
14117                                       <xs:restriction base="ns_p:BillPositionElementsType">
14118                                         <xs:sequence>
14119                                           <xs:element minOccurs="0" name="positionId"
14120 type="ns_p:ElementTagType"/>
14121                                           <xs:element minOccurs="0" name="positionType"
14122 type="ns_p:ElementTagType"/>
14123                                           <xs:element minOccurs="0" name="timePeriod"
14124 type="ns_p:TimePeriodElementsType"/>
14125                                           <xs:element minOccurs="0" name="value">
14126                                             <xs:complexType>
14127                                               <xs:complexContent>
14128                                                 <xs:restriction base="ns_p:BillValueElementsType">
14129                                                   <xs:sequence>
14130                                                     <xs:element minOccurs="0" name="valueId"
14131 type="ns_p:ElementTagType"/>
14132                                                     <xs:element minOccurs="0" name="value"
14133 type="ns_p:ScaledNumberElementsType"/>
14134                                                     <xs:element minOccurs="0"
14135 name="valuePercentage" type="ns_p:ScaledNumberElementsType"/>
14136                                                   </xs:sequence>
14137                                                 </xs:restriction>
14138                                               </xs:complexContent>
14139                                             </xs:complexType>
14140                                           </xs:element>
14141                                           <xs:element minOccurs="0" name="cost">
14142                                             <xs:complexType>
14143                                               <xs:complexContent>
14144                                                 <xs:restriction base="ns_p:BillCostElementsType">
14145                                                   <xs:sequence>
14146                                                     <xs:element minOccurs="0" name="costId"
14147 type="ns_p:ElementTagType"/>
14148                                                     <xs:element minOccurs="0" name="cost"
14149 type="ns_p:ScaledNumberElementsType"/>
14150                                                     <xs:element minOccurs="0"
14151 name="costPercentage" type="ns_p:ScaledNumberElementsType"/>
14152                                                   </xs:sequence>
14153                                                 </xs:restriction>
14154                                               </xs:complexContent>
14155                                             </xs:complexType>
14156                                           </xs:element>
14157                                           <xs:element minOccurs="0" name="label"
14158 type="ns_p:ElementTagType"/>
14159                                           <xs:element minOccurs="0" name="description"
14160 type="ns_p:ElementTagType"/>
14161                                         </xs:sequence>
14162                                         </xs:restriction>
14163                                         </xs:complexContent>
14164                                       </xs:complexType>
14165                                     </xs:element>
14166                                   </xs:sequence>
14167                                 </xs:complexType>

```

```
14168 <xs:element name="billDataElements" type="ns_p:BillDataElementsType"/>
14169 <xs:complexType name="BillListDataType">
14170 <xs:sequence>
14171 <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:billData"/>
14172 </xs:sequence>
14173 </xs:complexType>
14174 <xs:element name="billListData" type="ns_p:BillListDataType"/>
14175 <xs:complexType name="BillListDataSelectorsType">
14176 <xs:sequence>
14177 <xs:element minOccurs="0" name="billId" type="ns_p:BillIdType"/>
14178 <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
14179 </xs:sequence>
14180 </xs:complexType>
14181 <xs:element name="billListDataSelectors" type="ns_p:BillListDataSelectorsType"/>
14182 <xs:complexType name="BillConstraintsDataType">
14183 <xs:sequence>
14184 <xs:element minOccurs="0" name="billId" type="ns_p:BillIdType"/>
14185 <xs:element minOccurs="0" name="positionCountMin"
14186 type="ns_p:BillPositionCountType"/>
14187 <xs:element minOccurs="0" name="positionCountMax"
14188 type="ns_p:BillPositionCountType"/>
14189 </xs:sequence>
14190 </xs:complexType>
14191 <xs:element name="billConstraintsData" type="ns_p:BillConstraintsDataType"/>
14192 <xs:complexType name="BillConstraintsDataElementsType">
14193 <xs:sequence>
14194 <xs:element minOccurs="0" name="billId" type="ns_p:ElementTagType"/>
14195 <xs:element minOccurs="0" name="positionCountMin" type="ns_p:ElementTagType"/>
14196 <xs:element minOccurs="0" name="positionCountMax" type="ns_p:ElementTagType"/>
14197 </xs:sequence>
14198 </xs:complexType>
14199 <xs:element name="billConstraintsDataElements"
14200 type="ns_p:BillConstraintsDataElementsType"/>
14201 <xs:complexType name="BillConstraintsListDataType">
14202 <xs:sequence>
14203 <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:billConstraintsData"/>
14204 </xs:sequence>
14205 </xs:complexType>
14206 <xs:element name="billConstraintsListData" type="ns_p:BillConstraintsListDataType"/>
14207 <xs:complexType name="BillConstraintsListDataSelectorsType">
14208 <xs:sequence>
14209 <xs:element minOccurs="0" name="billId" type="ns_p:BillIdType"/>
14210 </xs:sequence>
14211 </xs:complexType>
14212 <xs:element name="billConstraintsListDataSelectors"
14213 type="ns_p:BillConstraintsListDataSelectorsType"/>
14214 <xs:complexType name="BillDescriptionDataType">
14215 <xs:sequence>
14216 <xs:element minOccurs="0" name="billId" type="ns_p:BillIdType"/>
14217 <xs:element minOccurs="0" name="billWriteable" type="xs:boolean"/>
14218 <xs:element minOccurs="0" name="updateRequired" type="xs:boolean"/>
14219 <xs:element maxOccurs="unbounded" minOccurs="0" name="supportedBillType"
14220 type="ns_p:BillTypeType"/>
14221 </xs:sequence>
14222 </xs:complexType>
14223 <xs:element name="billDescriptionData" type="ns_p:BillDescriptionDataType"/>
14224 <xs:complexType name="BillDescriptionDataElementsType">
14225 <xs:sequence>
14226 <xs:element minOccurs="0" name="billId" type="ns_p:ElementTagType"/>
14227 <xs:element minOccurs="0" name="billWriteable" type="ns_p:ElementTagType"/>
14228 <xs:element minOccurs="0" name="updateRequired" type="ns_p:ElementTagType"/>
14229 <xs:element minOccurs="0" name="supportedBillType" type="ns_p:ElementTagType"/>
14230 </xs:sequence>
14231 </xs:complexType>
14232 <xs:element name="billDescriptionDataElements"
14233 type="ns_p:BillDescriptionDataElementsType"/>
14234 <xs:complexType name="BillDescriptionListDataType">
14235 <xs:sequence>
14236 <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:billDescriptionData"/>
14237 </xs:sequence>
14238 </xs:complexType>
14239 <xs:element name="billDescriptionListData" type="ns_p:BillDescriptionListDataType"/>
14240 <xs:complexType name="BillDescriptionListDataSelectorsType">
14241 <xs:sequence>
14242 <xs:element minOccurs="0" name="billId" type="ns_p:BillIdType"/>
14243 </xs:sequence>
14244 </xs:complexType>
```

```
14245     <xs:element name="billDescriptionListDataSelectors"
14246 type="ns_p:BillDescriptionListDataSelectorsType"/>
14247 </xs:schema>
14248
14249
```

14251 A.2.5 BindingManagement

14252 File name: EEBus_SPINE_TS_BindingManagement.xsd

```
14253 <?xml version="1.0" encoding="UTF-8"?>
14254 <!--
14255     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
14256     Version 1.1.1
14257     2018-12-21
14258     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
14259     Source: https://www.eebus.org/en/specifications/
14260 -->
14261 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
14262 xmlns:xs="http://www.w3.org/2001/XMLSchema"
14263 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
14264 blockDefault="#all" elementFormDefault="qualified">
14265     <xs:annotation>
14266         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
14267 e.V. All rights reserved.</xs:documentation>
14268     </xs:annotation>
14269     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
14270     <xs:simpleType name="BindingIdType">
14271         <xs:restriction base="xs:unsignedInt"/>
14272     </xs:simpleType>
14273     <xs:complexType name="BindingManagementEntryDataType">
14274         <xs:sequence>
14275             <xs:element minOccurs="0" name="bindingId" type="ns_p:BindingIdType"/>
14276             <xs:element minOccurs="0" name="clientAddress" type="ns_p:FeatureAddressType"/>
14277             <xs:element minOccurs="0" name="serverAddress" type="ns_p:FeatureAddressType"/>
14278             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
14279             <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
14280         </xs:sequence>
14281     </xs:complexType>
14282     <xs:element name="bindingManagementEntryData" type="ns_p:BindingManagementEntryDataType"/>
14283     <xs:complexType name="BindingManagementEntryDataElementsType">
14284         <xs:sequence>
14285             <xs:element minOccurs="0" name="bindingId" type="ns_p:ElementTagType"/>
14286             <xs:element minOccurs="0" name="clientAddress"
14287 type="ns_p:FeatureAddressElementsType"/>
14288             <xs:element minOccurs="0" name="serverAddress"
14289 type="ns_p:FeatureAddressElementsType"/>
14290             <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
14291             <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
14292         </xs:sequence>
14293     </xs:complexType>
14294     <xs:element name="bindingManagementEntryDataElements"
14295 type="ns_p:BindingManagementEntryDataElementsType"/>
14296     <xs:complexType name="BindingManagementEntryListDataType">
14297         <xs:sequence>
14298             <xs:element maxOccurs="unbounded" minOccurs="0"
14299 ref="ns_p:bindingManagementEntryData"/>
14300         </xs:sequence>
14301     </xs:complexType>
14302     <xs:element name="bindingManagementEntryListData"
14303 type="ns_p:BindingManagementEntryListDataType"/>
14304     <xs:complexType name="BindingManagementEntryListDataSelectorsType">
14305         <xs:sequence>
14306             <xs:element minOccurs="0" name="bindingId" type="ns_p:BindingIdType"/>
14307             <xs:element minOccurs="0" name="clientAddress" type="ns_p:FeatureAddressType"/>
14308             <xs:element minOccurs="0" name="serverAddress" type="ns_p:FeatureAddressType"/>
14309         </xs:sequence>
14310     </xs:complexType>
14311     <xs:element name="bindingManagementEntryListDataSelectors"
14312 type="ns_p:BindingManagementEntryListDataSelectorsType"/>
14313     <xs:complexType name="BindingManagementRequestCallType">
14314         <xs:sequence>
14315             <xs:element minOccurs="0" name="clientAddress" type="ns_p:FeatureAddressType"/>
14316             <xs:element minOccurs="0" name="serverAddress" type="ns_p:FeatureAddressType"/>
14317
```

```

14318         <xs:element minOccurs="0" name="serverFeatureType" type="ns_p:FeatureTypeType"/>
14319     </xs:sequence>
14320 </xs:complexType>
14321     <xs:element name="bindingManagementRequestCall"
14322 type="ns_p:BindingManagementRequestCallType"/>
14323     <xs:complexType name="BindingManagementRequestCallElementsType">
14324         <xs:sequence>
14325             <xs:element minOccurs="0" name="clientAddress"
14326 type="ns_p:FeatureAddressElementsType"/>
14327             <xs:element minOccurs="0" name="serverAddress"
14328 type="ns_p:FeatureAddressElementsType"/>
14329             <xs:element minOccurs="0" name="serverFeatureType" type="ns_p:ElementTagType"/>
14330         </xs:sequence>
14331     </xs:complexType>
14332     <xs:element name="bindingManagementRequestCallElements"
14333 type="ns_p:BindingManagementRequestCallElementsType"/>
14334     <xs:complexType name="BindingManagementDeleteCallType">
14335         <xs:sequence>
14336             <xs:element minOccurs="0" name="bindingId" type="ns_p:BindingIdType"/>
14337             <xs:element minOccurs="0" name="clientAddress" type="ns_p:FeatureAddressType"/>
14338             <xs:element minOccurs="0" name="serverAddress" type="ns_p:FeatureAddressType"/>
14339         </xs:sequence>
14340     </xs:complexType>
14341     <xs:element name="bindingManagementDeleteCall"
14342 type="ns_p:BindingManagementDeleteCallType"/>
14343     <xs:complexType name="BindingManagementDeleteCallElementsType">
14344         <xs:sequence>
14345             <xs:element minOccurs="0" name="bindingId" type="ns_p:ElementTagType"/>
14346             <xs:element minOccurs="0" name="clientAddress"
14347 type="ns_p:FeatureAddressElementsType"/>
14348             <xs:element minOccurs="0" name="serverAddress"
14349 type="ns_p:FeatureAddressElementsType"/>
14350         </xs:sequence>
14351     </xs:complexType>
14352     <xs:element name="bindingManagementDeleteCallElements"
14353 type="ns_p:BindingManagementDeleteCallElementsType"/>
14354 </xs:schema>
14355
14356

```

14358 A.2.6 DataTunneling

14359 File name: EEBus_SPINE_TS_DataTunneling.xsd

```

14360 <?xml version="1.0" encoding="UTF-8"?>
14361 <!--
14362     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
14363     Version 1.1.1
14364     2018-12-21
14365     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
14366     Source: https://www.eebus.org/en/specifications/
14367 -->
14368 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
14369 xmlns:xs="http://www.w3.org/2001/XMLSchema"
14370 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
14371 blockDefault="#all" elementFormDefault="qualified">
14372     <xs:annotation>
14373         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
14374 e.V. All rights reserved.</xs:documentation>
14375     </xs:annotation>
14376     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
14377     <xs:simpleType name="PurposeIdType">
14378         <xs:restriction base="xs:string"/>
14379     </xs:simpleType>
14380     <xs:simpleType name="ChannelIdType">
14381         <xs:restriction base="xs:unsignedInt"/>
14382     </xs:simpleType>
14383     <xs:complexType name="DataTunnelingHeaderType">
14384         <xs:sequence>
14385             <xs:element name="purposeId" type="ns_p:PurposeIdType" minOccurs="0"/>
14386             <xs:element name="channelId" type="ns_p:ChannelIdType" minOccurs="0"/>
14387             <xs:element name="sequenceId" type="xs:unsignedInt" minOccurs="0"/>
14388         </xs:sequence>
14389     </xs:complexType>
14390

```

```

14391 <xs:complexType name="DataTunnelingHeaderElementsType">
14392 <xs:sequence>
14393 <xs:element name="purposeId" minOccurs="0" type="ns_p:ElementTagType"/>
14394 <xs:element name="channelId" minOccurs="0" type="ns_p:ElementTagType"/>
14395 <xs:element name="sequenceId" minOccurs="0" type="ns_p:ElementTagType"/>
14396 </xs:sequence>
14397 </xs:complexType>
14398 <xs:complexType name="DataTunnelingCallType">
14399 <xs:sequence>
14400 <xs:element name="header" minOccurs="0" type="ns_p:DataTunnelingHeaderType"/>
14401 <xs:element name="payload" type="xs:hexBinary" minOccurs="0"/>
14402 </xs:sequence>
14403 </xs:complexType>
14404 <xs:element name="dataTunnelingCall" type="ns_p:DataTunnelingCallType"/>
14405 <xs:complexType name="DataTunnelingCallElementsType">
14406 <xs:sequence>
14407 <xs:element name="header" minOccurs="0"
14408 type="ns_p:DataTunnelingHeaderElementsType"/>
14409 <xs:element name="payload" minOccurs="0" type="ns_p:ElementTagType"/>
14410 </xs:sequence>
14411 </xs:complexType>
14412 <xs:element name="dataTunnelingCallElements" type="ns_p:DataTunnelingCallElementsType"/>
14413 </xs:schema>
14414
14415

```

14416

14417 A.2.7 DeviceClassification

14418 File name: EEBus_SPINE_TS_DeviceClassification.xsd

```

14419 <?xml version="1.0" encoding="UTF-8"?>
14420 <!--
14421 Smart Premises Interoperable Neutral-Message Exchange (SPINE)
14422 Version 1.1.1
14423 2018-12-21
14424 Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
14425 Source: https://www.eebus.org/en/specifications/
14426 -->
14427 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
14428 xmlns:xs="http://www.w3.org/2001/XMLSchema"
14429 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
14430 blockDefault="#all" elementFormDefault="qualified">
14431 <xs:annotation>
14432 <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
14433 e.V. All rights reserved.</xs:documentation>
14434 </xs:annotation>
14435 <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
14436 <xs:simpleType name="DeviceClassificationStringType">
14437 <xs:restriction base="xs:string"/>
14438 </xs:simpleType>
14439 <xs:simpleType name="PowerSourceType">
14440 <xs:union memberTypes="ns_p:PowerSourceEnumType ns_p:EnumExtendType"/>
14441 </xs:simpleType>
14442 <xs:simpleType name="PowerSourceEnumType">
14443 <xs:restriction base="xs:string">
14444 <xs:enumeration value="unknown"/>
14445 <xs:enumeration value="mainsSinglePhase"/>
14446 <xs:enumeration value="mains3Phase"/>
14447 <xs:enumeration value="battery"/>
14448 <xs:enumeration value="dc"/>
14449 </xs:restriction>
14450 </xs:simpleType>
14451 <xs:complexType name="DeviceClassificationManufacturerDataType">
14452 <xs:sequence>
14453 <xs:element minOccurs="0" name="deviceName"
14454 type="ns_p:DeviceClassificationStringType"/>
14455 <xs:element minOccurs="0" name="deviceCode"
14456 type="ns_p:DeviceClassificationStringType"/>
14457 <xs:element minOccurs="0" name="serialNumber"
14458 type="ns_p:DeviceClassificationStringType"/>
14459 <xs:element minOccurs="0" name="softwareRevision"
14460 type="ns_p:DeviceClassificationStringType"/>
14461 <xs:element minOccurs="0" name="hardwareRevision"
14462 type="ns_p:DeviceClassificationStringType"/>
14463

```

```

14464         <xs:element minOccurs="0" name="vendorName"
14465 type="ns_p:DeviceClassificationStringType"/>
14466         <xs:element minOccurs="0" name="vendorCode"
14467 type="ns_p:DeviceClassificationStringType"/>
14468         <xs:element minOccurs="0" name="brandName"
14469 type="ns_p:DeviceClassificationStringType"/>
14470         <xs:element minOccurs="0" name="powerSource" type="ns_p:PowerSourceType"/>
14471         <xs:element minOccurs="0" name="manufacturerNodeIdentification"
14472 type="ns_p:DeviceClassificationStringType"/>
14473         <xs:element minOccurs="0" name="manufacturerLabel" type="ns_p:LabelType"/>
14474         <xs:element minOccurs="0" name="manufacturerDescription"
14475 type="ns_p:DescriptionType"/>
14476     </xs:sequence>
14477 </xs:complexType>
14478 <xs:element name="deviceClassificationManufacturerData"
14479 type="ns_p:DeviceClassificationManufacturerDataType"/>
14480 <xs:complexType name="DeviceClassificationManufacturerDataElementsType">
14481     <xs:sequence>
14482         <xs:element minOccurs="0" name="deviceName" type="ns_p:ElementTagType"/>
14483         <xs:element minOccurs="0" name="deviceCode" type="ns_p:ElementTagType"/>
14484         <xs:element minOccurs="0" name="serialNumber" type="ns_p:ElementTagType"/>
14485         <xs:element minOccurs="0" name="softwareRevision" type="ns_p:ElementTagType"/>
14486         <xs:element minOccurs="0" name="hardwareRevision" type="ns_p:ElementTagType"/>
14487         <xs:element minOccurs="0" name="vendorName" type="ns_p:ElementTagType"/>
14488         <xs:element minOccurs="0" name="vendorCode" type="ns_p:ElementTagType"/>
14489         <xs:element minOccurs="0" name="brandName" type="ns_p:ElementTagType"/>
14490         <xs:element minOccurs="0" name="powerSource" type="ns_p:ElementTagType"/>
14491         <xs:element minOccurs="0" name="manufacturerNodeIdentification"
14492 type="ns_p:ElementTagType"/>
14493         <xs:element minOccurs="0" name="manufacturerLabel" type="ns_p:ElementTagType"/>
14494         <xs:element minOccurs="0" name="manufacturerDescription"
14495 type="ns_p:ElementTagType"/>
14496     </xs:sequence>
14497 </xs:complexType>
14498 <xs:element name="deviceClassificationManufacturerDataElements"
14499 type="ns_p:DeviceClassificationManufacturerDataElementsType"/>
14500 <xs:complexType name="DeviceClassificationUserData">
14501     <xs:sequence>
14502         <xs:element minOccurs="0" name="userNodeIdentification"
14503 type="ns_p:DeviceClassificationStringType"/>
14504         <xs:element minOccurs="0" name="userLabel" type="ns_p:LabelType"/>
14505         <xs:element minOccurs="0" name="userDescription" type="ns_p:DescriptionType"/>
14506     </xs:sequence>
14507 </xs:complexType>
14508 <xs:element name="deviceClassificationUserData"
14509 type="ns_p:DeviceClassificationUserData">
14510 <xs:complexType name="DeviceClassificationUserDataElementsType">
14511     <xs:sequence>
14512         <xs:element minOccurs="0" name="userNodeIdentification"
14513 type="ns_p:ElementTagType"/>
14514         <xs:element minOccurs="0" name="userLabel" type="ns_p:ElementTagType"/>
14515         <xs:element minOccurs="0" name="userDescription" type="ns_p:ElementTagType"/>
14516     </xs:sequence>
14517 </xs:complexType>
14518 <xs:element name="deviceClassificationUserDataElements"
14519 type="ns_p:DeviceClassificationUserDataElementsType"/>
14520 </xs:schema>
14521
14522

```

A.2.8 DeviceConfiguration

File name: EEBus_SPINE_TS_DeviceConfiguration.xsd

```

14526 <?xml version="1.0" encoding="UTF-8"?>
14527 <!--
14528     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
14529     Version 1.1.1
14530     2018-12-21
14531     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
14532     Source: https://www.eebus.org/en/specifications/
14533 -->

```



```
14535 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
14536 xmlns:xs="http://www.w3.org/2001/XMLSchema"
14537 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
14538   blockDefault="#all" elementFormDefault="qualified">
14539   <xs:annotation>
14540     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
14541     e.V. All rights reserved.</xs:documentation>
14542   </xs:annotation>
14543   <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
14544   <xs:simpleType name="DeviceConfigurationKeyIdType">
14545     <xs:restriction base="xs:unsignedInt"/>
14546   </xs:simpleType>
14547   <xs:simpleType name="DeviceConfigurationKeyValueStringType">
14548     <xs:restriction base="xs:string"/>
14549   </xs:simpleType>
14550   <xs:simpleType name="DeviceConfigurationKeyNameType">
14551     <xs:union memberTypes="ns_p:DeviceConfigurationKeyNameEnumType ns_p:EnumExtendType"/>
14552   </xs:simpleType>
14553   <xs:simpleType name="DeviceConfigurationKeyNameEnumType">
14554     <xs:restriction base="xs:string">
14555       <xs:enumeration value="peakPowerOfPvSystem"/>
14556       <xs:enumeration value="pvCurtaimentLimitFactor"/>
14557       <xs:enumeration value="asymmetricChargingSupported"/>
14558       <xs:enumeration value="communicationsStandard"/>
14559     </xs:restriction>
14560   </xs:simpleType>
14561   <xs:simpleType name="DeviceConfigurationKeyValueValueType">
14562     <xs:restriction base="xs:string">
14563       <xs:enumeration value="boolean"/>
14564       <xs:enumeration value="date"/>
14565       <xs:enumeration value="dateTime"/>
14566       <xs:enumeration value="duration"/>
14567       <xs:enumeration value="string"/>
14568       <xs:enumeration value="time"/>
14569       <xs:enumeration value="scaledNumber"/>
14570     </xs:restriction>
14571   </xs:simpleType>
14572   <xs:complexType name="DeviceConfigurationKeyValueValueType">
14573     <xs:sequence>
14574       <xs:element minOccurs="0" name="boolean" type="xs:boolean"/>
14575       <xs:element minOccurs="0" name="date" type="xs:date"/>
14576       <xs:element minOccurs="0" name="dateTime" type="xs:dateTime"/>
14577       <xs:element minOccurs="0" name="duration" type="xs:duration"/>
14578       <xs:element minOccurs="0" name="string"
14579 type="ns_p:DeviceConfigurationKeyValueStringType"/>
14580       <xs:element minOccurs="0" name="time" type="xs:time"/>
14581       <xs:element minOccurs="0" name="scaledNumber" type="ns_p:ScaledNumberType"/>
14582     </xs:sequence>
14583   </xs:complexType>
14584   <xs:complexType name="DeviceConfigurationKeyValueValueElementsType">
14585     <xs:sequence>
14586       <xs:element minOccurs="0" name="boolean" type="ns_p:ElementTagType"/>
14587       <xs:element minOccurs="0" name="date" type="ns_p:ElementTagType"/>
14588       <xs:element minOccurs="0" name="dateTime" type="ns_p:ElementTagType"/>
14589       <xs:element minOccurs="0" name="duration" type="ns_p:ElementTagType"/>
14590       <xs:element minOccurs="0" name="string" type="ns_p:ElementTagType"/>
14591       <xs:element minOccurs="0" name="time" type="ns_p:ElementTagType"/>
14592       <xs:element minOccurs="0" name="scaledNumber"
14593 type="ns_p:ScaledNumberElementsType"/>
14594     </xs:sequence>
14595   </xs:complexType>
14596   <xs:complexType name="DeviceConfigurationKeyValueDataType">
14597     <xs:sequence>
14598       <xs:element minOccurs="0" name="keyId" type="ns_p:DeviceConfigurationKeyIdType"/>
14599       <xs:element minOccurs="0" name="value"
14600 type="ns_p:DeviceConfigurationKeyValueValueType"/>
14601       <xs:element minOccurs="0" name="isValueChangeable" type="xs:boolean"/>
14602     </xs:sequence>
14603   </xs:complexType>
14604   <xs:element name="deviceConfigurationKeyValueData"
14605 type="ns_p:DeviceConfigurationKeyValueDataType"/>
14606   <xs:complexType name="DeviceConfigurationKeyValueDataElementsType">
14607     <xs:sequence>
14608       <xs:element minOccurs="0" name="keyId" type="ns_p:ElementTagType"/>
14609       <xs:element minOccurs="0" name="value"
14610 type="ns_p:DeviceConfigurationKeyValueValueElementsType"/>
14611       <xs:element minOccurs="0" name="isValueChangeable" type="ns_p:ElementTagType"/>
14612     </xs:sequence>
```

```

14613     </xs:complexType>
14614     <xs:element name="deviceConfigurationKeyValueDataElements"
14615 type="ns_p:DeviceConfigurationKeyValueDataElementsType"/>
14616     <xs:complexType name="DeviceConfigurationKeyValueListDataType">
14617     <xs:sequence>
14618     <xs:element maxOccurs="unbounded" minOccurs="0"
14619 ref="ns_p:deviceConfigurationKeyValueData"/>
14620     </xs:sequence>
14621     </xs:complexType>
14622     <xs:element name="deviceConfigurationKeyValueListData"
14623 type="ns_p:DeviceConfigurationKeyValueListDataType"/>
14624     <xs:complexType name="DeviceConfigurationKeyValueListDataSelectorsType">
14625     <xs:sequence>
14626     <xs:element minOccurs="0" name="keyId" type="ns_p:DeviceConfigurationKeyIdType"/>
14627     </xs:sequence>
14628     </xs:complexType>
14629     <xs:element name="deviceConfigurationKeyValueListDataSelectors"
14630 type="ns_p:DeviceConfigurationKeyValueListDataSelectorsType"/>
14631     <xs:complexType name="DeviceConfigurationKeyValueDescriptionDataType">
14632     <xs:sequence>
14633     <xs:element minOccurs="0" name="keyId" type="ns_p:DeviceConfigurationKeyIdType"/>
14634     <xs:element minOccurs="0" name="keyName"
14635 type="ns_p:DeviceConfigurationKeyNameType"/>
14636     <xs:element minOccurs="0" name="valueType"
14637 type="ns_p:DeviceConfigurationKeyValueValueTypeType"/>
14638     <xs:element minOccurs="0" name="unit" type="ns_p:UnitOfMeasurementType"/>
14639     <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
14640     <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
14641     </xs:sequence>
14642     </xs:complexType>
14643     <xs:element name="deviceConfigurationKeyValueDescriptionData"
14644 type="ns_p:DeviceConfigurationKeyValueDescriptionDataType"/>
14645     <xs:complexType name="DeviceConfigurationKeyValueDescriptionDataElementsType">
14646     <xs:sequence>
14647     <xs:element minOccurs="0" name="keyId" type="ns_p:ElementTagType"/>
14648     <xs:element minOccurs="0" name="keyName" type="ns_p:ElementTagType"/>
14649     <xs:element minOccurs="0" name="valueType" type="ns_p:ElementTagType"/>
14650     <xs:element minOccurs="0" name="unit" type="ns_p:ElementTagType"/>
14651     <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
14652     <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
14653     </xs:sequence>
14654     </xs:complexType>
14655     <xs:element name="deviceConfigurationKeyValueDescriptionDataElements"
14656 type="ns_p:DeviceConfigurationKeyValueDescriptionDataElementsType"/>
14657     <xs:complexType name="DeviceConfigurationKeyValueDescriptionListDataType">
14658     <xs:sequence>
14659     <xs:element maxOccurs="unbounded" minOccurs="0"
14660 ref="ns_p:deviceConfigurationKeyValueDescriptionData"/>
14661     </xs:sequence>
14662     </xs:complexType>
14663     <xs:element name="deviceConfigurationKeyValueDescriptionListData"
14664 type="ns_p:DeviceConfigurationKeyValueDescriptionListDataType"/>
14665     <xs:complexType name="DeviceConfigurationKeyValueDescriptionListDataSelectorsType">
14666     <xs:sequence>
14667     <xs:element minOccurs="0" name="keyId" type="ns_p:DeviceConfigurationKeyIdType"/>
14668     <xs:element minOccurs="0" name="keyName"
14669 type="ns_p:DeviceConfigurationKeyNameType"/>
14670     </xs:sequence>
14671     </xs:complexType>
14672     <xs:element name="deviceConfigurationKeyValueDescriptionListDataSelectors"
14673 type="ns_p:DeviceConfigurationKeyValueDescriptionListDataSelectorsType"/>
14674     <xs:complexType name="DeviceConfigurationKeyValueConstraintsDataType">
14675     <xs:sequence>
14676     <xs:element minOccurs="0" name="keyId" type="ns_p:DeviceConfigurationKeyIdType"/>
14677     <xs:element minOccurs="0" name="valueRangeMin"
14678 type="ns_p:DeviceConfigurationKeyValueValueType"/>
14679     <xs:element minOccurs="0" name="valueRangeMax"
14680 type="ns_p:DeviceConfigurationKeyValueValueType"/>
14681     <xs:element minOccurs="0" name="valueStepSize"
14682 type="ns_p:DeviceConfigurationKeyValueValueType"/>
14683     </xs:sequence>
14684     </xs:complexType>
14685     <xs:element name="deviceConfigurationKeyValueConstraintsData"
14686 type="ns_p:DeviceConfigurationKeyValueConstraintsDataType"/>
14687     <xs:complexType name="DeviceConfigurationKeyValueConstraintsDataElementsType">
14688     <xs:sequence>
14689     <xs:element minOccurs="0" name="keyId" type="ns_p:ElementTagType"/>

```

```

14690         <xs:element minOccurs="0" name="valueRangeMin"
14691 type="ns_p:DeviceConfigurationKeyValueValueElementsType"/>
14692         <xs:element minOccurs="0" name="valueRangeMax"
14693 type="ns_p:DeviceConfigurationKeyValueValueElementsType"/>
14694         <xs:element minOccurs="0" name="valueStepSize"
14695 type="ns_p:DeviceConfigurationKeyValueValueElementsType"/>
14696     </xs:sequence>
14697 </xs:complexType>
14698     <xs:element name="deviceConfigurationKeyValueConstraintsDataElements"
14699 type="ns_p:DeviceConfigurationKeyValueConstraintsDataElementsType"/>
14700     <xs:complexType name="DeviceConfigurationKeyValueConstraintsListDataType">
14701         <xs:sequence>
14702             <xs:element maxOccurs="unbounded" minOccurs="0"
14703 ref="ns_p:deviceConfigurationKeyValueConstraintsData"/>
14704         </xs:sequence>
14705     </xs:complexType>
14706     <xs:element name="deviceConfigurationKeyValueConstraintsListData"
14707 type="ns_p:DeviceConfigurationKeyValueConstraintsListDataType"/>
14708     <xs:complexType name="DeviceConfigurationKeyValueConstraintsListDataSelectorsType">
14709         <xs:sequence>
14710             <xs:element minOccurs="0" name="keyId" type="ns_p:DeviceConfigurationKeyIdType"/>
14711         </xs:sequence>
14712     </xs:complexType>
14713     <xs:element name="deviceConfigurationKeyValueConstraintsListDataSelectors"
14714 type="ns_p:DeviceConfigurationKeyValueConstraintsListDataSelectorsType"/>
14715 </xs:schema>
14716
14717

```

14719 A.2.9 DeviceDiagnosis

14720 File name: EEBus_SPINE_TS_DeviceDiagnosis.xsd

```

14721 <?xml version="1.0" encoding="UTF-8"?>
14722 <!--
14723     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
14724     Version 1.1.1
14725     2018-12-21
14726     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
14727     Source: https://www.eebus.org/en/specifications/
14728 -->
14729 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
14730 xmlns:xs="http://www.w3.org/2001/XMLSchema"
14731 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
14732 blockDefault="#all" elementFormDefault="qualified">
14733     <xs:annotation>
14734         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
14735 e.V. All rights reserved.</xs:documentation>
14736     </xs:annotation>
14737     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
14738     <xs:simpleType name="VendorStateCodeType">
14739         <xs:restriction base="xs:string"/>
14740     </xs:simpleType>
14741     <xs:simpleType name="LastErrorCodeType">
14742         <xs:restriction base="xs:string"/>
14743     </xs:simpleType>
14744     <xs:simpleType name="DeviceDiagnosisOperatingStateType">
14745         <xs:union memberTypes="ns_p:DeviceDiagnosisOperatingStateEnumType
14746 ns_p:EnumExtendType"/>
14747     </xs:simpleType>
14748     <xs:simpleType name="DeviceDiagnosisOperatingStateEnumType">
14749         <xs:restriction base="xs:string">
14750             <xs:enumeration value="normalOperation"/>
14751             <xs:enumeration value="standby"/>
14752             <xs:enumeration value="failure"/>
14753             <xs:enumeration value="serviceNeeded"/>
14754             <xs:enumeration value="overrideDetected"/>
14755             <xs:enumeration value="inAlarm"/>
14756             <xs:enumeration value="notReachable"/>
14757             <xs:enumeration value="finished"/>
14758         </xs:restriction>
14759     </xs:simpleType>
14760     <xs:simpleType name="PowerSupplyConditionType">
14761         <xs:union memberTypes="ns_p:PowerSupplyConditionEnumType ns_p:EnumExtendType"/>
14762

```

```

14763     </xs:simpleType>
14764     <xs:simpleType name="PowerSupplyConditionEnumType">
14765         <xs:restriction base="xs:string">
14766             <xs:enumeration value="good"/>
14767             <xs:enumeration value="low"/>
14768             <xs:enumeration value="critical"/>
14769             <xs:enumeration value="unknown"/>
14770             <xs:enumeration value="error"/>
14771         </xs:restriction>
14772     </xs:simpleType>
14773     <xs:complexType name="DeviceDiagnosisStateDataType">
14774         <xs:sequence>
14775             <xs:element name="timestamp" type="ns_p:AbsoluteOrRelativeTimeType"
14776 minOccurs="0"/>
14777             <xs:element name="operatingState" type="ns_p:DeviceDiagnosisOperatingStateType"
14778 minOccurs="0"/>
14779             <xs:element name="vendorStateCode" type="ns_p:VendorStateCodeType" minOccurs="0"/>
14780             <xs:element name="lastErrorCode" type="ns_p:LastErrorCodeType" minOccurs="0"/>
14781             <xs:element name="upTime" type="xs:duration" minOccurs="0"/>
14782             <xs:element name="totalUpTime" type="xs:duration" minOccurs="0"/>
14783             <xs:element name="powerSupplyCondition" minOccurs="0"
14784 type="ns_p:PowerSupplyConditionType"/>
14785         </xs:sequence>
14786     </xs:complexType>
14787     <xs:element name="deviceDiagnosisStateData" type="ns_p:DeviceDiagnosisStateDataType"/>
14788     <xs:complexType name="DeviceDiagnosisStateDataElementsType">
14789         <xs:sequence>
14790             <xs:element name="timestamp" minOccurs="0" type="ns_p:ElementTagType"/>
14791             <xs:element name="operatingState" minOccurs="0" type="ns_p:ElementTagType"/>
14792             <xs:element name="vendorStateCode" minOccurs="0" type="ns_p:ElementTagType"/>
14793             <xs:element name="lastErrorCode" minOccurs="0" type="ns_p:ElementTagType"/>
14794             <xs:element name="upTime" minOccurs="0" type="ns_p:ElementTagType"/>
14795             <xs:element name="totalUpTime" minOccurs="0" type="ns_p:ElementTagType"/>
14796             <xs:element name="powerSupplyCondition" minOccurs="0" type="ns_p:ElementTagType"/>
14797         </xs:sequence>
14798     </xs:complexType>
14799     <xs:element name="deviceDiagnosisStateDataElements"
14800 type="ns_p:DeviceDiagnosisStateDataElementsType"/>
14801     <xs:complexType name="DeviceDiagnosisHeartbeatDataType">
14802         <xs:sequence>
14803             <xs:element name="timestamp" type="ns_p:AbsoluteOrRelativeTimeType"
14804 minOccurs="0"/>
14805             <xs:element name="heartbeatCounter" type="xs:unsignedLong" minOccurs="0"/>
14806             <xs:element name="heartbeatTimeout" type="xs:duration" minOccurs="0"/>
14807         </xs:sequence>
14808     </xs:complexType>
14809     <xs:element name="deviceDiagnosisHeartbeatData"
14810 type="ns_p:DeviceDiagnosisHeartbeatDataType"/>
14811     <xs:complexType name="DeviceDiagnosisHeartbeatDataElementsType">
14812         <xs:sequence>
14813             <xs:element name="timestamp" minOccurs="0" type="ns_p:ElementTagType"/>
14814             <xs:element name="heartbeatCounter" minOccurs="0" type="ns_p:ElementTagType"/>
14815             <xs:element name="heartbeatTimeout" minOccurs="0" type="ns_p:ElementTagType"/>
14816         </xs:sequence>
14817     </xs:complexType>
14818     <xs:element name="deviceDiagnosisHeartbeatDataElements"
14819 type="ns_p:DeviceDiagnosisHeartbeatDataElementsType"/>
14820     <xs:complexType name="DeviceDiagnosisServiceDataType">
14821         <xs:sequence>
14822             <xs:element name="timestamp" type="ns_p:AbsoluteOrRelativeTimeType"
14823 minOccurs="0"/>
14824             <xs:element name="installationTime" type="ns_p:AbsoluteOrRelativeTimeType"
14825 minOccurs="0"/>
14826             <xs:element name="bootCounter" type="xs:unsignedLong" minOccurs="0"/>
14827             <xs:element name="nextService" type="ns_p:AbsoluteOrRelativeTimeType"
14828 minOccurs="0"/>
14829         </xs:sequence>
14830     </xs:complexType>
14831     <xs:element name="deviceDiagnosisServiceData" type="ns_p:DeviceDiagnosisServiceDataType"/>
14832     <xs:complexType name="DeviceDiagnosisServiceDataElementsType">
14833         <xs:sequence>
14834             <xs:element name="timestamp" minOccurs="0" type="ns_p:ElementTagType"/>
14835             <xs:element name="installationTime" minOccurs="0" type="ns_p:ElementTagType"/>
14836             <xs:element name="bootCounter" minOccurs="0" type="ns_p:ElementTagType"/>
14837             <xs:element name="nextService" minOccurs="0" type="ns_p:ElementTagType"/>
14838         </xs:sequence>
14839     </xs:complexType>

```

```
14840     <xs:element name="deviceDiagnosisServiceDataElements"
14841 type="ns_p:DeviceDiagnosisServiceDataElementsType"/>
14842 </xs:schema>
```

14846 A.2.10 DirectControl

14847 File name: EEBus_SPINE_TS_DirectControl.xsd

```
14848 <?xml version="1.0" encoding="UTF-8"?>
14849 <!--
14850     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
14851     Version 1.1.1
14852     2018-12-21
14853     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
14854     Source: https://www.eebus.org/en/specifications/
14855 -->
14856 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
14857 xmlns:xs="http://www.w3.org/2001/XMLSchema"
14858 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
14859 blockDefault="#all" elementFormDefault="qualified">
14860     <xs:annotation>
14861         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
14862 e.V. All rights reserved.</xs:documentation>
14863     </xs:annotation>
14864     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
14865     <xs:include schemaLocation="EEBus_SPINE_TS_PowerSequences.xsd"/>
14866     <xs:simpleType name="DirectControlActivityStateType">
14867         <xs:union memberTypes="ns_p:DirectControlActivityStateEnumType ns_p:EnumExtendType"/>
14868     </xs:simpleType>
14869     <xs:simpleType name="DirectControlActivityStateEnumType">
14870         <xs:restriction base="xs:string">
14871             <xs:enumeration value="running"/>
14872             <xs:enumeration value="paused"/>
14873             <xs:enumeration value="inactive"/>
14874         </xs:restriction>
14875     </xs:simpleType>
14876     <xs:complexType name="DirectControlActivityDataType">
14877         <xs:sequence>
14878             <xs:element minOccurs="0" name="timestamp"
14879 type="ns_p:AbsoluteOrRelativeTimeType"/>
14880             <xs:element minOccurs="0" name="activityState"
14881 type="ns_p:DirectControlActivityStateType"/>
14882             <xs:element minOccurs="0" name="isActivityStateChangeable" type="xs:boolean"/>
14883             <xs:element minOccurs="0" name="energyMode" type="ns_p:EnergyModeType"/>
14884             <xs:element minOccurs="0" name="isEnergyModeChangeable" type="xs:boolean"/>
14885             <xs:element name="power" type="ns_p:ScaledNumberType" minOccurs="0"/>
14886             <xs:element minOccurs="0" name="isPowerChangeable" type="xs:boolean"/>
14887             <xs:element name="energy" type="ns_p:ScaledNumberType" minOccurs="0"/>
14888             <xs:element minOccurs="0" name="isEnergyChangeable" type="xs:boolean"/>
14889             <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
14890         </xs:sequence>
14891     </xs:complexType>
14892     <xs:element name="directControlActivityData" type="ns_p:DirectControlActivityDataType"/>
14893     <xs:complexType name="DirectControlActivityDataElementsType">
14894         <xs:sequence>
14895             <xs:element minOccurs="0" name="timestamp" type="ns_p:ElementTagType"/>
14896             <xs:element minOccurs="0" name="activityState" type="ns_p:ElementTagType"/>
14897             <xs:element minOccurs="0" name="isActivityStateChangeable"
14898 type="ns_p:ElementTagType"/>
14899             <xs:element minOccurs="0" name="energyMode" type="ns_p:ElementTagType"/>
14900             <xs:element minOccurs="0" name="isEnergyModeChangeable"
14901 type="ns_p:ElementTagType"/>
14902             <xs:element name="power" minOccurs="0" type="ns_p:ScaledNumberElementsType"/>
14903             <xs:element minOccurs="0" name="isPowerChangeable" type="ns_p:ElementTagType"/>
14904             <xs:element name="energy" minOccurs="0" type="ns_p:ScaledNumberElementsType"/>
14905             <xs:element minOccurs="0" name="isEnergyChangeable" type="ns_p:ElementTagType"/>
14906             <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
14907         </xs:sequence>
14908     </xs:complexType>
14909     <xs:element name="directControlActivityDataElements"
14910 type="ns_p:DirectControlActivityDataElementsType"/>
14911     <xs:complexType name="DirectControlActivityListDataType">
```

```

14913     <xs:sequence>
14914         <xs:element maxOccurs="unbounded" minOccurs="0"
14915 ref="ns_p:directControlActivityData"/>
14916     </xs:sequence>
14917 </xs:complexType>
14918 <xs:element name="directControlActivityListData"
14919 type="ns_p:DirectControlActivityListDataType"/>
14920 <xs:complexType name="DirectControlActivityListDataSelectorsType">
14921     <xs:sequence>
14922         <xs:element minOccurs="0" name="timestampInterval"
14923 type="ns_p:TimestampIntervalType"/>
14924     </xs:sequence>
14925 </xs:complexType>
14926 <xs:element name="directControlActivityListDataSelectors"
14927 type="ns_p:DirectControlActivityListDataSelectorsType"/>
14928 <xs:complexType name="DirectControlDescriptionDataType">
14929     <xs:sequence>
14930         <xs:element minOccurs="0" name="positiveEnergyDirection"
14931 type="ns_p:EnergyDirectionType"/>
14932         <xs:element minOccurs="0" name="powerUnit" type="ns_p:UnitOfMeasurementType"/>
14933         <xs:element minOccurs="0" name="energyUnit" type="ns_p:UnitOfMeasurementType"/>
14934     </xs:sequence>
14935 </xs:complexType>
14936 <xs:element name="directControlDescriptionData"
14937 type="ns_p:DirectControlDescriptionDataType"/>
14938 <xs:complexType name="DirectControlDescriptionDataElementsType">
14939     <xs:sequence>
14940         <xs:element minOccurs="0" name="positiveEnergyDirection"
14941 type="ns_p:ElementTagType"/>
14942         <xs:element minOccurs="0" name="powerUnit" type="ns_p:ElementTagType"/>
14943         <xs:element minOccurs="0" name="energyUnit" type="ns_p:ElementTagType"/>
14944     </xs:sequence>
14945 </xs:complexType>
14946 <xs:element name="directControlDescriptionDataElements"
14947 type="ns_p:DirectControlDescriptionDataElementsType"/>
14948 </xs:schema>
14949
14950

```

14952 A.2.11 ElectricalConnection

14953 File name: EEBus_SPINE_TS_ElectricalConnection.xsd

```

14954 <?xml version="1.0" encoding="UTF-8"?>
14955 <!--
14956     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
14957     Version 1.1.1
14958     2018-12-21
14959     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
14960     Source: https://www.eebus.org/en/specifications/
14961 -->
14962 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
14963 xmlns:xs="http://www.w3.org/2001/XMLSchema"
14964 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
14965 blockDefault="#all" elementFormDefault="qualified">
14966     <xs:annotation>
14967         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
14968 e.V. All rights reserved.</xs:documentation>
14969     </xs:annotation>
14970     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
14971     <xs:include schemaLocation="EEBus_SPINE_TS_Measurement.xsd"/>
14972     <xs:simpleType name="ElectricalConnectionIdType">
14973         <xs:restriction base="xs:unsignedInt"/>
14974     </xs:simpleType>
14975     <xs:simpleType name="ElectricalConnectionParameterIdType">
14976         <xs:restriction base="xs:unsignedInt"/>
14977     </xs:simpleType>
14978     <xs:simpleType name="ElectricalConnectionMeasurandVariantType">
14979         <xs:union memberTypes="ns_p:ElectricalConnectionMeasurandVariantEnumType
14980 ns_p:EnumExtendType"/>
14981     </xs:simpleType>
14982     <xs:simpleType name="ElectricalConnectionMeasurandVariantEnumType">
14983         <xs:restriction base="xs:string">
14984             <xs:enumeration value="amplitude"/>
14985

```

```

14986         <xs:enumeration value="rms"/>
14987         <xs:enumeration value="instantaneous"/>
14988         <xs:enumeration value="angle"/>
14989         <xs:enumeration value="cosPhi"/>
14990     </xs:restriction>
14991 </xs:simpleType>
14992 <xs:simpleType name="ElectricalConnectionVoltageTypeType">
14993     <xs:union memberTypes="ns_p:ElectricalConnectionVoltageTypeEnumType
14994 ns_p:EnumExtendType"/>
14995 </xs:simpleType>
14996 <xs:simpleType name="ElectricalConnectionVoltageTypeEnumType">
14997     <xs:restriction base="xs:string">
14998         <xs:enumeration value="ac"/>
14999         <xs:enumeration value="dc"/>
15000     </xs:restriction>
15001 </xs:simpleType>
15002 <xs:simpleType name="ElectricalConnectionAcMeasurementTypeType">
15003     <xs:union memberTypes="ns_p:ElectricalConnectionAcMeasurementTypeEnumType
15004 ns_p:EnumExtendType"/>
15005 </xs:simpleType>
15006 <xs:simpleType name="ElectricalConnectionAcMeasurementTypeEnumType">
15007     <xs:restriction base="xs:string">
15008         <xs:enumeration value="real"/>
15009         <xs:enumeration value="reactive"/>
15010         <xs:enumeration value="apparent"/>
15011         <xs:enumeration value="phase"/>
15012     </xs:restriction>
15013 </xs:simpleType>
15014 <xs:simpleType name="ElectricalConnectionPhaseNameType">
15015     <xs:union memberTypes="ns_p:ElectricalConnectionPhaseNameEnumType
15016 ns_p:EnumExtendType"/>
15017 </xs:simpleType>
15018 <xs:simpleType name="ElectricalConnectionPhaseNameEnumType">
15019     <xs:restriction base="xs:string">
15020         <xs:enumeration value="a"/>
15021         <xs:enumeration value="b"/>
15022         <xs:enumeration value="c"/>
15023         <xs:enumeration value="ab"/>
15024         <xs:enumeration value="bc"/>
15025         <xs:enumeration value="ac"/>
15026         <xs:enumeration value="abc"/>
15027         <xs:enumeration value="neutral"/>
15028         <xs:enumeration value="ground"/>
15029         <xs:enumeration value="none"/>
15030     </xs:restriction>
15031 </xs:simpleType>
15032 <xs:simpleType name="ElectricalConnectionConnectionPointType">
15033     <xs:restriction base="xs:string">
15034         <xs:enumeration value="grid"/>
15035         <xs:enumeration value="home"/>
15036         <xs:enumeration value="pv"/>
15037         <xs:enumeration value="sd"/>
15038         <xs:enumeration value="other"/>
15039     </xs:restriction>
15040 </xs:simpleType>
15041 <xs:complexType name="ElectricalConnectionParameterDescriptionDataType">
15042     <xs:sequence>
15043         <xs:element name="electricalConnectionId" minOccurs="0"
15044 type="ns_p:ElectricalConnectionIdType"/>
15045         <xs:element name="parameterId" minOccurs="0"
15046 type="ns_p:ElectricalConnectionParameterIdType"/>
15047         <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
15048         <xs:element minOccurs="0" name="voltageType"
15049 type="ns_p:ElectricalConnectionVoltageTypeType"/>
15050         <xs:element name="acMeasuredPhases" minOccurs="0"
15051 type="ns_p:ElectricalConnectionPhaseNameType"/>
15052         <xs:element name="acMeasuredInReferenceTo" minOccurs="0"
15053 type="ns_p:ElectricalConnectionPhaseNameType"/>
15054         <xs:element name="acMeasurementType"
15055 type="ns_p:ElectricalConnectionAcMeasurementTypeType" minOccurs="0"/>
15056         <xs:element name="acMeasurementVariant" minOccurs="0"
15057 type="ns_p:ElectricalConnectionMeasurementVariantType"/>
15058         <xs:element minOccurs="0" name="acMeasuredHarmonic" type="xs:unsignedByte"/>
15059         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
15060         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
15061         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
15062     </xs:sequence>
15063 </xs:complexType>

```

```

15064     <xs:element name="electricalConnectionParameterDescriptionData"
15065 type="ns_p:ElectricalConnectionParameterDescriptionDataType"/>
15066     <xs:complexType name="ElectricalConnectionParameterDescriptionDataElementsType">
15067       <xs:sequence>
15068         <xs:element name="electricalConnectionId" minOccurs="0"
15069 type="ns_p:ElementTagType"/>
15070         <xs:element name="parameterId" minOccurs="0" type="ns_p:ElementTagType"/>
15071         <xs:element minOccurs="0" name="measurementId" type="ns_p:ElementTagType"/>
15072         <xs:element minOccurs="0" name="voltageType" type="ns_p:ElementTagType"/>
15073         <xs:element name="acMeasuredPhases" minOccurs="0" type="ns_p:ElementTagType"/>
15074         <xs:element name="acMeasuredInReferenceTo" minOccurs="0"
15075 type="ns_p:ElementTagType"/>
15076         <xs:element name="acMeasurementType" minOccurs="0" type="ns_p:ElementTagType"/>
15077         <xs:element name="acMeasurementVariant" minOccurs="0" type="ns_p:ElementTagType"/>
15078         <xs:element minOccurs="0" name="acMeasuredHarmonic" type="ns_p:ElementTagType"/>
15079         <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
15080         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
15081         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
15082       </xs:sequence>
15083     </xs:complexType>
15084     <xs:element name="electricalConnectionParameterDescriptionDataElements"
15085 type="ns_p:ElectricalConnectionParameterDescriptionDataElementsType"/>
15086     <xs:complexType name="ElectricalConnectionParameterDescriptionListDataType">
15087       <xs:sequence>
15088         <xs:element maxOccurs="unbounded" minOccurs="0"
15089 ref="ns_p:electricalConnectionParameterDescriptionData"/>
15090       </xs:sequence>
15091     </xs:complexType>
15092     <xs:element name="electricalConnectionParameterDescriptionListData"
15093 type="ns_p:ElectricalConnectionParameterDescriptionListDataType"/>
15094     <xs:complexType name="ElectricalConnectionParameterDescriptionListDataSelectorsType">
15095       <xs:sequence>
15096         <xs:element name="electricalConnectionId" minOccurs="0"
15097 type="ns_p:ElectricalConnectionIdType"/>
15098         <xs:element name="parameterId" minOccurs="0"
15099 type="ns_p:ElectricalConnectionParameterIdType"/>
15100         <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
15101         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
15102       </xs:sequence>
15103     </xs:complexType>
15104     <xs:element name="electricalConnectionParameterDescriptionListDataSelectors"
15105 type="ns_p:ElectricalConnectionParameterDescriptionListDataSelectorsType"/>
15106     <xs:complexType name="ElectricalConnectionPermittedValueSetDataType">
15107       <xs:sequence>
15108         <xs:element name="electricalConnectionId" minOccurs="0"
15109 type="ns_p:ElectricalConnectionIdType"/>
15110         <xs:element name="parameterId" minOccurs="0"
15111 type="ns_p:ElectricalConnectionParameterIdType"/>
15112         <xs:element minOccurs="0" name="permittedValueSet" type="ns_p:ScaledNumberSetType"
15113 maxOccurs="unbounded"/>
15114       </xs:sequence>
15115     </xs:complexType>
15116     <xs:element name="electricalConnectionPermittedValueSetData"
15117 type="ns_p:ElectricalConnectionPermittedValueSetDataType"/>
15118     <xs:complexType name="ElectricalConnectionPermittedValueSetDataElementsType">
15119       <xs:sequence>
15120         <xs:element name="electricalConnectionId" minOccurs="0"
15121 type="ns_p:ElementTagType"/>
15122         <xs:element name="parameterId" minOccurs="0" type="ns_p:ElementTagType"/>
15123         <xs:element minOccurs="0" name="permittedValueSet"
15124 type="ns_p:ScaledNumberSetElementsType"/>
15125       </xs:sequence>
15126     </xs:complexType>
15127     <xs:element name="electricalConnectionPermittedValueSetDataElements"
15128 type="ns_p:ElectricalConnectionPermittedValueSetDataElementsType"/>
15129     <xs:complexType name="ElectricalConnectionPermittedValueSetListDataType">
15130       <xs:sequence>
15131         <xs:element maxOccurs="unbounded" minOccurs="0"
15132 ref="ns_p:electricalConnectionPermittedValueSetData"/>
15133       </xs:sequence>
15134     </xs:complexType>
15135     <xs:element name="electricalConnectionPermittedValueSetListData"
15136 type="ns_p:ElectricalConnectionPermittedValueSetListDataType"/>
15137     <xs:complexType name="ElectricalConnectionPermittedValueSetListDataSelectorsType">
15138       <xs:sequence>
15139         <xs:element name="electricalConnectionId" minOccurs="0"
15140 type="ns_p:ElectricalConnectionIdType"/>

```



```
15141         <xs:element name="parameterId" minOccurs="0"
15142 type="ns_p:ElectricalConnectionParameterIdType"/>
15143     </xs:sequence>
15144 </xs:complexType>
15145     <xs:element name="electricalConnectionPermittedValueSetListDataSelectors"
15146 type="ns_p:ElectricalConnectionPermittedValueSetListDataSelectorsType"/>
15147
15148     <xs:complexType name="ElectricalConnectionStateDataType">
15149         <xs:sequence>
15150             <xs:element name="electricalConnectionId" minOccurs="0"
15151 type="ns_p:ElectricalConnectionIdType"/>
15152             <xs:element minOccurs="0" name="timestamp"
15153 type="ns_p:AbsoluteOrRelativeTimeType"/>
15154             <xs:element name="currentEnergyMode" minOccurs="0" type="ns_p:EnergyModeType"/>
15155             <xs:element minOccurs="0" name="consumptionTime" type="xs:duration"/>
15156             <xs:element minOccurs="0" name="productionTime" type="xs:duration"/>
15157             <xs:element minOccurs="0" name="totalConsumptionTime" type="xs:duration"/>
15158             <xs:element minOccurs="0" name="totalProductionTime" type="xs:duration"/>
15159         </xs:sequence>
15160     </xs:complexType>
15161     <xs:element name="electricalConnectionStateData"
15162 type="ns_p:ElectricalConnectionStateDataType"/>
15163     <xs:complexType name="ElectricalConnectionStateDataElementsType">
15164         <xs:sequence>
15165             <xs:element name="electricalConnectionId" minOccurs="0"
15166 type="ns_p:ElementTagType"/>
15167             <xs:element minOccurs="0" name="timestamp" type="ns_p:ElementTagType"/>
15168             <xs:element name="currentEnergyMode" minOccurs="0" type="ns_p:ElementTagType"/>
15169             <xs:element minOccurs="0" name="consumptionTime" type="ns_p:ElementTagType"/>
15170             <xs:element minOccurs="0" name="productionTime" type="ns_p:ElementTagType"/>
15171             <xs:element minOccurs="0" name="totalConsumptionTime" type="ns_p:ElementTagType"/>
15172             <xs:element minOccurs="0" name="totalProductionTime" type="ns_p:ElementTagType"/>
15173         </xs:sequence>
15174     </xs:complexType>
15175     <xs:element name="electricalConnectionStateDataElements"
15176 type="ns_p:ElectricalConnectionStateDataElementsType"/>
15177     <xs:complexType name="ElectricalConnectionStateListDataType">
15178         <xs:sequence>
15179             <xs:element maxOccurs="unbounded" minOccurs="0"
15180 ref="ns_p:electricalConnectionStateData"/>
15181         </xs:sequence>
15182     </xs:complexType>
15183     <xs:element name="electricalConnectionStateListData"
15184 type="ns_p:ElectricalConnectionStateListDataType"/>
15185     <xs:complexType name="ElectricalConnectionStateListDataSelectorsType">
15186         <xs:sequence>
15187             <xs:element name="electricalConnectionId" minOccurs="0"
15188 type="ns_p:ElectricalConnectionIdType"/>
15189         </xs:sequence>
15190     </xs:complexType>
15191     <xs:element name="electricalConnectionStateListDataSelectors"
15192 type="ns_p:ElectricalConnectionStateListDataSelectorsType"/>
15193     <xs:complexType name="ElectricalConnectionDescriptionDataType">
15194         <xs:sequence>
15195             <xs:element name="electricalConnectionId" minOccurs="0"
15196 type="ns_p:ElectricalConnectionIdType"/>
15197             <xs:element name="powerSupplyType" type="ns_p:ElectricalConnectionVoltageTypeType"
15198 minOccurs="0"/>
15199             <xs:element name="acConnectedPhases" minOccurs="0" type="xs:unsignedInt"/>
15200             <xs:element minOccurs="0" name="acRmsPeriodDuration" type="xs:duration"/>
15201             <xs:element minOccurs="0" name="positiveEnergyDirection"
15202 type="ns_p:EnergyDirectionType"/>
15203             <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
15204             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
15205             <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
15206         </xs:sequence>
15207     </xs:complexType>
15208     <xs:element name="electricalConnectionDescriptionData"
15209 type="ns_p:ElectricalConnectionDescriptionDataType"/>
15210     <xs:complexType name="ElectricalConnectionDescriptionDataElementsType">
15211         <xs:sequence>
15212             <xs:element name="electricalConnectionId" minOccurs="0"
15213 type="ns_p:ElementTagType"/>
15214             <xs:element name="powerSupplyType" minOccurs="0" type="ns_p:ElementTagType"/>
15215             <xs:element name="acConnectedPhases" minOccurs="0" type="ns_p:ElementTagType"/>
15216             <xs:element minOccurs="0" name="acRmsPeriodDuration" type="ns_p:ElementTagType"/>
15217             <xs:element minOccurs="0" name="positiveEnergyDirection"
15218 type="ns_p:ElementTagType"/>
```

```

15219         <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
15220         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
15221         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
15222     </xs:sequence>
15223 </xs:complexType>
15224 <xs:element name="electricalConnectionDescriptionDataElements"
15225 type="ns_p:ElectricalConnectionDescriptionDataElementsType"/>
15226 <xs:complexType name="ElectricalConnectionDescriptionListDataType">
15227     <xs:sequence>
15228         <xs:element maxOccurs="unbounded" minOccurs="0"
15229 ref="ns_p:electricalConnectionDescriptionData"/>
15230     </xs:sequence>
15231 </xs:complexType>
15232 <xs:element name="electricalConnectionDescriptionListData"
15233 type="ns_p:ElectricalConnectionDescriptionListDataType"/>
15234 <xs:complexType name="ElectricalConnectionDescriptionListDataSelectorsType">
15235     <xs:sequence>
15236         <xs:element name="electricalConnectionId" minOccurs="0"
15237 type="ns_p:ElectricalConnectionIdType"/>
15238         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
15239     </xs:sequence>
15240 </xs:complexType>
15241 <xs:element name="electricalConnectionDescriptionListDataSelectors"
15242 type="ns_p:ElectricalConnectionDescriptionListDataSelectorsType"/>
15243 </xs:schema>
15244
15245

```

15246

15247 **A.2.12 HVAC**

15248 File name: EEBus_SPINE_TS_HVAC.xsd

```

15249
15250 <?xml version="1.0" encoding="UTF-8"?>
15251 <!--
15252     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
15253     Version 1.1.1
15254     2018-12-21
15255     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
15256     Source: https://www.eebus.org/en/specifications/
15257 -->
15258 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
15259 xmlns:xs="http://www.w3.org/2001/XMLSchema"
15260 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
15261 blockDefault="#all" elementFormDefault="qualified">
15262     <xs:annotation>
15263         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
15264 e.V. All rights reserved.</xs:documentation>
15265     </xs:annotation>
15266     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
15267     <xs:include schemaLocation="EEBus_SPINE_TS_PowerSequences.xsd"/>
15268     <xs:include schemaLocation="EEBus_SPINE_TS_Setpoint.xsd"/>
15269     <xs:simpleType name="HvacSystemFunctionIdType">
15270         <xs:restriction base="xs:unsignedInt"/>
15271     </xs:simpleType>
15272     <xs:simpleType name="HvacSystemFunctionTypeType">
15273         <xs:union memberTypes="ns_p:HvacSystemFunctionTypeEnumType ns_p:EnumExtendType"/>
15274     </xs:simpleType>
15275     <xs:simpleType name="HvacSystemFunctionTypeEnumType">
15276         <xs:restriction base="xs:string">
15277             <xs:enumeration value="heating"/>
15278             <xs:enumeration value="cooling"/>
15279             <xs:enumeration value="ventilation"/>
15280             <xs:enumeration value="dhw"/>
15281         </xs:restriction>
15282     </xs:simpleType>
15283     <xs:simpleType name="HvacOperationModeIdType">
15284         <xs:restriction base="xs:unsignedInt"/>
15285     </xs:simpleType>
15286     <xs:simpleType name="HvacOperationModeTypeType">
15287         <xs:union memberTypes="ns_p:HvacOperationModeTypeEnumType ns_p:EnumExtendType"/>
15288     </xs:simpleType>
15289     <xs:simpleType name="HvacOperationModeTypeEnumType">
15290         <xs:restriction base="xs:string">
15291             <xs:enumeration value="auto"/>

```

```

15292         <xs:enumeration value="on"/>
15293         <xs:enumeration value="off"/>
15294         <xs:enumeration value="eco"/>
15295     </xs:restriction>
15296 </xs:simpleType>
15297 <xs:simpleType name="HvacOverrunIdType">
15298     <xs:restriction base="xs:unsignedInt"/>
15299 </xs:simpleType>
15300 <xs:simpleType name="HvacOverrunTypeEnumType">
15301     <xs:union memberTypes="ns_p:HvacOverrunTypeEnumType ns_p:EnumExtendType"/>
15302 </xs:simpleType>
15303 <xs:simpleType name="HvacOverrunTypeEnumType">
15304     <xs:restriction base="xs:string">
15305         <xs:enumeration value="oneTimeDhw"/>
15306         <xs:enumeration value="party"/>
15307         <xs:enumeration value="sgReadyCondition1"/>
15308         <xs:enumeration value="sgReadyCondition3"/>
15309         <xs:enumeration value="sgReadyCondition4"/>
15310         <xs:enumeration value="oneDayAway"/>
15311         <xs:enumeration value="oneDayAtHome"/>
15312         <xs:enumeration value="oneTimeVentilation"/>
15313         <xs:enumeration value="hvacSystemOff"/>
15314         <xs:enumeration value="valveKick"/>
15315     </xs:restriction>
15316 </xs:simpleType>
15317 <xs:simpleType name="HvacOverrunStatusType">
15318     <xs:union memberTypes="ns_p:HvacOverrunStatusEnumType ns_p:EnumExtendType"/>
15319 </xs:simpleType>
15320 <xs:simpleType name="HvacOverrunStatusEnumType">
15321     <xs:restriction base="xs:string">
15322         <xs:enumeration value="active"/>
15323         <xs:enumeration value="running"/>
15324         <xs:enumeration value="finished"/>
15325         <xs:enumeration value="inactive"/>
15326     </xs:restriction>
15327 </xs:simpleType>
15328 <xs:complexType name="HvacSystemFunctionDataType">
15329     <xs:sequence>
15330         <xs:element minOccurs="0" name="systemFunctionId"
15331 type="ns_p:HvacSystemFunctionIdType"/>
15332         <xs:element minOccurs="0" name="currentOperationModeId"
15333 type="ns_p:HvacOperationModeIdType"/>
15334         <xs:element minOccurs="0" name="isOperationModeIdChangeable" type="xs:boolean"/>
15335         <xs:element minOccurs="0" name="currentSetpointId" type="ns_p:SetpointIdType"/>
15336         <xs:element minOccurs="0" name="isSetpointIdChangeable" type="xs:boolean"/>
15337         <xs:element minOccurs="0" name="isOverrunActive" type="xs:boolean"/>
15338     </xs:sequence>
15339 </xs:complexType>
15340 <xs:element name="hvacSystemFunctionData" type="ns_p:HvacSystemFunctionDataType"/>
15341 <xs:complexType name="HvacSystemFunctionDataElementsType">
15342     <xs:sequence>
15343         <xs:element minOccurs="0" name="systemFunctionId" type="ns_p:ElementTagType"/>
15344         <xs:element minOccurs="0" name="currentOperationModeId"
15345 type="ns_p:ElementTagType"/>
15346         <xs:element minOccurs="0" name="isOperationModeIdChangeable"
15347 type="ns_p:ElementTagType"/>
15348         <xs:element minOccurs="0" name="currentSetpointId" type="ns_p:ElementTagType"/>
15349         <xs:element minOccurs="0" name="isSetpointIdChangeable"
15350 type="ns_p:ElementTagType"/>
15351         <xs:element minOccurs="0" name="isOverrunActive" type="ns_p:ElementTagType"/>
15352     </xs:sequence>
15353 </xs:complexType>
15354 <xs:element name="hvacSystemFunctionDataElements"
15355 type="ns_p:HvacSystemFunctionDataElementsType"/>
15356 <xs:complexType name="HvacSystemFunctionListDataType">
15357     <xs:sequence>
15358         <xs:element maxOccurs="unbounded" minOccurs="0"
15359 ref="ns_p:hvacSystemFunctionData"/>
15360     </xs:sequence>
15361 </xs:complexType>
15362 <xs:element name="hvacSystemFunctionListData" type="ns_p:HvacSystemFunctionListDataType"/>
15363 <xs:complexType name="HvacSystemFunctionListDataSelectorsType">
15364     <xs:sequence>
15365         <xs:element minOccurs="0" name="systemFunctionId"
15366 type="ns_p:HvacSystemFunctionIdType"/>
15367     </xs:sequence>
15368 </xs:complexType>

```

```

15369     <xs:element name="hvacSystemFunctionListDataSelectors"
15370 type="ns_p:HvacSystemFunctionListDataSelectorsType"/>
15371     <xs:complexType name="HvacSystemFunctionOperationModeRelationDataType">
15372       <xs:sequence>
15373         <xs:element minOccurs="0" name="systemFunctionId"
15374 type="ns_p:HvacSystemFunctionIdType"/>
15375         <xs:element minOccurs="0" name="operationModeId"
15376 type="ns_p:HvacOperationModeIdType" maxOccurs="unbounded"/>
15377       </xs:sequence>
15378     </xs:complexType>
15379     <xs:element name="hvacSystemFunctionOperationModeRelationData"
15380 type="ns_p:HvacSystemFunctionOperationModeRelationDataType"/>
15381     <xs:complexType name="HvacSystemFunctionOperationModeRelationDataElementsType">
15382       <xs:sequence>
15383         <xs:element minOccurs="0" name="systemFunctionId" type="ns_p:ElementTagType"/>
15384         <xs:element minOccurs="0" name="operationModeId" type="ns_p:ElementTagType"/>
15385       </xs:sequence>
15386     </xs:complexType>
15387     <xs:element name="hvacSystemFunctionOperationModeRelationDataElements"
15388 type="ns_p:HvacSystemFunctionOperationModeRelationDataElementsType"/>
15389     <xs:complexType name="HvacSystemFunctionOperationModeRelationListDataType">
15390       <xs:sequence>
15391         <xs:element maxOccurs="unbounded" minOccurs="0"
15392 ref="ns_p:hvacSystemFunctionOperationModeRelationData"/>
15393       </xs:sequence>
15394     </xs:complexType>
15395     <xs:element name="hvacSystemFunctionOperationModeRelationListData"
15396 type="ns_p:HvacSystemFunctionOperationModeRelationListDataType"/>
15397     <xs:complexType name="HvacSystemFunctionOperationModeRelationListDataSelectorsType">
15398       <xs:sequence>
15399         <xs:element minOccurs="0" name="systemFunctionId"
15400 type="ns_p:HvacSystemFunctionIdType"/>
15401       </xs:sequence>
15402     </xs:complexType>
15403     <xs:element name="hvacSystemFunctionOperationModeRelationListDataSelectors"
15404 type="ns_p:HvacSystemFunctionOperationModeRelationListDataSelectorsType"/>
15405     <xs:complexType name="HvacSystemFunctionSetpointRelationDataType">
15406       <xs:sequence>
15407         <xs:element minOccurs="0" name="systemFunctionId"
15408 type="ns_p:HvacSystemFunctionIdType"/>
15409         <xs:element minOccurs="0" name="operationModeId"
15410 type="ns_p:HvacOperationModeIdType"/>
15411         <xs:element minOccurs="0" name="setpointId" type="ns_p:SetpointIdType"
15412 maxOccurs="unbounded"/>
15413       </xs:sequence>
15414     </xs:complexType>
15415     <xs:element name="hvacSystemFunctionSetpointRelationData"
15416 type="ns_p:HvacSystemFunctionSetpointRelationDataType"/>
15417     <xs:complexType name="HvacSystemFunctionSetpointRelationDataElementsType">
15418       <xs:sequence>
15419         <xs:element minOccurs="0" name="systemFunctionId" type="ns_p:ElementTagType"/>
15420         <xs:element minOccurs="0" name="operationModeId" type="ns_p:ElementTagType"/>
15421         <xs:element minOccurs="0" name="setpointId" type="ns_p:ElementTagType"/>
15422       </xs:sequence>
15423     </xs:complexType>
15424     <xs:element name="hvacSystemFunctionSetpointRelationDataElements"
15425 type="ns_p:HvacSystemFunctionSetpointRelationDataElementsType"/>
15426     <xs:complexType name="HvacSystemFunctionSetpointRelationListDataType">
15427       <xs:sequence>
15428         <xs:element maxOccurs="unbounded" minOccurs="0"
15429 ref="ns_p:hvacSystemFunctionSetpointRelationData"/>
15430       </xs:sequence>
15431     </xs:complexType>
15432     <xs:element name="hvacSystemFunctionSetpointRelationListData"
15433 type="ns_p:HvacSystemFunctionSetpointRelationListDataType"/>
15434     <xs:complexType name="HvacSystemFunctionSetpointRelationListDataSelectorsType">
15435       <xs:sequence>
15436         <xs:element minOccurs="0" name="systemFunctionId"
15437 type="ns_p:HvacSystemFunctionIdType"/>
15438         <xs:element minOccurs="0" name="operationModeId"
15439 type="ns_p:HvacOperationModeIdType"/>
15440       </xs:sequence>
15441     </xs:complexType>
15442     <xs:element name="hvacSystemFunctionSetpointRelationListDataSelectors"
15443 type="ns_p:HvacSystemFunctionSetpointRelationListDataSelectorsType"/>
15444     <xs:complexType name="HvacSystemFunctionPowerSequenceRelationDataType">
15445       <xs:sequence>

```

```

15446         <xs:element minOccurs="0" name="systemFunctionId"
15447 type="ns_p:HvacSystemFunctionIdType"/>
15448         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"
15449 maxOccurs="unbounded"/>
15450     </xs:sequence>
15451 </xs:complexType>
15452 <xs:element name="hvacSystemFunctionPowerSequenceRelationData"
15453 type="ns_p:HvacSystemFunctionPowerSequenceRelationDataType"/>
15454 <xs:complexType name="HvacSystemFunctionPowerSequenceRelationDataElementsType">
15455     <xs:sequence>
15456         <xs:element minOccurs="0" name="systemFunctionId" type="ns_p:ElementTagType"/>
15457         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
15458     </xs:sequence>
15459 </xs:complexType>
15460 <xs:element name="hvacSystemFunctionPowerSequenceRelationDataElements"
15461 type="ns_p:HvacSystemFunctionPowerSequenceRelationDataElementsType"/>
15462 <xs:complexType name="HvacSystemFunctionPowerSequenceRelationListDataType">
15463     <xs:sequence>
15464         <xs:element maxOccurs="unbounded" minOccurs="0"
15465 ref="ns_p:hvacSystemFunctionPowerSequenceRelationData"/>
15466     </xs:sequence>
15467 </xs:complexType>
15468 <xs:element name="hvacSystemFunctionPowerSequenceRelationListData"
15469 type="ns_p:HvacSystemFunctionPowerSequenceRelationListDataType"/>
15470 <xs:complexType name="HvacSystemFunctionPowerSequenceRelationListDataSelectorsType">
15471     <xs:sequence>
15472         <xs:element minOccurs="0" name="systemFunctionId"
15473 type="ns_p:HvacSystemFunctionIdType"/>
15474     </xs:sequence>
15475 </xs:complexType>
15476 <xs:element name="hvacSystemFunctionPowerSequenceRelationListDataSelectors"
15477 type="ns_p:HvacSystemFunctionPowerSequenceRelationListDataSelectorsType"/>
15478 <xs:complexType name="HvacSystemFunctionDescriptionDataType">
15479     <xs:sequence>
15480         <xs:element minOccurs="0" name="systemFunctionId"
15481 type="ns_p:HvacSystemFunctionIdType"/>
15482         <xs:element minOccurs="0" name="systemFunctionType"
15483 type="ns_p:HvacSystemFunctionTypeType"/>
15484         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
15485         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
15486     </xs:sequence>
15487 </xs:complexType>
15488 <xs:element name="hvacSystemFunctionDescriptionData"
15489 type="ns_p:HvacSystemFunctionDescriptionDataType"/>
15490 <xs:complexType name="HvacSystemFunctionDescriptionDataElementsType">
15491     <xs:sequence>
15492         <xs:element minOccurs="0" name="systemFunctionId" type="ns_p:ElementTagType"/>
15493         <xs:element minOccurs="0" name="systemFunctionType" type="ns_p:ElementTagType"/>
15494         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
15495         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
15496     </xs:sequence>
15497 </xs:complexType>
15498 <xs:element name="hvacSystemFunctionDescriptionDataElements"
15499 type="ns_p:HvacSystemFunctionDescriptionDataElementsType"/>
15500 <xs:complexType name="HvacSystemFunctionDescriptionListDataType">
15501     <xs:sequence>
15502         <xs:element maxOccurs="unbounded" minOccurs="0"
15503 ref="ns_p:hvacSystemFunctionDescriptionData"/>
15504     </xs:sequence>
15505 </xs:complexType>
15506 <xs:element name="hvacSystemFunctionDescriptionListData"
15507 type="ns_p:HvacSystemFunctionDescriptionListDataType"/>
15508 <xs:complexType name="HvacSystemFunctionDescriptionListDataSelectorsType">
15509     <xs:sequence>
15510         <xs:element minOccurs="0" name="systemFunctionId"
15511 type="ns_p:HvacSystemFunctionIdType"/>
15512     </xs:sequence>
15513 </xs:complexType>
15514 <xs:element name="hvacSystemFunctionDescriptionListDataSelectors"
15515 type="ns_p:HvacSystemFunctionDescriptionListDataSelectorsType"/>
15516 <xs:complexType name="HvacOperationModeDescriptionDataType">
15517     <xs:sequence>
15518         <xs:element minOccurs="0" name="operationModeId"
15519 type="ns_p:HvacOperationModeIdType"/>
15520         <xs:element minOccurs="0" name="operationModeType"
15521 type="ns_p:HvacOperationModeTypeType"/>
15522         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
15523         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>

```

```

15524         </xs:sequence>
15525     </xs:complexType>
15526     <xs:element name="hvacOperationModeDescriptionData"
15527 type="ns_p:HvacOperationModeDescriptionDataType"/>
15528     <xs:complexType name="HvacOperationModeDescriptionDataElementsType">
15529         <xs:sequence>
15530             <xs:element minOccurs="0" name="operationModeId" type="ns_p:ElementTagType"/>
15531             <xs:element minOccurs="0" name="operationModeType" type="ns_p:ElementTagType"/>
15532             <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
15533             <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
15534         </xs:sequence>
15535     </xs:complexType>
15536     <xs:element name="hvacOperationModeDescriptionDataElements"
15537 type="ns_p:HvacOperationModeDescriptionDataElementsType"/>
15538     <xs:complexType name="HvacOperationModeDescriptionListDataType">
15539         <xs:sequence>
15540             <xs:element maxOccurs="unbounded" minOccurs="0"
15541 ref="ns_p:hvacOperationModeDescriptionData"/>
15542         </xs:sequence>
15543     </xs:complexType>
15544     <xs:element name="hvacOperationModeDescriptionListData"
15545 type="ns_p:HvacOperationModeDescriptionListDataType"/>
15546     <xs:complexType name="HvacOperationModeDescriptionListDataSelectorsType">
15547         <xs:sequence>
15548             <xs:element minOccurs="0" name="operationModeId"
15549 type="ns_p:HvacOperationModeIdType"/>
15550         </xs:sequence>
15551     </xs:complexType>
15552     <xs:element name="hvacOperationModeDescriptionListDataSelectors"
15553 type="ns_p:HvacOperationModeDescriptionListDataSelectorsType"/>
15554     <xs:complexType name="HvacOverrunDataType">
15555         <xs:sequence>
15556             <xs:element name="overrunId" minOccurs="0" type="ns_p:HvacOverrunIdType"/>
15557             <xs:element name="overrunStatus" minOccurs="0" type="ns_p:HvacOverrunStatusType"/>
15558             <xs:element minOccurs="0" name="timeTableId" type="ns_p:TimeTableIdType"/>
15559             <xs:element minOccurs="0" name="isOverrunStatusChangeable" type="xs:boolean"/>
15560         </xs:sequence>
15561     </xs:complexType>
15562     <xs:element name="hvacOverrunData" type="ns_p:HvacOverrunDataType"/>
15563     <xs:complexType name="HvacOverrunDataElementsType">
15564         <xs:sequence>
15565             <xs:element name="overrunId" minOccurs="0" type="ns_p:ElementTagType"/>
15566             <xs:element name="overrunStatus" minOccurs="0" type="ns_p:ElementTagType"/>
15567             <xs:element minOccurs="0" name="timeTableId" type="ns_p:ElementTagType"/>
15568             <xs:element minOccurs="0" name="isOverrunStatusChangeable"
15569 type="ns_p:ElementTagType"/>
15570         </xs:sequence>
15571     </xs:complexType>
15572     <xs:element name="hvacOverrunDataElements" type="ns_p:HvacOverrunDataElementsType"/>
15573     <xs:complexType name="HvacOverrunListDataType">
15574         <xs:sequence>
15575             <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:hvacOverrunData"/>
15576         </xs:sequence>
15577     </xs:complexType>
15578     <xs:element name="hvacOverrunListData" type="ns_p:HvacOverrunListDataType"/>
15579     <xs:complexType name="HvacOverrunListDataSelectorsType">
15580         <xs:sequence>
15581             <xs:element name="overrunId" minOccurs="0" type="ns_p:HvacOverrunIdType"/>
15582         </xs:sequence>
15583     </xs:complexType>
15584     <xs:element name="hvacOverrunListDataSelectors"
15585 type="ns_p:HvacOverrunListDataSelectorsType"/>
15586     <xs:complexType name="HvacOverrunDescriptionDataType">
15587         <xs:sequence>
15588             <xs:element name="overrunId" minOccurs="0" type="ns_p:HvacOverrunIdType"/>
15589             <xs:element minOccurs="0" name="overrunType" type="ns_p:HvacOverrunTypeType"/>
15590             <xs:element minOccurs="0" name="affectedSystemFunctionId"
15591 type="ns_p:HvacSystemFunctionIdType" maxOccurs="unbounded"/>
15592             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
15593             <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
15594         </xs:sequence>
15595     </xs:complexType>
15596     <xs:element name="hvacOverrunDescriptionData" type="ns_p:HvacOverrunDescriptionDataType"/>
15597     <xs:complexType name="HvacOverrunDescriptionDataElementsType">
15598         <xs:sequence>
15599             <xs:element name="overrunId" minOccurs="0" type="ns_p:ElementTagType"/>
15600             <xs:element minOccurs="0" name="overrunType" type="ns_p:ElementTagType"/>

```

```

15601         <xs:element minOccurs="0" name="affectedSystemFunctionId"
15602 type="ns_p:ElementTagType"/>
15603         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
15604         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
15605     </xs:sequence>
15606 </xs:complexType>
15607 <xs:element name="hvacOverrunDescriptionDataElements"
15608 type="ns_p:HvacOverrunDescriptionDataElementsType"/>
15609 <xs:complexType name="HvacOverrunDescriptionListDataType">
15610     <xs:sequence>
15611         <xs:element maxOccurs="unbounded" minOccurs="0"
15612 ref="ns_p:hvacOverrunDescriptionData"/>
15613     </xs:sequence>
15614 </xs:complexType>
15615 <xs:element name="hvacOverrunDescriptionListData"
15616 type="ns_p:HvacOverrunDescriptionListDataType"/>
15617 <xs:complexType name="HvacOverrunDescriptionListDataSelectorsType">
15618     <xs:sequence>
15619         <xs:element name="overrunId" minOccurs="0" type="ns_p:HvacOverrunIdType"/>
15620     </xs:sequence>
15621 </xs:complexType>
15622 <xs:element name="hvacOverrunDescriptionListDataSelectors"
15623 type="ns_p:HvacOverrunDescriptionListDataSelectorsType"/>
15624 </xs:schema>
15625
15626

```

A.2.13 Identification

File name: EEBus_SPINE_TS_Identification.xsd

```

15630
15631 <?xml version="1.0" encoding="UTF-8"?>
15632 <!--
15633     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
15634     Version 1.1.1
15635     2018-12-21
15636     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
15637     Source: https://www.eebus.org/en/specifications/
15638 -->
15639 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
15640 xmlns:xs="http://www.w3.org/2001/XMLSchema"
15641 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
15642 blockDefault="#all" elementFormDefault="qualified">
15643     <xs:annotation>
15644         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
15645 e.V. All rights reserved.</xs:documentation>
15646     </xs:annotation>
15647     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
15648     <xs:simpleType name="IdentificationIdType">
15649         <xs:restriction base="xs:unsignedInt"/>
15650     </xs:simpleType>
15651     <xs:simpleType name="IdentificationTypeType">
15652         <xs:union memberTypes="ns_p:IdentificationTypeEnumType ns_p:EnumExtendType"/>
15653     </xs:simpleType>
15654     <xs:simpleType name="IdentificationTypeEnumType">
15655         <xs:restriction base="xs:string">
15656             <xs:enumeration value="eui48"/>
15657             <xs:enumeration value="eui64"/>
15658             <xs:enumeration value="userRfidTag"/>
15659         </xs:restriction>
15660     </xs:simpleType>
15661     <xs:simpleType name="IdentificationValueType">
15662         <xs:restriction base="xs:string"/>
15663     </xs:simpleType>
15664     <xs:complexType name="IdentificationDataType">
15665         <xs:sequence>
15666             <xs:element minOccurs="0" name="identificationId"
15667 type="ns_p:IdentificationIdType"/>
15668             <xs:element minOccurs="0" name="identificationType"
15669 type="ns_p:IdentificationTypeType"/>
15670             <xs:element minOccurs="0" name="identificationValue"
15671 type="ns_p:IdentificationValueType"/>
15672             <xs:element minOccurs="0" name="authorized" type="xs:boolean"/>
15673         </xs:sequence>

```

```

15674 </xs:complexType>
15675 <xs:element name="identificationData" type="ns_p:IdentificationDataType"/>
15676 <xs:complexType name="IdentificationDataElementsType">
15677   <xs:sequence>
15678     <xs:element minOccurs="0" name="identificationId" type="ns_p:ElementTagType"/>
15679     <xs:element minOccurs="0" name="identificationType" type="ns_p:ElementTagType"/>
15680     <xs:element minOccurs="0" name="identificationValue" type="ns_p:ElementTagType"/>
15681     <xs:element minOccurs="0" name="authorized" type="ns_p:ElementTagType"/>
15682   </xs:sequence>
15683 </xs:complexType>
15684 <xs:element name="identificationDataElements" type="ns_p:IdentificationDataElementsType"/>
15685 <xs:complexType name="IdentificationListDataType">
15686   <xs:sequence>
15687     <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:identificationData"/>
15688   </xs:sequence>
15689 </xs:complexType>
15690 <xs:element name="identificationListData" type="ns_p:IdentificationListDataType"/>
15691 <xs:complexType name="IdentificationListDataSelectorsType">
15692   <xs:sequence>
15693     <xs:element minOccurs="0" name="identificationId"
15694 type="ns_p:IdentificationIdType"/>
15695     <xs:element minOccurs="0" name="identificationType"
15696 type="ns_p:IdentificationTypeType"/>
15697   </xs:sequence>
15698 </xs:complexType>
15699 <xs:element name="identificationListDataSelectors"
15700 type="ns_p:IdentificationListDataSelectorsType"/>
15701 </xs:schema>
15702
15703

```

15704

15705 A.2.14 LoadControl

15706 File name: EEBus_SPINE_TS_LoadControl.xsd

```

15707 <?xml version="1.0" encoding="UTF-8"?>
15708 <!--
15709   Smart Premises Interoperable Neutral-Message Exchange (SPINE)
15710   Version 1.1.1
15711   2018-12-21
15712   Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
15713   Source: https://www.eebus.org/en/specifications/
15714 -->
15715 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
15716   xmlns:xs="http://www.w3.org/2001/XMLSchema"
15717   targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
15718   blockDefault="#all" elementFormDefault="qualified">
15719   <xs:annotation>
15720     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
15721 e.V. All rights reserved.</xs:documentation>
15722   </xs:annotation>
15723   <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
15724   <xs:include schemaLocation="EEBus_SPINE_TS_Measurement.xsd"/>
15725   <xs:simpleType name="LoadControlEventsIdType">
15726     <xs:restriction base="xs:unsignedInt"/>
15727   </xs:simpleType>
15728   <xs:simpleType name="LoadControlEventActionType">
15729     <xs:union memberTypes="ns_p:LoadControlEventActionEnumType ns_p:EnumExtendType"/>
15730   </xs:simpleType>
15731   <xs:simpleType name="LoadControlEventActionEnumType">
15732     <xs:restriction base="xs:string">
15733       <xs:enumeration value="pause"/>
15734       <xs:enumeration value="resume"/>
15735       <xs:enumeration value="reduce"/>
15736       <xs:enumeration value="increase"/>
15737       <xs:enumeration value="emergency"/>
15738       <xs:enumeration value="normal"/>
15739     </xs:restriction>
15740   </xs:simpleType>
15741   <xs:simpleType name="LoadControlEventStateType">
15742     <xs:union memberTypes="ns_p:LoadControlEventStateEnumType ns_p:EnumExtendType"/>
15743   </xs:simpleType>
15744   <xs:simpleType name="LoadControlEventStateEnumType">
15745     <xs:restriction base="xs:string">
15746

```



```

15747         <xs:enumeration value="eventAccepted"/>
15748         <xs:enumeration value="eventStarted"/>
15749         <xs:enumeration value="eventStopped"/>
15750         <xs:enumeration value="eventRejected"/>
15751         <xs:enumeration value="eventCancelled"/>
15752         <xs:enumeration value="eventError"/>
15753     </xs:restriction>
15754 </xs:simpleType>
15755 <xs:simpleType name="LoadControlLimitIdType">
15756     <xs:restriction base="xs:unsignedInt"/>
15757 </xs:simpleType>
15758 <xs:simpleType name="LoadControlLimitTypeType">
15759     <xs:union memberTypes="ns_p:LoadControlLimitTypeEnumType ns_p:EnumExtendType"/>
15760 </xs:simpleType>
15761 <xs:simpleType name="LoadControlLimitTypeEnumType">
15762     <xs:restriction base="xs:string">
15763         <xs:enumeration value="minValueLimit"/>
15764         <xs:enumeration value="maxValueLimit"/>
15765     </xs:restriction>
15766 </xs:simpleType>
15767 <xs:simpleType name="LoadControlCategoryType">
15768     <xs:union memberTypes="ns_p:LoadControlCategoryTypeEnumType ns_p:EnumExtendType"/>
15769 </xs:simpleType>
15770 <xs:simpleType name="LoadControlCategoryTypeEnumType">
15771     <xs:restriction base="xs:string">
15772         <xs:enumeration value="obligation"/>
15773         <xs:enumeration value="recommendation"/>
15774         <xs:enumeration value="optimization"/>
15775     </xs:restriction>
15776 </xs:simpleType>
15777 <xs:complexType name="LoadControlNodeDataType">
15778     <xs:sequence>
15779         <xs:element minOccurs="0" name="isNodeRemoteControllable" type="xs:boolean"/>
15780     </xs:sequence>
15781 </xs:complexType>
15782 <xs:element name="loadControlNodeData" type="ns_p:LoadControlNodeDataType"/>
15783 <xs:complexType name="LoadControlNodeDataElementsType">
15784     <xs:sequence>
15785         <xs:element minOccurs="0" name="isNodeRemoteControllable"
15786 type="ns_p:ElementTagType"/>
15787     </xs:sequence>
15788 </xs:complexType>
15789 <xs:element name="loadControlNodeDataElements"
15790 type="ns_p:LoadControlNodeDataElementsType"/>
15791 <xs:complexType name="LoadControlEventDataType">
15792     <xs:sequence>
15793         <xs:element minOccurs="0" name="timestamp"
15794 type="ns_p:AbsoluteOrRelativeTimeType"/>
15795         <xs:element minOccurs="0" name="eventId" type="ns_p:LoadControlEventIdType"/>
15796         <xs:element minOccurs="0" name="eventActionConsume"
15797 type="ns_p:LoadControlEventActionType"/>
15798         <xs:element minOccurs="0" name="eventActionProduce"
15799 type="ns_p:LoadControlEventActionType"/>
15800         <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodType"/>
15801     </xs:sequence>
15802 </xs:complexType>
15803 <xs:element name="loadControlEventData" type="ns_p:LoadControlEventDataTypes"/>
15804 <xs:complexType name="LoadControlEventDataElementsType">
15805     <xs:sequence>
15806         <xs:element minOccurs="0" name="timestamp" type="ns_p:ElementTagType"/>
15807         <xs:element minOccurs="0" name="eventId" type="ns_p:ElementTagType"/>
15808         <xs:element minOccurs="0" name="eventActionConsume" type="ns_p:ElementTagType"/>
15809         <xs:element minOccurs="0" name="eventActionProduce" type="ns_p:ElementTagType"/>
15810         <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodElementsType"/>
15811     </xs:sequence>
15812 </xs:complexType>
15813 <xs:element name="loadControlEventDataElements"
15814 type="ns_p:LoadControlEventDataElementsType"/>
15815 <xs:complexType name="LoadControlEventListDataType">
15816     <xs:sequence>
15817         <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:loadControlEventData"/>
15818     </xs:sequence>
15819 </xs:complexType>
15820 <xs:element name="loadControlEventListData" type="ns_p:LoadControlEventListDataType"/>
15821 <xs:complexType name="LoadControlEventListDataSelectorsType">
15822     <xs:sequence>
15823         <xs:element minOccurs="0" name="timestampInterval"
15824 type="ns_p:TimestampIntervalType"/>

```

```
15825         <xs:element minOccurs="0" name="eventId" type="ns_p:LoadControlEventIdType"/>
15826     </xs:sequence>
15827 </xs:complexType>
15828     <xs:element name="loadControlEventListDataSelectors"
15829 type="ns_p:LoadControlEventListDataSelectorsType"/>
15830     <xs:complexType name="LoadControlStateDataType">
15831         <xs:sequence>
15832             <xs:element minOccurs="0" name="timestamp"
15833 type="ns_p:AbsoluteOrRelativeTimeType"/>
15834             <xs:element minOccurs="0" name="eventId" type="ns_p:LoadControlEventIdType"/>
15835             <xs:element minOccurs="0" name="eventStateConsume"
15836 type="ns_p:LoadControlEventStateType"/>
15837             <xs:element minOccurs="0" name="appliedEventActionConsume"
15838 type="ns_p:LoadControlEventActionType"/>
15839             <xs:element minOccurs="0" name="eventStateProduce"
15840 type="ns_p:LoadControlEventStateType"/>
15841             <xs:element minOccurs="0" name="appliedEventActionProduce"
15842 type="ns_p:LoadControlEventActionType"/>
15843         </xs:sequence>
15844     </xs:complexType>
15845     <xs:element name="loadControlStateData" type="ns_p:LoadControlStateDataType"/>
15846     <xs:complexType name="LoadControlStateDataElementsType">
15847         <xs:sequence>
15848             <xs:element minOccurs="0" name="timestamp" type="ns_p:ElementTagType"/>
15849             <xs:element minOccurs="0" name="eventId" type="ns_p:ElementTagType"/>
15850             <xs:element minOccurs="0" name="eventStateConsume" type="ns_p:ElementTagType"/>
15851             <xs:element minOccurs="0" name="appliedEventActionConsume"
15852 type="ns_p:ElementTagType"/>
15853             <xs:element minOccurs="0" name="eventStateProduce" type="ns_p:ElementTagType"/>
15854             <xs:element minOccurs="0" name="appliedEventActionProduce"
15855 type="ns_p:ElementTagType"/>
15856         </xs:sequence>
15857     </xs:complexType>
15858     <xs:element name="loadControlStateDataElements"
15859 type="ns_p:LoadControlStateDataElementsType"/>
15860     <xs:complexType name="LoadControlStateListDataType">
15861         <xs:sequence>
15862             <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:loadControlStateData"/>
15863         </xs:sequence>
15864     </xs:complexType>
15865     <xs:element name="loadControlStateListData" type="ns_p:LoadControlStateListDataType"/>
15866     <xs:complexType name="LoadControlStateListDataSelectorsType">
15867         <xs:sequence>
15868             <xs:element minOccurs="0" name="timestampInterval"
15869 type="ns_p:TimestampIntervalType"/>
15870             <xs:element minOccurs="0" name="eventId" type="ns_p:LoadControlEventIdType"/>
15871         </xs:sequence>
15872     </xs:complexType>
15873     <xs:element name="loadControlStateListDataSelectors"
15874 type="ns_p:LoadControlStateListDataSelectorsType"/>
15875     <xs:complexType name="LoadControlLimitDataType">
15876         <xs:sequence>
15877             <xs:element minOccurs="0" name="limitId" type="ns_p:LoadControlLimitIdType"/>
15878             <xs:element name="isLimitChangeable" minOccurs="0" type="xs:boolean"/>
15879             <xs:element name="isLimitActive" minOccurs="0" type="xs:boolean"/>
15880             <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodType"/>
15881             <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberType"/>
15882         </xs:sequence>
15883     </xs:complexType>
15884     <xs:element name="loadControlLimitData" type="ns_p:LoadControlLimitDataType"/>
15885     <xs:complexType name="LoadControlLimitDataElementsType">
15886         <xs:sequence>
15887             <xs:element minOccurs="0" name="limitId" type="ns_p:ElementTagType"/>
15888             <xs:element name="isLimitChangeable" minOccurs="0" type="ns_p:ElementTagType"/>
15889             <xs:element name="isLimitActive" minOccurs="0" type="ns_p:ElementTagType"/>
15890             <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodElementsType"/>
15891             <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberElementsType"/>
15892         </xs:sequence>
15893     </xs:complexType>
15894     <xs:element name="loadControlLimitDataElements"
15895 type="ns_p:LoadControlLimitDataElementsType"/>
15896     <xs:complexType name="LoadControlLimitListDataType">
15897         <xs:sequence>
15898             <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:loadControlLimitData"/>
15899         </xs:sequence>
15900     </xs:complexType>
15901     <xs:element name="loadControlLimitListData" type="ns_p:LoadControlLimitListDataType"/>
15902     <xs:complexType name="LoadControlLimitListDataSelectorsType">
```

```
15903     <xs:sequence>
15904       <xs:element minOccurs="0" name="limitId" type="ns_p:LoadControlLimitIdType"/>
15905     </xs:sequence>
15906   </xs:complexType>
15907   <xs:element name="loadControlLimitListDataSelectors"
15908 type="ns_p:LoadControlLimitListDataSelectorsType"/>
15909   <xs:complexType name="LoadControlLimitConstraintsDataType">
15910     <xs:sequence>
15911       <xs:element minOccurs="0" name="limitId" type="ns_p:LoadControlLimitIdType"/>
15912       <xs:element minOccurs="0" name="valueRangeMin" type="ns_p:ScaledNumberType"/>
15913       <xs:element minOccurs="0" name="valueRangeMax" type="ns_p:ScaledNumberType"/>
15914       <xs:element minOccurs="0" name="valueStepSize" type="ns_p:ScaledNumberType"/>
15915     </xs:sequence>
15916   </xs:complexType>
15917   <xs:element name="loadControlLimitConstraintsData"
15918 type="ns_p:LoadControlLimitConstraintsDataType"/>
15919   <xs:complexType name="LoadControlLimitConstraintsDataElementsType">
15920     <xs:sequence>
15921       <xs:element minOccurs="0" name="limitId" type="ns_p:ElementTagType"/>
15922       <xs:element minOccurs="0" name="valueRangeMin"
15923 type="ns_p:ScaledNumberElementsType"/>
15924       <xs:element minOccurs="0" name="valueRangeMax"
15925 type="ns_p:ScaledNumberElementsType"/>
15926       <xs:element minOccurs="0" name="valueStepSize"
15927 type="ns_p:ScaledNumberElementsType"/>
15928     </xs:sequence>
15929   </xs:complexType>
15930   <xs:element name="loadControlLimitConstraintsDataElements"
15931 type="ns_p:LoadControlLimitConstraintsDataElementsType"/>
15932   <xs:complexType name="LoadControlLimitConstraintsListDataType">
15933     <xs:sequence>
15934       <xs:element maxOccurs="unbounded" minOccurs="0"
15935 ref="ns_p:loadControlLimitConstraintsData"/>
15936     </xs:sequence>
15937   </xs:complexType>
15938   <xs:element name="loadControlLimitConstraintsListData"
15939 type="ns_p:LoadControlLimitConstraintsListDataType"/>
15940   <xs:complexType name="LoadControlLimitConstraintsListDataSelectorsType">
15941     <xs:sequence>
15942       <xs:element minOccurs="0" name="limitId" type="ns_p:LoadControlLimitIdType"/>
15943     </xs:sequence>
15944   </xs:complexType>
15945   <xs:element name="loadControlLimitConstraintsListDataSelectors"
15946 type="ns_p:LoadControlLimitConstraintsListDataSelectorsType"/>
15947   <xs:complexType name="LoadControlLimitDescriptionDataType">
15948     <xs:sequence>
15949       <xs:element minOccurs="0" name="limitId" type="ns_p:LoadControlLimitIdType"/>
15950       <xs:element minOccurs="0" name="limitType" type="ns_p:LoadControlLimitTypeType"/>
15951       <xs:element minOccurs="0" name="limitCategory"
15952 type="ns_p:LoadControlCategoryType"/>
15953       <xs:element minOccurs="0" name="limitDirection" type="ns_p:EnergyDirectionType"/>
15954       <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
15955       <xs:element minOccurs="0" name="unit" type="ns_p:UnitOfMeasurementType"/>
15956       <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
15957       <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
15958       <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
15959     </xs:sequence>
15960   </xs:complexType>
15961   <xs:element name="loadControlLimitDescriptionData"
15962 type="ns_p:LoadControlLimitDescriptionDataType"/>
15963   <xs:complexType name="LoadControlLimitDescriptionDataElementsType">
15964     <xs:sequence>
15965       <xs:element minOccurs="0" name="limitId" type="ns_p:ElementTagType"/>
15966       <xs:element minOccurs="0" name="limitType" type="ns_p:ElementTagType"/>
15967       <xs:element minOccurs="0" name="limitCategory" type="ns_p:ElementTagType"/>
15968       <xs:element minOccurs="0" name="limitDirection" type="ns_p:ElementTagType"/>
15969       <xs:element minOccurs="0" name="measurementId" type="ns_p:ElementTagType"/>
15970       <xs:element minOccurs="0" name="unit" type="ns_p:ElementTagType"/>
15971       <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
15972       <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
15973       <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
15974     </xs:sequence>
15975   </xs:complexType>
15976   <xs:element name="loadControlLimitDescriptionDataElements"
15977 type="ns_p:LoadControlLimitDescriptionDataElementsType"/>
15978   <xs:complexType name="LoadControlLimitDescriptionListDataType">
15979     <xs:sequence>
```

```

15980         <xs:element maxOccurs="unbounded" minOccurs="0"
15981 ref="ns_p:loadControlLimitDescriptionData"/>
15982     </xs:sequence>
15983 </xs:complexType>
15984 <xs:element name="loadControlLimitDescriptionListData"
15985 type="ns_p:LoadControlLimitDescriptionListDataType"/>
15986 <xs:complexType name="LoadControlLimitDescriptionListDataSelectorsType">
15987     <xs:sequence>
15988         <xs:element minOccurs="0" name="limitId" type="ns_p:LoadControlLimitIdType"/>
15989         <xs:element minOccurs="0" name="limitType" type="ns_p:LoadControlLimitTypeType"/>
15990         <xs:element minOccurs="0" name="limitDirection" type="ns_p:EnergyDirectionType"/>
15991         <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
15992         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
15993     </xs:sequence>
15994 </xs:complexType>
15995 <xs:element name="loadControlLimitDescriptionListDataSelectors"
15996 type="ns_p:LoadControlLimitDescriptionListDataSelectorsType"/>
15997 </xs:schema>
15998
15999

```

16000

16001 A.2.15 Measurement

16002 File name: EEBus_SPINE_TS_Measurement.xsd

```

16003
16004 <?xml version="1.0" encoding="UTF-8"?>
16005 <!--
16006     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
16007     Version 1.1.1
16008     2018-12-21
16009     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
16010     Source: https://www.eebus.org/en/specifications/
16011 -->
16012 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
16013 xmlns:xs="http://www.w3.org/2001/XMLSchema"
16014 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
16015 blockDefault="#all" elementFormDefault="qualified">
16016     <xs:annotation>
16017         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
16018 e.V. All rights reserved.</xs:documentation>
16019     </xs:annotation>
16020     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
16021     <xs:include schemaLocation="EEBus_SPINE_TS_Threshold.xsd"/>
16022     <xs:simpleType name="MeasurementIdType">
16023         <xs:restriction base="xs:unsignedInt"/>
16024     </xs:simpleType>
16025     <xs:simpleType name="MeasurementTypeType">
16026         <xs:union memberTypes="ns_p:MeasurementTypeEnumType ns_p:EnumExtendType"/>
16027     </xs:simpleType>
16028     <xs:simpleType name="MeasurementTypeEnumType">
16029         <xs:restriction base="xs:string">
16030             <xs:enumeration value="acceleration"/>
16031             <xs:enumeration value="angle"/>
16032             <xs:enumeration value="angularVelocity"/>
16033             <xs:enumeration value="area"/>
16034             <xs:enumeration value="atmosphericPressure"/>
16035             <xs:enumeration value="capacity"/>
16036             <xs:enumeration value="concentration"/>
16037             <xs:enumeration value="count"/>
16038             <xs:enumeration value="current"/>
16039             <xs:enumeration value="density"/>
16040             <xs:enumeration value="distance"/>
16041             <xs:enumeration value="electricField"/>
16042             <xs:enumeration value="energy"/>
16043             <xs:enumeration value="force"/>
16044             <xs:enumeration value="frequency"/>
16045             <xs:enumeration value="harmonicDistortion"/>
16046             <xs:enumeration value="heat"/>
16047             <xs:enumeration value="heatFlux"/>
16048             <xs:enumeration value="illuminance"/>
16049             <xs:enumeration value="impulse"/>
16050             <xs:enumeration value="level"/>
16051             <xs:enumeration value="magneticField"/>
16052             <xs:enumeration value="mass"/>

```

```

16053         <xs:enumeration value="massFlow"/>
16054         <xs:enumeration value="particles"/>
16055         <xs:enumeration value="percentage"/>
16056         <xs:enumeration value="power"/>
16057         <xs:enumeration value="powerFactor"/>
16058         <xs:enumeration value="pressure"/>
16059         <xs:enumeration value="radonActivity"/>
16060         <xs:enumeration value="relativeHumidity"/>
16061         <xs:enumeration value="resistance"/>
16062         <xs:enumeration value="solarRadiation"/>
16063         <xs:enumeration value="speed"/>
16064         <xs:enumeration value="temperature"/>
16065         <xs:enumeration value="time"/>
16066         <xs:enumeration value="torque"/>
16067         <xs:enumeration value="unknown"/>
16068         <xs:enumeration value="velocity"/>
16069         <xs:enumeration value="voltage"/>
16070         <xs:enumeration value="volume"/>
16071         <xs:enumeration value="volumetricFlow"/>
16072     </xs:restriction>
16073 </xs:simpleType>
16074 <xs:simpleType name="MeasurementValueTypeType">
16075     <xs:union memberTypes="ns_p:MeasurementValueTypeEnumType ns_p:EnumExtendType"/>
16076 </xs:simpleType>
16077 <xs:simpleType name="MeasurementValueTypeEnumType">
16078     <xs:restriction base="xs:string">
16079         <xs:enumeration value="value"/>
16080         <xs:enumeration value="averageValue"/>
16081         <xs:enumeration value="minValue"/>
16082         <xs:enumeration value="maxValue"/>
16083         <xs:enumeration value="standardDeviation"/>
16084     </xs:restriction>
16085 </xs:simpleType>
16086 <xs:simpleType name="MeasurementValueSourceType">
16087     <xs:union memberTypes="ns_p:MeasurementValueSourceEnumType ns_p:EnumExtendType"/>
16088 </xs:simpleType>
16089 <xs:simpleType name="MeasurementValueSourceEnumType">
16090     <xs:restriction base="xs:string">
16091         <xs:enumeration value="measuredValue"/>
16092         <xs:enumeration value="calculatedValue"/>
16093         <xs:enumeration value="empiricalValue"/>
16094     </xs:restriction>
16095 </xs:simpleType>
16096 <xs:simpleType name="MeasurementValueTendencyType">
16097     <xs:union memberTypes="ns_p:MeasurementValueTendencyEnumType ns_p:EnumExtendType"/>
16098 </xs:simpleType>
16099 <xs:simpleType name="MeasurementValueTendencyEnumType">
16100     <xs:restriction base="xs:string">
16101         <xs:enumeration value="rising"/>
16102         <xs:enumeration value="stable"/>
16103         <xs:enumeration value="falling"/>
16104     </xs:restriction>
16105 </xs:simpleType>
16106 <xs:simpleType name="MeasurementValueStateType">
16107     <xs:union memberTypes="ns_p:MeasurementValueStateEnumType ns_p:EnumExtendType"/>
16108 </xs:simpleType>
16109 <xs:simpleType name="MeasurementValueStateEnumType">
16110     <xs:restriction base="xs:string">
16111         <xs:enumeration value="normal"/>
16112         <xs:enumeration value="outOfRange"/>
16113         <xs:enumeration value="error"/>
16114     </xs:restriction>
16115 </xs:simpleType>
16116 <xs:complexType name="MeasurementDataType">
16117     <xs:sequence>
16118         <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
16119         <xs:element minOccurs="0" name="valueType" type="ns_p:MeasurementValueTypeType"/>
16120         <xs:element minOccurs="0" name="timestamp"
16121 type="ns_p:AbsoluteOrRelativeTimeType"/>
16122         <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberType"/>
16123         <xs:element minOccurs="0" name="evaluationPeriod" type="ns_p:TimePeriodType"/>
16124         <xs:element minOccurs="0" name="valueSource"
16125 type="ns_p:MeasurementValueSourceType"/>
16126         <xs:element minOccurs="0" name="valueTendency"
16127 type="ns_p:MeasurementValueTendencyType"/>
16128         <xs:element minOccurs="0" name="valueState"
16129 type="ns_p:MeasurementValueStateType"/>
16130     </xs:sequence>

```

```

16131 </xs:complexType>
16132 <xs:element name="measurementData" type="ns_p:MeasurementDataType"/>
16133 <xs:complexType name="MeasurementDataElementsType">
16134   <xs:sequence>
16135     <xs:element minOccurs="0" name="measurementId" type="ns_p:ElementTagType"/>
16136     <xs:element minOccurs="0" name="valueType" type="ns_p:ElementTagType"/>
16137     <xs:element minOccurs="0" name="timestamp" type="ns_p:ElementTagType"/>
16138     <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberElementsType"/>
16139     <xs:element minOccurs="0" name="evaluationPeriod"
16140 type="ns_p:TimePeriodElementsType"/>
16141     <xs:element minOccurs="0" name="valueSource" type="ns_p:ElementTagType"/>
16142     <xs:element minOccurs="0" name="valueTendency" type="ns_p:ElementTagType"/>
16143     <xs:element minOccurs="0" name="valueState" type="ns_p:ElementTagType"/>
16144   </xs:sequence>
16145 </xs:complexType>
16146 <xs:element name="measurementDataElements" type="ns_p:MeasurementDataElementsType"/>
16147 <xs:complexType name="MeasurementListDataType">
16148   <xs:sequence>
16149     <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:measurementData"/>
16150   </xs:sequence>
16151 </xs:complexType>
16152 <xs:element name="measurementListData" type="ns_p:MeasurementListDataType"/>
16153 <xs:complexType name="MeasurementListDataSelectorsType">
16154   <xs:sequence>
16155     <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
16156     <xs:element minOccurs="0" name="valueType" type="ns_p:MeasurementValueTypeType"/>
16157     <xs:element minOccurs="0" name="timestampInterval"
16158 type="ns_p:TimestampIntervalType"/>
16159   </xs:sequence>
16160 </xs:complexType>
16161 <xs:element name="measurementListDataSelectors"
16162 type="ns_p:MeasurementListDataSelectorsType"/>
16163 <xs:complexType name="MeasurementConstraintsDataType">
16164   <xs:sequence>
16165     <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
16166     <xs:element minOccurs="0" name="valueRangeMin" type="ns_p:ScaledNumberType"/>
16167     <xs:element minOccurs="0" name="valueRangeMax" type="ns_p:ScaledNumberType"/>
16168     <xs:element minOccurs="0" name="valueStepSize" type="ns_p:ScaledNumberType"/>
16169   </xs:sequence>
16170 </xs:complexType>
16171 <xs:element name="measurementConstraintsData" type="ns_p:MeasurementConstraintsDataType"/>
16172 <xs:complexType name="MeasurementConstraintsDataElementsType">
16173   <xs:sequence>
16174     <xs:element minOccurs="0" name="measurementId" type="ns_p:ElementTagType"/>
16175     <xs:element minOccurs="0" name="valueRangeMin"
16176 type="ns_p:ScaledNumberElementsType"/>
16177     <xs:element minOccurs="0" name="valueRangeMax"
16178 type="ns_p:ScaledNumberElementsType"/>
16179     <xs:element minOccurs="0" name="valueStepSize"
16180 type="ns_p:ScaledNumberElementsType"/>
16181   </xs:sequence>
16182 </xs:complexType>
16183 <xs:element name="measurementConstraintsDataElements"
16184 type="ns_p:MeasurementConstraintsDataElementsType"/>
16185 <xs:complexType name="MeasurementConstraintsListDataType">
16186   <xs:sequence>
16187     <xs:element maxOccurs="unbounded" minOccurs="0"
16188 ref="ns_p:measurementConstraintsData"/>
16189   </xs:sequence>
16190 </xs:complexType>
16191 <xs:element name="measurementConstraintsListData"
16192 type="ns_p:MeasurementConstraintsListDataType"/>
16193 <xs:complexType name="MeasurementConstraintsListDataSelectorsType">
16194   <xs:sequence>
16195     <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
16196   </xs:sequence>
16197 </xs:complexType>
16198 <xs:element name="measurementConstraintsListDataSelectors"
16199 type="ns_p:MeasurementConstraintsListDataSelectorsType"/>
16200 <xs:complexType name="MeasurementDescriptionDataType">
16201   <xs:sequence>
16202     <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
16203     <xs:element minOccurs="0" name="measurementType" type="ns_p:MeasurementTypeType"/>
16204     <xs:element minOccurs="0" name="commodityType" type="ns_p:CommodityTypeType"/>
16205     <xs:element minOccurs="0" name="unit" type="ns_p:UnitOfMeasurementType"/>
16206     <xs:element minOccurs="0" name="calibrationValue" type="ns_p:ScaledNumberType"/>
16207     <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
16208     <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>

```

```
16209         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
16210     </xs:sequence>
16211 </xs:complexType>
16212 <xs:element name="measurementDescriptionData" type="ns_p:MeasurementDescriptionDataType"/>
16213 <xs:complexType name="MeasurementDescriptionDataElementsType">
16214     <xs:sequence>
16215         <xs:element minOccurs="0" name="measurementId" type="ns_p:ElementTagType"/>
16216         <xs:element minOccurs="0" name="measurementType" type="ns_p:ElementTagType"/>
16217         <xs:element minOccurs="0" name="commodityType" type="ns_p:ElementTagType"/>
16218         <xs:element minOccurs="0" name="unit" type="ns_p:ElementTagType"/>
16219         <xs:element minOccurs="0" name="calibrationValue"
16220 type="ns_p:ScaledNumberElementsType"/>
16221         <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
16222         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
16223         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
16224     </xs:sequence>
16225 </xs:complexType>
16226 <xs:element name="measurementDescriptionDataElements"
16227 type="ns_p:MeasurementDescriptionDataElementsType"/>
16228 <xs:complexType name="MeasurementDescriptionListDataType">
16229     <xs:sequence>
16230         <xs:element maxOccurs="unbounded" minOccurs="0"
16231 ref="ns_p:measurementDescriptionData"/>
16232     </xs:sequence>
16233 </xs:complexType>
16234 <xs:element name="measurementDescriptionListData"
16235 type="ns_p:MeasurementDescriptionListDataType"/>
16236 <xs:complexType name="MeasurementDescriptionListDataSelectorsType">
16237     <xs:sequence>
16238         <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
16239         <xs:element minOccurs="0" name="measurementType" type="ns_p:MeasurementTypeType"/>
16240         <xs:element minOccurs="0" name="commodityType" type="ns_p:CommodityTypeType"/>
16241         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
16242     </xs:sequence>
16243 </xs:complexType>
16244 <xs:element name="measurementDescriptionListDataSelectors"
16245 type="ns_p:MeasurementDescriptionListDataSelectorsType"/>
16246 <xs:complexType name="MeasurementThresholdRelationDataType">
16247     <xs:sequence>
16248         <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
16249         <xs:element maxOccurs="unbounded" minOccurs="0" name="thresholdId"
16250 type="ns_p:ThresholdIdType"/>
16251     </xs:sequence>
16252 </xs:complexType>
16253 <xs:element name="measurementThresholdRelationData"
16254 type="ns_p:MeasurementThresholdRelationDataType"/>
16255 <xs:complexType name="MeasurementThresholdRelationDataElementsType">
16256     <xs:sequence>
16257         <xs:element minOccurs="0" name="measurementId" type="ns_p:ElementTagType"/>
16258         <xs:element minOccurs="0" name="thresholdId" type="ns_p:ElementTagType"/>
16259     </xs:sequence>
16260 </xs:complexType>
16261 <xs:element name="measurementThresholdRelationDataElements"
16262 type="ns_p:MeasurementThresholdRelationDataElementsType"/>
16263 <xs:complexType name="MeasurementThresholdRelationListDataType">
16264     <xs:sequence>
16265         <xs:element maxOccurs="unbounded" minOccurs="0"
16266 ref="ns_p:measurementThresholdRelationData"/>
16267     </xs:sequence>
16268 </xs:complexType>
16269 <xs:element name="measurementThresholdRelationListData"
16270 type="ns_p:MeasurementThresholdRelationListDataType"/>
16271 <xs:complexType name="MeasurementThresholdRelationListDataSelectorsType">
16272     <xs:sequence>
16273         <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
16274         <xs:element minOccurs="0" name="thresholdId" type="ns_p:ThresholdIdType"/>
16275     </xs:sequence>
16276 </xs:complexType>
16277 <xs:element name="measurementThresholdRelationListDataSelectors"
16278 type="ns_p:MeasurementThresholdRelationListDataSelectorsType"/>
16279 </xs:schema>
```

A.2.16 Messaging

File name: EEBus_SPINE_TS_Messaging.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    Smart Premises Interoperable Neutral-Message Exchange (SPINE)
    Version 1.1.1
    2018-12-21
    Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
    Source: https://www.eebus.org/en/specifications/
-->
<xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
  blockDefault="#all" elementFormDefault="qualified">
  <xs:annotation>
    <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
e.V. All rights reserved.</xs:documentation>
  </xs:annotation>
  <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
  <xs:simpleType name="MessagingNumberType">
    <xs:restriction base="xs:unsignedInt"/>
  </xs:simpleType>
  <xs:simpleType name="MessagingTypeType">
    <xs:union memberTypes="ns_p:MessagingTypeEnumType ns_p:EnumExtendType"/>
  </xs:simpleType>
  <xs:simpleType name="MessagingDataTextType">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
  <xs:simpleType name="MessagingTypeEnumType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="logging"/>
      <xs:enumeration value="information"/>
      <xs:enumeration value="warning"/>
      <xs:enumeration value="alarm"/>
      <xs:enumeration value="emergency"/>
      <xs:enumeration value="obsolete"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="MessagingDataType">
    <xs:sequence>
      <xs:element minOccurs="0" name="timestamp"
type="ns_p:AbsoluteOrRelativeTimeType"/>
      <xs:element minOccurs="0" name="messagingNumber" type="ns_p:MessagingNumberType"/>
      <xs:element minOccurs="0" name="type" type="ns_p:MessagingTypeType"/>
      <xs:element minOccurs="0" name="text" type="ns_p:MessagingDataTextType"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="messagingData" type="ns_p:MessagingDataType"/>
  <xs:complexType name="MessagingDataElementsType">
    <xs:sequence>
      <xs:element minOccurs="0" name="timestamp" type="ns_p:ElementTagType"/>
      <xs:element minOccurs="0" name="messagingNumber" type="ns_p:ElementTagType"/>
      <xs:element minOccurs="0" name="type" type="ns_p:ElementTagType"/>
      <xs:element minOccurs="0" name="text" type="ns_p:ElementTagType"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="messagingDataElements" type="ns_p:MessagingDataElementsType"/>
  <xs:complexType name="MessagingListDataType">
    <xs:sequence>
      <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:messagingData"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="messagingListData" type="ns_p:MessagingListDataType"/>
  <xs:complexType name="MessagingListDataSelectorsType">
    <xs:sequence>
      <xs:element minOccurs="0" name="timestampInterval"
type="ns_p:TimestampIntervalType"/>
      <xs:element minOccurs="0" name="messagingNumber" type="ns_p:MessagingNumberType"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="messagingListDataSelectors" type="ns_p:MessagingListDataSelectorsType"/>
</xs:schema>
```


16358

16359 **A.2.17 NetworkManagement**

16360 File name: EEBus_SPINE_TS_NetworkManagement.xsd

```
16361
16362 <?xml version="1.0" encoding="UTF-8"?>
16363 <!--
16364     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
16365     Version 1.1.1
16366     2018-12-21
16367     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
16368     Source: https://www.eebus.org/en/specifications/
16369 -->
16370 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
16371   xmlns:xs="http://www.w3.org/2001/XMLSchema"
16372   targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
16373     blockDefault="#all" elementFormDefault="qualified">
16374   <xs:annotation>
16375     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
16376     e.V. All rights reserved.</xs:documentation>
16377   </xs:annotation>
16378   <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
16379   <xs:simpleType name="NetworkManagementNativeSetupType">
16380     <xs:restriction base="xs:string"/>
16381   </xs:simpleType>
16382   <xs:simpleType name="NetworkManagementScanSetupType">
16383     <xs:restriction base="xs:string"/>
16384   </xs:simpleType>
16385   <xs:simpleType name="NetworkManagementSetupType">
16386     <xs:restriction base="xs:string"/>
16387   </xs:simpleType>
16388   <xs:simpleType name="NetworkManagementCandidateSetupType">
16389     <xs:restriction base="xs:string"/>
16390   </xs:simpleType>
16391   <xs:simpleType name="NetworkManagementTechnologyAddressType">
16392     <xs:restriction base="xs:string"/>
16393   </xs:simpleType>
16394   <xs:simpleType name="NetworkManagementCommunicationsTechnologyInformationType">
16395     <xs:restriction base="xs:string"/>
16396   </xs:simpleType>
16397   <xs:simpleType name="NetworkManagementMinimumTrustLevelType">
16398     <xs:restriction base="xs:string"/>
16399   </xs:simpleType>
16400   <xs:simpleType name="NetworkManagementProcessTimeoutType">
16401     <xs:restriction base="xs:duration"/>
16402   </xs:simpleType>
16403   <xs:simpleType name="NetworkManagementFeatureSetType">
16404     <xs:restriction base="xs:string">
16405       <xs:enumeration value="gateway"/>
16406       <xs:enumeration value="router"/>
16407       <xs:enumeration value="smart"/>
16408       <xs:enumeration value="simple"/>
16409     </xs:restriction>
16410   </xs:simpleType>
16411   <xs:simpleType name="NetworkManagementProcessStateType">
16412     <xs:restriction base="xs:string">
16413       <xs:enumeration value="succeeded"/>
16414       <xs:enumeration value="failed"/>
16415       <xs:enumeration value="aborted"/>
16416     </xs:restriction>
16417   </xs:simpleType>
16418   <xs:simpleType name="NetworkManagementStateChangeType">
16419     <xs:restriction base="xs:string">
16420       <xs:enumeration value="added"/>
16421       <xs:enumeration value="removed"/>
16422       <xs:enumeration value="modified"/>
16423     </xs:restriction>
16424   </xs:simpleType>
16425   <xs:complexType name="NetworkManagementAddNodeCallType">
16426     <xs:sequence>
16427       <xs:element minOccurs="0" name="nodeAddress" type="ns_p:FeatureAddressType"/>
16428       <xs:element minOccurs="0" name="nativeSetup"
16429 type="ns_p:NetworkManagementNativeSetupType"/>
16430       <xs:element minOccurs="0" name="timeout"
16431 type="ns_p:NetworkManagementProcessTimeoutType"/>

```

```
16432         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
16433         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
16434       </xs:sequence>
16435     </xs:complexType>
16436     <xs:element name="networkManagementAddNodeCall"
16437 type="ns_p:NetworkManagementAddNodeCallType"/>
16438     <xs:complexType name="NetworkManagementAddNodeCallElementsType">
16439       <xs:sequence>
16440         <xs:element minOccurs="0" name="nodeAddress"
16441 type="ns_p:FeatureAddressElementsType"/>
16442         <xs:element minOccurs="0" name="nativeSetup" type="ns_p:ElementTagType"/>
16443         <xs:element minOccurs="0" name="timeout" type="ns_p:ElementTagType"/>
16444         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
16445         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
16446       </xs:sequence>
16447     </xs:complexType>
16448     <xs:element name="networkManagementAddNodeCallElements"
16449 type="ns_p:NetworkManagementAddNodeCallElementsType"/>
16450     <xs:complexType name="NetworkManagementRemoveNodeCallType">
16451       <xs:sequence>
16452         <xs:element minOccurs="0" name="nodeAddress" type="ns_p:FeatureAddressType"/>
16453         <xs:element minOccurs="0" name="timeout"
16454 type="ns_p:NetworkManagementProcessTimeoutType"/>
16455       </xs:sequence>
16456     </xs:complexType>
16457     <xs:element name="networkManagementRemoveNodeCall"
16458 type="ns_p:NetworkManagementRemoveNodeCallType"/>
16459     <xs:complexType name="NetworkManagementRemoveNodeCallElementsType">
16460       <xs:sequence>
16461         <xs:element minOccurs="0" name="nodeAddress"
16462 type="ns_p:FeatureAddressElementsType"/>
16463         <xs:element minOccurs="0" name="timeout" type="ns_p:ElementTagType"/>
16464       </xs:sequence>
16465     </xs:complexType>
16466     <xs:element name="networkManagementRemoveNodeCallElements"
16467 type="ns_p:NetworkManagementRemoveNodeCallElementsType"/>
16468     <xs:complexType name="NetworkManagementModifyNodeCallType">
16469       <xs:sequence>
16470         <xs:element minOccurs="0" name="nodeAddress" type="ns_p:FeatureAddressType"/>
16471         <xs:element minOccurs="0" name="nativeSetup"
16472 type="ns_p:NetworkManagementNativeSetupType"/>
16473         <xs:element minOccurs="0" name="timeout"
16474 type="ns_p:NetworkManagementProcessTimeoutType"/>
16475         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
16476         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
16477       </xs:sequence>
16478     </xs:complexType>
16479     <xs:element name="networkManagementModifyNodeCall"
16480 type="ns_p:NetworkManagementModifyNodeCallType"/>
16481     <xs:complexType name="NetworkManagementModifyNodeCallElementsType">
16482       <xs:sequence>
16483         <xs:element minOccurs="0" name="nodeAddress"
16484 type="ns_p:FeatureAddressElementsType"/>
16485         <xs:element minOccurs="0" name="nativeSetup" type="ns_p:ElementTagType"/>
16486         <xs:element minOccurs="0" name="timeout" type="ns_p:ElementTagType"/>
16487         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
16488         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
16489       </xs:sequence>
16490     </xs:complexType>
16491     <xs:element name="networkManagementModifyNodeCallElements"
16492 type="ns_p:NetworkManagementModifyNodeCallElementsType"/>
16493     <xs:complexType name="NetworkManagementScanNetworkCallType">
16494       <xs:sequence>
16495         <xs:element minOccurs="0" name="scanSetup"
16496 type="ns_p:NetworkManagementScanSetupType"/>
16497         <xs:element minOccurs="0" name="timeout"
16498 type="ns_p:NetworkManagementProcessTimeoutType"/>
16499       </xs:sequence>
16500     </xs:complexType>
16501     <xs:element name="networkManagementScanNetworkCall"
16502 type="ns_p:NetworkManagementScanNetworkCallType"/>
16503     <xs:complexType name="NetworkManagementScanNetworkCallElementsType">
16504       <xs:sequence>
16505         <xs:element minOccurs="0" name="scanSetup" type="ns_p:ElementTagType"/>
16506         <xs:element minOccurs="0" name="timeout" type="ns_p:ElementTagType"/>
16507       </xs:sequence>
16508     </xs:complexType>
```

```
16509     <xs:element name="networkManagementScanNetworkCallElements"
16510 type="ns_p:NetworkManagementScanNetworkCallElementsType"/>
16511     <xs:complexType name="NetworkManagementDiscoverCallType">
16512       <xs:sequence>
16513         <xs:element minOccurs="0" name="discoverAddress" type="ns_p:FeatureAddressType"/>
16514       </xs:sequence>
16515     </xs:complexType>
16516     <xs:element name="networkManagementDiscoverCall"
16517 type="ns_p:NetworkManagementDiscoverCallType"/>
16518     <xs:complexType name="NetworkManagementDiscoverCallElementsType">
16519       <xs:sequence>
16520         <xs:element minOccurs="0" name="discoverAddress"
16521 type="ns_p:FeatureAddressElementsType"/>
16522       </xs:sequence>
16523     </xs:complexType>
16524     <xs:element name="networkManagementDiscoverCallElements"
16525 type="ns_p:NetworkManagementDiscoverCallElementsType"/>
16526     <xs:complexType name="NetworkManagementAbortCallType"/>
16527     <xs:element name="networkManagementAbortCall" type="ns_p:NetworkManagementAbortCallType"/>
16528     <xs:complexType name="NetworkManagementAbortCallElementsType"/>
16529     <xs:element name="networkManagementAbortCallElements"
16530 type="ns_p:NetworkManagementAbortCallElementsType"/>
16531     <xs:complexType name="NetworkManagementProcessStateDataType">
16532       <xs:sequence>
16533         <xs:element minOccurs="0" name="state"
16534 type="ns_p:NetworkManagementProcessStateStateType"/>
16535         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
16536       </xs:sequence>
16537     </xs:complexType>
16538     <xs:element name="networkManagementProcessStateData"
16539 type="ns_p:NetworkManagementProcessStateDataType"/>
16540     <xs:complexType name="NetworkManagementProcessStateDataElementsType">
16541       <xs:sequence>
16542         <xs:element minOccurs="0" name="state" type="ns_p:ElementTagType"/>
16543         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
16544       </xs:sequence>
16545     </xs:complexType>
16546     <xs:element name="networkManagementProcessStateDataElements"
16547 type="ns_p:NetworkManagementProcessStateDataElementsType"/>
16548     <xs:complexType name="NetworkManagementJoiningModeDataType">
16549       <xs:sequence>
16550         <xs:element minOccurs="0" name="setup" type="ns_p:NetworkManagementSetupType"/>
16551       </xs:sequence>
16552     </xs:complexType>
16553     <xs:element name="networkManagementJoiningModeData"
16554 type="ns_p:NetworkManagementJoiningModeDataType"/>
16555     <xs:complexType name="NetworkManagementJoiningModeDataElementsType">
16556       <xs:sequence>
16557         <xs:element minOccurs="0" name="setup" type="ns_p:ElementTagType"/>
16558       </xs:sequence>
16559     </xs:complexType>
16560     <xs:element name="networkManagementJoiningModeDataElements"
16561 type="ns_p:NetworkManagementJoiningModeDataElementsType"/>
16562     <xs:complexType name="NetworkManagementReportCandidateDataType">
16563       <xs:sequence>
16564         <xs:element minOccurs="0" name="candidateSetup"
16565 type="ns_p:NetworkManagementCandidateSetupType"/>
16566         <xs:element minOccurs="0" name="setupUsableForAdd" type="xs:boolean"/>
16567         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
16568         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
16569       </xs:sequence>
16570     </xs:complexType>
16571     <xs:element name="networkManagementReportCandidateData"
16572 type="ns_p:NetworkManagementReportCandidateDataType"/>
16573     <xs:complexType name="NetworkManagementReportCandidateDataElementsType">
16574       <xs:sequence>
16575         <xs:element minOccurs="0" name="candidateSetup" type="ns_p:ElementTagType"/>
16576         <xs:element minOccurs="0" name="setupUsableForAdd" type="ns_p:ElementTagType"/>
16577         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
16578         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
16579       </xs:sequence>
16580     </xs:complexType>
16581     <xs:element name="networkManagementReportCandidateDataElements"
16582 type="ns_p:NetworkManagementReportCandidateDataElementsType"/>
16583     <xs:complexType name="NetworkManagementDeviceDescriptionDataType">
16584       <xs:sequence>
16585         <xs:element minOccurs="0" name="deviceAddress" type="ns_p:DeviceAddressType"/>
16586         <xs:element minOccurs="0" name="deviceType" type="ns_p:DeviceTypeType"/>
```

```

16587         <xs:element minOccurs="0" name="networkManagementResponsibleAddress"
16588 type="ns_p:FeatureAddressType"/>
16589         <xs:element minOccurs="0" name="nativeSetup"
16590 type="ns_p:NetworkManagementNativeSetupType"/>
16591         <xs:element minOccurs="0" name="technologyAddress"
16592 type="ns_p:NetworkManagementTechnologyAddressType"/>
16593         <xs:element minOccurs="0" name="communicationsTechnologyInformation"
16594 type="ns_p:NetworkManagementCommunicationsTechnologyInformationType"/>
16595         <xs:element minOccurs="0" name="networkFeatureSet"
16596 type="ns_p:NetworkManagementFeatureSetType"/>
16597         <xs:element minOccurs="0" name="lastStateChange"
16598 type="ns_p:NetworkManagementStateChangeType"/>
16599         <xs:element minOccurs="0" name="minimumTrustLevel"
16600 type="ns_p:NetworkManagementMinimumTrustLevelType"/>
16601         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
16602         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
16603     </xs:sequence>
16604 </xs:complexType>
16605 <xs:element name="networkManagementDeviceDescriptionData"
16606 type="ns_p:NetworkManagementDeviceDescriptionDataType"/>
16607 <xs:complexType name="NetworkManagementDeviceDescriptionDataElementsType">
16608     <xs:sequence>
16609         <xs:element minOccurs="0" name="deviceAddress"
16610 type="ns_p:DeviceAddressElementsType"/>
16611         <xs:element minOccurs="0" name="deviceType" type="ns_p:ElementTagType"/>
16612         <xs:element minOccurs="0" name="networkManagementResponsibleAddress"
16613 type="ns_p:ElementTagType"/>
16614         <xs:element minOccurs="0" name="nativeSetup" type="ns_p:ElementTagType"/>
16615         <xs:element minOccurs="0" name="technologyAddress" type="ns_p:ElementTagType"/>
16616         <xs:element minOccurs="0" name="communicationsTechnologyInformation"
16617 type="ns_p:ElementTagType"/>
16618         <xs:element minOccurs="0" name="networkFeatureSet" type="ns_p:ElementTagType"/>
16619         <xs:element minOccurs="0" name="lastStateChange" type="ns_p:ElementTagType"/>
16620         <xs:element minOccurs="0" name="minimumTrustLevel" type="ns_p:ElementTagType"/>
16621         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
16622         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
16623     </xs:sequence>
16624 </xs:complexType>
16625 <xs:element name="networkManagementDeviceDescriptionDataElements"
16626 type="ns_p:NetworkManagementDeviceDescriptionDataElementsType"/>
16627 <xs:complexType name="NetworkManagementDeviceDescriptionListDataType">
16628     <xs:sequence>
16629         <xs:element maxOccurs="unbounded" minOccurs="0"
16630 ref="ns_p:networkManagementDeviceDescriptionData"/>
16631     </xs:sequence>
16632 </xs:complexType>
16633 <xs:element name="networkManagementDeviceDescriptionListData"
16634 type="ns_p:NetworkManagementDeviceDescriptionListDataType"/>
16635 <xs:complexType name="NetworkManagementDeviceDescriptionListDataSelectorsType">
16636     <xs:sequence>
16637         <xs:element minOccurs="0" name="deviceAddress" type="ns_p:DeviceAddressType"/>
16638         <xs:element minOccurs="0" name="deviceType" type="ns_p:DeviceTypeType"/>
16639     </xs:sequence>
16640 </xs:complexType>
16641 <xs:element name="networkManagementDeviceDescriptionListDataSelectors"
16642 type="ns_p:NetworkManagementDeviceDescriptionListDataSelectorsType"/>
16643 <xs:complexType name="NetworkManagementEntityDescriptionDataType">
16644     <xs:sequence>
16645         <xs:element minOccurs="0" name="entityAddress" type="ns_p:EntityAddressType"/>
16646         <xs:element minOccurs="0" name="entityType" type="ns_p:EntityTypeType"/>
16647         <xs:element minOccurs="0" name="lastStateChange"
16648 type="ns_p:NetworkManagementStateChangeType"/>
16649         <xs:element minOccurs="0" name="minimumTrustLevel"
16650 type="ns_p:NetworkManagementMinimumTrustLevelType"/>
16651         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
16652         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
16653     </xs:sequence>
16654 </xs:complexType>
16655 <xs:element name="networkManagementEntityDescriptionData"
16656 type="ns_p:NetworkManagementEntityDescriptionDataType"/>
16657 <xs:complexType name="NetworkManagementEntityDescriptionDataElementsType">
16658     <xs:sequence>
16659         <xs:element minOccurs="0" name="entityAddress"
16660 type="ns_p:EntityAddressElementsType"/>
16661         <xs:element minOccurs="0" name="entityType" type="ns_p:ElementTagType"/>
16662         <xs:element minOccurs="0" name="lastStateChange" type="ns_p:ElementTagType"/>
16663         <xs:element minOccurs="0" name="minimumTrustLevel" type="ns_p:ElementTagType"/>
16664         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>

```

```
16665         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
16666     </xs:sequence>
16667 </xs:complexType>
16668     <xs:element name="networkManagementEntityDescriptionDataElements"
16669 type="ns_p:NetworkManagementEntityDescriptionDataElementsType"/>
16670     <xs:complexType name="NetworkManagementEntityDescriptionListDataType">
16671         <xs:sequence>
16672             <xs:element maxOccurs="unbounded" minOccurs="0"
16673 ref="ns_p:networkManagementEntityDescriptionData"/>
16674         </xs:sequence>
16675     </xs:complexType>
16676     <xs:element name="networkManagementEntityDescriptionListData"
16677 type="ns_p:NetworkManagementEntityDescriptionListDataType"/>
16678     <xs:complexType name="NetworkManagementEntityDescriptionListDataSelectorsType">
16679         <xs:sequence>
16680             <xs:element minOccurs="0" name="entityAddress" type="ns_p:EntityAddressType"/>
16681             <xs:element minOccurs="0" name="entityType" type="ns_p:EntityTypeType"/>
16682         </xs:sequence>
16683     </xs:complexType>
16684     <xs:element name="networkManagementEntityDescriptionListDataSelectors"
16685 type="ns_p:NetworkManagementEntityDescriptionListDataSelectorsType"/>
16686     <xs:complexType name="NetworkManagementFeatureDescriptionDataType">
16687         <xs:sequence>
16688             <xs:element minOccurs="0" name="featureAddress" type="ns_p:FeatureAddressType"/>
16689             <xs:element minOccurs="0" name="featureType" type="ns_p:FeatureTypeType"/>
16690             <xs:element maxOccurs="unbounded" minOccurs="0" name="specificUsage"
16691 type="ns_p:FeatureSpecificUsageType"/>
16692             <xs:element minOccurs="0" name="featureGroup" type="ns_p:FeatureGroupType"/>
16693             <xs:element minOccurs="0" name="role" type="ns_p:RoleType"/>
16694             <xs:element maxOccurs="unbounded" minOccurs="0" name="supportedFunction"
16695 type="ns_p:FunctionPropertyType"/>
16696             <xs:element minOccurs="0" name="lastStateChange"
16697 type="ns_p:NetworkManagementStateChangeType"/>
16698             <xs:element minOccurs="0" name="minimumTrustLevel"
16699 type="ns_p:NetworkManagementMinimumTrustLevelType"/>
16700             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
16701             <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
16702             <xs:element minOccurs="0" name="maxResponseDelay"
16703 type="ns_p:MaxResponseDelayType"/>
16704         </xs:sequence>
16705     </xs:complexType>
16706     <xs:element name="networkManagementFeatureDescriptionData"
16707 type="ns_p:NetworkManagementFeatureDescriptionDataType"/>
16708     <xs:complexType name="NetworkManagementFeatureDescriptionDataElementsType">
16709         <xs:sequence>
16710             <xs:element minOccurs="0" name="featureAddress"
16711 type="ns_p:FeatureAddressElementsType"/>
16712             <xs:element minOccurs="0" name="featureType" type="ns_p:ElementTagType"/>
16713             <xs:element minOccurs="0" name="specificUsage" type="ns_p:ElementTagType"/>
16714             <xs:element minOccurs="0" name="featureGroup" type="ns_p:ElementTagType"/>
16715             <xs:element minOccurs="0" name="role" type="ns_p:ElementTagType"/>
16716             <xs:element minOccurs="0" name="supportedFunction"
16717 type="ns_p:FunctionPropertyElementsType"/>
16718             <xs:element minOccurs="0" name="lastStateChange" type="ns_p:ElementTagType"/>
16719             <xs:element minOccurs="0" name="minimumTrustLevel" type="ns_p:ElementTagType"/>
16720             <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
16721             <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
16722             <xs:element minOccurs="0" name="maxResponseDelay" type="ns_p:ElementTagType"/>
16723         </xs:sequence>
16724     </xs:complexType>
16725     <xs:element name="networkManagementFeatureDescriptionDataElements"
16726 type="ns_p:NetworkManagementFeatureDescriptionDataElementsType"/>
16727     <xs:complexType name="NetworkManagementFeatureDescriptionListDataType">
16728         <xs:sequence>
16729             <xs:element maxOccurs="unbounded" minOccurs="0"
16730 ref="ns_p:networkManagementFeatureDescriptionData"/>
16731         </xs:sequence>
16732     </xs:complexType>
16733     <xs:element name="networkManagementFeatureDescriptionListData"
16734 type="ns_p:NetworkManagementFeatureDescriptionListDataType"/>
16735     <xs:complexType name="NetworkManagementFeatureDescriptionListDataSelectorsType">
16736         <xs:sequence>
16737             <xs:element minOccurs="0" name="featureAddress" type="ns_p:FeatureAddressType"/>
16738             <xs:element minOccurs="0" name="featureType" type="ns_p:FeatureTypeType"/>
16739         </xs:sequence>
16740     </xs:complexType>
16741     <xs:element name="networkManagementFeatureDescriptionListDataSelectors"
16742 type="ns_p:NetworkManagementFeatureDescriptionListDataSelectorsType"/>
```

```
</xs:schema>
```

A.2.18 OperatingConstraints

File name: EEBus_SPINE_TS_OperatingConstraints.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    Smart Premises Interoperable Neutral-Message Exchange (SPINE)
    Version 1.1.1
    2018-12-21
    Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
    Source: https://www.eebus.org/en/specifications/
-->
<xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
  blockDefault="#all" elementFormDefault="qualified">
  <xs:annotation>
    <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
    e.V. All rights reserved.</xs:documentation>
  </xs:annotation>
  <xs:include schemaLocation="EEBus_SPINE_TS_PowerSequences.xsd"/>
  <xs:complexType name="OperatingConstraintsInterruptDataType">
    <xs:sequence>
      <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
      <xs:element name="isPausable" type="xs:boolean" minOccurs="0"/>
      <xs:element name="isStoppable" type="xs:boolean" minOccurs="0"/>
      <xs:element name="notInterruptibleAtHighPower" type="xs:boolean" minOccurs="0"/>
      <xs:element name="maxCyclesPerDay" type="xs:unsignedInt" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="operatingConstraintsInterruptData"
    type="ns_p:OperatingConstraintsInterruptDataType"/>
  <xs:complexType name="OperatingConstraintsInterruptDataElementsType">
    <xs:sequence>
      <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
      <xs:element name="isPausable" minOccurs="0" type="ns_p:ElementTagType"/>
      <xs:element name="isStoppable" minOccurs="0" type="ns_p:ElementTagType"/>
      <xs:element name="notInterruptibleAtHighPower" minOccurs="0"
    type="ns_p:ElementTagType"/>
      <xs:element name="maxCyclesPerDay" minOccurs="0" type="ns_p:ElementTagType"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="operatingConstraintsInterruptDataElements"
    type="ns_p:OperatingConstraintsInterruptDataElementsType"/>
  <xs:complexType name="OperatingConstraintsInterruptListDataType">
    <xs:sequence>
      <xs:element maxOccurs="unbounded" minOccurs="0"
    ref="ns_p:operatingConstraintsInterruptData"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="operatingConstraintsInterruptListData"
    type="ns_p:OperatingConstraintsInterruptListDataType"/>
  <xs:complexType name="OperatingConstraintsInterruptListDataSelectorsType">
    <xs:sequence>
      <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="operatingConstraintsInterruptListDataSelectors"
    type="ns_p:OperatingConstraintsInterruptListDataSelectorsType"/>
  <xs:complexType name="OperatingConstraintsDurationDataType">
    <xs:sequence>
      <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
      <xs:element name="activeDurationMin" type="xs:duration" minOccurs="0"/>
      <xs:element name="activeDurationMax" type="xs:duration" minOccurs="0"/>
      <xs:element name="pauseDurationMin" type="xs:duration" minOccurs="0"/>
      <xs:element name="pauseDurationMax" type="xs:duration" minOccurs="0"/>
      <xs:element name="activeDurationSumMin" type="xs:duration" minOccurs="0"/>
      <xs:element name="activeDurationSumMax" type="xs:duration" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
```

```
16816 <xs:element name="operatingConstraintsDurationData"
16817 type="ns_p:OperatingConstraintsDurationDataType"/>
16818 <xs:complexType name="OperatingConstraintsDurationDataElementsType">
16819 <xs:sequence>
16820 <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
16821 <xs:element name="activeDurationMin" minOccurs="0" type="ns_p:ElementTagType"/>
16822 <xs:element name="activeDurationMax" minOccurs="0" type="ns_p:ElementTagType"/>
16823 <xs:element name="pauseDurationMin" minOccurs="0" type="ns_p:ElementTagType"/>
16824 <xs:element name="pauseDurationMax" minOccurs="0" type="ns_p:ElementTagType"/>
16825 <xs:element name="activeDurationSumMin" minOccurs="0" type="ns_p:ElementTagType"/>
16826 <xs:element name="activeDurationSumMax" minOccurs="0" type="ns_p:ElementTagType"/>
16827 </xs:sequence>
16828 </xs:complexType>
16829 <xs:element name="operatingConstraintsDurationDataElements"
16830 type="ns_p:OperatingConstraintsDurationDataElementsType"/>
16831 <xs:complexType name="OperatingConstraintsDurationListDataType">
16832 <xs:sequence>
16833 <xs:element maxOccurs="unbounded" minOccurs="0"
16834 ref="ns_p:operatingConstraintsDurationData"/>
16835 </xs:sequence>
16836 </xs:complexType>
16837 <xs:element name="operatingConstraintsDurationListData"
16838 type="ns_p:OperatingConstraintsDurationListDataType"/>
16839 <xs:complexType name="OperatingConstraintsDurationListDataSelectorsType">
16840 <xs:sequence>
16841 <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
16842 </xs:sequence>
16843 </xs:complexType>
16844 <xs:element name="operatingConstraintsDurationListDataSelectors"
16845 type="ns_p:OperatingConstraintsDurationListDataSelectorsType"/>
16846 <xs:complexType name="OperatingConstraintsPowerDescriptionDataType">
16847 <xs:sequence>
16848 <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
16849 <xs:element minOccurs="0" name="positiveEnergyDirection"
16850 type="ns_p:EnergyDirectionType"/>
16851 <xs:element minOccurs="0" name="powerUnit" type="ns_p:UnitOfMeasurementType"/>
16852 <xs:element minOccurs="0" name="energyUnit" type="ns_p:UnitOfMeasurementType"/>
16853 <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
16854 </xs:sequence>
16855 </xs:complexType>
16856 <xs:element name="operatingConstraintsPowerDescriptionData"
16857 type="ns_p:OperatingConstraintsPowerDescriptionDataType"/>
16858 <xs:complexType name="OperatingConstraintsPowerDescriptionDataElementsType">
16859 <xs:sequence>
16860 <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
16861 <xs:element minOccurs="0" name="positiveEnergyDirection"
16862 type="ns_p:ElementTagType"/>
16863 <xs:element minOccurs="0" name="powerUnit" type="ns_p:ElementTagType"/>
16864 <xs:element minOccurs="0" name="energyUnit" type="ns_p:ElementTagType"/>
16865 <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
16866 </xs:sequence>
16867 </xs:complexType>
16868 <xs:element name="operatingConstraintsPowerDescriptionDataElements"
16869 type="ns_p:OperatingConstraintsPowerDescriptionDataElementsType"/>
16870 <xs:complexType name="OperatingConstraintsPowerDescriptionListDataType">
16871 <xs:sequence>
16872 <xs:element maxOccurs="unbounded" minOccurs="0"
16873 ref="ns_p:operatingConstraintsPowerDescriptionData"/>
16874 </xs:sequence>
16875 </xs:complexType>
16876 <xs:element name="operatingConstraintsPowerDescriptionListData"
16877 type="ns_p:OperatingConstraintsPowerDescriptionListDataType"/>
16878 <xs:complexType name="OperatingConstraintsPowerDescriptionListDataSelectorsType">
16879 <xs:sequence>
16880 <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
16881 </xs:sequence>
16882 </xs:complexType>
16883 <xs:element name="operatingConstraintsPowerDescriptionListDataSelectors"
16884 type="ns_p:OperatingConstraintsPowerDescriptionListDataSelectorsType"/>
16885 <xs:complexType name="OperatingConstraintsPowerRangeDataType">
16886 <xs:sequence>
16887 <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
16888 <xs:element name="powerMin" type="ns_p:ScaledNumberType" minOccurs="0"/>
16889 <xs:element name="powerMax" type="ns_p:ScaledNumberType" minOccurs="0"/>
16890 <xs:element name="energyMin" type="ns_p:ScaledNumberType" minOccurs="0"/>
16891 <xs:element name="energyMax" type="ns_p:ScaledNumberType" minOccurs="0"/>
16892 </xs:sequence>
16893 </xs:complexType>
```

```

16894     <xs:element name="operatingConstraintsPowerRangeData"
16895 type="ns_p:OperatingConstraintsPowerRangeDataType"/>
16896     <xs:complexType name="OperatingConstraintsPowerRangeDataElementsType">
16897       <xs:sequence>
16898         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
16899         <xs:element name="powerMin" minOccurs="0" type="ns_p:ScaledNumberElementsType"/>
16900         <xs:element name="powerMax" minOccurs="0" type="ns_p:ScaledNumberElementsType"/>
16901         <xs:element name="energyMin" minOccurs="0" type="ns_p:ScaledNumberElementsType"/>
16902         <xs:element name="energyMax" minOccurs="0" type="ns_p:ScaledNumberElementsType"/>
16903       </xs:sequence>
16904     </xs:complexType>
16905     <xs:element name="operatingConstraintsPowerRangeDataElements"
16906 type="ns_p:OperatingConstraintsPowerRangeDataElementsType"/>
16907     <xs:complexType name="OperatingConstraintsPowerRangeListDataType">
16908       <xs:sequence>
16909         <xs:element maxOccurs="unbounded" minOccurs="0"
16910 ref="ns_p:operatingConstraintsPowerRangeData"/>
16911       </xs:sequence>
16912     </xs:complexType>
16913     <xs:element name="operatingConstraintsPowerRangeListData"
16914 type="ns_p:OperatingConstraintsPowerRangeListDataType"/>
16915     <xs:complexType name="OperatingConstraintsPowerRangeListDataSelectorsType">
16916       <xs:sequence>
16917         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
16918       </xs:sequence>
16919     </xs:complexType>
16920     <xs:element name="operatingConstraintsPowerRangeListDataSelectors"
16921 type="ns_p:OperatingConstraintsPowerRangeListDataSelectorsType"/>
16922     <xs:complexType name="OperatingConstraintsPowerLevelDataType">
16923       <xs:sequence>
16924         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
16925         <xs:element name="power" type="ns_p:ScaledNumberType" minOccurs="0"
16926 maxOccurs="unbounded"/>
16927       </xs:sequence>
16928     </xs:complexType>
16929     <xs:element name="operatingConstraintsPowerLevelData"
16930 type="ns_p:OperatingConstraintsPowerLevelDataType"/>
16931     <xs:complexType name="OperatingConstraintsPowerLevelDataElementsType">
16932       <xs:sequence>
16933         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
16934         <xs:element name="power" minOccurs="0" type="ns_p:ScaledNumberElementsType"/>
16935       </xs:sequence>
16936     </xs:complexType>
16937     <xs:element name="operatingConstraintsPowerLevelDataElements"
16938 type="ns_p:OperatingConstraintsPowerLevelDataElementsType"/>
16939     <xs:complexType name="OperatingConstraintsPowerLevelListDataType">
16940       <xs:sequence>
16941         <xs:element maxOccurs="unbounded" minOccurs="0"
16942 ref="ns_p:operatingConstraintsPowerLevelData"/>
16943       </xs:sequence>
16944     </xs:complexType>
16945     <xs:element name="operatingConstraintsPowerLevelListData"
16946 type="ns_p:OperatingConstraintsPowerLevelListDataType"/>
16947     <xs:complexType name="OperatingConstraintsPowerLevelListDataSelectorsType">
16948       <xs:sequence>
16949         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
16950       </xs:sequence>
16951     </xs:complexType>
16952     <xs:element name="operatingConstraintsPowerLevelListDataSelectors"
16953 type="ns_p:OperatingConstraintsPowerLevelListDataSelectorsType"/>
16954     <xs:complexType name="OperatingConstraintsResumeImplicationDataType">
16955       <xs:sequence>
16956         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
16957         <xs:element name="resumeEnergyEstimated" type="ns_p:ScaledNumberType"
16958 minOccurs="0"/>
16959         <xs:element name="energyUnit" minOccurs="0" type="ns_p:UnitOfMeasurementType"/>
16960         <xs:element name="resumeCostEstimated" type="ns_p:ScaledNumberType"
16961 minOccurs="0"/>
16962         <xs:element minOccurs="0" name="currency" type="ns_p:CurrencyType"/>
16963       </xs:sequence>
16964     </xs:complexType>
16965     <xs:element name="operatingConstraintsResumeImplicationData"
16966 type="ns_p:OperatingConstraintsResumeImplicationDataType"/>
16967     <xs:complexType name="OperatingConstraintsResumeImplicationDataElementsType">
16968       <xs:sequence>
16969         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
16970         <xs:element name="resumeEnergyEstimated" minOccurs="0"
16971 type="ns_p:ScaledNumberElementsType"/>

```



```

16972         <xs:element name="energyUnit" minOccurs="0" type="ns_p:ElementTagType"/>
16973         <xs:element name="resumeCostEstimated" minOccurs="0"
16974 type="ns_p:ScaledNumberElementsType"/>
16975         <xs:element minOccurs="0" name="currency" type="ns_p:ElementTagType"/>
16976     </xs:sequence>
16977 </xs:complexType>
16978 <xs:element name="operatingConstraintsResumeImplicationDataElements"
16979 type="ns_p:OperatingConstraintsResumeImplicationDataElementsType"/>
16980 <xs:complexType name="OperatingConstraintsResumeImplicationListDataType">
16981     <xs:sequence>
16982         <xs:element maxOccurs="unbounded" minOccurs="0"
16983 ref="ns_p:operatingConstraintsResumeImplicationData"/>
16984     </xs:sequence>
16985 </xs:complexType>
16986 <xs:element name="operatingConstraintsResumeImplicationListData"
16987 type="ns_p:OperatingConstraintsResumeImplicationListDataType"/>
16988 <xs:complexType name="OperatingConstraintsResumeImplicationListDataSelectorsType">
16989     <xs:sequence>
16990         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
16991     </xs:sequence>
16992 </xs:complexType>
16993 <xs:element name="operatingConstraintsResumeImplicationListDataSelectors"
16994 type="ns_p:OperatingConstraintsResumeImplicationListDataSelectorsType"/>
16995 </xs:schema>
16996
16997

```

16998

16999 **A.2.19 PowerSequences**

17000 File name: EEBus_SPINE_TS_PowerSequences.xsd

```

17001 <?xml version="1.0" encoding="UTF-8"?>
17002 <!--
17003     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
17004     Version 1.1.1
17005     2018-12-21
17006     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
17007     Source: https://www.eebus.org/en/specifications/
17008 -->
17009 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
17010 xmlns:xs="http://www.w3.org/2001/XMLSchema"
17011 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
17012 blockDefault="#all" elementFormDefault="qualified">
17013     <xs:annotation>
17014         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
17015 e.V. All rights reserved.</xs:documentation>
17016     </xs:annotation>
17017     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
17018     <xs:include schemaLocation="EEBus_SPINE_TS_Measurement.xsd"/>
17019     <xs:simpleType name="AlternativesIdType">
17020         <xs:restriction base="xs:unsignedInt"/>
17021     </xs:simpleType>
17022     <xs:simpleType name="PowerSequenceIdType">
17023         <xs:restriction base="xs:unsignedInt"/>
17024     </xs:simpleType>
17025     <xs:simpleType name="PowerTimeSlotNumberType">
17026         <xs:restriction base="xs:unsignedInt"/>
17027     </xs:simpleType>
17028     <xs:simpleType name="PowerTimeSlotValueTypeType">
17029         <xs:union memberTypes="ns_p:PowerTimeSlotValueTypeEnumType ns_p:EnumExtendType"/>
17030     </xs:simpleType>
17031     <xs:simpleType name="PowerTimeSlotValueTypeEnumType">
17032         <xs:restriction base="xs:string">
17033             <xs:enumeration value="power"/>
17034             <xs:enumeration value="powerMin"/>
17035             <xs:enumeration value="powerMax"/>
17036             <xs:enumeration value="powerExpectedValue"/>
17037             <xs:enumeration value="powerStandardDeviation"/>
17038             <xs:enumeration value="powerSkewness"/>
17039             <xs:enumeration value="energy"/>
17040             <xs:enumeration value="energyMin"/>
17041             <xs:enumeration value="energyMax"/>
17042             <xs:enumeration value="energyExpectedValue"/>
17043             <xs:enumeration value="energyStandardDeviation"/>
17044

```

```
17045         <xs:enumeration value="energySkewness"/>
17046     </xs:restriction>
17047 </xs:simpleType>
17048 <xs:simpleType name="PowerSequenceScopeType">
17049     <xs:union memberTypes="ns_p:PowerSequenceScopeEnumType ns_p:EnumExtendType"/>
17050 </xs:simpleType>
17051 <xs:simpleType name="PowerSequenceScopeEnumType">
17052     <xs:restriction base="xs:string">
17053         <xs:enumeration value="forecast"/>
17054         <xs:enumeration value="measurement"/>
17055         <xs:enumeration value="recommendation"/>
17056     </xs:restriction>
17057 </xs:simpleType>
17058 <xs:simpleType name="PowerSequenceStateType">
17059     <xs:union memberTypes="ns_p:PowerSequenceStateEnumType ns_p:EnumExtendType"/>
17060 </xs:simpleType>
17061 <xs:simpleType name="PowerSequenceStateEnumType">
17062     <xs:restriction base="xs:string">
17063         <xs:enumeration value="running"/>
17064         <xs:enumeration value="paused"/>
17065         <xs:enumeration value="scheduled"/>
17066         <xs:enumeration value="scheduledPaused"/>
17067         <xs:enumeration value="pending"/>
17068         <xs:enumeration value="inactive"/>
17069         <xs:enumeration value="completed"/>
17070         <xs:enumeration value="invalid"/>
17071     </xs:restriction>
17072 </xs:simpleType>
17073 <xs:complexType name="PowerTimeSlotScheduleDataType">
17074     <xs:sequence>
17075         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17076         <xs:element minOccurs="0" name="slotNumber" type="ns_p:PowerTimeSlotNumberType"/>
17077         <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodType"/>
17078         <xs:element minOccurs="0" name="defaultDuration" type="xs:duration"/>
17079         <xs:element minOccurs="0" name="durationUncertainty" type="xs:duration"/>
17080         <xs:element minOccurs="0" name="slotActivated" type="xs:boolean"/>
17081         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
17082     </xs:sequence>
17083 </xs:complexType>
17084 <xs:element name="powerTimeSlotScheduleData" type="ns_p:PowerTimeSlotScheduleDataType"/>
17085 <xs:complexType name="PowerTimeSlotScheduleDataElementsType">
17086     <xs:sequence>
17087         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
17088         <xs:element minOccurs="0" name="slotNumber" type="ns_p:ElementTagType"/>
17089         <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodElementsType"/>
17090         <xs:element minOccurs="0" name="defaultDuration" type="ns_p:ElementTagType"/>
17091         <xs:element minOccurs="0" name="durationUncertainty" type="ns_p:ElementTagType"/>
17092         <xs:element minOccurs="0" name="slotActivated" type="ns_p:ElementTagType"/>
17093         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
17094     </xs:sequence>
17095 </xs:complexType>
17096 <xs:element name="powerTimeSlotScheduleDataElements"
17097 type="ns_p:PowerTimeSlotScheduleDataElementsType"/>
17098 <xs:complexType name="PowerTimeSlotScheduleListDataType">
17099     <xs:sequence>
17100         <xs:element minOccurs="0" maxOccurs="unbounded" minOccurs="0"
17101 ref="ns_p:powerTimeSlotScheduleData"/>
17102     </xs:sequence>
17103 </xs:complexType>
17104 <xs:element name="powerTimeSlotScheduleListData"
17105 type="ns_p:PowerTimeSlotScheduleListDataType"/>
17106 <xs:complexType name="PowerTimeSlotScheduleListDataSelectorsType">
17107     <xs:sequence>
17108         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17109         <xs:element minOccurs="0" name="slotNumber" type="ns_p:PowerTimeSlotNumberType"/>
17110     </xs:sequence>
17111 </xs:complexType>
17112 <xs:element name="powerTimeSlotScheduleListDataSelectors"
17113 type="ns_p:PowerTimeSlotScheduleListDataSelectorsType"/>
17114 <xs:complexType name="PowerTimeSlotValueDataType">
17115     <xs:sequence>
17116         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17117         <xs:element minOccurs="0" name="slotNumber" type="ns_p:PowerTimeSlotNumberType"/>
17118         <xs:element minOccurs="0" name="valueType"
17119 type="ns_p:PowerTimeSlotValueTypeType"/>
17120         <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberType"/>
17121     </xs:sequence>
17122 </xs:complexType>
```

```

17123 <xs:element name="powerTimeSlotValueData" type="ns_p:PowerTimeSlotValueDataType"/>
17124 <xs:complexType name="PowerTimeSlotValueDataElementsType">
17125 <xs:sequence>
17126 <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
17127 <xs:element minOccurs="0" name="slotNumber" type="ns_p:ElementTagType"/>
17128 <xs:element minOccurs="0" name="valueType" type="ns_p:ElementTagType"/>
17129 <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberElementsType"/>
17130 </xs:sequence>
17131 </xs:complexType>
17132 <xs:element name="powerTimeSlotValueDataElements"
17133 type="ns_p:PowerTimeSlotValueDataElementsType"/>
17134 <xs:complexType name="PowerTimeSlotValueListDataType">
17135 <xs:sequence>
17136 <xs:element maxOccurs="unbounded" minOccurs="0"
17137 ref="ns_p:powerTimeSlotValueData"/>
17138 </xs:sequence>
17139 </xs:complexType>
17140 <xs:element name="powerTimeSlotValueListData" type="ns_p:PowerTimeSlotValueListDataType"/>
17141 <xs:complexType name="PowerTimeSlotValueListDataSelectorsType">
17142 <xs:sequence>
17143 <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17144 <xs:element minOccurs="0" name="slotNumber" type="ns_p:PowerTimeSlotNumberType"/>
17145 <xs:element minOccurs="0" name="valueType"
17146 type="ns_p:PowerTimeSlotValueTypeType"/>
17147 </xs:sequence>
17148 </xs:complexType>
17149 <xs:element name="powerTimeSlotValueListDataSelectors"
17150 type="ns_p:PowerTimeSlotValueListDataSelectorsType"/>
17151 <xs:complexType name="PowerTimeSlotScheduleConstraintsDataType">
17152 <xs:sequence>
17153 <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17154 <xs:element minOccurs="0" name="slotNumber" type="ns_p:PowerTimeSlotNumberType"/>
17155 <xs:element minOccurs="0" name="earliestStartTime"
17156 type="ns_p:AbsoluteOrRelativeTimeType"/>
17157 <xs:element minOccurs="0" name="latestEndTime"
17158 type="ns_p:AbsoluteOrRelativeTimeType"/>
17159 <xs:element minOccurs="0" name="minDuration" type="xs:duration"/>
17160 <xs:element minOccurs="0" name="maxDuration" type="xs:duration"/>
17161 <xs:element minOccurs="0" name="optionalSlot" type="xs:boolean"/>
17162 </xs:sequence>
17163 </xs:complexType>
17164 <xs:element name="powerTimeSlotScheduleConstraintsData"
17165 type="ns_p:PowerTimeSlotScheduleConstraintsDataType"/>
17166 <xs:complexType name="PowerTimeSlotScheduleConstraintsDataElementsType">
17167 <xs:sequence>
17168 <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
17169 <xs:element minOccurs="0" name="slotNumber" type="ns_p:ElementTagType"/>
17170 <xs:element minOccurs="0" name="earliestStartTime" type="ns_p:ElementTagType"/>
17171 <xs:element minOccurs="0" name="latestEndTime" type="ns_p:ElementTagType"/>
17172 <xs:element minOccurs="0" name="minDuration" type="ns_p:ElementTagType"/>
17173 <xs:element minOccurs="0" name="maxDuration" type="ns_p:ElementTagType"/>
17174 <xs:element minOccurs="0" name="optionalSlot" type="ns_p:ElementTagType"/>
17175 </xs:sequence>
17176 </xs:complexType>
17177 <xs:element name="powerTimeSlotScheduleConstraintsDataElements"
17178 type="ns_p:PowerTimeSlotScheduleConstraintsDataElementsType"/>
17179 <xs:complexType name="PowerTimeSlotScheduleConstraintsListDataType">
17180 <xs:sequence>
17181 <xs:element maxOccurs="unbounded" minOccurs="0"
17182 ref="ns_p:powerTimeSlotScheduleConstraintsData"/>
17183 </xs:sequence>
17184 </xs:complexType>
17185 <xs:element name="powerTimeSlotScheduleConstraintsListData"
17186 type="ns_p:PowerTimeSlotScheduleConstraintsListDataType"/>
17187 <xs:complexType name="PowerTimeSlotScheduleConstraintsListDataSelectorsType">
17188 <xs:sequence>
17189 <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17190 <xs:element minOccurs="0" name="slotNumber" type="ns_p:PowerTimeSlotNumberType"/>
17191 </xs:sequence>
17192 </xs:complexType>
17193 <xs:element name="powerTimeSlotScheduleConstraintsListDataSelectors"
17194 type="ns_p:PowerTimeSlotScheduleConstraintsListDataSelectorsType"/>
17195 <xs:complexType name="PowerSequenceAlternativesRelationDataType">
17196 <xs:sequence>
17197 <xs:element minOccurs="0" name="alternativesId" type="ns_p:AlternativesIdType"/>
17198 <xs:element name="sequenceId" type="ns_p:PowerSequenceIdType" minOccurs="0"
17199 maxOccurs="unbounded"/>
17200 </xs:sequence>

```

```

17201     </xs:complexType>
17202     <xs:element name="powerSequenceAlternativesRelationData"
17203 type="ns_p:PowerSequenceAlternativesRelationDataType"/>
17204     <xs:complexType name="PowerSequenceAlternativesRelationDataElementsType">
17205       <xs:sequence>
17206         <xs:element minOccurs="0" name="alternativesId" type="ns_p:ElementTagType"/>
17207         <xs:element name="sequenceId" minOccurs="0" type="ns_p:ElementTagType"/>
17208       </xs:sequence>
17209     </xs:complexType>
17210     <xs:element name="powerSequenceAlternativesRelationDataElements"
17211 type="ns_p:PowerSequenceAlternativesRelationDataElementsType"/>
17212     <xs:complexType name="PowerSequenceAlternativesRelationListDataType">
17213       <xs:sequence>
17214         <xs:element maxOccurs="unbounded" minOccurs="0"
17215 ref="ns_p:powerSequenceAlternativesRelationData"/>
17216       </xs:sequence>
17217     </xs:complexType>
17218     <xs:element name="powerSequenceAlternativesRelationListData"
17219 type="ns_p:PowerSequenceAlternativesRelationListDataType"/>
17220     <xs:complexType name="PowerSequenceAlternativesRelationListDataSelectorsType">
17221       <xs:sequence>
17222         <xs:element minOccurs="0" name="alternativesId" type="ns_p:AlternativesIdType"/>
17223         <xs:element name="sequenceId" type="ns_p:PowerSequenceIdType" minOccurs="0"/>
17224       </xs:sequence>
17225     </xs:complexType>
17226     <xs:element name="powerSequenceAlternativesRelationListDataSelectors"
17227 type="ns_p:PowerSequenceAlternativesRelationListDataSelectorsType"/>
17228     <xs:complexType name="PowerSequenceDescriptionDataType">
17229       <xs:sequence>
17230         <xs:element name="sequenceId" type="ns_p:PowerSequenceIdType" minOccurs="0"/>
17231         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
17232         <xs:element minOccurs="0" name="positiveEnergyDirection"
17233 type="ns_p:EnergyDirectionType"/>
17234         <xs:element minOccurs="0" name="powerUnit" type="ns_p:UnitOfMeasurementType"/>
17235         <xs:element minOccurs="0" name="energyUnit" type="ns_p:UnitOfMeasurementType"/>
17236         <xs:element minOccurs="0" name="valueSource"
17237 type="ns_p:MeasurementValueSourceType"/>
17238         <xs:element minOccurs="0" name="scope" type="ns_p:PowerSequenceScopeType"/>
17239         <xs:element minOccurs="0" name="taskIdentifier" type="xs:unsignedInt"/>
17240         <xs:element minOccurs="0" name="repetitionsTotal" type="xs:unsignedInt"/>
17241       </xs:sequence>
17242     </xs:complexType>
17243     <xs:element name="powerSequenceDescriptionData"
17244 type="ns_p:PowerSequenceDescriptionDataType"/>
17245     <xs:complexType name="PowerSequenceDescriptionDataElementsType">
17246       <xs:sequence>
17247         <xs:element name="sequenceId" minOccurs="0" type="ns_p:ElementTagType"/>
17248         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
17249         <xs:element minOccurs="0" name="positiveEnergyDirection"
17250 type="ns_p:ElementTagType"/>
17251         <xs:element minOccurs="0" name="powerUnit" type="ns_p:ElementTagType"/>
17252         <xs:element minOccurs="0" name="energyUnit" type="ns_p:ElementTagType"/>
17253         <xs:element minOccurs="0" name="valueSource" type="ns_p:ElementTagType"/>
17254         <xs:element minOccurs="0" name="scope" type="ns_p:ElementTagType"/>
17255         <xs:element minOccurs="0" name="taskIdentifier" type="ns_p:ElementTagType"/>
17256         <xs:element minOccurs="0" name="repetitionsTotal" type="ns_p:ElementTagType"/>
17257       </xs:sequence>
17258     </xs:complexType>
17259     <xs:element name="powerSequenceDescriptionDataElements"
17260 type="ns_p:PowerSequenceDescriptionDataElementsType"/>
17261     <xs:complexType name="PowerSequenceDescriptionListDataType">
17262       <xs:sequence>
17263         <xs:element maxOccurs="unbounded" minOccurs="0"
17264 ref="ns_p:powerSequenceDescriptionData"/>
17265       </xs:sequence>
17266     </xs:complexType>
17267     <xs:element name="powerSequenceDescriptionListData"
17268 type="ns_p:PowerSequenceDescriptionListDataType"/>
17269     <xs:complexType name="PowerSequenceDescriptionListDataSelectorsType">
17270       <xs:sequence>
17271         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17272       </xs:sequence>
17273     </xs:complexType>
17274     <xs:element name="powerSequenceDescriptionListDataSelectors"
17275 type="ns_p:PowerSequenceDescriptionListDataSelectorsType"/>
17276     <xs:complexType name="PowerSequenceStateDataType">
17277       <xs:sequence>
17278         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>

```

```
17279         <xs:element minOccurs="0" name="state" type="ns_p:PowerSequenceStateType"/>
17280         <xs:element minOccurs="0" name="activeSlotNumber"
17281 type="ns_p:PowerTimeSlotNumberType"/>
17282         <xs:element minOccurs="0" name="elapsedSlotTime" type="xs:duration"/>
17283         <xs:element minOccurs="0" name="remainingSlotTime" type="xs:duration"/>
17284         <xs:element minOccurs="0" name="sequenceRemoteControllable" type="xs:boolean"/>
17285         <xs:element minOccurs="0" name="activeRepetitionNumber" type="xs:unsignedInt"/>
17286         <xs:element minOccurs="0" name="remainingPauseTime" type="xs:duration"/>
17287     </xs:sequence>
17288 </xs:complexType>
17289 <xs:element name="powerSequenceStateData" type="ns_p:PowerSequenceStateDataType"/>
17290 <xs:complexType name="PowerSequenceStateDataElementsType">
17291     <xs:sequence>
17292         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
17293         <xs:element minOccurs="0" name="state" type="ns_p:ElementTagType"/>
17294         <xs:element minOccurs="0" name="activeSlotNumber" type="ns_p:ElementTagType"/>
17295         <xs:element minOccurs="0" name="elapsedSlotTime" type="ns_p:ElementTagType"/>
17296         <xs:element minOccurs="0" name="remainingSlotTime" type="ns_p:ElementTagType"/>
17297         <xs:element minOccurs="0" name="sequenceRemoteControllable"
17298 type="ns_p:ElementTagType"/>
17299         <xs:element minOccurs="0" name="activeRepetitionNumber"
17300 type="ns_p:ElementTagType"/>
17301         <xs:element minOccurs="0" name="remainingPauseTime" type="ns_p:ElementTagType"/>
17302     </xs:sequence>
17303 </xs:complexType>
17304 <xs:element name="powerSequenceStateDataElements"
17305 type="ns_p:PowerSequenceStateDataElementsType"/>
17306 <xs:complexType name="PowerSequenceStateListDataType">
17307     <xs:sequence>
17308         <xs:element maxOccurs="unbounded" minOccurs="0"
17309 ref="ns_p:powerSequenceStateData"/>
17310     </xs:sequence>
17311 </xs:complexType>
17312 <xs:element name="powerSequenceStateListData" type="ns_p:PowerSequenceStateListDataType"/>
17313 <xs:complexType name="PowerSequenceStateListDataSelectorsType">
17314     <xs:sequence>
17315         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17316     </xs:sequence>
17317 </xs:complexType>
17318 <xs:element name="powerSequenceStateListDataSelectors"
17319 type="ns_p:PowerSequenceStateListDataSelectorsType"/>
17320 <xs:complexType name="PowerSequenceScheduleDataType">
17321     <xs:sequence>
17322         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17323         <xs:element minOccurs="0" name="startTime"
17324 type="ns_p:AbsoluteOrRelativeTimeType"/>
17325         <xs:element minOccurs="0" name="endTime" type="ns_p:AbsoluteOrRelativeTimeType"/>
17326     </xs:sequence>
17327 </xs:complexType>
17328 <xs:element name="powerSequenceScheduleData" type="ns_p:PowerSequenceScheduleDataType"/>
17329 <xs:complexType name="PowerSequenceScheduleDataElementsType">
17330     <xs:sequence>
17331         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
17332         <xs:element minOccurs="0" name="startTime" type="ns_p:ElementTagType"/>
17333         <xs:element minOccurs="0" name="endTime" type="ns_p:ElementTagType"/>
17334     </xs:sequence>
17335 </xs:complexType>
17336 <xs:element name="powerSequenceScheduleDataElements"
17337 type="ns_p:PowerSequenceScheduleDataElementsType"/>
17338 <xs:complexType name="PowerSequenceScheduleListDataType">
17339     <xs:sequence>
17340         <xs:element maxOccurs="unbounded" minOccurs="0"
17341 ref="ns_p:powerSequenceScheduleData"/>
17342     </xs:sequence>
17343 </xs:complexType>
17344 <xs:element name="powerSequenceScheduleListData"
17345 type="ns_p:PowerSequenceScheduleListDataType"/>
17346 <xs:complexType name="PowerSequenceScheduleListDataSelectorsType">
17347     <xs:sequence>
17348         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17349     </xs:sequence>
17350 </xs:complexType>
17351 <xs:element name="powerSequenceScheduleListDataSelectors"
17352 type="ns_p:PowerSequenceScheduleListDataSelectorsType"/>
17353 <xs:complexType name="PowerSequenceScheduleConstraintsDataType">
17354     <xs:sequence>
17355         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
```

```

17356         <xs:element minOccurs="0" name="earliestStartTime"
17357 type="ns_p:AbsoluteOrRelativeTimeType"/>
17358         <xs:element minOccurs="0" name="latestStartTime"
17359 type="ns_p:AbsoluteOrRelativeTimeType"/>
17360         <xs:element minOccurs="0" name="earliestEndTime"
17361 type="ns_p:AbsoluteOrRelativeTimeType"/>
17362         <xs:element minOccurs="0" name="latestEndTime"
17363 type="ns_p:AbsoluteOrRelativeTimeType"/>
17364         <xs:element minOccurs="0" name="optionalSequence" type="xs:boolean"/>
17365     </xs:sequence>
17366 </xs:complexType>
17367 <xs:element name="powerSequenceScheduleConstraintsData"
17368 type="ns_p:PowerSequenceScheduleConstraintsDataType"/>
17369 <xs:complexType name="PowerSequenceScheduleConstraintsDataElementsType">
17370     <xs:sequence>
17371         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
17372         <xs:element minOccurs="0" name="earliestStartTime" type="ns_p:ElementTagType"/>
17373         <xs:element minOccurs="0" name="latestStartTime" type="ns_p:ElementTagType"/>
17374         <xs:element minOccurs="0" name="earliestEndTime" type="ns_p:ElementTagType"/>
17375         <xs:element minOccurs="0" name="latestEndTime" type="ns_p:ElementTagType"/>
17376         <xs:element minOccurs="0" name="optionalSequence" type="ns_p:ElementTagType"/>
17377     </xs:sequence>
17378 </xs:complexType>
17379 <xs:element name="powerSequenceScheduleConstraintsDataElements"
17380 type="ns_p:PowerSequenceScheduleConstraintsDataElementsType"/>
17381 <xs:complexType name="PowerSequenceScheduleConstraintsListDataType">
17382     <xs:sequence>
17383         <xs:element maxOccurs="unbounded" minOccurs="0"
17384 ref="ns_p:powerSequenceScheduleConstraintsData"/>
17385     </xs:sequence>
17386 </xs:complexType>
17387 <xs:element name="powerSequenceScheduleConstraintsListData"
17388 type="ns_p:PowerSequenceScheduleConstraintsListDataType"/>
17389 <xs:complexType name="PowerSequenceScheduleConstraintsListDataSelectorsType">
17390     <xs:sequence>
17391         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17392     </xs:sequence>
17393 </xs:complexType>
17394 <xs:element name="powerSequenceScheduleConstraintsListDataSelectors"
17395 type="ns_p:PowerSequenceScheduleConstraintsListDataSelectorsType"/>
17396 <xs:complexType name="PowerSequencePriceDataType">
17397     <xs:sequence>
17398         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17399         <xs:element minOccurs="0" name="potentialStartTime"
17400 type="ns_p:AbsoluteOrRelativeTimeType"/>
17401         <xs:element minOccurs="0" name="price" type="ns_p:ScaledNumberType"/>
17402         <xs:element minOccurs="0" name="currency" type="ns_p:CurrencyType"/>
17403     </xs:sequence>
17404 </xs:complexType>
17405 <xs:element name="powerSequencePriceData" type="ns_p:PowerSequencePriceDataType"/>
17406 <xs:complexType name="PowerSequencePriceDataElementsType">
17407     <xs:sequence>
17408         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
17409         <xs:element minOccurs="0" name="potentialStartTime" type="ns_p:ElementTagType"/>
17410         <xs:element minOccurs="0" name="price" type="ns_p:ScaledNumberElementsType"/>
17411         <xs:element minOccurs="0" name="currency" type="ns_p:ElementTagType"/>
17412     </xs:sequence>
17413 </xs:complexType>
17414 <xs:element name="powerSequencePriceDataElements"
17415 type="ns_p:PowerSequencePriceDataElementsType"/>
17416 <xs:complexType name="PowerSequencePriceListDataType">
17417     <xs:sequence>
17418         <xs:element maxOccurs="unbounded" minOccurs="0"
17419 ref="ns_p:powerSequencePriceData"/>
17420     </xs:sequence>
17421 </xs:complexType>
17422 <xs:element name="powerSequencePriceListData" type="ns_p:PowerSequencePriceListDataType"/>
17423 <xs:complexType name="PowerSequencePriceListDataSelectorsType">
17424     <xs:sequence>
17425         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17426         <xs:element minOccurs="0" name="potentialStartTimeInterval"
17427 type="ns_p:TimestampIntervalType"/>
17428     </xs:sequence>
17429 </xs:complexType>
17430 <xs:element name="powerSequencePriceListDataSelectors"
17431 type="ns_p:PowerSequencePriceListDataSelectorsType"/>
17432 <xs:complexType name="PowerSequenceSchedulePreferenceDataType">
17433     <xs:sequence>

```

```

17434         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17435         <xs:element minOccurs="0" name="greenest" type="xs:boolean"/>
17436         <xs:element minOccurs="0" name="cheapest" type="xs:boolean"/>
17437     </xs:sequence>
17438 </xs:complexType>
17439 <xs:element name="powerSequenceSchedulePreferenceData"
17440 type="ns_p:PowerSequenceSchedulePreferenceDataType"/>
17441 <xs:complexType name="PowerSequenceSchedulePreferenceDataElementsType">
17442     <xs:sequence>
17443         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
17444         <xs:element minOccurs="0" name="greenest" type="ns_p:ElementTagType"/>
17445         <xs:element minOccurs="0" name="cheapest" type="ns_p:ElementTagType"/>
17446     </xs:sequence>
17447 </xs:complexType>
17448 <xs:element name="powerSequenceSchedulePreferenceDataElements"
17449 type="ns_p:PowerSequenceSchedulePreferenceDataElementsType"/>
17450 <xs:complexType name="PowerSequenceSchedulePreferenceListDataType">
17451     <xs:sequence>
17452         <xs:element maxOccurs="unbounded" minOccurs="0"
17453 ref="ns_p:powerSequenceSchedulePreferenceData"/>
17454     </xs:sequence>
17455 </xs:complexType>
17456 <xs:element name="powerSequenceSchedulePreferenceListData"
17457 type="ns_p:PowerSequenceSchedulePreferenceListDataType"/>
17458 <xs:complexType name="PowerSequenceSchedulePreferenceListDataSelectorsType">
17459     <xs:sequence>
17460         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17461     </xs:sequence>
17462 </xs:complexType>
17463 <xs:element name="powerSequenceSchedulePreferenceListDataSelectors"
17464 type="ns_p:PowerSequenceSchedulePreferenceListDataSelectorsType"/>
17465 <xs:complexType name="PowerSequenceNodeScheduleInformationDataType">
17466     <xs:sequence>
17467         <xs:element minOccurs="0" name="nodeRemoteControllable" type="xs:boolean"/>
17468         <xs:element minOccurs="0" name="supportsSingleSlotSchedulingOnly"
17469 type="xs:boolean"/>
17470         <xs:element minOccurs="0" name="alternativesCount" type="xs:unsignedInt"/>
17471         <xs:element minOccurs="0" name="totalSequencesCountMax" type="xs:unsignedInt"/>
17472         <xs:element minOccurs="0" name="supportsReselection" type="xs:boolean"/>
17473     </xs:sequence>
17474 </xs:complexType>
17475 <xs:element name="powerSequenceNodeScheduleInformationData"
17476 type="ns_p:PowerSequenceNodeScheduleInformationDataType"/>
17477 <xs:complexType name="PowerSequenceNodeScheduleInformationDataElementsType">
17478     <xs:sequence>
17479         <xs:element minOccurs="0" name="nodeRemoteControllable"
17480 type="ns_p:ElementTagType"/>
17481         <xs:element minOccurs="0" name="supportsSingleSlotSchedulingOnly"
17482 type="ns_p:ElementTagType"/>
17483         <xs:element minOccurs="0" name="alternativesCount" type="ns_p:ElementTagType"/>
17484         <xs:element minOccurs="0" name="totalSequencesCountMax"
17485 type="ns_p:ElementTagType"/>
17486         <xs:element minOccurs="0" name="supportsReselection" type="ns_p:ElementTagType"/>
17487     </xs:sequence>
17488 </xs:complexType>
17489 <xs:element name="powerSequenceNodeScheduleInformationDataElements"
17490 type="ns_p:PowerSequenceNodeScheduleInformationDataElementsType"/>
17491 <xs:complexType name="PowerSequenceScheduleConfigurationRequestCallType">
17492     <xs:sequence>
17493         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17494     </xs:sequence>
17495 </xs:complexType>
17496 <xs:element name="powerSequenceScheduleConfigurationRequestCall"
17497 type="ns_p:PowerSequenceScheduleConfigurationRequestCallType"/>
17498 <xs:complexType name="PowerSequenceScheduleConfigurationRequestCallElementsType">
17499     <xs:sequence>
17500         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
17501     </xs:sequence>
17502 </xs:complexType>
17503 <xs:element name="powerSequenceScheduleConfigurationRequestCallElements"
17504 type="ns_p:PowerSequenceScheduleConfigurationRequestCallElementsType"/>
17505 <xs:complexType name="PowerSequencePriceCalculationRequestCallType">
17506     <xs:sequence>
17507         <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
17508         <xs:element minOccurs="0" name="potentialStartTime"
17509 type="ns_p:AbsoluteOrRelativeTimeType"/>
17510     </xs:sequence>
17511 </xs:complexType>

```

```

17512     <xs:element name="powerSequencePriceCalculationRequestCall"
17513 type="ns_p:PowerSequencePriceCalculationRequestCallType"/>
17514     <xs:complexType name="PowerSequencePriceCalculationRequestCallElementsType">
17515       <xs:sequence>
17516         <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
17517         <xs:element minOccurs="0" name="potentialStartTime" type="ns_p:ElementTagType"/>
17518       </xs:sequence>
17519     </xs:complexType>
17520     <xs:element name="powerSequencePriceCalculationRequestCallElements"
17521 type="ns_p:PowerSequencePriceCalculationRequestCallElementsType"/>
17522   </xs:schema>
17523
17524

```

17525

17526 A.2.20 Sensing

17527 File name: EEBus_SPINE_TS_Sensing.xsd

```

17528
17529 <?xml version="1.0" encoding="UTF-8"?>
17530 <!--
17531   Smart Premises Interoperable Neutral-Message Exchange (SPINE)
17532   Version 1.1.1
17533   2018-12-21
17534   Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
17535   Source: https://www.eebus.org/en/specifications/
17536 -->
17537 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
17538 xmlns:xs="http://www.w3.org/2001/XMLSchema"
17539 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
17540 blockDefault="#all" elementFormDefault="qualified">
17541   <xs:annotation>
17542     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
17543 e.V. All rights reserved.</xs:documentation>
17544   </xs:annotation>
17545   <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
17546   <xs:simpleType name="SensingStateType">
17547     <xs:union memberTypes="ns_p:SensingStateEnumType ns_p:EnumExtendType"/>
17548   </xs:simpleType>
17549   <xs:simpleType name="SensingStateEnumType">
17550     <xs:restriction base="xs:string">
17551       <xs:enumeration value="on"/>
17552       <xs:enumeration value="off"/>
17553       <xs:enumeration value="toggle"/>
17554       <xs:enumeration value="level"/>
17555       <xs:enumeration value="levelUp"/>
17556       <xs:enumeration value="levelDown"/>
17557       <xs:enumeration value="levelStart"/>
17558       <xs:enumeration value="levelStop"/>
17559       <xs:enumeration value="levelAbsolute"/>
17560       <xs:enumeration value="levelRelative"/>
17561       <xs:enumeration value="levelPercentageAbsolute"/>
17562       <xs:enumeration value="levelPercentageRelative"/>
17563       <xs:enumeration value="pressed"/>
17564       <xs:enumeration value="longPressed"/>
17565       <xs:enumeration value="released"/>
17566       <xs:enumeration value="changed"/>
17567       <xs:enumeration value="started"/>
17568       <xs:enumeration value="stopped"/>
17569       <xs:enumeration value="paused"/>
17570       <xs:enumeration value="middle"/>
17571       <xs:enumeration value="up"/>
17572       <xs:enumeration value="down"/>
17573       <xs:enumeration value="forward"/>
17574       <xs:enumeration value="backwards"/>
17575       <xs:enumeration value="open"/>
17576       <xs:enumeration value="closed"/>
17577       <xs:enumeration value="opening"/>
17578       <xs:enumeration value="closing"/>
17579       <xs:enumeration value="high"/>
17580       <xs:enumeration value="low"/>
17581       <xs:enumeration value="day"/>
17582       <xs:enumeration value="night"/>
17583       <xs:enumeration value="detected"/>
17584       <xs:enumeration value="notDetected"/>

```



```

17585         <xs:enumeration value="alarmed"/>
17586         <xs:enumeration value="notAlarmed"/>
17587     </xs:restriction>
17588 </xs:simpleType>
17589 <xs:simpleType name="SensingTypeType">
17590     <xs:union memberTypes="ns_p:SensingTypeEnumType ns_p:EnumExtendType"/>
17591 </xs:simpleType>
17592 <xs:simpleType name="SensingTypeEnumType">
17593     <xs:restriction base="xs:string">
17594         <xs:enumeration value="switch"/>
17595         <xs:enumeration value="button"/>
17596         <xs:enumeration value="level"/>
17597         <xs:enumeration value="levelSwitch"/>
17598         <xs:enumeration value="windowHandle"/>
17599         <xs:enumeration value="contactSensor"/>
17600         <xs:enumeration value="occupancySensor"/>
17601         <xs:enumeration value="motionDetector"/>
17602         <xs:enumeration value="fireDetector"/>
17603         <xs:enumeration value="smokeDetector"/>
17604         <xs:enumeration value="heatDetector"/>
17605         <xs:enumeration value="waterDetector"/>
17606         <xs:enumeration value="gasDetector"/>
17607         <xs:enumeration value="alarmSensor"/>
17608         <xs:enumeration value="powerAlarmSensor"/>
17609         <xs:enumeration value="dayNightIndicator"/>
17610     </xs:restriction>
17611 </xs:simpleType>
17612 <xs:complexType name="SensingDataType">
17613     <xs:sequence>
17614         <xs:element minOccurs="0" name="timestamp"
17615 type="ns_p:AbsoluteOrRelativeTimeType"/>
17616         <xs:element minOccurs="0" name="state" type="ns_p:SensingStateType"/>
17617         <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberType"/>
17618     </xs:sequence>
17619 </xs:complexType>
17620 <xs:element name="sensingData" type="ns_p:SensingDataType"/>
17621 <xs:complexType name="SensingDataElementsType">
17622     <xs:sequence>
17623         <xs:element minOccurs="0" name="timestamp" type="ns_p:ElementTagType"/>
17624         <xs:element minOccurs="0" name="state" type="ns_p:ElementTagType"/>
17625         <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberElementsType"/>
17626     </xs:sequence>
17627 </xs:complexType>
17628 <xs:element name="sensingDataElements" type="ns_p:SensingDataElementsType"/>
17629 <xs:complexType name="SensingListDataType">
17630     <xs:sequence>
17631         <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:sensingData"/>
17632     </xs:sequence>
17633 </xs:complexType>
17634 <xs:element name="sensingListData" type="ns_p:SensingListDataType"/>
17635 <xs:complexType name="SensingListDataSelectorsType">
17636     <xs:sequence>
17637         <xs:element minOccurs="0" name="timestampInterval"
17638 type="ns_p:TimestampIntervalType"/>
17639     </xs:sequence>
17640 </xs:complexType>
17641 <xs:element name="sensingListDataSelectors" type="ns_p:SensingListDataSelectorsType"/>
17642 <xs:complexType name="SensingDescriptionDataType">
17643     <xs:sequence>
17644         <xs:element minOccurs="0" name="sensingType" type="ns_p:SensingTypeType"/>
17645         <xs:element minOccurs="0" name="unit" type="ns_p:UnitOfMeasurementType"/>
17646         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
17647         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
17648         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
17649     </xs:sequence>
17650 </xs:complexType>
17651 <xs:element name="sensingDescriptionData" type="ns_p:SensingDescriptionDataType"/>
17652 <xs:complexType name="SensingDescriptionDataElementsType">
17653     <xs:sequence>
17654         <xs:element minOccurs="0" name="sensingType" type="ns_p:ElementTagType"/>
17655         <xs:element minOccurs="0" name="unit" type="ns_p:ElementTagType"/>
17656         <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
17657         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
17658         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
17659     </xs:sequence>
17660 </xs:complexType>
17661 <xs:element name="sensingDescriptionDataElements"
17662 type="ns_p:SensingDescriptionDataElementsType"/>

```

17663 </xs:schema>
17664
17665

17666

17667 A.2.21 Setpoint

17668 File name: EEBus_SPINE_TS_Setpoint.xsd

```
17669 <?xml version="1.0" encoding="UTF-8"?>
17670 <!--
17671     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
17672     Version 1.1.1
17673     2018-12-21
17674     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
17675     Source: https://www.eebus.org/en/specifications/
17676 -->
17677 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
17678   xmlns:xs="http://www.w3.org/2001/XMLSchema"
17679   targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
17680   blockDefault="#all" elementFormDefault="qualified">
17681   <xs:annotation>
17682     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
17683     e.V. All rights reserved.</xs:documentation>
17684   </xs:annotation>
17685   <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
17686   <xs:include schemaLocation="EEBus_SPINE_TS_Measurement.xsd"/>
17687   <xs:include schemaLocation="EEBus_SPINE_TS_TimeTable.xsd"/>
17688   <xs:simpleType name="SetpointIdType">
17689     <xs:restriction base="xs:unsignedInt"/>
17690   </xs:simpleType>
17691   <xs:simpleType name="SetpointTypeType">
17692     <xs:union memberTypes="ns_p:SetpointTypeEnumType ns_p:EnumExtendType"/>
17693   </xs:simpleType>
17694   <xs:simpleType name="SetpointTypeEnumType">
17695     <xs:restriction base="xs:string">
17696       <xs:enumeration value="valueAbsolute"/>
17697       <xs:enumeration value="valueRelative"/>
17698     </xs:restriction>
17699   </xs:simpleType>
17700   <xs:complexType name="SetpointDataType">
17701     <xs:sequence>
17702       <xs:element minOccurs="0" name="setpointId" type="ns_p:SetpointIdType"/>
17703       <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberType"/>
17704       <xs:element minOccurs="0" name="valueMin" type="ns_p:ScaledNumberType"/>
17705       <xs:element minOccurs="0" name="valueMax" type="ns_p:ScaledNumberType"/>
17706       <xs:element minOccurs="0" name="valueToleranceAbsolute"
17707         type="ns_p:ScaledNumberType"/>
17708       <xs:element minOccurs="0" name="valueTolerancePercentage"
17709         type="ns_p:ScaledNumberType"/>
17710     </xs:sequence>
17711   </xs:complexType>
17712   <xs:element name="setpointData" type="ns_p:SetpointDataType"/>
17713   <xs:complexType name="SetpointDataElementsType">
17714     <xs:sequence>
17715       <xs:element minOccurs="0" name="setpointId" type="ns_p:ElementTagType"/>
17716       <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberElementsType"/>
17717       <xs:element minOccurs="0" name="valueMin" type="ns_p:ScaledNumberElementsType"/>
17718       <xs:element minOccurs="0" name="valueMax" type="ns_p:ScaledNumberElementsType"/>
17719       <xs:element minOccurs="0" name="valueToleranceAbsolute"
17720         type="ns_p:ScaledNumberElementsType"/>
17721       <xs:element minOccurs="0" name="valueTolerancePercentage"
17722         type="ns_p:ScaledNumberElementsType"/>
17723     </xs:sequence>
17724   </xs:complexType>
17725   <xs:element name="setpointDataElements" type="ns_p:SetpointDataElementsType"/>
17726   <xs:complexType name="SetpointListDataType">
17727     <xs:sequence>
17728       <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:setpointData"/>
17729     </xs:sequence>
17730   </xs:complexType>
17731   <xs:element name="setpointListData" type="ns_p:SetpointListDataType"/>
17732   <xs:complexType name="SetpointListDataSelectorsType">
17733     <xs:sequence>
17734       <xs:element minOccurs="0" name="setpointId" type="ns_p:SetpointIdType"/>
17735
```

```
17736         </xs:sequence>
17737     </xs:complexType>
17738     <xs:element name="setpointListDataSelectors" type="ns_p:SetpointListDataSelectorsType"/>
17739     <xs:complexType name="SetpointConstraintsDataType">
17740         <xs:sequence>
17741             <xs:element minOccurs="0" name="setpointId" type="ns_p:SetpointIdType"/>
17742             <xs:element minOccurs="0" name="setpointRangeMin" type="ns_p:ScaledNumberType"/>
17743             <xs:element minOccurs="0" name="setpointRangeMax" type="ns_p:ScaledNumberType"/>
17744             <xs:element minOccurs="0" name="setpointStepSize" type="ns_p:ScaledNumberType"/>
17745         </xs:sequence>
17746     </xs:complexType>
17747     <xs:element name="setpointConstraintsData" type="ns_p:SetpointConstraintsDataType"/>
17748     <xs:complexType name="SetpointConstraintsDataElementsType">
17749         <xs:sequence>
17750             <xs:element minOccurs="0" name="setpointId" type="ns_p:ElementTagType"/>
17751             <xs:element minOccurs="0" name="setpointRangeMin"
17752 type="ns_p:ScaledNumberElementsType"/>
17753             <xs:element minOccurs="0" name="setpointRangeMax"
17754 type="ns_p:ScaledNumberElementsType"/>
17755             <xs:element minOccurs="0" name="setpointStepSize"
17756 type="ns_p:ScaledNumberElementsType"/>
17757         </xs:sequence>
17758     </xs:complexType>
17759     <xs:element name="setpointConstraintsDataElements"
17760 type="ns_p:SetpointConstraintsDataElementsType"/>
17761     <xs:complexType name="SetpointConstraintsListDataType">
17762         <xs:sequence>
17763             <xs:element maxOccurs="unbounded" minOccurs="0"
17764 ref="ns_p:setpointConstraintsData"/>
17765         </xs:sequence>
17766     </xs:complexType>
17767     <xs:element name="setpointConstraintsListData"
17768 type="ns_p:SetpointConstraintsListDataType"/>
17769     <xs:complexType name="SetpointConstraintsListDataSelectorsType">
17770         <xs:sequence>
17771             <xs:element minOccurs="0" name="setpointId" type="ns_p:SetpointIdType"/>
17772         </xs:sequence>
17773     </xs:complexType>
17774     <xs:element name="setpointConstraintsListDataSelectors"
17775 type="ns_p:SetpointConstraintsListDataSelectorsType"/>
17776     <xs:complexType name="SetpointDescriptionDataType">
17777         <xs:sequence>
17778             <xs:element minOccurs="0" name="setpointId" type="ns_p:SetpointIdType"/>
17779             <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
17780             <xs:element minOccurs="0" name="timeTableId" type="ns_p:TimeTableIdType"/>
17781             <xs:element minOccurs="0" name="setpointType" type="ns_p:SetpointTypeType"/>
17782             <xs:element minOccurs="0" name="unit" type="ns_p:UnitOfMeasurementType"/>
17783             <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
17784             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
17785             <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
17786         </xs:sequence>
17787     </xs:complexType>
17788     <xs:element name="setpointDescriptionData" type="ns_p:SetpointDescriptionDataType"/>
17789     <xs:complexType name="SetpointDescriptionDataElementsType">
17790         <xs:sequence>
17791             <xs:element minOccurs="0" name="setpointId" type="ns_p:ElementTagType"/>
17792             <xs:element minOccurs="0" name="measurementId" type="ns_p:ElementTagType"/>
17793             <xs:element minOccurs="0" name="timeTableId" type="ns_p:ElementTagType"/>
17794             <xs:element minOccurs="0" name="setpointType" type="ns_p:ElementTagType"/>
17795             <xs:element minOccurs="0" name="unit" type="ns_p:ElementTagType"/>
17796             <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
17797             <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
17798             <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
17799         </xs:sequence>
17800     </xs:complexType>
17801     <xs:element name="setpointDescriptionDataElements"
17802 type="ns_p:SetpointDescriptionDataElementsType"/>
17803     <xs:complexType name="SetpointDescriptionListDataType">
17804         <xs:sequence>
17805             <xs:element maxOccurs="unbounded" minOccurs="0"
17806 ref="ns_p:setpointDescriptionData"/>
17807         </xs:sequence>
17808     </xs:complexType>
17809     <xs:element name="setpointDescriptionListData"
17810 type="ns_p:SetpointDescriptionListDataType"/>
17811     <xs:complexType name="SetpointDescriptionListDataSelectorsType">
17812         <xs:sequence>
17813             <xs:element minOccurs="0" name="setpointId" type="ns_p:SetpointIdType"/>
```

```

17814         <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
17815         <xs:element minOccurs="0" name="timeTableId" type="ns_p:TimeTableIdType"/>
17816         <xs:element minOccurs="0" name="setpointType" type="ns_p:SetpointTypeType"/>
17817         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
17818     </xs:sequence>
17819 </xs:complexType>
17820 <xs:element name="setpointDescriptionListDataSelectors"
17821 type="ns_p:SetpointDescriptionListDataSelectorsType"/>
17822 </xs:schema>
17823
17824

```

A.2.22 SubscriptionManagement

File name: EEBus_SPINE_TS_SubscriptionManagement.xsd

```

17828
17829 <?xml version="1.0" encoding="UTF-8"?>
17830 <!--
17831     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
17832     Version 1.1.1
17833     2018-12-21
17834     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
17835     Source: https://www.eebus.org/en/specifications/
17836 -->
17837 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
17838 xmlns:xs="http://www.w3.org/2001/XMLSchema"
17839 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
17840 blockDefault="#all" elementFormDefault="qualified">
17841     <xs:annotation>
17842         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
17843 e.V. All rights reserved.</xs:documentation>
17844     </xs:annotation>
17845     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
17846     <xs:simpleType name="SubscriptionIdType">
17847         <xs:restriction base="xs:unsignedInt"/>
17848     </xs:simpleType>
17849     <xs:complexType name="SubscriptionManagementEntryDataType">
17850         <xs:sequence>
17851             <xs:element minOccurs="0" name="subscriptionId" type="ns_p:SubscriptionIdType"/>
17852             <xs:element minOccurs="0" name="clientAddress" type="ns_p:FeatureAddressType"/>
17853             <xs:element minOccurs="0" name="serverAddress" type="ns_p:FeatureAddressType"/>
17854             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
17855             <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
17856         </xs:sequence>
17857     </xs:complexType>
17858     <xs:element name="subscriptionManagementEntryData"
17859 type="ns_p:SubscriptionManagementEntryDataType"/>
17860     <xs:complexType name="SubscriptionManagementEntryDataElementsType">
17861         <xs:sequence>
17862             <xs:element minOccurs="0" name="subscriptionId" type="ns_p:ElementTagType"/>
17863             <xs:element minOccurs="0" name="clientAddress"
17864 type="ns_p:FeatureAddressElementsType"/>
17865             <xs:element minOccurs="0" name="serverAddress"
17866 type="ns_p:FeatureAddressElementsType"/>
17867             <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
17868             <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
17869         </xs:sequence>
17870     </xs:complexType>
17871     <xs:element name="subscriptionManagementEntryDataElements"
17872 type="ns_p:SubscriptionManagementEntryDataElementsType"/>
17873     <xs:complexType name="SubscriptionManagementEntryListDataType">
17874         <xs:sequence>
17875             <xs:element maxOccurs="unbounded" minOccurs="0"
17876 ref="ns_p:subscriptionManagementEntryData"/>
17877         </xs:sequence>
17878     </xs:complexType>
17879     <xs:element name="subscriptionManagementEntryListData"
17880 type="ns_p:SubscriptionManagementEntryListDataType"/>
17881     <xs:complexType name="SubscriptionManagementEntryListDataSelectorsType">
17882         <xs:sequence>
17883             <xs:element minOccurs="0" name="subscriptionId" type="ns_p:SubscriptionIdType"/>
17884             <xs:element minOccurs="0" name="clientAddress" type="ns_p:FeatureAddressType"/>
17885             <xs:element minOccurs="0" name="serverAddress" type="ns_p:FeatureAddressType"/>
17886         </xs:sequence>

```

```

17887     </xs:complexType>
17888     <xs:element name="subscriptionManagementEntryListDataSelectors"
17889 type="ns_p:SubscriptionManagementEntryListDataSelectorsType"/>
17890     <xs:complexType name="SubscriptionManagementRequestCallType">
17891       <xs:sequence>
17892         <xs:element minOccurs="0" name="clientAddress" type="ns_p:FeatureAddressType"/>
17893         <xs:element minOccurs="0" name="serverAddress" type="ns_p:FeatureAddressType"/>
17894         <xs:element minOccurs="0" name="serverFeatureType" type="ns_p:FeatureTypeType"/>
17895       </xs:sequence>
17896     </xs:complexType>
17897     <xs:element name="subscriptionManagementRequestCall"
17898 type="ns_p:SubscriptionManagementRequestCallType"/>
17899     <xs:complexType name="SubscriptionManagementRequestCallElementsType">
17900       <xs:sequence>
17901         <xs:element minOccurs="0" name="clientAddress"
17902 type="ns_p:FeatureAddressElementsType"/>
17903         <xs:element minOccurs="0" name="serverAddress"
17904 type="ns_p:FeatureAddressElementsType"/>
17905         <xs:element minOccurs="0" name="serverFeatureType" type="ns_p:ElementTagType"/>
17906       </xs:sequence>
17907     </xs:complexType>
17908     <xs:element name="subscriptionManagementRequestCallElements"
17909 type="ns_p:SubscriptionManagementRequestCallElementsType"/>
17910     <xs:complexType name="SubscriptionManagementDeleteCallType">
17911       <xs:sequence>
17912         <xs:element minOccurs="0" name="subscriptionId" type="ns_p:SubscriptionIdType"/>
17913         <xs:element minOccurs="0" name="clientAddress" type="ns_p:FeatureAddressType"/>
17914         <xs:element minOccurs="0" name="serverAddress" type="ns_p:FeatureAddressType"/>
17915       </xs:sequence>
17916     </xs:complexType>
17917     <xs:element name="subscriptionManagementDeleteCall"
17918 type="ns_p:SubscriptionManagementDeleteCallType"/>
17919     <xs:complexType name="SubscriptionManagementDeleteCallElementsType">
17920       <xs:sequence>
17921         <xs:element minOccurs="0" name="subscriptionId" type="ns_p:ElementTagType"/>
17922         <xs:element minOccurs="0" name="clientAddress"
17923 type="ns_p:FeatureAddressElementsType"/>
17924         <xs:element minOccurs="0" name="serverAddress"
17925 type="ns_p:FeatureAddressElementsType"/>
17926       </xs:sequence>
17927     </xs:complexType>
17928     <xs:element name="subscriptionManagementDeleteCallElements"
17929 type="ns_p:SubscriptionManagementDeleteCallElementsType"/>
17930 </xs:schema>
17931
17932

```

17933

17934 A.2.23 SupplyCondition

17935 File name: EEBus_SPINE_TS_SupplyCondition.xsd

```

17936
17937 <?xml version="1.0" encoding="UTF-8"?>
17938 <!--
17939   Smart Premises Interoperable Neutral-Message Exchange (SPINE)
17940   Version 1.1.1
17941   2018-12-21
17942   Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
17943   Source: https://www.eebus.org/en/specifications/
17944 -->
17945 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
17946 xmlns:xs="http://www.w3.org/2001/XMLSchema"
17947 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
17948 blockDefault="#all" elementFormDefault="qualified">
17949   <xs:annotation>
17950     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
17951 e.V. All rights reserved.</xs:documentation>
17952   </xs:annotation>
17953   <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
17954   <xs:include schemaLocation="EEBus_SPINE_TS_Threshold.xsd"/>
17955   <xs:simpleType name="ConditionIdType">
17956     <xs:restriction base="xs:unsignedInt"/>
17957   </xs:simpleType>
17958   <xs:simpleType name="SupplyConditionEventTypeType">
17959     <xs:union memberTypes="ns_p:SupplyConditionEventTypeEnumType ns_p:EnumExtendType"/>

```

```

17960 </xs:simpleType>
17961 <xs:simpleType name="SupplyConditionEventTypeEnumType">
17962   <xs:restriction base="xs:string">
17963     <xs:enumeration value="thresholdExceeded"/>
17964     <xs:enumeration value="fallenBelowThreshold"/>
17965     <xs:enumeration value="supplyInterrupt"/>
17966     <xs:enumeration value="releaseOfLimitations"/>
17967     <xs:enumeration value="otherProblem"/>
17968     <xs:enumeration value="gridConditionUpdate"/>
17969   </xs:restriction>
17970 </xs:simpleType>
17971 <xs:simpleType name="SupplyConditionOriginatorType">
17972   <xs:union memberTypes="ns_p:SupplyConditionOriginatorEnumType ns_p:EnumExtendType"/>
17973 </xs:simpleType>
17974 <xs:simpleType name="SupplyConditionOriginatorEnumType">
17975   <xs:restriction base="xs:string">
17976     <xs:enumeration value="externDSO"/>
17977     <xs:enumeration value="externSupplier"/>
17978     <xs:enumeration value="internalLimit"/>
17979     <xs:enumeration value="internalService"/>
17980     <xs:enumeration value="internalUser"/>
17981   </xs:restriction>
17982 </xs:simpleType>
17983 <xs:simpleType name="GridConditionType">
17984   <xs:union memberTypes="ns_p:GridConditionEnumType ns_p:EnumExtendType"/>
17985 </xs:simpleType>
17986 <xs:simpleType name="GridConditionEnumType">
17987   <xs:restriction base="xs:string">
17988     <xs:enumeration value="consumptionRed"/>
17989     <xs:enumeration value="consumptionYellow"/>
17990     <xs:enumeration value="good"/>
17991     <xs:enumeration value="productionYellow"/>
17992     <xs:enumeration value="productionRed"/>
17993   </xs:restriction>
17994 </xs:simpleType>
17995 <xs:complexType name="SupplyConditionDataType">
17996   <xs:sequence>
17997     <xs:element minOccurs="0" name="conditionId" type="ns_p:ConditionIdType"/>
17998     <xs:element minOccurs="0" name="timestamp"
17999 type="ns_p:AbsoluteOrRelativeTimeType"/>
18000     <xs:element minOccurs="0" name="eventType"
18001 type="ns_p:SupplyConditionEventTypeType"/>
18002     <xs:element minOccurs="0" name="originator"
18003 type="ns_p:SupplyConditionOriginatorType"/>
18004     <xs:element minOccurs="0" name="thresholdId" type="ns_p:ThresholdIdType"/>
18005     <xs:element minOccurs="0" name="thresholdPercentage"
18006 type="ns_p:ScaledNumberType"/>
18007     <xs:element minOccurs="0" name="relevantPeriod" type="ns_p:TimePeriodType"/>
18008     <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
18009     <xs:element minOccurs="0" name="gridCondition" type="ns_p:GridConditionType"/>
18010   </xs:sequence>
18011 </xs:complexType>
18012 <xs:element name="supplyConditionData" type="ns_p:SupplyConditionDataType"/>
18013 <xs:complexType name="SupplyConditionDataElementsType">
18014   <xs:sequence>
18015     <xs:element minOccurs="0" name="conditionId" type="ns_p:ElementTagType"/>
18016     <xs:element minOccurs="0" name="timestamp" type="ns_p:ElementTagType"/>
18017     <xs:element minOccurs="0" name="eventType" type="ns_p:ElementTagType"/>
18018     <xs:element minOccurs="0" name="originator" type="ns_p:ElementTagType"/>
18019     <xs:element minOccurs="0" name="thresholdId" type="ns_p:ElementTagType"/>
18020     <xs:element minOccurs="0" name="thresholdPercentage"
18021 type="ns_p:ScaledNumberElementsType"/>
18022     <xs:element minOccurs="0" name="relevantPeriod"
18023 type="ns_p:TimePeriodElementsType"/>
18024     <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
18025     <xs:element minOccurs="0" name="gridCondition" type="ns_p:ElementTagType"/>
18026   </xs:sequence>
18027 </xs:complexType>
18028 <xs:element name="supplyConditionDataElements"
18029 type="ns_p:SupplyConditionDataElementsType"/>
18030 <xs:complexType name="SupplyConditionListDataType">
18031   <xs:sequence>
18032     <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:supplyConditionData"/>
18033   </xs:sequence>
18034 </xs:complexType>
18035 <xs:element name="supplyConditionListData" type="ns_p:SupplyConditionListDataType"/>
18036 <xs:complexType name="SupplyConditionListDataSelectorsType">
18037   <xs:sequence>

```

```
18038         <xs:element minOccurs="0" name="conditionId" type="ns_p:ConditionIdType"/>
18039         <xs:element minOccurs="0" name="timestampInterval"
18040 type="ns_p:TimestampIntervalType"/>
18041         <xs:element minOccurs="0" name="eventType"
18042 type="ns_p:SupplyConditionEventTypeType"/>
18043         <xs:element minOccurs="0" name="originator"
18044 type="ns_p:SupplyConditionOriginatorType"/>
18045     </xs:sequence>
18046 </xs:complexType>
18047 <xs:element name="supplyConditionListDataSelectors"
18048 type="ns_p:SupplyConditionListDataSelectorsType"/>
18049 <xs:complexType name="SupplyConditionDescriptionDataType">
18050     <xs:sequence>
18051         <xs:element minOccurs="0" name="conditionId" type="ns_p:ConditionIdType"/>
18052         <xs:element minOccurs="0" name="commodityType" type="ns_p:CommodityTypeType"/>
18053         <xs:element minOccurs="0" name="positiveEnergyDirection"
18054 type="ns_p:EnergyDirectionType"/>
18055         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
18056         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
18057     </xs:sequence>
18058 </xs:complexType>
18059 <xs:element name="supplyConditionDescriptionData"
18060 type="ns_p:SupplyConditionDescriptionDataType"/>
18061 <xs:complexType name="SupplyConditionDescriptionDataElementsType">
18062     <xs:sequence>
18063         <xs:element minOccurs="0" name="conditionId" type="ns_p:ElementTagType"/>
18064         <xs:element minOccurs="0" name="commodityType" type="ns_p:ElementTagType"/>
18065         <xs:element minOccurs="0" name="positiveEnergyDirection"
18066 type="ns_p:ElementTagType"/>
18067         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
18068         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
18069     </xs:sequence>
18070 </xs:complexType>
18071 <xs:element name="supplyConditionDescriptionDataElements"
18072 type="ns_p:SupplyConditionDescriptionDataElementsType"/>
18073 <xs:complexType name="SupplyConditionDescriptionListDataType">
18074     <xs:sequence>
18075         <xs:element maxOccurs="unbounded" minOccurs="0"
18076 ref="ns_p:supplyConditionDescriptionData"/>
18077     </xs:sequence>
18078 </xs:complexType>
18079 <xs:element name="supplyConditionDescriptionListData"
18080 type="ns_p:SupplyConditionDescriptionListDataType"/>
18081 <xs:complexType name="SupplyConditionDescriptionListDataSelectorsType">
18082     <xs:sequence>
18083         <xs:element minOccurs="0" name="conditionId" type="ns_p:ConditionIdType"/>
18084     </xs:sequence>
18085 </xs:complexType>
18086 <xs:element name="supplyConditionDescriptionListDataSelectors"
18087 type="ns_p:SupplyConditionDescriptionListDataSelectorsType"/>
18088 <xs:complexType name="SupplyConditionThresholdRelationDataType">
18089     <xs:sequence>
18090         <xs:element minOccurs="0" name="conditionId" type="ns_p:ConditionIdType"/>
18091         <xs:element maxOccurs="unbounded" minOccurs="0" name="thresholdId"
18092 type="ns_p:ThresholdIdType"/>
18093     </xs:sequence>
18094 </xs:complexType>
18095 <xs:element name="supplyConditionThresholdRelationData"
18096 type="ns_p:SupplyConditionThresholdRelationDataType"/>
18097 <xs:complexType name="SupplyConditionThresholdRelationDataElementsType">
18098     <xs:sequence>
18099         <xs:element minOccurs="0" name="conditionId" type="ns_p:ElementTagType"/>
18100         <xs:element minOccurs="0" name="thresholdId" type="ns_p:ElementTagType"/>
18101     </xs:sequence>
18102 </xs:complexType>
18103 <xs:element name="supplyConditionThresholdRelationDataElements"
18104 type="ns_p:SupplyConditionThresholdRelationDataElementsType"/>
18105 <xs:complexType name="SupplyConditionThresholdRelationListDataType">
18106     <xs:sequence>
18107         <xs:element maxOccurs="unbounded" minOccurs="0"
18108 ref="ns_p:supplyConditionThresholdRelationData"/>
18109     </xs:sequence>
18110 </xs:complexType>
18111 <xs:element name="supplyConditionThresholdRelationListData"
18112 type="ns_p:SupplyConditionThresholdRelationListDataType"/>
18113 <xs:complexType name="SupplyConditionThresholdRelationListDataSelectorsType">
18114     <xs:sequence>
18115         <xs:element minOccurs="0" name="conditionId" type="ns_p:ConditionIdType"/>
```

```
18116         <xs:element minOccurs="0" name="thresholdId" type="ns_p:ThresholdIdType"/>
18117     </xs:sequence>
18118 </xs:complexType>
18119     <xs:element name="supplyConditionThresholdRelationListDataSelectors"
18120 type="ns_p:SupplyConditionThresholdRelationListDataSelectorsType"/>
18121 </xs:schema>
18122
18123
```

18124

18125 A.2.24 TariffInformation

18126 File name: EEBus_SPINE_TS_TariffInformation.xsd

```
18127
18128 <?xml version="1.0" encoding="UTF-8"?>
18129 <!--
18130     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
18131     Version 1.1.1
18132     2018-12-21
18133     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
18134     Source: https://www.eebus.org/en/specifications/
18135 -->
18136 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
18137 xmlns:xs="http://www.w3.org/2001/XMLSchema"
18138 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
18139 blockDefault="#all" elementFormDefault="qualified">
18140     <xs:annotation>
18141         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
18142 e.V. All rights reserved.</xs:documentation>
18143     </xs:annotation>
18144     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
18145     <xs:include schemaLocation="EEBus_SPINE_TS_Measurement.xsd"/>
18146     <xs:include schemaLocation="EEBus_SPINE_TS_TimeTable.xsd"/>
18147     <xs:simpleType name="TariffIdType">
18148         <xs:restriction base="xs:unsignedInt"/>
18149     </xs:simpleType>
18150     <xs:simpleType name="TariffCountType">
18151         <xs:restriction base="ns_p:TariffIdType"/>
18152     </xs:simpleType>
18153     <xs:simpleType name="TierBoundaryIdType">
18154         <xs:restriction base="xs:unsignedInt"/>
18155     </xs:simpleType>
18156     <xs:simpleType name="TierBoundaryCountType">
18157         <xs:restriction base="ns_p:TierBoundaryIdType"/>
18158     </xs:simpleType>
18159     <xs:simpleType name="TierBoundaryTypeEnumType">
18160         <xs:union memberTypes="ns_p:TierBoundaryTypeEnumType ns_p:EnumExtendType"/>
18161     </xs:simpleType>
18162     <xs:simpleType name="TierBoundaryTypeEnumType">
18163         <xs:restriction base="xs:string">
18164             <xs:enumeration value="powerBoundary"/>
18165             <xs:enumeration value="energyBoundary"/>
18166             <xs:enumeration value="countBoundary"/>
18167         </xs:restriction>
18168     </xs:simpleType>
18169     <xs:simpleType name="CommodityIdType">
18170         <xs:restriction base="xs:unsignedInt"/>
18171     </xs:simpleType>
18172     <xs:simpleType name="TierIdType">
18173         <xs:restriction base="xs:unsignedInt"/>
18174     </xs:simpleType>
18175     <xs:simpleType name="TierCountType">
18176         <xs:restriction base="ns_p:TierIdType"/>
18177     </xs:simpleType>
18178     <xs:simpleType name="TierTypeEnumType">
18179         <xs:union memberTypes="ns_p:TierTypeEnumType ns_p:EnumExtendType"/>
18180     </xs:simpleType>
18181     <xs:simpleType name="TierTypeEnumType">
18182         <xs:restriction base="xs:string">
18183             <xs:enumeration value="fixedCost"/>
18184             <xs:enumeration value="dynamicCost"/>
18185         </xs:restriction>
18186     </xs:simpleType>
18187     <xs:simpleType name="IncentiveIdType">
18188         <xs:restriction base="xs:unsignedInt"/>
```



```
18189 </xs:simpleType>
18190 <xs:simpleType name="IncentiveCountType">
18191   <xs:restriction base="ns_p:IncentiveIdType"/>
18192 </xs:simpleType>
18193 <xs:simpleType name="IncentiveTypeType">
18194   <xs:union memberTypes="ns_p:IncentiveTypeEnumType ns_p:EnumExtendType"/>
18195 </xs:simpleType>
18196 <xs:simpleType name="IncentiveTypeEnumType">
18197   <xs:restriction base="xs:string">
18198     <xs:enumeration value="absoluteCost"/>
18199     <xs:enumeration value="relativeCost"/>
18200     <xs:enumeration value="renewableEnergyPercentage"/>
18201     <xs:enumeration value="co2Emission"/>
18202   </xs:restriction>
18203 </xs:simpleType>
18204 <xs:simpleType name="IncentivePriorityType">
18205   <xs:restriction base="xs:unsignedInt"/>
18206 </xs:simpleType>
18207 <xs:simpleType name="IncentiveValueTypeType">
18208   <xs:union memberTypes="ns_p:IncentiveValueTypeEnumType ns_p:EnumExtendType"/>
18209 </xs:simpleType>
18210 <xs:simpleType name="IncentiveValueTypeEnumType">
18211   <xs:restriction base="xs:string">
18212     <xs:enumeration value="value"/>
18213     <xs:enumeration value="averageValue"/>
18214     <xs:enumeration value="minValue"/>
18215     <xs:enumeration value="maxValue"/>
18216   </xs:restriction>
18217 </xs:simpleType>
18218 <xs:complexType name="TariffOverallConstraintsDataType">
18219   <xs:sequence>
18220     <xs:element minOccurs="0" name="maxTariffCount" type="ns_p:TariffCountType"/>
18221     <xs:element minOccurs="0" name="maxBoundaryCount"
18222 type="ns_p:TierBoundaryCountType"/>
18223     <xs:element minOccurs="0" name="maxTierCount" type="ns_p:TierCountType"/>
18224     <xs:element minOccurs="0" name="maxIncentiveCount"
18225 type="ns_p:IncentiveCountType"/>
18226     <xs:element minOccurs="0" name="maxBoundariesPerTariff"
18227 type="ns_p:TierBoundaryCountType"/>
18228     <xs:element minOccurs="0" name="maxTiersPerTariff" type="ns_p:TierCountType"/>
18229     <xs:element minOccurs="0" name="maxBoundariesPerTier"
18230 type="ns_p:TierBoundaryCountType"/>
18231     <xs:element minOccurs="0" name="maxIncentivesPerTier"
18232 type="ns_p:IncentiveCountType"/>
18233   </xs:sequence>
18234 </xs:complexType>
18235 <xs:element name="tariffOverallConstraintsData"
18236 type="ns_p:TariffOverallConstraintsDataType"/>
18237 <xs:complexType name="TariffOverallConstraintsDataElementsType">
18238   <xs:sequence>
18239     <xs:element minOccurs="0" name="maxTariffCount" type="ns_p:ElementTagType"/>
18240     <xs:element minOccurs="0" name="maxBoundaryCount" type="ns_p:ElementTagType"/>
18241     <xs:element minOccurs="0" name="maxTierCount" type="ns_p:ElementTagType"/>
18242     <xs:element minOccurs="0" name="maxIncentiveCount" type="ns_p:ElementTagType"/>
18243     <xs:element minOccurs="0" name="maxBoundariesPerTariff"
18244 type="ns_p:ElementTagType"/>
18245     <xs:element minOccurs="0" name="maxTiersPerTariff" type="ns_p:ElementTagType"/>
18246     <xs:element minOccurs="0" name="maxBoundariesPerTier" type="ns_p:ElementTagType"/>
18247     <xs:element minOccurs="0" name="maxIncentivesPerTier" type="ns_p:ElementTagType"/>
18248   </xs:sequence>
18249 </xs:complexType>
18250 <xs:element name="tariffOverallConstraintsDataElements"
18251 type="ns_p:TariffOverallConstraintsDataElementsType"/>
18252 <xs:complexType name="TariffDataType">
18253   <xs:sequence>
18254     <xs:element name="tariffId" type="ns_p:TariffIdType" minOccurs="0"/>
18255     <xs:element name="activeTierId" type="ns_p:TierIdType" minOccurs="0"
18256 maxOccurs="unbounded"/>
18257   </xs:sequence>
18258 </xs:complexType>
18259 <xs:element name="tariffData" type="ns_p:TariffDataType"/>
18260 <xs:complexType name="TariffDataElementsType">
18261   <xs:sequence>
18262     <xs:element name="tariffId" type="ns_p:ElementTagType" minOccurs="0"/>
18263     <xs:element name="activeTierId" type="ns_p:ElementTagType" minOccurs="0"/>
18264   </xs:sequence>
18265 </xs:complexType>
18266 <xs:element name="tariffDataElements" type="ns_p:TariffDataElementsType"/>
```

```
18267 <xs:complexType name="TariffListDataType">
18268   <xs:sequence>
18269     <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:tariffData"/>
18270   </xs:sequence>
18271 </xs:complexType>
18272 <xs:element name="tariffListData" type="ns_p:TariffListDataType"/>
18273 <xs:complexType name="TariffListDataSelectorsType">
18274   <xs:sequence>
18275     <xs:element name="tariffId" type="ns_p:TariffIdType" minOccurs="0"/>
18276     <xs:element name="activeTierId" type="ns_p:TierIdType" minOccurs="0"/>
18277   </xs:sequence>
18278 </xs:complexType>
18279 <xs:element name="tariffListDataSelectors" type="ns_p:TariffListDataSelectorsType"/>
18280 <xs:complexType name="TariffTierRelationDataType">
18281   <xs:sequence>
18282     <xs:element name="tariffId" type="ns_p:TariffIdType" minOccurs="0"/>
18283     <xs:element name="tierId" maxOccurs="unbounded" type="ns_p:TierIdType"
18284 minOccurs="0"/>
18285   </xs:sequence>
18286 </xs:complexType>
18287 <xs:element name="tariffTierRelationData" type="ns_p:TariffTierRelationDataType"/>
18288 <xs:complexType name="TariffTierRelationDataElementsType">
18289   <xs:sequence>
18290     <xs:element name="tariffId" type="ns_p:ElementTagType" minOccurs="0"/>
18291     <xs:element name="tierId" type="ns_p:ElementTagType" minOccurs="0"/>
18292   </xs:sequence>
18293 </xs:complexType>
18294 <xs:element name="tariffTierRelationDataElements"
18295 type="ns_p:TariffTierRelationDataElementsType"/>
18296 <xs:complexType name="TariffTierRelationListDataType">
18297   <xs:sequence>
18298     <xs:element maxOccurs="unbounded" minOccurs="0"
18299 ref="ns_p:tariffTierRelationData"/>
18300   </xs:sequence>
18301 </xs:complexType>
18302 <xs:element name="tariffTierRelationListData" type="ns_p:TariffTierRelationListDataType"/>
18303 <xs:complexType name="TariffTierRelationListDataSelectorsType">
18304   <xs:sequence>
18305     <xs:element name="tariffId" type="ns_p:TariffIdType" minOccurs="0"/>
18306     <xs:element name="tierId" type="ns_p:TierIdType" minOccurs="0"/>
18307   </xs:sequence>
18308 </xs:complexType>
18309 <xs:element name="tariffTierRelationListDataSelectors"
18310 type="ns_p:TariffTierRelationListDataSelectorsType"/>
18311 <xs:complexType name="TariffBoundaryRelationDataType">
18312   <xs:sequence>
18313     <xs:element name="tariffId" type="ns_p:TariffIdType" minOccurs="0"/>
18314     <xs:element minOccurs="0" name="boundaryId" type="ns_p:TierBoundaryIdType"
18315 maxOccurs="unbounded"/>
18316   </xs:sequence>
18317 </xs:complexType>
18318 <xs:element name="tariffBoundaryRelationData" type="ns_p:TariffBoundaryRelationDataType"/>
18319 <xs:complexType name="TariffBoundaryRelationDataElementsType">
18320   <xs:sequence>
18321     <xs:element name="tariffId" type="ns_p:ElementTagType" minOccurs="0"/>
18322     <xs:element name="boundaryId" type="ns_p:ElementTagType" minOccurs="0"/>
18323   </xs:sequence>
18324 </xs:complexType>
18325 <xs:element name="tariffBoundaryRelationDataElements"
18326 type="ns_p:TariffBoundaryRelationDataElementsType"/>
18327 <xs:complexType name="TariffBoundaryRelationListDataType">
18328   <xs:sequence>
18329     <xs:element maxOccurs="unbounded" minOccurs="0"
18330 ref="ns_p:tariffBoundaryRelationData"/>
18331   </xs:sequence>
18332 </xs:complexType>
18333 <xs:element name="tariffBoundaryRelationListData"
18334 type="ns_p:TariffBoundaryRelationListDataType"/>
18335 <xs:complexType name="TariffBoundaryRelationListDataSelectorsType">
18336   <xs:sequence>
18337     <xs:element name="tariffId" type="ns_p:TariffIdType" minOccurs="0"/>
18338     <xs:element minOccurs="0" name="boundaryId" type="ns_p:TierBoundaryIdType"/>
18339   </xs:sequence>
18340 </xs:complexType>
18341 <xs:element name="tariffBoundaryRelationListDataSelectors"
18342 type="ns_p:TariffBoundaryRelationListDataSelectorsType"/>
18343 <xs:complexType name="TariffDescriptionDataType">
18344   <xs:sequence>
```

```
18345      <xs:element name="tariffId" type="ns_p:TariffIdType" minOccurs="0"/>
18346      <xs:element name="commodityId" type="ns_p:CommodityIdType" minOccurs="0"/>
18347      <xs:element name="measurementId" type="ns_p:MeasurementIdType" minOccurs="0"/>
18348      <xs:element name="tariffWriteable" type="xs:boolean" minOccurs="0"/>
18349      <xs:element name="updateRequired" type="xs:boolean" minOccurs="0"/>
18350      <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
18351      <xs:element name="label" type="ns_p:LabelType" minOccurs="0"/>
18352      <xs:element name="description" type="ns_p:DescriptionType" minOccurs="0"/>
18353      <xs:element name="slotIdSupport" type="xs:boolean" minOccurs="0"/>
18354    </xs:sequence>
18355  </xs:complexType>
18356  <xs:element name="tariffDescriptionData" type="ns_p:TariffDescriptionDataType"/>
18357  <xs:complexType name="TariffDescriptionDataElementsType">
18358    <xs:sequence>
18359      <xs:element name="tariffId" type="ns_p:ElementTagType" minOccurs="0"/>
18360      <xs:element name="commodityId" type="ns_p:ElementTagType" minOccurs="0"/>
18361      <xs:element name="measurementId" type="ns_p:ElementTagType" minOccurs="0"/>
18362      <xs:element name="tariffWriteable" type="ns_p:ElementTagType" minOccurs="0"/>
18363      <xs:element name="updateRequired" type="ns_p:ElementTagType" minOccurs="0"/>
18364      <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
18365      <xs:element name="label" type="ns_p:ElementTagType" minOccurs="0"/>
18366      <xs:element name="description" type="ns_p:ElementTagType" minOccurs="0"/>
18367      <xs:element name="slotIdSupport" type="ns_p:ElementTagType" minOccurs="0"/>
18368    </xs:sequence>
18369  </xs:complexType>
18370  <xs:element name="tariffDescriptionDataElements"
18371    type="ns_p:TariffDescriptionDataElementsType"/>
18372  <xs:complexType name="TariffDescriptionListDataType">
18373    <xs:sequence>
18374      <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:tariffDescriptionData"/>
18375    </xs:sequence>
18376  </xs:complexType>
18377  <xs:element name="tariffDescriptionListData" type="ns_p:TariffDescriptionListDataType"/>
18378  <xs:complexType name="TariffDescriptionListDataSelectorsType">
18379    <xs:sequence>
18380      <xs:element name="tariffId" type="ns_p:TariffIdType" minOccurs="0"/>
18381      <xs:element name="commodityId" type="ns_p:CommodityIdType" minOccurs="0"/>
18382      <xs:element name="measurementId" type="ns_p:MeasurementIdType" minOccurs="0"/>
18383      <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
18384    </xs:sequence>
18385  </xs:complexType>
18386  <xs:element name="tariffDescriptionListDataSelectors"
18387    type="ns_p:TariffDescriptionListDataSelectorsType"/>
18388  <xs:complexType name="TierBoundaryDataType">
18389    <xs:sequence>
18390      <xs:element minOccurs="0" name="boundaryId" type="ns_p:TierBoundaryIdType"/>
18391      <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodType"/>
18392      <xs:element minOccurs="0" name="timeTableId" type="ns_p:TimeTableIdType"/>
18393      <xs:element minOccurs="0" name="lowerBoundaryValue" type="ns_p:ScaledNumberType"/>
18394      <xs:element minOccurs="0" name="upperBoundaryValue" type="ns_p:ScaledNumberType"/>
18395    </xs:sequence>
18396  </xs:complexType>
18397  <xs:element name="tierBoundaryData" type="ns_p:TierBoundaryDataType"/>
18398  <xs:complexType name="TierBoundaryDataElementsType">
18399    <xs:sequence>
18400      <xs:element minOccurs="0" name="boundaryId" type="ns_p:ElementTagType"/>
18401      <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodElementsType"/>
18402      <xs:element minOccurs="0" name="timeTableId" type="ns_p:ElementTagType"/>
18403      <xs:element minOccurs="0" name="lowerBoundaryValue"
18404        type="ns_p:ScaledNumberElementsType"/>
18405      <xs:element minOccurs="0" name="upperBoundaryValue"
18406        type="ns_p:ScaledNumberElementsType"/>
18407    </xs:sequence>
18408  </xs:complexType>
18409  <xs:element name="tierBoundaryDataElements" type="ns_p:TierBoundaryDataElementsType"/>
18410  <xs:complexType name="TierBoundaryListDataType">
18411    <xs:sequence>
18412      <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:tierBoundaryData"/>
18413    </xs:sequence>
18414  </xs:complexType>
18415  <xs:element name="tierBoundaryListData" type="ns_p:TierBoundaryListDataType"/>
18416  <xs:complexType name="TierBoundaryListDataSelectorsType">
18417    <xs:sequence>
18418      <xs:element minOccurs="0" name="boundaryId" type="ns_p:TierBoundaryIdType"/>
18419    </xs:sequence>
18420  </xs:complexType>
18421  <xs:element name="tierBoundaryListDataSelectors"
18422    type="ns_p:TierBoundaryListDataSelectorsType"/>
```

```

18423 <xs:complexType name="TierBoundaryDescriptionDataType">
18424 <xs:sequence>
18425 <xs:element minOccurs="0" name="boundaryId" type="ns_p:TierBoundaryIdType"/>
18426 <xs:element minOccurs="0" name="boundaryType" type="ns_p:TierBoundaryTypeType"/>
18427 <xs:element minOccurs="0" name="validForTierId" type="ns_p:TierIdType"/>
18428 <xs:element minOccurs="0" name="switchToTierIdWhenLower" type="ns_p:TierIdType"/>
18429 <xs:element minOccurs="0" name="switchToTierIdWhenHigher" type="ns_p:TierIdType"/>
18430 <xs:element minOccurs="0" name="boundaryUnit" type="ns_p:UnitOfMeasurementType"/>
18431 <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
18432 <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
18433 </xs:sequence>
18434 </xs:complexType>
18435 <xs:element name="tierBoundaryDescriptionData"
18436 type="ns_p:TierBoundaryDescriptionDataType"/>
18437 <xs:complexType name="TierBoundaryDescriptionDataElementsType">
18438 <xs:sequence>
18439 <xs:element minOccurs="0" name="boundaryId" type="ns_p:ElementTagType"/>
18440 <xs:element minOccurs="0" name="boundaryType" type="ns_p:ElementTagType"/>
18441 <xs:element minOccurs="0" name="validForTierId" type="ns_p:ElementTagType"/>
18442 <xs:element minOccurs="0" name="switchToTierIdWhenLower"
18443 type="ns_p:ElementTagType"/>
18444 <xs:element minOccurs="0" name="switchToTierIdWhenHigher"
18445 type="ns_p:ElementTagType"/>
18446 <xs:element minOccurs="0" name="boundaryUnit" type="ns_p:ElementTagType"/>
18447 <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
18448 <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
18449 </xs:sequence>
18450 </xs:complexType>
18451 <xs:element name="tierBoundaryDescriptionDataElements"
18452 type="ns_p:TierBoundaryDescriptionDataElementsType"/>
18453 <xs:complexType name="TierBoundaryDescriptionListDataType">
18454 <xs:sequence>
18455 <xs:element maxOccurs="unbounded" minOccurs="0"
18456 ref="ns_p:tierBoundaryDescriptionData"/>
18457 </xs:sequence>
18458 </xs:complexType>
18459 <xs:element name="tierBoundaryDescriptionListData"
18460 type="ns_p:TierBoundaryDescriptionListDataType"/>
18461 <xs:complexType name="TierBoundaryDescriptionListDataSelectorsType">
18462 <xs:sequence>
18463 <xs:element minOccurs="0" name="boundaryId" type="ns_p:TierBoundaryIdType"/>
18464 <xs:element minOccurs="0" name="boundaryType" type="ns_p:TierBoundaryTypeType"/>
18465 </xs:sequence>
18466 </xs:complexType>
18467 <xs:element name="tierBoundaryDescriptionListDataSelectors"
18468 type="ns_p:TierBoundaryDescriptionListDataSelectorsType"/>
18469 <xs:complexType name="CommodityDataType">
18470 <xs:sequence>
18471 <xs:element name="commodityId" type="ns_p:CommodityIdType" minOccurs="0"/>
18472 <xs:element name="commodityType" type="ns_p:CommodityTypeType" minOccurs="0"/>
18473 <xs:element name="positiveEnergyDirection" type="ns_p:EnergyDirectionType"
18474 minOccurs="0"/>
18475 <xs:element name="label" type="ns_p:LabelType" minOccurs="0"/>
18476 <xs:element name="description" type="ns_p:DescriptionType" minOccurs="0"/>
18477 </xs:sequence>
18478 </xs:complexType>
18479 <xs:element name="commodityData" type="ns_p:CommodityDataType"/>
18480 <xs:complexType name="CommodityDataElementsType">
18481 <xs:sequence>
18482 <xs:element name="commodityId" type="ns_p:ElementTagType" minOccurs="0"/>
18483 <xs:element name="commodityType" type="ns_p:ElementTagType" minOccurs="0"/>
18484 <xs:element name="positiveEnergyDirection" type="ns_p:ElementTagType"
18485 minOccurs="0"/>
18486 <xs:element name="label" type="ns_p:ElementTagType" minOccurs="0"/>
18487 <xs:element name="description" type="ns_p:ElementTagType" minOccurs="0"/>
18488 </xs:sequence>
18489 </xs:complexType>
18490 <xs:element name="commodityDataElements" type="ns_p:CommodityDataElementsType"/>
18491 <xs:complexType name="CommodityListDataType">
18492 <xs:sequence>
18493 <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:commodityData"/>
18494 </xs:sequence>
18495 </xs:complexType>
18496 <xs:element name="commodityListData" type="ns_p:CommodityListDataType"/>
18497 <xs:complexType name="CommodityListDataSelectorsType">
18498 <xs:sequence>
18499 <xs:element name="commodityId" type="ns_p:CommodityIdType" minOccurs="0"/>
18500 <xs:element name="commodityType" type="ns_p:CommodityTypeType" minOccurs="0"/>

```

```

18501         </xs:sequence>
18502     </xs:complexType>
18503     <xs:element name="commodityListDataSelectors" type="ns_p:CommodityListDataSelectorsType"/>
18504     <xs:complexType name="TierDataType">
18505         <xs:sequence>
18506             <xs:element name="tierId" type="ns_p:TierIdType" minOccurs="0"/>
18507             <xs:element name="timePeriod" type="ns_p:TimePeriodType" minOccurs="0"/>
18508             <xs:element minOccurs="0" name="timeTableId" type="ns_p:TimeTableIdType"/>
18509             <xs:element name="activeIncentiveId" type="ns_p:IncentiveIdType" minOccurs="0"
18510 maxOccurs="unbounded"/>
18511         </xs:sequence>
18512     </xs:complexType>
18513     <xs:element name="tierData" type="ns_p:TierDataType"/>
18514     <xs:complexType name="TierDataElementsType">
18515         <xs:sequence>
18516             <xs:element name="tierId" type="ns_p:ElementTagType" minOccurs="0"/>
18517             <xs:element name="timePeriod" type="ns_p:TimePeriodElementsType" minOccurs="0"/>
18518             <xs:element minOccurs="0" name="timeTableId" type="ns_p:ElementTagType"/>
18519             <xs:element name="activeIncentiveId" type="ns_p:ElementTagType" minOccurs="0"/>
18520         </xs:sequence>
18521     </xs:complexType>
18522     <xs:element name="tierDataElements" type="ns_p:TierDataElementsType"/>
18523     <xs:complexType name="TierListDataType">
18524         <xs:sequence>
18525             <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:tierData"/>
18526         </xs:sequence>
18527     </xs:complexType>
18528     <xs:element name="tierListData" type="ns_p:TierListDataType"/>
18529     <xs:complexType name="TierListDataSelectorsType">
18530         <xs:sequence>
18531             <xs:element name="tierId" type="ns_p:TierIdType" minOccurs="0"/>
18532             <xs:element name="activeIncentiveId" type="ns_p:IncentiveIdType" minOccurs="0"/>
18533         </xs:sequence>
18534     </xs:complexType>
18535     <xs:element name="tierListDataSelectors" type="ns_p:TierListDataSelectorsType"/>
18536     <xs:complexType name="TierIncentiveRelationDataType">
18537         <xs:sequence>
18538             <xs:element name="tierId" type="ns_p:TierIdType" minOccurs="0"/>
18539             <xs:element name="incentiveId" maxOccurs="unbounded" type="ns_p:IncentiveIdType"
18540 minOccurs="0"/>
18541         </xs:sequence>
18542     </xs:complexType>
18543     <xs:element name="tierIncentiveRelationData" type="ns_p:TierIncentiveRelationDataType"/>
18544     <xs:complexType name="TierIncentiveRelationDataElementsType">
18545         <xs:sequence>
18546             <xs:element name="tierId" type="ns_p:ElementTagType" minOccurs="0"/>
18547             <xs:element name="incentiveId" type="ns_p:ElementTagType" minOccurs="0"/>
18548         </xs:sequence>
18549     </xs:complexType>
18550     <xs:element name="tierIncentiveRelationDataElements"
18551 type="ns_p:TierIncentiveRelationDataElementsType"/>
18552     <xs:complexType name="TierIncentiveRelationListDataType">
18553         <xs:sequence>
18554             <xs:element maxOccurs="unbounded" minOccurs="0"
18555 ref="ns_p:tierIncentiveRelationData"/>
18556         </xs:sequence>
18557     </xs:complexType>
18558     <xs:element name="tierIncentiveRelationListData"
18559 type="ns_p:TierIncentiveRelationListDataType"/>
18560     <xs:complexType name="TierIncentiveRelationListDataSelectorsType">
18561         <xs:sequence>
18562             <xs:element name="tierId" type="ns_p:TierIdType" minOccurs="0"/>
18563             <xs:element name="incentiveId" type="ns_p:IncentiveIdType" minOccurs="0"/>
18564         </xs:sequence>
18565     </xs:complexType>
18566     <xs:element name="tierIncentiveRelationListDataSelectors"
18567 type="ns_p:TierIncentiveRelationListDataSelectorsType"/>
18568     <xs:complexType name="TierDescriptionDataType">
18569         <xs:sequence>
18570             <xs:element name="tierId" type="ns_p:TierIdType" minOccurs="0"/>
18571             <xs:element name="tierType" minOccurs="0" type="ns_p:TierTypeType"/>
18572             <xs:element name="label" type="ns_p:LabelType" minOccurs="0"/>
18573             <xs:element name="description" type="ns_p:DescriptionType" minOccurs="0"/>
18574         </xs:sequence>
18575     </xs:complexType>
18576     <xs:element name="tierDescriptionData" type="ns_p:TierDescriptionDataType"/>
18577     <xs:complexType name="TierDescriptionDataElementsType">
18578         <xs:sequence>

```

```
18579         <xs:element name="tierId" type="ns_p:ElementTagType" minOccurs="0"/>
18580         <xs:element name="tierType" minOccurs="0" type="ns_p:ElementTagType"/>
18581         <xs:element name="label" type="ns_p:ElementTagType" minOccurs="0"/>
18582         <xs:element name="description" type="ns_p:ElementTagType" minOccurs="0"/>
18583     </xs:sequence>
18584 </xs:complexType>
18585 <xs:element name="tierDescriptionDataElements"
18586 type="ns_p:TierDescriptionDataElementsType"/>
18587 <xs:complexType name="TierDescriptionListDataType">
18588     <xs:sequence>
18589         <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:tierDescriptionData"/>
18590     </xs:sequence>
18591 </xs:complexType>
18592 <xs:element name="tierDescriptionListData" type="ns_p:TierDescriptionListDataType"/>
18593 <xs:complexType name="TierDescriptionListDataSelectorsType">
18594     <xs:sequence>
18595         <xs:element name="tierId" type="ns_p:TierIdType" minOccurs="0"/>
18596         <xs:element name="tierType" minOccurs="0" type="ns_p:TierTypeType"/>
18597     </xs:sequence>
18598 </xs:complexType>
18599 <xs:element name="tierDescriptionListDataSelectors"
18600 type="ns_p:TierDescriptionListDataSelectorsType"/>
18601 <xs:complexType name="IncentiveDataType">
18602     <xs:sequence>
18603         <xs:element name="incentiveId" type="ns_p:IncentiveIdType" minOccurs="0"/>
18604         <xs:element name="valueType" type="ns_p:IncentiveValueTypeType" minOccurs="0"/>
18605         <xs:element name="timestamp" type="ns_p:AbsoluteOrRelativeTimeType"
18606 minOccurs="0"/>
18607         <xs:element name="timePeriod" type="ns_p:TimePeriodType" minOccurs="0"/>
18608         <xs:element minOccurs="0" name="timeTableId" type="ns_p:TimeTableIdType"/>
18609         <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberType"/>
18610     </xs:sequence>
18611 </xs:complexType>
18612 <xs:element name="incentiveData" type="ns_p:IncentiveDataType"/>
18613 <xs:complexType name="IncentiveDataElementsType">
18614     <xs:sequence>
18615         <xs:element name="incentiveId" type="ns_p:ElementTagType" minOccurs="0"/>
18616         <xs:element name="valueType" type="ns_p:ElementTagType" minOccurs="0"/>
18617         <xs:element name="timestamp" type="ns_p:ElementTagType" minOccurs="0"/>
18618         <xs:element name="timePeriod" type="ns_p:TimePeriodElementsType" minOccurs="0"/>
18619         <xs:element minOccurs="0" name="timeTableId" type="ns_p:ElementTagType"/>
18620         <xs:element minOccurs="0" name="value" type="ns_p:ElementTagType"/>
18621     </xs:sequence>
18622 </xs:complexType>
18623 <xs:element name="incentiveDataElements" type="ns_p:IncentiveDataElementsType"/>
18624 <xs:complexType name="IncentiveListDataType">
18625     <xs:sequence>
18626         <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:incentiveData"/>
18627     </xs:sequence>
18628 </xs:complexType>
18629 <xs:element name="incentiveListData" type="ns_p:IncentiveListDataType"/>
18630 <xs:complexType name="IncentiveListDataSelectorsType">
18631     <xs:sequence>
18632         <xs:element name="incentiveId" type="ns_p:IncentiveIdType" minOccurs="0"/>
18633         <xs:element name="valueType" type="ns_p:IncentiveValueTypeType" minOccurs="0"/>
18634         <xs:element minOccurs="0" name="timestampInterval"
18635 type="ns_p:TimestampIntervalType"/>
18636     </xs:sequence>
18637 </xs:complexType>
18638 <xs:element name="incentiveListDataSelectors" type="ns_p:IncentiveListDataSelectorsType"/>
18639 <xs:complexType name="IncentiveDescriptionDataType">
18640     <xs:sequence>
18641         <xs:element name="incentiveId" type="ns_p:IncentiveIdType" minOccurs="0"/>
18642         <xs:element name="incentiveType" type="ns_p:IncentiveTypeType" minOccurs="0"/>
18643         <xs:element name="incentivePriority" type="ns_p:IncentivePriorityType"
18644 minOccurs="0"/>
18645         <xs:element name="currency" type="ns_p:CurrencyType" minOccurs="0"/>
18646         <xs:element name="unit" minOccurs="0" type="ns_p:UnitOfMeasurementType"/>
18647         <xs:element name="label" type="ns_p:LabelType" minOccurs="0"/>
18648         <xs:element name="description" type="ns_p:DescriptionType" minOccurs="0"/>
18649     </xs:sequence>
18650 </xs:complexType>
18651 <xs:element name="incentiveDescriptionData" type="ns_p:IncentiveDescriptionDataType"/>
18652 <xs:complexType name="IncentiveDescriptionDataElementsType">
18653     <xs:sequence>
18654         <xs:element name="incentiveId" type="ns_p:ElementTagType" minOccurs="0"/>
18655         <xs:element name="incentiveType" type="ns_p:ElementTagType" minOccurs="0"/>
18656         <xs:element name="incentivePriority" type="ns_p:ElementTagType" minOccurs="0"/>
```

```

18657         <xs:element name="currency" type="ns_p:ElementTagType" minOccurs="0"/>
18658         <xs:element name="unit" minOccurs="0" type="ns_p:ElementTagType"/>
18659         <xs:element name="label" type="ns_p:ElementTagType" minOccurs="0"/>
18660         <xs:element name="description" type="ns_p:ElementTagType" minOccurs="0"/>
18661     </xs:sequence>
18662 </xs:complexType>
18663 <xs:element name="incentiveDescriptionDataElements"
18664 type="ns_p:IncentiveDescriptionDataElementsType"/>
18665 <xs:complexType name="IncentiveDescriptionListDataType">
18666     <xs:sequence>
18667         <xs:element maxOccurs="unbounded" minOccurs="0"
18668 ref="ns_p:incentiveDescriptionData"/>
18669     </xs:sequence>
18670 </xs:complexType>
18671 <xs:element name="incentiveDescriptionListData"
18672 type="ns_p:IncentiveDescriptionListDataType"/>
18673 <xs:complexType name="IncentiveDescriptionListDataSelectorsType">
18674     <xs:sequence>
18675         <xs:element name="incentiveId" type="ns_p:IncentiveIdType" minOccurs="0"/>
18676         <xs:element name="incentiveType" type="ns_p:IncentiveTypeType" minOccurs="0"/>
18677     </xs:sequence>
18678 </xs:complexType>
18679 <xs:element name="incentiveDescriptionListDataSelectors"
18680 type="ns_p:IncentiveDescriptionListDataSelectorsType"/>
18681 </xs:schema>
18682
18683
18684

```

A.2.25 TaskManagement

File name: EEBus_SPINE_TS_TaskManagement.xsd

```

18687 <?xml version="1.0" encoding="UTF-8"?>
18688 <!--
18689     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
18690     Version 1.1.1
18691     2018-12-21
18692     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
18693     Source: https://www.eebus.org/en/specifications/
18694 -->
18695 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
18696 xmlns:xs="http://www.w3.org/2001/XMLSchema"
18697 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
18698 blockDefault="#all" elementFormDefault="qualified">
18699     <xs:annotation>
18700         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
18701 e.V. All rights reserved.</xs:documentation>
18702     </xs:annotation>
18703     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
18704     <xs:include schemaLocation="EEBus_SPINE_TS_DirectControl.xsd"/>
18705     <xs:include schemaLocation="EEBus_SPINE_TS_HVAC.xsd"/>
18706     <xs:include schemaLocation="EEBus_SPINE_TS_LoadControl.xsd"/>
18707     <xs:include schemaLocation="EEBus_SPINE_TS_PowerSequences.xsd"/>
18708     <xs:simpleType name="TaskManagementJobIdType">
18709         <xs:restriction base="xs:unsignedInt"/>
18710     </xs:simpleType>
18711     <xs:simpleType name="TaskManagementJobStateType">
18712         <xs:union memberTypes="ns_p:DirectControlActivityStateEnumType
18713 ns_p:HvacOverrunStatusEnumType ns_p:LoadControlEventStateEnumType
18714 ns_p:PowerSequenceStateEnumType ns_p:EnumExtendType"/>
18715     </xs:simpleType>
18716     <xs:simpleType name="TaskManagementJobSourceType">
18717         <xs:union memberTypes="ns_p:TaskManagementJobSourceEnumType ns_p:EnumExtendType"/>
18718     </xs:simpleType>
18719     <xs:simpleType name="TaskManagementJobSourceEnumType">
18720         <xs:restriction base="xs:string">
18721             <xs:enumeration value="internalMechanism"/>
18722             <xs:enumeration value="userInteraction"/>
18723             <xs:enumeration value="externalConfiguration"/>
18724         </xs:restriction>
18725     </xs:simpleType>
18726     <xs:complexType name="TaskManagementDirectControlRelatedType"/>
18727     <xs:complexType name="TaskManagementDirectControlRelatedElementsType"/>
18728     <xs:complexType name="TaskManagementHvacRelatedType">
18729

```

```

18730         <xs:sequence>
18731             <xs:element minOccurs="0" name="overrunId" type="ns_p:HvacOverrunIdType"/>
18732         </xs:sequence>
18733     </xs:complexType>
18734     <xs:complexType name="TaskManagementHvacRelatedElementsType">
18735         <xs:sequence>
18736             <xs:element minOccurs="0" name="overrunId" type="ns_p:ElementTagType"/>
18737         </xs:sequence>
18738     </xs:complexType>
18739     <xs:complexType name="TaskManagementLoadControlRelatedType">
18740         <xs:sequence>
18741             <xs:element minOccurs="0" name="eventId" type="ns_p:LoadControlEventIdType"/>
18742         </xs:sequence>
18743     </xs:complexType>
18744     <xs:complexType name="TaskManagementLoadControlRelatedElementsType">
18745         <xs:sequence>
18746             <xs:element minOccurs="0" name="eventId" type="ns_p:ElementTagType"/>
18747         </xs:sequence>
18748     </xs:complexType>
18749     <xs:complexType name="TaskManagementPowerSequencesRelatedType">
18750         <xs:sequence>
18751             <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
18752         </xs:sequence>
18753     </xs:complexType>
18754     <xs:complexType name="TaskManagementPowerSequencesRelatedElementsType">
18755         <xs:sequence>
18756             <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
18757         </xs:sequence>
18758     </xs:complexType>
18759     <xs:complexType name="TaskManagementSmartEnergyManagementPsRelatedType">
18760         <xs:sequence>
18761             <xs:element minOccurs="0" name="sequenceId" type="ns_p:PowerSequenceIdType"/>
18762         </xs:sequence>
18763     </xs:complexType>
18764     <xs:complexType name="TaskManagementSmartEnergyManagementPsRelatedElementsType">
18765         <xs:sequence>
18766             <xs:element minOccurs="0" name="sequenceId" type="ns_p:ElementTagType"/>
18767         </xs:sequence>
18768     </xs:complexType>
18769     <xs:complexType name="TaskManagementJobDataType">
18770         <xs:sequence>
18771             <xs:element minOccurs="0" name="jobId" type="ns_p:TaskManagementJobIdType"/>
18772             <xs:element name="timestamp" type="ns_p:AbsoluteOrRelativeTimeType"
18773 minOccurs="0"/>
18774             <xs:element minOccurs="0" name="jobState" type="ns_p:TaskManagementJobStateType"/>
18775             <xs:element name="elapsedTime" type="xs:duration" minOccurs="0"/>
18776             <xs:element name="remainingTime" type="xs:duration" minOccurs="0"/>
18777         </xs:sequence>
18778     </xs:complexType>
18779     <xs:element name="taskManagementJobData" type="ns_p:TaskManagementJobDataType"/>
18780     <xs:complexType name="TaskManagementJobDataElementsType">
18781         <xs:sequence>
18782             <xs:element minOccurs="0" name="jobId" type="ns_p:ElementTagType"/>
18783             <xs:element name="timestamp" minOccurs="0" type="ns_p:ElementTagType"/>
18784             <xs:element minOccurs="0" name="jobState" type="ns_p:ElementTagType"/>
18785             <xs:element name="elapsedTime" minOccurs="0" type="ns_p:ElementTagType"/>
18786             <xs:element name="remainingTime" minOccurs="0" type="ns_p:ElementTagType"/>
18787         </xs:sequence>
18788     </xs:complexType>
18789     <xs:element name="taskManagementJobDataElements"
18790 type="ns_p:TaskManagementJobDataElementsType"/>
18791     <xs:complexType name="TaskManagementJobListDataType">
18792         <xs:sequence>
18793             <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:taskManagementJobData"/>
18794         </xs:sequence>
18795     </xs:complexType>
18796     <xs:element name="taskManagementJobListData" type="ns_p:TaskManagementJobListDataType"/>
18797     <xs:complexType name="TaskManagementJobListDataSelectorsType">
18798         <xs:sequence>
18799             <xs:element minOccurs="0" name="jobId" type="ns_p:TaskManagementJobIdType"/>
18800             <xs:element minOccurs="0" name="jobState" type="ns_p:TaskManagementJobStateType"/>
18801         </xs:sequence>
18802     </xs:complexType>
18803     <xs:element name="taskManagementJobListDataSelectors"
18804 type="ns_p:TaskManagementJobListDataSelectorsType"/>
18805     <xs:complexType name="TaskManagementJobRelationDataType">
18806         <xs:sequence>
18807             <xs:element minOccurs="0" name="jobId" type="ns_p:TaskManagementJobIdType"/>

```



```
18808         <xs:element minOccurs="0" name="directControlRelated"
18809 type="ns_p:TaskManagementDirectControlRelatedType"/>
18810         <xs:element minOccurs="0" name="hvacRelated"
18811 type="ns_p:TaskManagementHvacRelatedType"/>
18812         <xs:element minOccurs="0" name="loadControlRelated"
18813 type="ns_p:TaskManagementLoadControlRelatedType"/>
18814         <xs:element minOccurs="0" name="powerSequencesRelated"
18815 type="ns_p:TaskManagementPowerSequencesRelatedType"/>
18816         <xs:element minOccurs="0" name="smartEnergyManagementPsRelated"
18817 type="ns_p:TaskManagementSmartEnergyManagementPsRelatedType"/>
18818     </xs:sequence>
18819 </xs:complexType>
18820     <xs:element name="taskManagementJobRelationData"
18821 type="ns_p:TaskManagementJobRelationDataType"/>
18822     <xs:complexType name="TaskManagementJobRelationDataElementsType">
18823         <xs:sequence>
18824             <xs:element minOccurs="0" name="jobId" type="ns_p:ElementTagType"/>
18825             <xs:element minOccurs="0" name="directControlRelated"
18826 type="ns_p:TaskManagementDirectControlRelatedElementsType"/>
18827             <xs:element minOccurs="0" name="hvacRelated"
18828 type="ns_p:TaskManagementHvacRelatedElementsType"/>
18829             <xs:element minOccurs="0" name="loadControlRelated"
18830 type="ns_p:TaskManagementLoadControlRelatedElementsType"/>
18831             <xs:element minOccurs="0" name="powerSequencesRelated"
18832 type="ns_p:TaskManagementPowerSequencesRelatedElementsType"/>
18833             <xs:element minOccurs="0" name="smartEnergyManagementPsRelated"
18834 type="ns_p:TaskManagementSmartEnergyManagementPsRelatedElementsType"/>
18835         </xs:sequence>
18836     </xs:complexType>
18837     <xs:element name="taskManagementJobRelationDataElements"
18838 type="ns_p:TaskManagementJobRelationDataElementsType"/>
18839     <xs:complexType name="TaskManagementJobRelationListDataType">
18840         <xs:sequence>
18841             <xs:element maxOccurs="unbounded" minOccurs="0"
18842 ref="ns_p:taskManagementJobRelationData"/>
18843         </xs:sequence>
18844     </xs:complexType>
18845     <xs:element name="taskManagementJobRelationListData"
18846 type="ns_p:TaskManagementJobRelationListDataType"/>
18847     <xs:complexType name="TaskManagementJobRelationListDataSelectorsType">
18848         <xs:sequence>
18849             <xs:element minOccurs="0" name="jobId" type="ns_p:TaskManagementJobIdType"/>
18850         </xs:sequence>
18851     </xs:complexType>
18852     <xs:element name="taskManagementJobRelationListDataSelectors"
18853 type="ns_p:TaskManagementJobRelationListDataSelectorsType"/>
18854     <xs:complexType name="TaskManagementJobDescriptionDataType">
18855         <xs:sequence>
18856             <xs:element minOccurs="0" name="jobId" type="ns_p:TaskManagementJobIdType"/>
18857             <xs:element minOccurs="0" name="jobSource"
18858 type="ns_p:TaskManagementJobSourceType"/>
18859             <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
18860             <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
18861         </xs:sequence>
18862     </xs:complexType>
18863     <xs:element name="taskManagementJobDescriptionData"
18864 type="ns_p:TaskManagementJobDescriptionDataType"/>
18865     <xs:complexType name="TaskManagementJobDescriptionDataElementsType">
18866         <xs:sequence>
18867             <xs:element minOccurs="0" name="jobId" type="ns_p:ElementTagType"/>
18868             <xs:element minOccurs="0" name="jobSource" type="ns_p:ElementTagType"/>
18869             <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
18870             <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
18871         </xs:sequence>
18872     </xs:complexType>
18873     <xs:element name="taskManagementJobDescriptionDataElements"
18874 type="ns_p:TaskManagementJobDescriptionDataElementsType"/>
18875     <xs:complexType name="TaskManagementJobDescriptionListDataType">
18876         <xs:sequence>
18877             <xs:element maxOccurs="unbounded" minOccurs="0"
18878 ref="ns_p:taskManagementJobDescriptionData"/>
18879         </xs:sequence>
18880     </xs:complexType>
18881     <xs:element name="taskManagementJobDescriptionListData"
18882 type="ns_p:TaskManagementJobDescriptionListDataType"/>
18883     <xs:complexType name="TaskManagementJobDescriptionListDataSelectorsType">
18884         <xs:sequence>
18885             <xs:element minOccurs="0" name="jobId" type="ns_p:TaskManagementJobIdType"/>
```

```

18886         <xs:element minOccurs="0" name="jobSource"
18887 type="ns_p:TaskManagementJobSourceType"/>
18888     </xs:sequence>
18889 </xs:complexType>
18890 <xs:element name="taskManagementJobDescriptionListDataSelectors"
18891 type="ns_p:TaskManagementJobDescriptionListDataSelectorsType"/>
18892 <xs:complexType name="TaskManagementOverviewDataType">
18893     <xs:sequence>
18894         <xs:element minOccurs="0" name="remoteControllable" type="xs:boolean"/>
18895         <xs:element minOccurs="0" name="jobsActive" type="xs:boolean"/>
18896     </xs:sequence>
18897 </xs:complexType>
18898 <xs:element name="taskManagementOverviewData" type="ns_p:TaskManagementOverviewDataType"/>
18899 <xs:complexType name="TaskManagementOverviewDataElementsType">
18900     <xs:sequence>
18901         <xs:element minOccurs="0" name="remoteControllable" type="ns_p:ElementTagType"/>
18902         <xs:element minOccurs="0" name="jobsActive" type="ns_p:ElementTagType"/>
18903     </xs:sequence>
18904 </xs:complexType>
18905 <xs:element name="taskManagementOverviewDataElements"
18906 type="ns_p:TaskManagementOverviewDataElementsType"/>
18907 </xs:schema>
18908
18909

```

18910

18911 A.2.26 Threshold

18912 File name: EEBus_SPINE_TS_Threshold.xsd

```

18913 <?xml version="1.0" encoding="UTF-8"?>
18914 <!--
18915     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
18916     Version 1.1.1
18917     2018-12-21
18918     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
18919     Source: https://www.eebus.org/en/specifications/
18920 -->
18921 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
18922 xmlns:xs="http://www.w3.org/2001/XMLSchema"
18923 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
18924 blockDefault="#all" elementFormDefault="qualified">
18925     <xs:annotation>
18926         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
18927 e.V. All rights reserved.</xs:documentation>
18928     </xs:annotation>
18929     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
18930     <xs:simpleType name="ThresholdIdType">
18931         <xs:restriction base="xs:unsignedInt"/>
18932     </xs:simpleType>
18933     <xs:simpleType name="ThresholdTypeEnumType">
18934         <xs:union memberTypes="ns_p:ThresholdTypeEnumType ns_p:EnumExtendType"/>
18935     </xs:simpleType>
18936     <xs:simpleType name="ThresholdTypeEnumType">
18937         <xs:restriction base="xs:string">
18938             <xs:enumeration value="goodAbove"/>
18939             <xs:enumeration value="badAbove"/>
18940             <xs:enumeration value="goodBelow"/>
18941             <xs:enumeration value="badBelow"/>
18942             <xs:enumeration value="minValueThreshold"/>
18943             <xs:enumeration value="maxValueThreshold"/>
18944             <xs:enumeration value="minValueThresholdExtreme"/>
18945             <xs:enumeration value="maxValueThresholdExtreme"/>
18946             <xs:enumeration value="sagThreshold"/>
18947             <xs:enumeration value="swellThreshold"/>
18948         </xs:restriction>
18949     </xs:simpleType>
18950     <xs:complexType name="ThresholdDataType">
18951         <xs:sequence>
18952             <xs:element name="thresholdId" minOccurs="0" type="ns_p:ThresholdIdType"/>
18953             <xs:element minOccurs="0" name="thresholdValue" type="ns_p:ScaledNumberType"/>
18954         </xs:sequence>
18955     </xs:complexType>
18956     <xs:element name="thresholdData" type="ns_p:ThresholdDataType"/>
18957     <xs:complexType name="ThresholdDataElementsType">

```

```
18959     <xs:sequence>
18960         <xs:element name="thresholdId" minOccurs="0" type="ns_p:ElementTagType"/>
18961         <xs:element minOccurs="0" name="thresholdValue"
18962 type="ns_p:ScaledNumberElementsType"/>
18963     </xs:sequence>
18964 </xs:complexType>
18965 <xs:element name="thresholdDataElements" type="ns_p:ThresholdDataElementsType"/>
18966 <xs:complexType name="ThresholdListDataType">
18967     <xs:sequence>
18968         <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:thresholdData"/>
18969     </xs:sequence>
18970 </xs:complexType>
18971 <xs:element name="thresholdListData" type="ns_p:ThresholdListDataType"/>
18972 <xs:complexType name="ThresholdListDataSelectorsType">
18973     <xs:sequence>
18974         <xs:element name="thresholdId" minOccurs="0" type="ns_p:ThresholdIdType"/>
18975     </xs:sequence>
18976 </xs:complexType>
18977 <xs:element name="thresholdListDataSelectors" type="ns_p:ThresholdListDataSelectorsType"/>
18978 <xs:complexType name="ThresholdConstraintsDataType">
18979     <xs:sequence>
18980         <xs:element minOccurs="0" name="thresholdId" type="ns_p:ThresholdIdType"/>
18981         <xs:element minOccurs="0" name="thresholdRangeMin" type="ns_p:ScaledNumberType"/>
18982         <xs:element minOccurs="0" name="thresholdRangeMax" type="ns_p:ScaledNumberType"/>
18983         <xs:element minOccurs="0" name="thresholdStepSize" type="ns_p:ScaledNumberType"/>
18984     </xs:sequence>
18985 </xs:complexType>
18986 <xs:element name="thresholdConstraintsData" type="ns_p:ThresholdConstraintsDataType"/>
18987 <xs:complexType name="ThresholdConstraintsDataElementsType">
18988     <xs:sequence>
18989         <xs:element minOccurs="0" name="thresholdId" type="ns_p:ElementTagType"/>
18990         <xs:element minOccurs="0" name="thresholdRangeMin"
18991 type="ns_p:ScaledNumberElementsType"/>
18992         <xs:element minOccurs="0" name="thresholdRangeMax"
18993 type="ns_p:ScaledNumberElementsType"/>
18994         <xs:element minOccurs="0" name="thresholdStepSize"
18995 type="ns_p:ScaledNumberElementsType"/>
18996     </xs:sequence>
18997 </xs:complexType>
18998 <xs:element name="thresholdConstraintsDataElements"
18999 type="ns_p:ThresholdConstraintsDataElementsType"/>
19000 <xs:complexType name="ThresholdConstraintsListDataType">
19001     <xs:sequence>
19002         <xs:element maxOccurs="unbounded" minOccurs="0"
19003 ref="ns_p:thresholdConstraintsData"/>
19004     </xs:sequence>
19005 </xs:complexType>
19006 <xs:element name="thresholdConstraintsListData"
19007 type="ns_p:ThresholdConstraintsListDataType"/>
19008 <xs:complexType name="ThresholdConstraintsListDataSelectorsType">
19009     <xs:sequence>
19010         <xs:element name="thresholdId" minOccurs="0" type="ns_p:ThresholdIdType"/>
19011     </xs:sequence>
19012 </xs:complexType>
19013 <xs:element name="thresholdConstraintsListDataSelectors"
19014 type="ns_p:ThresholdConstraintsListDataSelectorsType"/>
19015 <xs:complexType name="ThresholdDescriptionDataType">
19016     <xs:sequence>
19017         <xs:element name="thresholdId" minOccurs="0" type="ns_p:ThresholdIdType"/>
19018         <xs:element name="thresholdType" minOccurs="0" type="ns_p:ThresholdType"/>
19019         <xs:element minOccurs="0" name="unit" type="ns_p:UnitOfMeasurementType"/>
19020         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
19021         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
19022         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
19023     </xs:sequence>
19024 </xs:complexType>
19025 <xs:element name="thresholdDescriptionData" type="ns_p:ThresholdDescriptionDataType"/>
19026 <xs:complexType name="ThresholdDescriptionDataElementsType">
19027     <xs:sequence>
19028         <xs:element name="thresholdId" minOccurs="0" type="ns_p:ElementTagType"/>
19029         <xs:element name="thresholdType" minOccurs="0" type="ns_p:ElementTagType"/>
19030         <xs:element minOccurs="0" name="unit" type="ns_p:ElementTagType"/>
19031         <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
19032         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
19033         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
19034     </xs:sequence>
19035 </xs:complexType>
```

```

19036     <xs:element name="thresholdDescriptionDataElements"
19037 type="ns_p:ThresholdDescriptionDataElementsType"/>
19038     <xs:complexType name="ThresholdDescriptionListDataType">
19039       <xs:sequence>
19040         <xs:element maxOccurs="unbounded" minOccurs="0"
19041 ref="ns_p:thresholdDescriptionData"/>
19042       </xs:sequence>
19043     </xs:complexType>
19044     <xs:element name="thresholdDescriptionListData"
19045 type="ns_p:ThresholdDescriptionListDataType"/>
19046     <xs:complexType name="ThresholdDescriptionListDataSelectorsType">
19047       <xs:sequence>
19048         <xs:element name="thresholdId" minOccurs="0" type="ns_p:ThresholdIdType"/>
19049         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
19050       </xs:sequence>
19051     </xs:complexType>
19052     <xs:element name="thresholdDescriptionListDataSelectors"
19053 type="ns_p:ThresholdDescriptionListDataSelectorsType"/>
19054   </xs:schema>
19055
19056

```

19057

19058 A.2.27 TimeInformation

19059 File name: EEBus_SPINE_TS_TimeInformation.xsd

```

19060
19061 <?xml version="1.0" encoding="UTF-8"?>
19062 <!--
19063   Smart Premises Interoperable Neutral-Message Exchange (SPINE)
19064   Version 1.1.1
19065   2018-12-21
19066   Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
19067   Source: https://www.eebus.org/en/specifications/
19068 -->
19069 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
19070 xmlns:xs="http://www.w3.org/2001/XMLSchema"
19071 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1" blockDefault="#all"
19072 elementFormDefault="qualified">
19073   <xs:annotation>
19074     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
19075 e.V. All rights reserved.</xs:documentation>
19076   </xs:annotation>
19077   <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
19078   <xs:complexType name="TimeInformationDataType">
19079     <xs:sequence>
19080       <xs:element minOccurs="0" name="utc" type="xs:dateTime"/>
19081       <xs:element minOccurs="0" name="utcOffset" type="xs:duration"/>
19082       <xs:element minOccurs="0" name="dayOfWeek" type="ns_p:DayOfWeekType"/>
19083       <xs:element minOccurs="0" name="calendarWeek" type="ns_p:CalendarWeekType"/>
19084     </xs:sequence>
19085   </xs:complexType>
19086   <xs:element name="timeInformationData" type="ns_p:TimeInformationDataType"/>
19087   <xs:complexType name="TimeInformationDataElementsType">
19088     <xs:sequence>
19089       <xs:element minOccurs="0" name="utc" type="ns_p:ElementTagType"/>
19090       <xs:element minOccurs="0" name="utcOffset" type="ns_p:ElementTagType"/>
19091       <xs:element minOccurs="0" name="dayOfWeek" type="ns_p:ElementTagType"/>
19092       <xs:element minOccurs="0" name="calendarWeek" type="ns_p:ElementTagType"/>
19093     </xs:sequence>
19094   </xs:complexType>
19095   <xs:element name="timeInformationDataElements"
19096 type="ns_p:TimeInformationDataElementsType"/>
19097   <xs:complexType name="TimeDistributorDataType">
19098     <xs:sequence>
19099       <xs:element minOccurs="0" name="isTimeDistributor" type="xs:boolean"/>
19100       <xs:element minOccurs="0" name="distributorPriority" type="xs:unsignedInt"/>
19101     </xs:sequence>
19102   </xs:complexType>
19103   <xs:element name="timeDistributorData" type="ns_p:TimeDistributorDataType"/>
19104   <xs:complexType name="TimeDistributorDataElementsType">
19105     <xs:sequence>
19106       <xs:element minOccurs="0" name="isTimeDistributor" type="ns_p:ElementTagType"/>
19107       <xs:element minOccurs="0" name="distributorPriority" type="ns_p:ElementTagType"/>
19108     </xs:sequence>

```

```

19109     </xs:complexType>
19110     <xs:element name="timeDistributorDataElements"
19111 type="ns_p:TimeDistributorDataElementsType"/>
19112     <xs:complexType name="TimePrecisionDataType">
19113       <xs:sequence>
19114         <xs:element minOccurs="0" name="isSynchronised" type="xs:boolean"/>
19115         <xs:element minOccurs="0" name="lastSyncAt" type="xs:dateTime"/>
19116         <xs:element minOccurs="0" name="clockDrift" type="xs:integer"/>
19117       </xs:sequence>
19118     </xs:complexType>
19119     <xs:element name="timePrecisionData" type="ns_p:TimePrecisionDataType"/>
19120     <xs:complexType name="TimePrecisionDataElementsType">
19121       <xs:sequence>
19122         <xs:element minOccurs="0" name="isSynchronised" type="ns_p:ElementTagType"/>
19123         <xs:element minOccurs="0" name="lastSyncAt" type="ns_p:ElementTagType"/>
19124         <xs:element minOccurs="0" name="clockDrift" type="ns_p:ElementTagType"/>
19125       </xs:sequence>
19126     </xs:complexType>
19127     <xs:element name="timePrecisionDataElements" type="ns_p:TimePrecisionDataElementsType"/>
19128     <xs:complexType name="TimeDistributorEnquiryCallType"/>
19129     <xs:element name="timeDistributorEnquiryCall" type="ns_p:TimeDistributorEnquiryCallType"/>
19130     <xs:complexType name="TimeDistributorEnquiryCallElementsType"/>
19131     <xs:element name="timeDistributorEnquiryCallElements"
19132 type="ns_p:TimeDistributorEnquiryCallElementsType"/>
19133   </xs:schema>
19134
19135

```

19136

19137 A.2.28 TimeSeries

19138 File name: EEBus_SPINE_TS_TimeSeries.xsd

```

19139
19140 <?xml version="1.0" encoding="UTF-8"?>
19141 <!--
19142   Smart Premises Interoperable Neutral-Message Exchange (SPINE)
19143   Version 1.1.1
19144   2018-12-21
19145   Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
19146   Source: https://www.eebus.org/en/specifications/
19147 -->
19148 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
19149 xmlns:xs="http://www.w3.org/2001/XMLSchema"
19150 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
19151 blockDefault="#all" elementFormDefault="qualified">
19152   <xs:annotation>
19153     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
19154 e.V. All rights reserved.</xs:documentation>
19155   </xs:annotation>
19156   <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
19157   <xs:include schemaLocation="EEBus_SPINE_TS_Measurement.xsd"/>
19158   <xs:simpleType name="TimeSeriesIdType">
19159     <xs:restriction base="xs:unsignedInt"/>
19160   </xs:simpleType>
19161   <xs:simpleType name="TimeSeriesSlotIdType">
19162     <xs:restriction base="xs:unsignedInt"/>
19163   </xs:simpleType>
19164   <xs:simpleType name="TimeSeriesSlotCountType">
19165     <xs:restriction base="ns_p:TimeSeriesSlotIdType"/>
19166   </xs:simpleType>
19167   <xs:simpleType name="TimeSeriesTypeType">
19168     <xs:union memberTypes="ns_p:TimeSeriesTypeEnumType ns_p:EnumExtendType"/>
19169   </xs:simpleType>
19170   <xs:simpleType name="TimeSeriesTypeEnumType">
19171     <xs:restriction base="xs:string">
19172       <xs:enumeration value="plan"/>
19173       <xs:enumeration value="singleDemand"/>
19174       <xs:enumeration value="constraints"/>
19175     </xs:restriction>
19176   </xs:simpleType>
19177   <xs:complexType name="TimeSeriesSlotType">
19178     <xs:sequence>
19179       <xs:element minOccurs="0" name="timeSeriesSlotId"
19180 type="ns_p:TimeSeriesSlotIdType"/>
19181       <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodType"/>

```

```
19182         <xs:element minOccurs="0" name="duration" type="xs:duration"/>
19183         <xs:element minOccurs="0" name="recurrenceInformation"
19184 type="ns_p:AbsoluteOrRecurringTimeType"/>
19185         <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberType"/>
19186         <xs:element minOccurs="0" name="minValue" type="ns_p:ScaledNumberType"/>
19187         <xs:element minOccurs="0" name="maxValue" type="ns_p:ScaledNumberType"/>
19188     </xs:sequence>
19189 </xs:complexType>
19190 <xs:complexType name="TimeSeriesSlotElementsType">
19191     <xs:sequence>
19192         <xs:element minOccurs="0" name="timeSeriesSlotId" type="ns_p:ElementTagType"/>
19193         <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodElementsType"/>
19194         <xs:element minOccurs="0" name="duration" type="ns_p:ElementTagType"/>
19195         <xs:element minOccurs="0" name="recurrenceInformation"
19196 type="ns_p:AbsoluteOrRecurringTimeElementsType"/>
19197         <xs:element minOccurs="0" name="value" type="ns_p:ElementTagType"/>
19198         <xs:element minOccurs="0" name="minValue" type="ns_p:ElementTagType"/>
19199         <xs:element minOccurs="0" name="maxValue" type="ns_p:ElementTagType"/>
19200     </xs:sequence>
19201 </xs:complexType>
19202 <xs:complexType name="TimeSeriesDataType">
19203     <xs:sequence>
19204         <xs:element minOccurs="0" name="timeSeriesId" type="ns_p:TimeSeriesIdType"/>
19205         <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodType"/>
19206         <xs:element minOccurs="0" name="timeSeriesSlot"
19207 type="ns_p:TimeSeriesSlotType"/>
19208     </xs:sequence>
19209 </xs:complexType>
19210 <xs:element name="timeSeriesData" type="ns_p:TimeSeriesDataType"/>
19211 <xs:complexType name="TimeSeriesDataElementsType">
19212     <xs:sequence>
19213         <xs:element minOccurs="0" name="timeSeriesId" type="ns_p:ElementTagType"/>
19214         <xs:element minOccurs="0" name="timePeriod" type="ns_p:TimePeriodElementsType"/>
19215         <xs:element minOccurs="0" name="timeSeriesSlot"
19216 type="ns_p:TimeSeriesSlotElementsType"/>
19217     </xs:sequence>
19218 </xs:complexType>
19219 <xs:element name="timeSeriesDataElements" type="ns_p:TimeSeriesDataElementsType"/>
19220 <xs:complexType name="TimeSeriesListDataType">
19221     <xs:sequence>
19222         <xs:element minOccurs="0" name="timeSeriesData" ref="ns_p:timeSeriesData"/>
19223     </xs:sequence>
19224 </xs:complexType>
19225 <xs:element name="timeSeriesListData" type="ns_p:TimeSeriesListDataType"/>
19226 <xs:complexType name="TimeSeriesListDataSelectorsType">
19227     <xs:sequence>
19228         <xs:element minOccurs="0" name="timeSeriesId" type="ns_p:TimeSeriesIdType"/>
19229         <xs:element minOccurs="0" name="timeSeriesSlotId"
19230 type="ns_p:TimeSeriesSlotIdType"/>
19231     </xs:sequence>
19232 </xs:complexType>
19233 <xs:element name="timeSeriesListDataSelectors"
19234 type="ns_p:TimeSeriesListDataSelectorsType"/>
19235 <xs:complexType name="TimeSeriesDescriptionDataType">
19236     <xs:sequence>
19237         <xs:element minOccurs="0" name="timeSeriesId" type="ns_p:TimeSeriesIdType"/>
19238         <xs:element minOccurs="0" name="timeSeriesType" type="ns_p:TimeSeriesTypeType"/>
19239         <xs:element minOccurs="0" name="timeSeriesWriteable" type="xs:boolean"/>
19240         <xs:element minOccurs="0" name="updateRequired" type="xs:boolean"/>
19241         <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
19242         <xs:element minOccurs="0" name="currency" type="ns_p:CurrencyType"/>
19243         <xs:element minOccurs="0" name="unit" type="ns_p:UnitOfMeasurementType"/>
19244         <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
19245         <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
19246         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
19247     </xs:sequence>
19248 </xs:complexType>
19249 <xs:element name="timeSeriesDescriptionData" type="ns_p:TimeSeriesDescriptionDataType"/>
19250 <xs:complexType name="TimeSeriesDescriptionDataElementsType">
19251     <xs:sequence>
19252         <xs:element minOccurs="0" name="timeSeriesId" type="ns_p:ElementTagType"/>
19253         <xs:element minOccurs="0" name="timeSeriesType" type="ns_p:ElementTagType"/>
19254         <xs:element minOccurs="0" name="timeSeriesWriteable" type="ns_p:ElementTagType"/>
19255         <xs:element minOccurs="0" name="updateRequired" type="ns_p:ElementTagType"/>
19256         <xs:element minOccurs="0" name="measurementId" type="ns_p:ElementTagType"/>
19257         <xs:element minOccurs="0" name="currency" type="ns_p:ElementTagType"/>
19258         <xs:element minOccurs="0" name="unit" type="ns_p:ElementTagType"/>
19259         <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
```

```
19260         <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
19261         <xs:element minOccurs="0" name="scopeType" type="ns_p:ElementTagType"/>
19262     </xs:sequence>
19263 </xs:complexType>
19264 <xs:element name="timeSeriesDescriptionDataElements"
19265 type="ns_p:TimeSeriesDescriptionDataElementsType"/>
19266 <xs:complexType name="TimeSeriesDescriptionListDataType">
19267     <xs:sequence>
19268         <xs:element maxOccurs="unbounded" minOccurs="0"
19269 ref="ns_p:timeSeriesDescriptionData"/>
19270     </xs:sequence>
19271 </xs:complexType>
19272 <xs:element name="timeSeriesDescriptionListData"
19273 type="ns_p:TimeSeriesDescriptionListDataType"/>
19274 <xs:complexType name="TimeSeriesDescriptionListDataSelectorsType">
19275     <xs:sequence>
19276         <xs:element minOccurs="0" name="timeSeriesId" type="ns_p:TimeSeriesIdType"/>
19277         <xs:element minOccurs="0" name="timeSeriesType" type="ns_p:TimeSeriesTypeType"/>
19278         <xs:element minOccurs="0" name="measurementId" type="ns_p:MeasurementIdType"/>
19279         <xs:element minOccurs="0" name="scopeType" type="ns_p:ScopeTypeType"/>
19280     </xs:sequence>
19281 </xs:complexType>
19282 <xs:element name="timeSeriesDescriptionListDataSelectors"
19283 type="ns_p:TimeSeriesDescriptionListDataSelectorsType"/>
19284 <xs:complexType name="TimeSeriesConstraintsDataType">
19285     <xs:sequence>
19286         <xs:element minOccurs="0" name="timeSeriesId" type="ns_p:TimeSeriesIdType"/>
19287         <xs:element minOccurs="0" name="slotCountMin"
19288 type="ns_p:TimeSeriesSlotCountType"/>
19289         <xs:element minOccurs="0" name="slotCountMax"
19290 type="ns_p:TimeSeriesSlotCountType"/>
19291         <xs:element minOccurs="0" name="slotDurationMin" type="xs:duration"/>
19292         <xs:element minOccurs="0" name="slotDurationMax" type="xs:duration"/>
19293         <xs:element minOccurs="0" name="slotDurationStepSize" type="xs:duration"/>
19294         <xs:element minOccurs="0" name="earliestTimeSeriesStartTime"
19295 type="ns_p:AbsoluteOrRelativeTimeType"/>
19296         <xs:element minOccurs="0" name="latestTimeSeriesEndTime"
19297 type="ns_p:AbsoluteOrRelativeTimeType"/>
19298         <xs:element minOccurs="0" name="slotValueMin" type="ns_p:ScaledNumberType"/>
19299         <xs:element minOccurs="0" name="slotValueMax" type="ns_p:ScaledNumberType"/>
19300         <xs:element minOccurs="0" name="slotValueStepSize" type="ns_p:ScaledNumberType"/>
19301     </xs:sequence>
19302 </xs:complexType>
19303 <xs:element name="timeSeriesConstraintsData" type="ns_p:TimeSeriesConstraintsDataType"/>
19304 <xs:complexType name="TimeSeriesConstraintsDataElementsType">
19305     <xs:sequence>
19306         <xs:element minOccurs="0" name="timeSeriesId" type="ns_p:ElementTagType"/>
19307         <xs:element minOccurs="0" name="slotCountMin" type="ns_p:ElementTagType"/>
19308         <xs:element minOccurs="0" name="slotCountMax" type="ns_p:ElementTagType"/>
19309         <xs:element minOccurs="0" name="slotDurationMin" type="ns_p:ElementTagType"/>
19310         <xs:element minOccurs="0" name="slotDurationMax" type="ns_p:ElementTagType"/>
19311         <xs:element minOccurs="0" name="slotDurationStepSize" type="ns_p:ElementTagType"/>
19312         <xs:element minOccurs="0" name="earliestTimeSeriesStartTime"
19313 type="ns_p:ElementTagType"/>
19314         <xs:element minOccurs="0" name="latestTimeSeriesEndTime"
19315 type="ns_p:ElementTagType"/>
19316         <xs:element minOccurs="0" name="slotValueMin"
19317 type="ns_p:ScaledNumberElementsType"/>
19318         <xs:element minOccurs="0" name="slotValueMax"
19319 type="ns_p:ScaledNumberElementsType"/>
19320         <xs:element minOccurs="0" name="slotValueStepSize"
19321 type="ns_p:ScaledNumberElementsType"/>
19322     </xs:sequence>
19323 </xs:complexType>
19324 <xs:element name="timeSeriesConstraintsDataElements"
19325 type="ns_p:TimeSeriesConstraintsDataElementsType"/>
19326 <xs:complexType name="TimeSeriesConstraintsListDataType">
19327     <xs:sequence>
19328         <xs:element maxOccurs="unbounded" minOccurs="0"
19329 ref="ns_p:timeSeriesConstraintsData"/>
19330     </xs:sequence>
19331 </xs:complexType>
19332 <xs:element name="timeSeriesConstraintsListData"
19333 type="ns_p:TimeSeriesConstraintsListDataType"/>
19334 <xs:complexType name="TimeSeriesConstraintsListDataSelectorsType">
19335     <xs:sequence>
19336         <xs:element minOccurs="0" name="timeSeriesId" type="ns_p:TimeSeriesIdType"/>
19337     </xs:sequence>
```

```

19338     </xs:complexType>
19339     <xs:element name="timeSeriesConstraintsListDataSelectors"
19340     type="ns_p:TimeSeriesConstraintsListDataSelectorsType"/>
19341 </xs:schema>
19342
19343

```

19345 A.2.29 TimeTable

19346 File name: EEBus_SPINE_TS_TimeTable.xsd

```

19347 <?xml version="1.0" encoding="UTF-8"?>
19348 <!--
19349     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
19350     Version 1.1.1
19351     2018-12-21
19352     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
19353     Source: https://www.eebus.org/en/specifications/
19354 -->
19355 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
19356     xmlns:xs="http://www.w3.org/2001/XMLSchema"
19357     targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
19358     blockDefault="#all" elementFormDefault="qualified">
19359     <xs:annotation>
19360     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
19361     e.V. All rights reserved.</xs:documentation>
19362     </xs:annotation>
19363     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
19364     <xs:simpleType name="TimeTableIdType">
19365     <xs:restriction base="xs:unsignedInt"/>
19366     </xs:simpleType>
19367     <xs:simpleType name="TimeSlotIdType">
19368     <xs:restriction base="xs:unsignedInt"/>
19369     </xs:simpleType>
19370     <xs:simpleType name="TimeSlotCountType">
19371     <xs:restriction base="ns_p:TimeSlotIdType"/>
19372     </xs:simpleType>
19373     <xs:simpleType name="TimeSlotTimeModeType">
19374     <xs:union memberTypes="ns_p:TimeSlotTimeModeEnumType ns_p:EnumExtendType"/>
19375     </xs:simpleType>
19376     <xs:simpleType name="TimeSlotTimeModeEnumType">
19377     <xs:restriction base="xs:string">
19378     <xs:enumeration value="absolute"/>
19379     <xs:enumeration value="recurring"/>
19380     <xs:enumeration value="both"/>
19381     </xs:restriction>
19382     </xs:simpleType>
19383     <xs:complexType name="TimeTableDataType">
19384     <xs:sequence>
19385     <xs:element minOccurs="0" name="timeTableId" type="ns_p:TimeTableIdType"/>
19386     <xs:element minOccurs="0" name="timeSlotId" type="ns_p:TimeSlotIdType"/>
19387     <xs:element name="recurrenceInformation" minOccurs="0"
19388     type="ns_p:RecurrenceInformationType"/>
19389     <xs:element minOccurs="0" name="startTime"
19390     type="ns_p:AbsoluteOrRecurringTimeType"/>
19391     <xs:element minOccurs="0" name="endTime" type="ns_p:AbsoluteOrRecurringTimeType"/>
19392     </xs:sequence>
19393     </xs:complexType>
19394     <xs:element name="timeTableData" type="ns_p:TimeTableDataType"/>
19395     <xs:complexType name="TimeTableDataElementsType">
19396     <xs:sequence>
19397     <xs:element minOccurs="0" name="timeTableId" type="ns_p:ElementTagType"/>
19398     <xs:element minOccurs="0" name="timeSlotId" type="ns_p:ElementTagType"/>
19399     <xs:element minOccurs="0" name="recurrenceInformation"
19400     type="ns_p:RecurrenceInformationElementsType"/>
19401     <xs:element minOccurs="0" name="startTime"
19402     type="ns_p:AbsoluteOrRecurringTimeElementsType"/>
19403     <xs:element minOccurs="0" name="endTime"
19404     type="ns_p:AbsoluteOrRecurringTimeElementsType"/>
19405     </xs:sequence>
19406     </xs:complexType>
19407     <xs:element name="timeTableDataElements" type="ns_p:TimeTableDataElementsType"/>
19408     <xs:complexType name="TimeTableListDataType">
19409     <xs:sequence>
19410

```



```
19411      <xs:element maxOccurs="unbounded" minOccurs="0" ref="ns_p:timeTableData"/>
19412    </xs:sequence>
19413  </xs:complexType>
19414  <xs:element name="timeTableListData" type="ns_p:TimeTableListDataType"/>
19415  <xs:complexType name="TimeTableListDataSelectorsType">
19416    <xs:sequence>
19417      <xs:element minOccurs="0" name="timeTableId" type="ns_p:TimeTableIdType"/>
19418      <xs:element minOccurs="0" name="timeSlotId" type="ns_p:TimeSlotIdType"/>
19419    </xs:sequence>
19420  </xs:complexType>
19421  <xs:element name="timeTableListDataSelectors" type="ns_p:TimeTableListDataSelectorsType"/>
19422  <xs:complexType name="TimeTableConstraintsDataType">
19423    <xs:sequence>
19424      <xs:element minOccurs="0" name="timeTableId" type="xs:unsignedInt"/>
19425      <xs:element minOccurs="0" name="slotCountMin" type="ns_p:TimeSlotCountType"/>
19426      <xs:element minOccurs="0" name="slotCountMax" type="ns_p:TimeSlotCountType"/>
19427      <xs:element minOccurs="0" name="slotDurationMin" type="xs:duration"/>
19428      <xs:element minOccurs="0" name="slotDurationMax" type="xs:duration"/>
19429      <xs:element minOccurs="0" name="slotDurationStepSize" type="xs:duration"/>
19430      <xs:element minOccurs="0" name="slotShiftStepSize" type="xs:duration"/>
19431      <xs:element minOccurs="0" name="firstSlotBeginsAt" type="xs:time"/>
19432    </xs:sequence>
19433  </xs:complexType>
19434  <xs:element name="timeTableConstraintsData" type="ns_p:TimeTableConstraintsDataType"/>
19435  <xs:complexType name="TimeTableConstraintsDataElementsType">
19436    <xs:sequence>
19437      <xs:element minOccurs="0" name="timeTableId" type="ns_p:ElementTagType"/>
19438      <xs:element minOccurs="0" name="slotCountMin" type="ns_p:ElementTagType"/>
19439      <xs:element minOccurs="0" name="slotCountMax" type="ns_p:ElementTagType"/>
19440      <xs:element minOccurs="0" name="slotDurationMin" type="ns_p:ElementTagType"/>
19441      <xs:element minOccurs="0" name="slotDurationMax" type="ns_p:ElementTagType"/>
19442      <xs:element minOccurs="0" name="slotDurationStepSize" type="ns_p:ElementTagType"/>
19443      <xs:element minOccurs="0" name="slotShiftStepSize" type="ns_p:ElementTagType"/>
19444      <xs:element minOccurs="0" name="firstSlotBeginsAt" type="ns_p:ElementTagType"/>
19445    </xs:sequence>
19446  </xs:complexType>
19447  <xs:element name="timeTableConstraintsDataElements"
19448  type="ns_p:TimeTableConstraintsDataElementsType"/>
19449  <xs:complexType name="TimeTableConstraintsListDataType">
19450    <xs:sequence>
19451      <xs:element maxOccurs="unbounded" minOccurs="0"
19452  ref="ns_p:timeTableConstraintsData"/>
19453    </xs:sequence>
19454  </xs:complexType>
19455  <xs:element name="timeTableConstraintsListData"
19456  type="ns_p:TimeTableConstraintsListDataType"/>
19457  <xs:complexType name="TimeTableConstraintsListDataSelectorsType">
19458    <xs:sequence>
19459      <xs:element minOccurs="0" name="timeTableId" type="ns_p:TimeTableIdType"/>
19460    </xs:sequence>
19461  </xs:complexType>
19462  <xs:element name="timeTableConstraintsListDataSelectors"
19463  type="ns_p:TimeTableConstraintsListDataSelectorsType"/>
19464  <xs:complexType name="TimeTableDescriptionDataType">
19465    <xs:sequence>
19466      <xs:element minOccurs="0" name="timeTableId" type="xs:unsignedInt"/>
19467      <xs:element minOccurs="0" name="timeSlotCountChangeable" type="xs:boolean"/>
19468      <xs:element minOccurs="0" name="timeSlotTimesChangeable" type="xs:boolean"/>
19469      <xs:element minOccurs="0" name="timeSlotTimeMode"
19470  type="ns_p:TimeSlotTimeModeType"/>
19471      <xs:element minOccurs="0" name="label" type="ns_p:LabelType"/>
19472      <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
19473    </xs:sequence>
19474  </xs:complexType>
19475  <xs:element name="timeTableDescriptionData" type="ns_p:TimeTableDescriptionDataType"/>
19476  <xs:complexType name="TimeTableDescriptionDataElementsType">
19477    <xs:sequence>
19478      <xs:element minOccurs="0" name="timeTableId" type="ns_p:ElementTagType"/>
19479      <xs:element minOccurs="0" name="timeSlotCountChangeable"
19480  type="ns_p:ElementTagType"/>
19481      <xs:element minOccurs="0" name="timeSlotTimesChangeable"
19482  type="ns_p:ElementTagType"/>
19483      <xs:element minOccurs="0" name="timeSlotTimeMode" type="ns_p:ElementTagType"/>
19484      <xs:element minOccurs="0" name="label" type="ns_p:ElementTagType"/>
19485      <xs:element minOccurs="0" name="description" type="ns_p:ElementTagType"/>
19486    </xs:sequence>
19487  </xs:complexType>
```

```

19488     <xs:element name="timeTableDescriptionDataElements"
19489 type="ns_p:TimeTableDescriptionDataElementsType"/>
19490     <xs:complexType name="TimeTableDescriptionListDataType">
19491       <xs:sequence>
19492         <xs:element maxOccurs="unbounded" minOccurs="0"
19493 ref="ns_p:timeTableDescriptionData"/>
19494       </xs:sequence>
19495     </xs:complexType>
19496     <xs:element name="timeTableDescriptionListData"
19497 type="ns_p:TimeTableDescriptionListDataType"/>
19498     <xs:complexType name="TimeTableDescriptionListDataSelectorsType">
19499       <xs:sequence>
19500         <xs:element minOccurs="0" name="timeTableId" type="ns_p:TimeTableIdType"/>
19501       </xs:sequence>
19502     </xs:complexType>
19503     <xs:element name="timeTableDescriptionListDataSelectors"
19504 type="ns_p:TimeTableDescriptionListDataSelectorsType"/>
19505   </xs:schema>
19506
19507

```

19508

19509 A.2.30 UseCaseInformation

19510 File name: EEBus_SPINE_TS_UseCaseInformation.xsd

```

19511 <?xml version="1.0" encoding="UTF-8"?>
19512 <!--
19513   Smart Premises Interoperable Neutral-Message Exchange (SPINE)
19514   Version 1.1.1
19515   2018-12-21
19516   Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
19517   Source: https://www.eebus.org/en/specifications/
19518 -->
19519 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
19520 xmlns:xs="http://www.w3.org/2001/XMLSchema"
19521 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
19522 blockDefault="#all" elementFormDefault="qualified">
19523   <xs:annotation>
19524     <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
19525 e.V. All rights reserved.</xs:documentation>
19526   </xs:annotation>
19527   <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
19528   <xs:include schemaLocation="EEBus_SPINE_TS_Version.xsd"/>
19529   <xs:simpleType name="UseCaseActorType">
19530     <xs:union memberTypes="ns_p:UseCaseActorEnumType ns_p:EnumExtendType"/>
19531   </xs:simpleType>
19532   <xs:simpleType name="UseCaseActorEnumType">
19533     <xs:restriction base="xs:string"/>
19534   </xs:simpleType>
19535   <xs:simpleType name="UseCaseNameType">
19536     <xs:union memberTypes="ns_p:UseCaseNameEnumType ns_p:EnumExtendType"/>
19537   </xs:simpleType>
19538   <xs:simpleType name="UseCaseNameEnumType">
19539     <xs:restriction base="xs:string"/>
19540   </xs:simpleType>
19541   <xs:simpleType name="UseCaseScenarioSupportType">
19542     <xs:restriction base="xs:unsignedInt"/>
19543   </xs:simpleType>
19544   <xs:complexType name="UseCaseSupportType">
19545     <xs:sequence>
19546       <xs:element minOccurs="0" name="useCaseName" type="ns_p:UseCaseNameType"/>
19547       <xs:element minOccurs="0" name="useCaseVersion"
19548 type="ns_p:SpecificationVersionType"/>
19549       <xs:element minOccurs="0" name="useCaseAvailable" type="xs:boolean"/>
19550       <xs:element maxOccurs="unbounded" minOccurs="0" name="scenarioSupport"
19551 type="ns_p:UseCaseScenarioSupportType"/>
19552     </xs:sequence>
19553   </xs:complexType>
19554   <xs:complexType name="UseCaseSupportElementsType">
19555     <xs:sequence>
19556       <xs:element minOccurs="0" name="useCaseName" type="ns_p:ElementTagType"/>
19557       <xs:element minOccurs="0" name="useCaseVersion" type="ns_p:ElementTagType"/>
19558       <xs:element minOccurs="0" name="useCaseAvailable" type="ns_p:ElementTagType"/>
19559       <xs:element minOccurs="0" name="scenarioSupport" type="ns_p:ElementTagType"/>
19560

```

```

19561         </xs:sequence>
19562     </xs:complexType>
19563     <xs:complexType name="UseCaseSupportSelectorsType">
19564         <xs:sequence>
19565             <xs:element minOccurs="0" name="useCaseName" type="ns_p:UseCaseNameType"/>
19566             <xs:element minOccurs="0" name="useCaseVersion"
19567 type="ns_p:SpecificationVersionType"/>
19568             <xs:element minOccurs="0" name="scenarioSupport"
19569 type="ns_p:UseCaseScenarioSupportType"/>
19570         </xs:sequence>
19571     </xs:complexType>
19572     <xs:complexType name="UseCaseInformationDataType">
19573         <xs:sequence>
19574             <xs:element minOccurs="0" name="address" type="ns_p:FeatureAddressType"/>
19575             <xs:element minOccurs="0" name="actor" type="ns_p:UseCaseActorType"/>
19576             <xs:element maxOccurs="unbounded" minOccurs="0" name="useCaseSupport"
19577 type="ns_p:UseCaseSupportType"/>
19578         </xs:sequence>
19579     </xs:complexType>
19580     <xs:element name="useCaseInformationData" type="ns_p:UseCaseInformationDataType"/>
19581     <xs:complexType name="UseCaseInformationDataElementsType">
19582         <xs:sequence>
19583             <xs:element minOccurs="0" name="address" type="ns_p:FeatureAddressElementsType"/>
19584             <xs:element minOccurs="0" name="actor" type="ns_p:ElementTagType"/>
19585             <xs:element minOccurs="0" name="useCaseSupport"
19586 type="ns_p:UseCaseSupportElementsType"/>
19587         </xs:sequence>
19588     </xs:complexType>
19589     <xs:element name="useCaseInformationDataElements"
19590 type="ns_p:UseCaseInformationDataElementsType"/>
19591     <xs:complexType name="UseCaseInformationListDataType">
19592         <xs:sequence>
19593             <xs:element maxOccurs="unbounded" minOccurs="0"
19594 ref="ns_p:useCaseInformationData"/>
19595         </xs:sequence>
19596     </xs:complexType>
19597     <xs:element name="useCaseInformationListData" type="ns_p:UseCaseInformationListDataType"/>
19598     <xs:complexType name="UseCaseInformationListDataSelectorsType">
19599         <xs:sequence>
19600             <xs:element minOccurs="0" name="address" type="ns_p:FeatureAddressType"/>
19601             <xs:element minOccurs="0" name="actor" type="ns_p:UseCaseActorType"/>
19602             <xs:element minOccurs="0" name="useCaseSupport"
19603 type="ns_p:UseCaseSupportSelectorsType"/>
19604         </xs:sequence>
19605     </xs:complexType>
19606     <xs:element name="useCaseInformationListDataSelectors"
19607 type="ns_p:UseCaseInformationListDataSelectorsType"/>
19608 </xs:schema>
19609
19610

```

19611

19612 A.2.31 Version

19613 File name: EEBus_SPINE_TS_Version.xsd

```

19614
19615 <?xml version="1.0" encoding="UTF-8"?>
19616 <!--
19617     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
19618     Version 1.1.1
19619     2018-12-21
19620     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
19621     Source: https://www.eebus.org/en/specifications/
19622 -->
19623 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
19624 xmlns:xs="http://www.w3.org/2001/XMLSchema"
19625 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1" blockDefault="#all"
19626 elementFormDefault="qualified">
19627     <xs:annotation>
19628         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
19629 e.V. All rights reserved.</xs:documentation>
19630     </xs:annotation>
19631     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
19632     <xs:complexType name="SpecificationVersionDataType">
19633         <xs:simpleContent>

```

```

19634         <xs:extension base="ns_p:SpecificationVersionType"/>
19635     </xs:simpleContent>
19636 </xs:complexType>
19637 <xs:element name="specificationVersionData" type="ns_p:SpecificationVersionDataType"/>
19638 <xs:complexType name="SpecificationVersionDataElementsType"/>
19639 <xs:element name="specificationVersionDataElements"
19640 type="ns_p:SpecificationVersionDataElementsType"/>
19641 <xs:complexType name="SpecificationVersionListDataType">
19642     <xs:sequence>
19643         <xs:element maxOccurs="unbounded" minOccurs="0"
19644 ref="ns_p:specificationVersionData"/>
19645     </xs:sequence>
19646 </xs:complexType>
19647 <xs:element name="specificationVersionListData"
19648 type="ns_p:SpecificationVersionListDataType"/>
19649 <xs:complexType name="SpecificationVersionListDataSelectorsType"/>
19650 <xs:element name="specificationVersionListDataSelectors"
19651 type="ns_p:SpecificationVersionListDataSelectorsType"/>
19652 </xs:schema>
19653
19654

```

19656 A.3 Further XSDs

19657 A.3.1 CommandCommonDefinitions

19658 File name: EEBus_SPINE_TS_CommandCommonDefinitions.xsd

```

19659 <?xml version="1.0" encoding="UTF-8"?>
19660 <!--
19661     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
19662     Version 1.1.1
19663     2018-12-21
19664     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
19665     Source: https://www.eebus.org/en/specifications/
19666 -->
19667 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
19668 xmlns:xs="http://www.w3.org/2001/XMLSchema"
19669 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
19670 blockDefault="#all" elementFormDefault="qualified">
19671     <xs:annotation>
19672         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
19673 e.V. All rights reserved.</xs:documentation>
19674     </xs:annotation>
19675     <xs:include schemaLocation="EEBus_SPINE_TS_ActuatorLevel.xsd"/>
19676     <xs:include schemaLocation="EEBus_SPINE_TS_ActuatorSwitch.xsd"/>
19677     <xs:include schemaLocation="EEBus_SPINE_TS_Alarm.xsd"/>
19678     <xs:include schemaLocation="EEBus_SPINE_TS_Bill.xsd"/>
19679     <xs:include schemaLocation="EEBus_SPINE_TS_BindingManagement.xsd"/>
19680     <xs:include schemaLocation="EEBus_SPINE_TS_DataTunneling.xsd"/>
19681     <xs:include schemaLocation="EEBus_SPINE_TS_DeviceClassification.xsd"/>
19682     <xs:include schemaLocation="EEBus_SPINE_TS_DeviceConfiguration.xsd"/>
19683     <xs:include schemaLocation="EEBus_SPINE_TS_DeviceDiagnosis.xsd"/>
19684     <xs:include schemaLocation="EEBus_SPINE_TS_DirectControl.xsd"/>
19685     <xs:include schemaLocation="EEBus_SPINE_TS_ElectricalConnection.xsd"/>
19686     <xs:include schemaLocation="EEBus_SPINE_TS_HVAC.xsd"/>
19687     <xs:include schemaLocation="EEBus_SPINE_TS_Identification.xsd"/>
19688     <xs:include schemaLocation="EEBus_SPINE_TS_IncentiveTable.xsd"/>
19689     <xs:include schemaLocation="EEBus_SPINE_TS_LoadControl.xsd"/>
19690     <xs:include schemaLocation="EEBus_SPINE_TS_Measurement.xsd"/>
19691     <xs:include schemaLocation="EEBus_SPINE_TS_Messaging.xsd"/>
19692     <xs:include schemaLocation="EEBus_SPINE_TS_NetworkManagement.xsd"/>
19693     <xs:include schemaLocation="EEBus_SPINE_TS_NodeManagement.xsd"/>
19694     <xs:include schemaLocation="EEBus_SPINE_TS_OperatingConstraints.xsd"/>
19695     <xs:include schemaLocation="EEBus_SPINE_TS_PowerSequences.xsd"/>
19696     <xs:include schemaLocation="EEBus_SPINE_TS_Result.xsd"/>
19697     <xs:include schemaLocation="EEBus_SPINE_TS_Sensing.xsd"/>
19698     <xs:include schemaLocation="EEBus_SPINE_TS_Setpoint.xsd"/>
19699     <xs:include schemaLocation="EEBus_SPINE_TS_SmartEnergyManagementPs.xsd"/>
19700     <xs:include schemaLocation="EEBus_SPINE_TS_SubscriptionManagement.xsd"/>
19701     <xs:include schemaLocation="EEBus_SPINE_TS_SupplyCondition.xsd"/>
19702     <xs:include schemaLocation="EEBus_SPINE_TS_TariffInformation.xsd"/>
19703     <xs:include schemaLocation="EEBus_SPINE_TS_TaskManagement.xsd"/>
19704

```

```

19705 <xs:include schemaLocation="EEBus_SPINE_TS_Threshold.xsd"/>
19706 <xs:include schemaLocation="EEBus_SPINE_TS_TimeInformation.xsd"/>
19707 <xs:include schemaLocation="EEBus_SPINE_TS_TimeSeries.xsd"/>
19708 <xs:include schemaLocation="EEBus_SPINE_TS_TimeTable.xsd"/>
19709 <xs:include schemaLocation="EEBus_SPINE_TS_UseCaseInformation.xsd"/>
19710 <xs:include schemaLocation="EEBus_SPINE_TS_Version.xsd"/>
19711 <xs:group name="DataChoiceGroup">
19712   <xs:choice>
19713     <xs:element ref="ns_p:actuatorLevelData"/>
19714     <xs:element ref="ns_p:actuatorLevelDescriptionData"/>
19715     <xs:element ref="ns_p:actuatorSwitchData"/>
19716     <xs:element ref="ns_p:actuatorSwitchDescriptionData"/>
19717     <xs:element ref="ns_p:alarmListData"/>
19718     <xs:element ref="ns_p:billConstraintsListData"/>
19719     <xs:element ref="ns_p:billDescriptionListData"/>
19720     <xs:element ref="ns_p:billListData"/>
19721     <xs:element ref="ns_p:bindingManagementDeleteCall"/>
19722     <xs:element ref="ns_p:bindingManagementEntryListData"/>
19723     <xs:element ref="ns_p:bindingManagementRequestCall"/>
19724     <xs:element ref="ns_p:commodityListData"/>
19725     <xs:element ref="ns_p:dataTunnelingCall"/>
19726     <xs:element ref="ns_p:deviceClassificationManufacturerData"/>
19727     <xs:element ref="ns_p:deviceClassificationUserData"/>
19728     <xs:element ref="ns_p:deviceConfigurationKeyValueConstraintsListData"/>
19729     <xs:element ref="ns_p:deviceConfigurationKeyValueDescriptionListData"/>
19730     <xs:element ref="ns_p:deviceConfigurationKeyValueListData"/>
19731     <xs:element ref="ns_p:deviceDiagnosisHeartbeatData"/>
19732     <xs:element ref="ns_p:deviceDiagnosisServiceData"/>
19733     <xs:element ref="ns_p:deviceDiagnosisStateData"/>
19734     <xs:element ref="ns_p:directControlActivityListData"/>
19735     <xs:element ref="ns_p:directControlDescriptionData"/>
19736     <xs:element ref="ns_p:electricalConnectionDescriptionListData"/>
19737     <xs:element ref="ns_p:electricalConnectionParameterDescriptionListData"/>
19738     <xs:element ref="ns_p:electricalConnectionPermittedValueSetListData"/>
19739     <xs:element ref="ns_p:electricalConnectionStateListData"/>
19740     <xs:element ref="ns_p:hvacOperationModeDescriptionListData"/>
19741     <xs:element ref="ns_p:hvacOverrunDescriptionListData"/>
19742     <xs:element ref="ns_p:hvacOverrunListData"/>
19743     <xs:element ref="ns_p:hvacSystemFunctionDescriptionListData"/>
19744     <xs:element ref="ns_p:hvacSystemFunctionListData"/>
19745     <xs:element ref="ns_p:hvacSystemFunctionOperationModeRelationListData"/>
19746     <xs:element ref="ns_p:hvacSystemFunctionPowerSequenceRelationListData"/>
19747     <xs:element ref="ns_p:hvacSystemFunctionSetpointRelationListData"/>
19748     <xs:element ref="ns_p:identificationListData"/>
19749     <xs:element ref="ns_p:incentiveDescriptionListData"/>
19750     <xs:element ref="ns_p:incentiveListData"/>
19751     <xs:element ref="ns_p:incentiveTableConstraintsData"/>
19752     <xs:element ref="ns_p:incentiveTableData"/>
19753     <xs:element ref="ns_p:incentiveTableDescriptionData"/>
19754     <xs:element ref="ns_p:loadControlEventListData"/>
19755     <xs:element ref="ns_p:loadControlLimitConstraintsListData"/>
19756     <xs:element ref="ns_p:loadControlLimitDescriptionListData"/>
19757     <xs:element ref="ns_p:loadControlLimitListData"/>
19758     <xs:element ref="ns_p:loadControlNodeData"/>
19759     <xs:element ref="ns_p:loadControlStateListData"/>
19760     <xs:element ref="ns_p:measurementConstraintsListData"/>
19761     <xs:element ref="ns_p:measurementDescriptionListData"/>
19762     <xs:element ref="ns_p:measurementListData"/>
19763     <xs:element ref="ns_p:measurementThresholdRelationListData"/>
19764     <xs:element ref="ns_p:messagingListData"/>
19765     <xs:element ref="ns_p:networkManagementAbortCall"/>
19766     <xs:element ref="ns_p:networkManagementAddNodeCall"/>
19767     <xs:element ref="ns_p:networkManagementDeviceDescriptionListData"/>
19768     <xs:element ref="ns_p:networkManagementDiscoverCall"/>
19769     <xs:element ref="ns_p:networkManagementEntityDescriptionListData"/>
19770     <xs:element ref="ns_p:networkManagementFeatureDescriptionListData"/>
19771     <xs:element ref="ns_p:networkManagementJoiningModeData"/>
19772     <xs:element ref="ns_p:networkManagementModifyNodeCall"/>
19773     <xs:element ref="ns_p:networkManagementProcessStateData"/>
19774     <xs:element ref="ns_p:networkManagementRemoveNodeCall"/>
19775     <xs:element ref="ns_p:networkManagementReportCandidateData"/>
19776     <xs:element ref="ns_p:networkManagementScanNetworkCall"/>
19777     <xs:element ref="ns_p:nodeManagementBindingData"/>
19778     <xs:element ref="ns_p:nodeManagementBindingDeleteCall"/>
19779     <xs:element ref="ns_p:nodeManagementBindingRequestCall"/>
19780     <xs:element ref="ns_p:nodeManagementDestinationListData"/>
19781     <xs:element ref="ns_p:nodeManagementDetailedDiscoveryData"/>
19782     <xs:element ref="ns_p:nodeManagementSubscriptionData"/>

```

```

19783 <xs:element ref="ns_p:nodeManagementSubscriptionDeleteCall"/>
19784 <xs:element ref="ns_p:nodeManagementSubscriptionRequestCall"/>
19785 <xs:element ref="ns_p:nodeManagementUseCaseData"/>
19786 <xs:element ref="ns_p:operatingConstraintsDurationListData"/>
19787 <xs:element ref="ns_p:operatingConstraintsInterruptListData"/>
19788 <xs:element ref="ns_p:operatingConstraintsPowerDescriptionListData"/>
19789 <xs:element ref="ns_p:operatingConstraintsPowerLevelListData"/>
19790 <xs:element ref="ns_p:operatingConstraintsPowerRangeListData"/>
19791 <xs:element ref="ns_p:operatingConstraintsResumeImplicationListData"/>
19792 <xs:element ref="ns_p:powerSequenceAlternativesRelationListData"/>
19793 <xs:element ref="ns_p:powerSequenceDescriptionListData"/>
19794 <xs:element ref="ns_p:powerSequenceNodeScheduleInformationData"/>
19795 <xs:element ref="ns_p:powerSequencePriceCalculationRequestCall"/>
19796 <xs:element ref="ns_p:powerSequencePriceListData"/>
19797 <xs:element ref="ns_p:powerSequenceScheduleConfigurationRequestCall"/>
19798 <xs:element ref="ns_p:powerSequenceScheduleConstraintsListData"/>
19799 <xs:element ref="ns_p:powerSequenceScheduleListData"/>
19800 <xs:element ref="ns_p:powerSequenceSchedulePreferenceListData"/>
19801 <xs:element ref="ns_p:powerSequenceStateListData"/>
19802 <xs:element ref="ns_p:powerTimeSlotScheduleConstraintsListData"/>
19803 <xs:element ref="ns_p:powerTimeSlotScheduleListData"/>
19804 <xs:element ref="ns_p:powerTimeSlotValueListData"/>
19805 <xs:element ref="ns_p:resultData"/>
19806 <xs:element ref="ns_p:sensingDescriptionData"/>
19807 <xs:element ref="ns_p:sensingListData"/>
19808 <xs:element ref="ns_p:setpointConstraintsListData"/>
19809 <xs:element ref="ns_p:setpointDescriptionListData"/>
19810 <xs:element ref="ns_p:setpointListData"/>
19811 <xs:element ref="ns_p:smartEnergyManagementPsConfigurationRequestCall"/>
19812 <xs:element ref="ns_p:smartEnergyManagementPsData"/>
19813 <xs:element ref="ns_p:smartEnergyManagementPsPriceCalculationRequestCall"/>
19814 <xs:element ref="ns_p:smartEnergyManagementPsPriceData"/>
19815 <xs:element ref="ns_p:specificationVersionListData"/>
19816 <xs:element ref="ns_p:subscriptionManagementDeleteCall"/>
19817 <xs:element ref="ns_p:subscriptionManagementEntryListData"/>
19818 <xs:element ref="ns_p:subscriptionManagementRequestCall"/>
19819 <xs:element ref="ns_p:supplyConditionDescriptionListData"/>
19820 <xs:element ref="ns_p:supplyConditionListData"/>
19821 <xs:element ref="ns_p:supplyConditionThresholdRelationListData"/>
19822 <xs:element ref="ns_p:tariffBoundaryRelationListData"/>
19823 <xs:element ref="ns_p:tariffDescriptionListData"/>
19824 <xs:element ref="ns_p:tariffListData"/>
19825 <xs:element ref="ns_p:tariffOverallConstraintsData"/>
19826 <xs:element ref="ns_p:tariffTierRelationListData"/>
19827 <xs:element ref="ns_p:taskManagementJobDescriptionListData"/>
19828 <xs:element ref="ns_p:taskManagementJobListData"/>
19829 <xs:element ref="ns_p:taskManagementJobRelationListData"/>
19830 <xs:element ref="ns_p:taskManagementOverviewData"/>
19831 <xs:element ref="ns_p:thresholdConstraintsListData"/>
19832 <xs:element ref="ns_p:thresholdDescriptionListData"/>
19833 <xs:element ref="ns_p:thresholdListData"/>
19834 <xs:element ref="ns_p:tierBoundaryDescriptionListData"/>
19835 <xs:element ref="ns_p:tierBoundaryListData"/>
19836 <xs:element ref="ns_p:tierDescriptionListData"/>
19837 <xs:element ref="ns_p:tierIncentiveRelationListData"/>
19838 <xs:element ref="ns_p:tierListData"/>
19839 <xs:element ref="ns_p:timeDistributorData"/>
19840 <xs:element ref="ns_p:timeDistributorEnquiryCall"/>
19841 <xs:element ref="ns_p:timeInformationData"/>
19842 <xs:element ref="ns_p:timePrecisionData"/>
19843 <xs:element ref="ns_p:timeSeriesConstraintsListData"/>
19844 <xs:element ref="ns_p:timeSeriesDescriptionListData"/>
19845 <xs:element ref="ns_p:timeSeriesListData"/>
19846 <xs:element ref="ns_p:timeTableConstraintsListData"/>
19847 <xs:element ref="ns_p:timeTableDescriptionListData"/>
19848 <xs:element ref="ns_p:timeTableListData"/>
19849 <xs:element ref="ns_p:useCaseInformationListData"/>
19850 </xs:choice>
19851 </xs:group>
19852 <xs:group name="DataElementsChoiceGroup">
19853 <xs:choice>
19854 <xs:element ref="ns_p:actuatorLevelDataElements"/>
19855 <xs:element ref="ns_p:actuatorLevelDescriptionDataElements"/>
19856 <xs:element ref="ns_p:actuatorSwitchDataElements"/>
19857 <xs:element ref="ns_p:actuatorSwitchDescriptionDataElements"/>
19858 <xs:element ref="ns_p:alarmDataElements"/>
19859 <xs:element ref="ns_p:billConstraintsDataElements"/>
19860 <xs:element ref="ns_p:billDataElements"/>

```

```
19861 <xs:element ref="ns_p:billDescriptionDataElements"/>
19862 <xs:element ref="ns_p:bindingManagementDeleteCallElements"/>
19863 <xs:element ref="ns_p:bindingManagementEntryDataElements"/>
19864 <xs:element ref="ns_p:bindingManagementRequestCallElements"/>
19865 <xs:element ref="ns_p:commodityDataElements"/>
19866 <xs:element ref="ns_p:dataTunnelingCallElements"/>
19867 <xs:element ref="ns_p:deviceClassificationManufacturerDataElements"/>
19868 <xs:element ref="ns_p:deviceClassificationUserDataElements"/>
19869 <xs:element ref="ns_p:deviceConfigurationKeyValueConstraintsDataElements"/>
19870 <xs:element ref="ns_p:deviceConfigurationKeyValueDataElements"/>
19871 <xs:element ref="ns_p:deviceConfigurationKeyValueDescriptionDataElements"/>
19872 <xs:element ref="ns_p:deviceDiagnosisHeartbeatDataElements"/>
19873 <xs:element ref="ns_p:deviceDiagnosisServiceDataElements"/>
19874 <xs:element ref="ns_p:deviceDiagnosisStateDataElements"/>
19875 <xs:element ref="ns_p:directControlActivityDataElements"/>
19876 <xs:element ref="ns_p:directControlDescriptionDataElements"/>
19877 <xs:element ref="ns_p:electricalConnectionDescriptionDataElements"/>
19878 <xs:element ref="ns_p:electricalConnectionParameterDescriptionDataElements"/>
19879 <xs:element ref="ns_p:electricalConnectionPermittedValueSetDataElements"/>
19880 <xs:element ref="ns_p:electricalConnectionStateDataElements"/>
19881 <xs:element ref="ns_p:hvacOperationModeDescriptionDataElements"/>
19882 <xs:element ref="ns_p:hvacOverrunDataElements"/>
19883 <xs:element ref="ns_p:hvacOverrunDescriptionDataElements"/>
19884 <xs:element ref="ns_p:hvacSystemFunctionDataElements"/>
19885 <xs:element ref="ns_p:hvacSystemFunctionDescriptionDataElements"/>
19886 <xs:element ref="ns_p:hvacSystemFunctionOperationModeRelationDataElements"/>
19887 <xs:element ref="ns_p:hvacSystemFunctionPowerSequenceRelationDataElements"/>
19888 <xs:element ref="ns_p:hvacSystemFunctionSetpointRelationDataElements"/>
19889 <xs:element ref="ns_p:identificationDataElements"/>
19890 <xs:element ref="ns_p:incentiveDataElements"/>
19891 <xs:element ref="ns_p:incentiveDescriptionDataElements"/>
19892 <xs:element ref="ns_p:incentiveTableConstraintsDataElements"/>
19893 <xs:element ref="ns_p:incentiveTableDataElements"/>
19894 <xs:element ref="ns_p:incentiveTableDescriptionDataElements"/>
19895 <xs:element ref="ns_p:loadControlEventDataElements"/>
19896 <xs:element ref="ns_p:loadControlLimitConstraintsDataElements"/>
19897 <xs:element ref="ns_p:loadControlLimitDataElements"/>
19898 <xs:element ref="ns_p:loadControlLimitDescriptionDataElements"/>
19899 <xs:element ref="ns_p:loadControlNodeDataElements"/>
19900 <xs:element ref="ns_p:loadControlStateDataElements"/>
19901 <xs:element ref="ns_p:measurementConstraintsDataElements"/>
19902 <xs:element ref="ns_p:measurementDataElements"/>
19903 <xs:element ref="ns_p:measurementDescriptionDataElements"/>
19904 <xs:element ref="ns_p:measurementThresholdRelationDataElements"/>
19905 <xs:element ref="ns_p:messagingDataElements"/>
19906 <xs:element ref="ns_p:networkManagementAbortCallElements"/>
19907 <xs:element ref="ns_p:networkManagementAddNodeCallElements"/>
19908 <xs:element ref="ns_p:networkManagementDeviceDescriptionDataElements"/>
19909 <xs:element ref="ns_p:networkManagementDiscoverCallElements"/>
19910 <xs:element ref="ns_p:networkManagementEntityDescriptionDataElements"/>
19911 <xs:element ref="ns_p:networkManagementFeatureDescriptionDataElements"/>
19912 <xs:element ref="ns_p:networkManagementJoiningModeDataElements"/>
19913 <xs:element ref="ns_p:networkManagementModifyNodeCallElements"/>
19914 <xs:element ref="ns_p:networkManagementProcessStateDataElements"/>
19915 <xs:element ref="ns_p:networkManagementRemoveNodeCallElements"/>
19916 <xs:element ref="ns_p:networkManagementReportCandidateDataElements"/>
19917 <xs:element ref="ns_p:networkManagementScanNetworkCallElements"/>
19918 <xs:element ref="ns_p:nodeManagementBindingDataElements"/>
19919 <xs:element ref="ns_p:nodeManagementBindingDeleteCallElements"/>
19920 <xs:element ref="ns_p:nodeManagementBindingRequestCallElements"/>
19921 <xs:element ref="ns_p:nodeManagementDestinationDataElements"/>
19922 <xs:element ref="ns_p:nodeManagementDetailedDiscoveryDataElements"/>
19923 <xs:element ref="ns_p:nodeManagementSubscriptionDataElements"/>
19924 <xs:element ref="ns_p:nodeManagementSubscriptionDeleteCallElements"/>
19925 <xs:element ref="ns_p:nodeManagementSubscriptionRequestCallElements"/>
19926 <xs:element ref="ns_p:nodeManagementUseCaseDataElements"/>
19927 <xs:element ref="ns_p:operatingConstraintsDurationDataElements"/>
19928 <xs:element ref="ns_p:operatingConstraintsInterruptDataElements"/>
19929 <xs:element ref="ns_p:operatingConstraintsPowerDescriptionDataElements"/>
19930 <xs:element ref="ns_p:operatingConstraintsPowerLevelDataElements"/>
19931 <xs:element ref="ns_p:operatingConstraintsPowerRangeDataElements"/>
19932 <xs:element ref="ns_p:operatingConstraintsResumeImplicationDataElements"/>
19933 <xs:element ref="ns_p:powerSequenceAlternativesRelationDataElements"/>
19934 <xs:element ref="ns_p:powerSequenceDescriptionDataElements"/>
19935 <xs:element ref="ns_p:powerSequenceNodeScheduleInformationDataElements"/>
19936 <xs:element ref="ns_p:powerSequencePriceCalculationRequestCallElements"/>
19937 <xs:element ref="ns_p:powerSequencePriceDataElements"/>
19938 <xs:element ref="ns_p:powerSequenceScheduleConfigurationRequestCallElements"/>
```

```

19939      <xs:element ref="ns_p:powerSequenceScheduleConstraintsDataElements"/>
19940      <xs:element ref="ns_p:powerSequenceScheduleDataElements"/>
19941      <xs:element ref="ns_p:powerSequenceSchedulePreferenceDataElements"/>
19942      <xs:element ref="ns_p:powerSequenceStateDataElements"/>
19943      <xs:element ref="ns_p:powerTimeSlotScheduleConstraintsDataElements"/>
19944      <xs:element ref="ns_p:powerTimeSlotScheduleDataElements"/>
19945      <xs:element ref="ns_p:powerTimeSlotValueDataElements"/>
19946      <xs:element ref="ns_p:sensingDataElements"/>
19947      <xs:element ref="ns_p:sensingDescriptionDataElements"/>
19948      <xs:element ref="ns_p:setpointConstraintsDataElements"/>
19949      <xs:element ref="ns_p:setpointDataElements"/>
19950      <xs:element ref="ns_p:setpointDescriptionDataElements"/>
19951      <xs:element ref="ns_p:smartEnergyManagementPsConfigurationRequestCallElements"/>
19952      <xs:element ref="ns_p:smartEnergyManagementPsDataElements"/>
19953      <xs:element
19954      ref="ns_p:smartEnergyManagementPsPriceCalculationRequestCallElements"/>
19955      <xs:element ref="ns_p:smartEnergyManagementPsPriceDataElements"/>
19956      <xs:element ref="ns_p:specificationVersionDataElements"/>
19957      <xs:element ref="ns_p:subscriptionManagementDeleteCallElements"/>
19958      <xs:element ref="ns_p:subscriptionManagementEntryDataElements"/>
19959      <xs:element ref="ns_p:subscriptionManagementRequestCallElements"/>
19960      <xs:element ref="ns_p:supplyConditionDataElements"/>
19961      <xs:element ref="ns_p:supplyConditionDescriptionDataElements"/>
19962      <xs:element ref="ns_p:supplyConditionThresholdRelationDataElements"/>
19963      <xs:element ref="ns_p:tariffBoundaryRelationDataElements"/>
19964      <xs:element ref="ns_p:tariffDataElements"/>
19965      <xs:element ref="ns_p:tariffDescriptionDataElements"/>
19966      <xs:element ref="ns_p:tariffOverallConstraintsDataElements"/>
19967      <xs:element ref="ns_p:tariffTierRelationDataElements"/>
19968      <xs:element ref="ns_p:taskManagementJobDataElements"/>
19969      <xs:element ref="ns_p:taskManagementJobDescriptionDataElements"/>
19970      <xs:element ref="ns_p:taskManagementJobRelationDataElements"/>
19971      <xs:element ref="ns_p:taskManagementOverviewDataElements"/>
19972      <xs:element ref="ns_p:thresholdConstraintsDataElements"/>
19973      <xs:element ref="ns_p:thresholdDataElements"/>
19974      <xs:element ref="ns_p:thresholdDescriptionDataElements"/>
19975      <xs:element ref="ns_p:tierBoundaryDataElements"/>
19976      <xs:element ref="ns_p:tierBoundaryDescriptionDataElements"/>
19977      <xs:element ref="ns_p:tierDataElements"/>
19978      <xs:element ref="ns_p:tierDescriptionDataElements"/>
19979      <xs:element ref="ns_p:tierIncentiveRelationDataElements"/>
19980      <xs:element ref="ns_p:timeDistributorDataElements"/>
19981      <xs:element ref="ns_p:timeDistributorEnquiryCallElements"/>
19982      <xs:element ref="ns_p:timeInformationDataElements"/>
19983      <xs:element ref="ns_p:timePrecisionDataElements"/>
19984      <xs:element ref="ns_p:timeSeriesConstraintsDataElements"/>
19985      <xs:element ref="ns_p:timeSeriesDataElements"/>
19986      <xs:element ref="ns_p:timeSeriesDescriptionDataElements"/>
19987      <xs:element ref="ns_p:timeTableConstraintsDataElements"/>
19988      <xs:element ref="ns_p:timeTableDataElements"/>
19989      <xs:element ref="ns_p:timeTableDescriptionDataElements"/>
19990      <xs:element ref="ns_p:useCaseInformationDataElements"/>
19991    </xs:choice>
19992  </xs:group>
19993  <xs:group name="DataSelectorsChoiceGroup">
19994    <xs:choice>
19995      <xs:element ref="ns_p:alarmListDataSelectors"/>
19996      <xs:element ref="ns_p:billConstraintsListDataSelectors"/>
19997      <xs:element ref="ns_p:billDescriptionListDataSelectors"/>
19998      <xs:element ref="ns_p:billListDataSelectors"/>
19999      <xs:element ref="ns_p:bindingManagementEntryListDataSelectors"/>
20000      <xs:element ref="ns_p:commodityListDataSelectors"/>
20001      <xs:element ref="ns_p:deviceConfigurationKeyValueConstraintsListDataSelectors"/>
20002      <xs:element ref="ns_p:deviceConfigurationKeyValueDescriptionListDataSelectors"/>
20003      <xs:element ref="ns_p:deviceConfigurationKeyValueListDataSelectors"/>
20004      <xs:element ref="ns_p:directControlActivityListDataSelectors"/>
20005      <xs:element ref="ns_p:electricalConnectionDescriptionListDataSelectors"/>
20006      <xs:element ref="ns_p:electricalConnectionParameterDescriptionListDataSelectors"/>
20007      <xs:element ref="ns_p:electricalConnectionPermittedValueSetListDataSelectors"/>
20008      <xs:element ref="ns_p:electricalConnectionStateListDataSelectors"/>
20009      <xs:element ref="ns_p:hvacOperationModeDescriptionListDataSelectors"/>
20010      <xs:element ref="ns_p:hvacOverrunDescriptionListDataSelectors"/>
20011      <xs:element ref="ns_p:hvacOverrunListDataSelectors"/>
20012      <xs:element ref="ns_p:hvacSystemFunctionDescriptionListDataSelectors"/>
20013      <xs:element ref="ns_p:hvacSystemFunctionListDataSelectors"/>
20014      <xs:element ref="ns_p:hvacSystemFunctionOperationModeRelationListDataSelectors"/>
20015      <xs:element ref="ns_p:hvacSystemFunctionPowerSequenceRelationListDataSelectors"/>
20016      <xs:element ref="ns_p:hvacSystemFunctionSetpointRelationListDataSelectors"/>

```



```
20017 <xs:element ref="ns_p:identificationListDataSelectors"/>
20018 <xs:element ref="ns_p:incentiveDescriptionListDataSelectors"/>
20019 <xs:element ref="ns_p:incentiveListDataSelectors"/>
20020 <xs:element ref="ns_p:incentiveTableConstraintsDataSelectors"/>
20021 <xs:element ref="ns_p:incentiveTableDataSelectors"/>
20022 <xs:element ref="ns_p:incentiveTableDescriptionDataSelectors"/>
20023 <xs:element ref="ns_p:loadControlEventListDataSelectors"/>
20024 <xs:element ref="ns_p:loadControlLimitConstraintsListDataSelectors"/>
20025 <xs:element ref="ns_p:loadControlLimitDescriptionListDataSelectors"/>
20026 <xs:element ref="ns_p:loadControlLimitListDataSelectors"/>
20027 <xs:element ref="ns_p:loadControlStateListDataSelectors"/>
20028 <xs:element ref="ns_p:measurementConstraintsListDataSelectors"/>
20029 <xs:element ref="ns_p:measurementDescriptionListDataSelectors"/>
20030 <xs:element ref="ns_p:measurementListDataSelectors"/>
20031 <xs:element ref="ns_p:measurementThresholdRelationListDataSelectors"/>
20032 <xs:element ref="ns_p:messagingListDataSelectors"/>
20033 <xs:element ref="ns_p:networkManagementDeviceDescriptionListDataSelectors"/>
20034 <xs:element ref="ns_p:networkManagementEntityDescriptionListDataSelectors"/>
20035 <xs:element ref="ns_p:networkManagementFeatureDescriptionListDataSelectors"/>
20036 <xs:element ref="ns_p:nodeManagementBindingDataSelectors"/>
20037 <xs:element ref="ns_p:nodeManagementDestinationListDataSelectors"/>
20038 <xs:element ref="ns_p:nodeManagementDetailedDiscoveryDataSelectors"/>
20039 <xs:element ref="ns_p:nodeManagementSubscriptionDataSelectors"/>
20040 <xs:element ref="ns_p:nodeManagementUseCaseDataSelectors"/>
20041 <xs:element ref="ns_p:operatingConstraintsDurationListDataSelectors"/>
20042 <xs:element ref="ns_p:operatingConstraintsInterruptListDataSelectors"/>
20043 <xs:element ref="ns_p:operatingConstraintsPowerDescriptionListDataSelectors"/>
20044 <xs:element ref="ns_p:operatingConstraintsPowerLevelListDataSelectors"/>
20045 <xs:element ref="ns_p:operatingConstraintsPowerRangeListDataSelectors"/>
20046 <xs:element ref="ns_p:operatingConstraintsResumeImplicationListDataSelectors"/>
20047 <xs:element ref="ns_p:powerSequenceAlternativesRelationListDataSelectors"/>
20048 <xs:element ref="ns_p:powerSequenceDescriptionListDataSelectors"/>
20049 <xs:element ref="ns_p:powerSequencePriceListDataSelectors"/>
20050 <xs:element ref="ns_p:powerSequenceScheduleConstraintsListDataSelectors"/>
20051 <xs:element ref="ns_p:powerSequenceScheduleListDataSelectors"/>
20052 <xs:element ref="ns_p:powerSequenceSchedulePreferenceListDataSelectors"/>
20053 <xs:element ref="ns_p:powerSequenceStateListDataSelectors"/>
20054 <xs:element ref="ns_p:powerTimeSlotScheduleConstraintsListDataSelectors"/>
20055 <xs:element ref="ns_p:powerTimeSlotScheduleListDataSelectors"/>
20056 <xs:element ref="ns_p:powerTimeSlotValueListDataSelectors"/>
20057 <xs:element ref="ns_p:sensingListDataSelectors"/>
20058 <xs:element ref="ns_p:setpointConstraintsListDataSelectors"/>
20059 <xs:element ref="ns_p:setpointDescriptionListDataSelectors"/>
20060 <xs:element ref="ns_p:setpointListDataSelectors"/>
20061 <xs:element ref="ns_p:smartEnergyManagementPsDataSelectors"/>
20062 <xs:element ref="ns_p:smartEnergyManagementPsPriceDataSelectors"/>
20063 <xs:element ref="ns_p:specificationVersionListDataSelectors"/>
20064 <xs:element ref="ns_p:subscriptionManagementEntryListDataSelectors"/>
20065 <xs:element ref="ns_p:supplyConditionDescriptionListDataSelectors"/>
20066 <xs:element ref="ns_p:supplyConditionListDataSelectors"/>
20067 <xs:element ref="ns_p:supplyConditionThresholdRelationListDataSelectors"/>
20068 <xs:element ref="ns_p:tariffBoundaryRelationListDataSelectors"/>
20069 <xs:element ref="ns_p:tariffDescriptionListDataSelectors"/>
20070 <xs:element ref="ns_p:tariffListDataSelectors"/>
20071 <xs:element ref="ns_p:tariffTierRelationListDataSelectors"/>
20072 <xs:element ref="ns_p:taskManagementJobDescriptionListDataSelectors"/>
20073 <xs:element ref="ns_p:taskManagementJobListDataSelectors"/>
20074 <xs:element ref="ns_p:taskManagementJobRelationListDataSelectors"/>
20075 <xs:element ref="ns_p:thresholdConstraintsListDataSelectors"/>
20076 <xs:element ref="ns_p:thresholdDescriptionListDataSelectors"/>
20077 <xs:element ref="ns_p:thresholdListDataSelectors"/>
20078 <xs:element ref="ns_p:tierBoundaryDescriptionListDataSelectors"/>
20079 <xs:element ref="ns_p:tierBoundaryListDataSelectors"/>
20080 <xs:element ref="ns_p:tierDescriptionListDataSelectors"/>
20081 <xs:element ref="ns_p:tierIncentiveRelationListDataSelectors"/>
20082 <xs:element ref="ns_p:tierListDataSelectors"/>
20083 <xs:element ref="ns_p:timeSeriesConstraintsListDataSelectors"/>
20084 <xs:element ref="ns_p:timeSeriesDescriptionListDataSelectors"/>
20085 <xs:element ref="ns_p:timeSeriesListDataSelectors"/>
20086 <xs:element ref="ns_p:timeTableConstraintsListDataSelectors"/>
20087 <xs:element ref="ns_p:timeTableDescriptionListDataSelectors"/>
20088 <xs:element ref="ns_p:timeTableListDataSelectors"/>
20089 <xs:element ref="ns_p:useCaseInformationListDataSelectors"/>
20090 </xs:choice>
20091 </xs:group>
20092 </xs:schema>
20093
20094
```

20095

20096 **A.3.2 CommandFrame**

20097 File name: EEBus_SPINE_TS_CommandFrame.xsd

```
20098
20099 <?xml version="1.0" encoding="UTF-8"?>
20100 <!--
20101     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
20102     Version 1.1.1
20103     2018-12-21
20104     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
20105     Source: https://www.eebus.org/en/specifications/
20106 -->
20107 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
20108     xmlns:xs="http://www.w3.org/2001/XMLSchema"
20109     targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
20110     blockDefault="#all" elementFormDefault="qualified">
20111     <xs:annotation>
20112         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
20113     e.V. All rights reserved.</xs:documentation>
20114     </xs:annotation>
20115     <xs:include schemaLocation="EEBus_SPINE_TS_CommandCommonDefinitions.xsd"/>
20116     <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
20117     <xs:simpleType name="MsgCounterType">
20118         <xs:restriction base="xs:unsignedLong"> </xs:restriction>
20119     </xs:simpleType>
20120     <xs:simpleType name="CmdClassifierType">
20121         <xs:restriction base="xs:string">
20122             <xs:enumeration value="read"/>
20123             <xs:enumeration value="reply"/>
20124             <xs:enumeration value="notify"/>
20125             <xs:enumeration value="write"/>
20126             <xs:enumeration value="call"/>
20127             <xs:enumeration value="result"/>
20128         </xs:restriction>
20129     </xs:simpleType>
20130     <xs:element name="cmdClassifier" type="ns_p:CmdClassifierType"/>
20131     <xs:element name="manufacturerSpecificExtension" type="xs:hexBinary"/>
20132     <xs:element name="lastUpdateAt" type="ns_p:AbsoluteOrRelativeTimeType"/>
20133     <xs:element name="function" type="ns_p:FunctionType"/>
20134     <xs:simpleType name="FilterIdType">
20135         <xs:restriction base="xs:unsignedInt"/>
20136     </xs:simpleType>
20137     <xs:complexType name="FilterType">
20138         <xs:sequence>
20139             <xs:element minOccurs="0" name="filterId" type="ns_p:FilterIdType"/>
20140             <xs:element minOccurs="0" ref="ns_p:cmdControl"/>
20141             <xs:group ref="ns_p:DataSelectorsChoiceGroup" minOccurs="0"
20142     maxOccurs="unbounded"/>
20143             <xs:group ref="ns_p:DataElementsChoiceGroup" minOccurs="0"/>
20144         </xs:sequence>
20145     </xs:complexType>
20146     <xs:element name="filter" type="ns_p:FilterType"/>
20147     <xs:complexType name="CmdControlType">
20148         <xs:sequence>
20149             <xs:element name="delete" minOccurs="0" type="ns_p:ElementTagType"/>
20150             <xs:element name="partial" minOccurs="0" type="ns_p:ElementTagType"/>
20151         </xs:sequence>
20152     </xs:complexType>
20153     <xs:element name="cmdControl" type="ns_p:CmdControlType"/>
20154     <xs:group name="CmdOptionGroup">
20155         <xs:sequence>
20156             <xs:element minOccurs="0" ref="ns_p:function"/>
20157             <xs:element minOccurs="0" ref="ns_p:filter" maxOccurs="unbounded"/>
20158         </xs:sequence>
20159     </xs:group>
20160     <xs:group name="DataExtendGroup">
20161         <xs:sequence>
20162             <xs:element minOccurs="0" ref="ns_p:manufacturerSpecificExtension"/>
20163             <xs:element minOccurs="0" ref="ns_p:lastUpdateAt"/>
20164         </xs:sequence>
20165     </xs:group>
20166     <xs:group name="PayloadContributionGroup">
20167         <xs:sequence>
20168             <xs:group ref="ns_p:CmdOptionGroup" minOccurs="0"/>
```

```
20169         <xs:group ref="ns_p:DataChoiceGroup" minOccurs="0"/>
20170         <xs:group minOccurs="0" ref="ns_p:DataExtendGroup"/>
20171     </xs:sequence>
20172 </xs:group>
20173 <xs:complexType name="CmdType">
20174     <xs:sequence>
20175         <xs:group ref="ns_p:PayloadContributionGroup"/>
20176     </xs:sequence>
20177 </xs:complexType>
20178 </xs:schema>
20179
20180
```

20181

20182 A.3.3 CommonDataTypes

20183 File name: EEBus_SPINE_TS_CommonDataTypes.xsd

```
20184
20185 <?xml version="1.0" encoding="UTF-8"?>
20186 <!--
20187     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
20188     Version 1.1.1
20189     2018-12-21
20190     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
20191     Source: https://www.eebus.org/en/specifications/
20192 -->
20193 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
20194     xmlns:xs="http://www.w3.org/2001/XMLSchema"
20195     targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
20196     blockDefault="#all" elementFormDefault="qualified">
20197     <xs:annotation>
20198         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
20199     e.V. All rights reserved.</xs:documentation>
20200     </xs:annotation>
20201     <xs:complexType name="ElementTagType"/>
20202     <xs:simpleType name="LabelType">
20203         <xs:restriction base="xs:string"> </xs:restriction>
20204     </xs:simpleType>
20205     <xs:simpleType name="DescriptionType">
20206         <xs:restriction base="xs:string"> </xs:restriction>
20207     </xs:simpleType>
20208     <xs:simpleType name="SpecificationVersionType">
20209         <xs:restriction base="xs:string">
20210             <xs:pattern value="[1-9][0-9]*\.[0-9]+\.[0-9]+"/>
20211             <xs:maxLength value="128"/>
20212         </xs:restriction>
20213     </xs:simpleType>
20214     <xs:complexType name="TimePeriodType">
20215         <xs:sequence>
20216             <xs:element minOccurs="0" name="startTime"
20217 type="ns_p:AbsoluteOrRelativeTimeType"/>
20218             <xs:element minOccurs="0" name="endTime" type="ns_p:AbsoluteOrRelativeTimeType"/>
20219         </xs:sequence>
20220     </xs:complexType>
20221     <xs:complexType name="TimePeriodElementsType">
20222         <xs:sequence>
20223             <xs:element minOccurs="0" name="startTime" type="ns_p:ElementTagType"/>
20224             <xs:element minOccurs="0" name="endTime" type="ns_p:ElementTagType"/>
20225         </xs:sequence>
20226     </xs:complexType>
20227     <xs:complexType name="TimestampIntervalType">
20228         <xs:sequence>
20229             <xs:element name="startTime" type="ns_p:AbsoluteOrRelativeTimeType"
20230 minOccurs="0"/>
20231             <xs:element name="endTime" type="ns_p:AbsoluteOrRelativeTimeType" minOccurs="0"/>
20232         </xs:sequence>
20233     </xs:complexType>
20234     <xs:simpleType name="EnumExtendType">
20235         <xs:restriction base="xs:string">
20236             <xs:pattern value="_(:[1-9][0-9]*|n:[a-zA-Z0-9-]+)_([^\p{Cc}\p{Cf}\p{Z}])+"/>
20237             <xs:maxLength value="256"/>
20238         </xs:restriction>
20239     </xs:simpleType>
20240     <xs:simpleType name="AbsoluteOrRelativeTimeType">
20241         <xs:union memberTypes="xs:duration xs:dateTime"/>
```

```

20242 </xs:simpleType>
20243 <xs:simpleType name="RecurringIntervalType">
20244   <xs:union memberTypes="ns_p:RecurringIntervalEnumType ns_p:EnumExtendType"/>
20245 </xs:simpleType>
20246 <xs:simpleType name="RecurringIntervalEnumType">
20247   <xs:restriction base="xs:string">
20248     <xs:enumeration value="yearly"/>
20249     <xs:enumeration value="monthly"/>
20250     <xs:enumeration value="weekly"/>
20251     <xs:enumeration value="daily"/>
20252     <xs:enumeration value="hourly"/>
20253     <xs:enumeration value="everyMinute"/>
20254     <xs:enumeration value="everySecond"/>
20255   </xs:restriction>
20256 </xs:simpleType>
20257 <xs:simpleType name="MonthType">
20258   <xs:restriction base="xs:string">
20259     <xs:enumeration value="january"/>
20260     <xs:enumeration value="february"/>
20261     <xs:enumeration value="march"/>
20262     <xs:enumeration value="april"/>
20263     <xs:enumeration value="may"/>
20264     <xs:enumeration value="june"/>
20265     <xs:enumeration value="july"/>
20266     <xs:enumeration value="august"/>
20267     <xs:enumeration value="september"/>
20268     <xs:enumeration value="october"/>
20269     <xs:enumeration value="november"/>
20270     <xs:enumeration value="december"/>
20271   </xs:restriction>
20272 </xs:simpleType>
20273 <xs:simpleType name="DayOfMonthType">
20274   <xs:restriction base="xs:unsignedByte">
20275     <xs:minInclusive value="1"/>
20276     <xs:maxInclusive value="31"/>
20277   </xs:restriction>
20278 </xs:simpleType>
20279 <xs:simpleType name="CalendarWeekType">
20280   <xs:restriction base="xs:unsignedByte">
20281     <xs:minInclusive value="1"/>
20282     <xs:maxInclusive value="53"/>
20283   </xs:restriction>
20284 </xs:simpleType>
20285 <xs:simpleType name="DayOfWeekType">
20286   <xs:restriction base="xs:string">
20287     <xs:enumeration value="monday"/>
20288     <xs:enumeration value="tuesday"/>
20289     <xs:enumeration value="wednesday"/>
20290     <xs:enumeration value="thursday"/>
20291     <xs:enumeration value="friday"/>
20292     <xs:enumeration value="saturday"/>
20293     <xs:enumeration value="sunday"/>
20294   </xs:restriction>
20295 </xs:simpleType>
20296 <xs:complexType name="DaysOfWeekType">
20297   <xs:sequence>
20298     <xs:element name="monday" minOccurs="0" type="ns_p:ElementTagType"/>
20299     <xs:element name="tuesday" minOccurs="0" type="ns_p:ElementTagType"/>
20300     <xs:element name="wednesday" minOccurs="0" type="ns_p:ElementTagType"/>
20301     <xs:element name="thursday" minOccurs="0" type="ns_p:ElementTagType"/>
20302     <xs:element name="friday" minOccurs="0" type="ns_p:ElementTagType"/>
20303     <xs:element name="saturday" minOccurs="0" type="ns_p:ElementTagType"/>
20304     <xs:element name="sunday" minOccurs="0" type="ns_p:ElementTagType"/>
20305   </xs:sequence>
20306 </xs:complexType>
20307 <xs:simpleType name="OccurrenceType">
20308   <xs:union memberTypes="ns_p:OccurrenceEnumType ns_p:EnumExtendType"/>
20309 </xs:simpleType>
20310 <xs:simpleType name="OccurrenceEnumType">
20311   <xs:restriction base="xs:string">
20312     <xs:enumeration value="first"/>
20313     <xs:enumeration value="second"/>
20314     <xs:enumeration value="third"/>
20315     <xs:enumeration value="fourth"/>
20316     <xs:enumeration value="last"/>
20317   </xs:restriction>
20318 </xs:simpleType>
20319 <xs:complexType name="AbsoluteOrRecurringTimeType">

```

```
20320     <xs:sequence>
20321       <xs:element minOccurs="0" name="dateTime" type="xs:dateTime"/>
20322       <xs:element name="month" minOccurs="0" type="ns_p:MonthType"/>
20323       <xs:element name="dayOfMonth" minOccurs="0" type="ns_p:DayOfMonthType"/>
20324       <xs:element minOccurs="0" name="calendarWeek" type="ns_p:CalendarWeekType"/>
20325       <xs:element minOccurs="0" name="dayOfWeekOccurrence" type="ns_p:OccurrenceType"/>
20326       <xs:element minOccurs="0" name="daysOfWeek" type="ns_p:DaysOfWeekType"/>
20327       <xs:element minOccurs="0" name="time" type="xs:time"/>
20328       <xs:element minOccurs="0" name="relative" type="xs:duration"/>
20329     </xs:sequence>
20330   </xs:complexType>
20331   <xs:complexType name="AbsoluteOrRecurringTimeElementsType">
20332     <xs:sequence>
20333       <xs:element minOccurs="0" name="dateTime" type="ns_p:ElementTagType"/>
20334       <xs:element name="month" minOccurs="0" type="ns_p:ElementTagType"/>
20335       <xs:element name="dayOfMonth" minOccurs="0" type="ns_p:ElementTagType"/>
20336       <xs:element minOccurs="0" name="calendarWeek" type="ns_p:ElementTagType"/>
20337       <xs:element minOccurs="0" name="dayOfWeekOccurrence" type="ns_p:ElementTagType"/>
20338       <xs:element minOccurs="0" name="daysOfWeek" type="ns_p:ElementTagType"/>
20339       <xs:element minOccurs="0" name="time" type="ns_p:ElementTagType"/>
20340       <xs:element minOccurs="0" name="relative" type="ns_p:ElementTagType"/>
20341     </xs:sequence>
20342   </xs:complexType>
20343   <xs:complexType name="RecurrenceInformationType">
20344     <xs:sequence>
20345       <xs:element minOccurs="0" name="recurringInterval"
20346 type="ns_p:RecurringIntervalType"/>
20347       <xs:element minOccurs="0" name="recurringIntervalStep" type="xs:unsignedInt"/>
20348       <xs:element minOccurs="0" name="firstExecution" type="xs:dateTime"/>
20349       <xs:element minOccurs="0" name="executionCount" type="xs:unsignedInt"/>
20350       <xs:element minOccurs="0" name="lastExecution" type="xs:dateTime"/>
20351     </xs:sequence>
20352   </xs:complexType>
20353   <xs:complexType name="RecurrenceInformationElementsType">
20354     <xs:sequence>
20355       <xs:element minOccurs="0" name="recurringInterval" type="ns_p:ElementTagType"/>
20356       <xs:element minOccurs="0" name="recurringIntervalStep"
20357 type="ns_p:ElementTagType"/>
20358       <xs:element minOccurs="0" name="firstExecution" type="ns_p:ElementTagType"/>
20359       <xs:element minOccurs="0" name="executionCount" type="ns_p:ElementTagType"/>
20360       <xs:element minOccurs="0" name="lastExecution" type="ns_p:ElementTagType"/>
20361     </xs:sequence>
20362   </xs:complexType>
20363   <xs:complexType name="ScaledNumberRangeType">
20364     <xs:sequence>
20365       <xs:element minOccurs="0" name="min" type="ns_p:ScaledNumberType"/>
20366       <xs:element minOccurs="0" name="max" type="ns_p:ScaledNumberType"/>
20367     </xs:sequence>
20368   </xs:complexType>
20369   <xs:complexType name="ScaledNumberRangeElementsType">
20370     <xs:sequence>
20371       <xs:element minOccurs="0" name="min" type="ns_p:ScaledNumberElementsType"/>
20372       <xs:element minOccurs="0" name="max" type="ns_p:ScaledNumberElementsType"/>
20373     </xs:sequence>
20374   </xs:complexType>
20375   <xs:complexType name="ScaledNumberSetType">
20376     <xs:sequence>
20377       <xs:element maxOccurs="unbounded" minOccurs="0" name="value"
20378 type="ns_p:ScaledNumberType"/>
20379       <xs:element maxOccurs="unbounded" minOccurs="0" name="range"
20380 type="ns_p:ScaledNumberRangeType"/>
20381     </xs:sequence>
20382   </xs:complexType>
20383   <xs:complexType name="ScaledNumberSetElementsType">
20384     <xs:sequence>
20385       <xs:element minOccurs="0" name="value" type="ns_p:ScaledNumberElementsType"/>
20386       <xs:element minOccurs="0" name="range" type="ns_p:ScaledNumberRangeElementsType"/>
20387     </xs:sequence>
20388   </xs:complexType>
20389   <xs:simpleType name="NumberType">
20390     <xs:restriction base="xs:long"/>
20391   </xs:simpleType>
20392   <xs:simpleType name="ScaleType">
20393     <xs:restriction base="xs:short"/>
20394   </xs:simpleType>
20395   <xs:complexType name="ScaledNumberType">
20396     <xs:sequence>
20397       <xs:element minOccurs="0" name="number" type="ns_p:NumberType"/>
```

```
20398         <xs:element minOccurs="0" name="scale" type="ns_p:ScaleType"/>
20399     </xs:sequence>
20400 </xs:complexType>
20401 <xs:complexType name="ScaledNumberElementsType">
20402     <xs:sequence>
20403         <xs:element minOccurs="0" name="number" type="ns_p:ElementTagType"/>
20404         <xs:element minOccurs="0" name="scale" type="ns_p:ElementTagType"/>
20405     </xs:sequence>
20406 </xs:complexType>
20407 <xs:simpleType name="MaxResponseDelayType">
20408     <xs:restriction base="xs:duration"/>
20409 </xs:simpleType>
20410 <xs:simpleType name="CommodityTypeType">
20411     <xs:union memberTypes="ns_p:CommodityTypeEnumType ns_p:EnumExtendType"/>
20412 </xs:simpleType>
20413 <xs:simpleType name="CommodityTypeEnumType">
20414     <xs:restriction base="xs:string">
20415         <xs:enumeration value="electricity"/>
20416         <xs:enumeration value="gas"/>
20417         <xs:enumeration value="oil"/>
20418         <xs:enumeration value="water"/>
20419         <xs:enumeration value="wasteWater"/>
20420         <xs:enumeration value="domesticHotWater"/>
20421         <xs:enumeration value="heatingWater"/>
20422         <xs:enumeration value="steam"/>
20423         <xs:enumeration value="heat"/>
20424         <xs:enumeration value="coolingLoad"/>
20425         <xs:enumeration value="air"/>
20426     </xs:restriction>
20427 </xs:simpleType>
20428 <xs:simpleType name="EnergyDirectionType">
20429     <xs:union memberTypes="ns_p:EnergyDirectionEnumType ns_p:EnumExtendType"/>
20430 </xs:simpleType>
20431 <xs:simpleType name="EnergyDirectionEnumType">
20432     <xs:restriction base="xs:string">
20433         <xs:enumeration value="consume"/>
20434         <xs:enumeration value="produce"/>
20435     </xs:restriction>
20436 </xs:simpleType>
20437 <xs:simpleType name="EnergyModeType">
20438     <xs:union memberTypes="ns_p:EnergyModeEnumType ns_p:EnumExtendType"/>
20439 </xs:simpleType>
20440 <xs:simpleType name="EnergyModeEnumType">
20441     <xs:restriction base="xs:string">
20442         <xs:enumeration value="consume"/>
20443         <xs:enumeration value="produce"/>
20444         <xs:enumeration value="idle"/>
20445         <xs:enumeration value="auto"/>
20446     </xs:restriction>
20447 </xs:simpleType>
20448 <xs:simpleType name="UnitOfMeasurementType">
20449     <xs:union memberTypes="ns_p:UnitOfMeasurementEnumType ns_p:EnumExtendType"/>
20450 </xs:simpleType>
20451 <xs:simpleType name="UnitOfMeasurementEnumType">
20452     <xs:restriction base="xs:string">
20453         <xs:enumeration value="unknown"/>
20454         <xs:enumeration value="1"/>
20455         <xs:enumeration value="m"/>
20456         <xs:enumeration value="kg"/>
20457         <xs:enumeration value="s"/>
20458         <xs:enumeration value="A"/>
20459         <xs:enumeration value="K"/>
20460         <xs:enumeration value="mol"/>
20461         <xs:enumeration value="cd"/>
20462         <xs:enumeration value="V"/>
20463         <xs:enumeration value="W"/>
20464         <xs:enumeration value="Wh"/>
20465         <xs:enumeration value="VA"/>
20466         <xs:enumeration value="VAh"/>
20467         <xs:enumeration value="var"/>
20468         <xs:enumeration value="varh"/>
20469         <xs:enumeration value="degC"/>
20470         <xs:enumeration value="degF"/>
20471         <xs:enumeration value="lm"/>
20472         <xs:enumeration value="lx"/>
20473         <xs:enumeration value="Ohm"/>
20474         <xs:enumeration value="Hz"/>
20475         <xs:enumeration value="dB"/>
```

```

20476      <xs:enumeration value="dBm"/>
20477      <xs:enumeration value="pct"/>
20478      <xs:enumeration value="ppm"/>
20479      <xs:enumeration value="l"/>
20480      <xs:enumeration value="l/s"/>
20481      <xs:enumeration value="l/h"/>
20482      <xs:enumeration value="deg"/>
20483      <xs:enumeration value="rad"/>
20484      <xs:enumeration value="rad/s"/>
20485      <xs:enumeration value="sr"/>
20486      <xs:enumeration value="Gy"/>
20487      <xs:enumeration value="Bq"/>
20488      <xs:enumeration value="Bq/m^3"/>
20489      <xs:enumeration value="Sv"/>
20490      <xs:enumeration value="Rd"/>
20491      <xs:enumeration value="C"/>
20492      <xs:enumeration value="F"/>
20493      <xs:enumeration value="H"/>
20494      <xs:enumeration value="J"/>
20495      <xs:enumeration value="N"/>
20496      <xs:enumeration value="N_m"/>
20497      <xs:enumeration value="N_s"/>
20498      <xs:enumeration value="Wb"/>
20499      <xs:enumeration value="T"/>
20500      <xs:enumeration value="Pa"/>
20501      <xs:enumeration value="bar"/>
20502      <xs:enumeration value="atm"/>
20503      <xs:enumeration value="psi"/>
20504      <xs:enumeration value="mmHg"/>
20505      <xs:enumeration value="m^2"/>
20506      <xs:enumeration value="m^3"/>
20507      <xs:enumeration value="m^3/h"/>
20508      <xs:enumeration value="m/s"/>
20509      <xs:enumeration value="m/s^2"/>
20510      <xs:enumeration value="m^3/s"/>
20511      <xs:enumeration value="m/m^3"/>
20512      <xs:enumeration value="kg/m^3"/>
20513      <xs:enumeration value="kg_m"/>
20514      <xs:enumeration value="m^2/s"/>
20515      <xs:enumeration value="W/m_K"/>
20516      <xs:enumeration value="J/K"/>
20517      <xs:enumeration value="l/s"/>
20518      <xs:enumeration value="W/m^2"/>
20519      <xs:enumeration value="J/m^2"/>
20520      <xs:enumeration value="S"/>
20521      <xs:enumeration value="S/m"/>
20522      <xs:enumeration value="K/s"/>
20523      <xs:enumeration value="Pa/s"/>
20524      <xs:enumeration value="J/kg_K"/>
20525      <xs:enumeration value="Vs"/>
20526      <xs:enumeration value="V/m"/>
20527      <xs:enumeration value="V/Hz"/>
20528      <xs:enumeration value="As"/>
20529      <xs:enumeration value="A/m"/>
20530      <xs:enumeration value="Hz/s"/>
20531      <xs:enumeration value="kg/s"/>
20532      <xs:enumeration value="kg_m^2"/>
20533      <xs:enumeration value="J/Wh"/>
20534      <xs:enumeration value="W/s"/>
20535      <xs:enumeration value="ft^3"/>
20536      <xs:enumeration value="ft^3/h"/>
20537      <xs:enumeration value="ccf"/>
20538      <xs:enumeration value="ccf/h"/>
20539      <xs:enumeration value="US.liq.gal"/>
20540      <xs:enumeration value="US.liq.gal/h"/>
20541      <xs:enumeration value="Imp.gal"/>
20542      <xs:enumeration value="Imp.gal/h"/>
20543      <xs:enumeration value="Btu"/>
20544      <xs:enumeration value="Btu/h"/>
20545      <xs:enumeration value="Ah"/>
20546      <xs:enumeration value="kg/Wh"/>
20547    </xs:restriction>
20548  </xs:simpleType>
20549  <xs:simpleType name="CurrencyType">
20550    <xs:union memberTypes="ns_p:CurrencyEnumType ns_p:EnumExtendType"/>
20551  </xs:simpleType>
20552  <xs:simpleType name="CurrencyEnumType">
20553    <xs:restriction base="xs:string">

```

20554	<xs:enumeration value="AED"/>
20555	<xs:enumeration value="AFN"/>
20556	<xs:enumeration value="ALL"/>
20557	<xs:enumeration value="AMD"/>
20558	<xs:enumeration value="ANG"/>
20559	<xs:enumeration value="AOA"/>
20560	<xs:enumeration value="ARS"/>
20561	<xs:enumeration value="AUD"/>
20562	<xs:enumeration value="AWG"/>
20563	<xs:enumeration value="AZN"/>
20564	<xs:enumeration value="BAM"/>
20565	<xs:enumeration value="BBD"/>
20566	<xs:enumeration value="BDT"/>
20567	<xs:enumeration value="BGN"/>
20568	<xs:enumeration value="BHD"/>
20569	<xs:enumeration value="BIF"/>
20570	<xs:enumeration value="BMD"/>
20571	<xs:enumeration value="BND"/>
20572	<xs:enumeration value="BOB"/>
20573	<xs:enumeration value="BOV"/>
20574	<xs:enumeration value="BRL"/>
20575	<xs:enumeration value="BSD"/>
20576	<xs:enumeration value="BTN"/>
20577	<xs:enumeration value="BWP"/>
20578	<xs:enumeration value="BYR"/>
20579	<xs:enumeration value="BZD"/>
20580	<xs:enumeration value="CAD"/>
20581	<xs:enumeration value="CDF"/>
20582	<xs:enumeration value="CHE"/>
20583	<xs:enumeration value="CHF"/>
20584	<xs:enumeration value="CHW"/>
20585	<xs:enumeration value="CLF"/>
20586	<xs:enumeration value="CLP"/>
20587	<xs:enumeration value="CNY"/>
20588	<xs:enumeration value="COP"/>
20589	<xs:enumeration value="COU"/>
20590	<xs:enumeration value="CRC"/>
20591	<xs:enumeration value="CUC"/>
20592	<xs:enumeration value="CUP"/>
20593	<xs:enumeration value="CVE"/>
20594	<xs:enumeration value="CZK"/>
20595	<xs:enumeration value="DJF"/>
20596	<xs:enumeration value="DKK"/>
20597	<xs:enumeration value="DOP"/>
20598	<xs:enumeration value="DZD"/>
20599	<xs:enumeration value="EGP"/>
20600	<xs:enumeration value="ERN"/>
20601	<xs:enumeration value="ETB"/>
20602	<xs:enumeration value="EUR"/>
20603	<xs:enumeration value="FJD"/>
20604	<xs:enumeration value="FKP"/>
20605	<xs:enumeration value="GBP"/>
20606	<xs:enumeration value="GEL"/>
20607	<xs:enumeration value="GHS"/>
20608	<xs:enumeration value="GIP"/>
20609	<xs:enumeration value="GMD"/>
20610	<xs:enumeration value="GNF"/>
20611	<xs:enumeration value="GTQ"/>
20612	<xs:enumeration value="GYD"/>
20613	<xs:enumeration value="HKD"/>
20614	<xs:enumeration value="HNL"/>
20615	<xs:enumeration value="HRK"/>
20616	<xs:enumeration value="HTG"/>
20617	<xs:enumeration value="HUF"/>
20618	<xs:enumeration value="IDR"/>
20619	<xs:enumeration value="ILS"/>
20620	<xs:enumeration value="INR"/>
20621	<xs:enumeration value="IQD"/>
20622	<xs:enumeration value="IRR"/>
20623	<xs:enumeration value="ISK"/>
20624	<xs:enumeration value="JMD"/>
20625	<xs:enumeration value="JOD"/>
20626	<xs:enumeration value="JPY"/>
20627	<xs:enumeration value="KES"/>
20628	<xs:enumeration value="KGS"/>
20629	<xs:enumeration value="KHR"/>
20630	<xs:enumeration value="KMF"/>
20631	<xs:enumeration value="KPW"/>

20632	<xs:enumeration value="KRW"/>
20633	<xs:enumeration value="KWD"/>
20634	<xs:enumeration value="KYD"/>
20635	<xs:enumeration value="KZT"/>
20636	<xs:enumeration value="LAK"/>
20637	<xs:enumeration value="LBP"/>
20638	<xs:enumeration value="LKR"/>
20639	<xs:enumeration value="LRD"/>
20640	<xs:enumeration value="LSL"/>
20641	<xs:enumeration value="LYD"/>
20642	<xs:enumeration value="MAD"/>
20643	<xs:enumeration value="MDL"/>
20644	<xs:enumeration value="MGA"/>
20645	<xs:enumeration value="MKD"/>
20646	<xs:enumeration value="MMK"/>
20647	<xs:enumeration value="MNT"/>
20648	<xs:enumeration value="MOP"/>
20649	<xs:enumeration value="MRO"/>
20650	<xs:enumeration value="MUR"/>
20651	<xs:enumeration value="MVR"/>
20652	<xs:enumeration value="MWK"/>
20653	<xs:enumeration value="MXN"/>
20654	<xs:enumeration value="MXV"/>
20655	<xs:enumeration value="MYR"/>
20656	<xs:enumeration value="MZN"/>
20657	<xs:enumeration value="NAD"/>
20658	<xs:enumeration value="NGN"/>
20659	<xs:enumeration value="NIO"/>
20660	<xs:enumeration value="NOK"/>
20661	<xs:enumeration value="NPR"/>
20662	<xs:enumeration value="NZD"/>
20663	<xs:enumeration value="OMR"/>
20664	<xs:enumeration value="PAB"/>
20665	<xs:enumeration value="PEN"/>
20666	<xs:enumeration value="PGK"/>
20667	<xs:enumeration value="PHP"/>
20668	<xs:enumeration value="PKR"/>
20669	<xs:enumeration value="PLN"/>
20670	<xs:enumeration value="PYG"/>
20671	<xs:enumeration value="QAR"/>
20672	<xs:enumeration value="RON"/>
20673	<xs:enumeration value="RSD"/>
20674	<xs:enumeration value="RUB"/>
20675	<xs:enumeration value="RWF"/>
20676	<xs:enumeration value="SAR"/>
20677	<xs:enumeration value="SBD"/>
20678	<xs:enumeration value="SCR"/>
20679	<xs:enumeration value="SDG"/>
20680	<xs:enumeration value="SEK"/>
20681	<xs:enumeration value="SGD"/>
20682	<xs:enumeration value="SHP"/>
20683	<xs:enumeration value="SLL"/>
20684	<xs:enumeration value="SOS"/>
20685	<xs:enumeration value="SRD"/>
20686	<xs:enumeration value="SSP"/>
20687	<xs:enumeration value="STD"/>
20688	<xs:enumeration value="SVC"/>
20689	<xs:enumeration value="SYP"/>
20690	<xs:enumeration value="SZL"/>
20691	<xs:enumeration value="THB"/>
20692	<xs:enumeration value="TJS"/>
20693	<xs:enumeration value="TMT"/>
20694	<xs:enumeration value="TND"/>
20695	<xs:enumeration value="TOP"/>
20696	<xs:enumeration value="TRY"/>
20697	<xs:enumeration value="TTD"/>
20698	<xs:enumeration value="TWD"/>
20699	<xs:enumeration value="TZS"/>
20700	<xs:enumeration value="UAH"/>
20701	<xs:enumeration value="UGX"/>
20702	<xs:enumeration value="USD"/>
20703	<xs:enumeration value="USN"/>
20704	<xs:enumeration value="UYI"/>
20705	<xs:enumeration value="UYU"/>
20706	<xs:enumeration value="UZS"/>
20707	<xs:enumeration value="VEF"/>
20708	<xs:enumeration value="VND"/>
20709	<xs:enumeration value="VUV"/>

```

20710      <xs:enumeration value="WST"/>
20711      <xs:enumeration value="XAF"/>
20712      <xs:enumeration value="XAG"/>
20713      <xs:enumeration value="XAU"/>
20714      <xs:enumeration value="XBA"/>
20715      <xs:enumeration value="XBB"/>
20716      <xs:enumeration value="XBC"/>
20717      <xs:enumeration value="XBD"/>
20718      <xs:enumeration value="XCD"/>
20719      <xs:enumeration value="XDR"/>
20720      <xs:enumeration value="XOF"/>
20721      <xs:enumeration value="XPD"/>
20722      <xs:enumeration value="XPF"/>
20723      <xs:enumeration value="XPT"/>
20724      <xs:enumeration value="XSU"/>
20725      <xs:enumeration value="XTS"/>
20726      <xs:enumeration value="XUA"/>
20727      <xs:enumeration value="XXX"/>
20728      <xs:enumeration value="YER"/>
20729      <xs:enumeration value="ZAR"/>
20730      <xs:enumeration value="ZMW"/>
20731      <xs:enumeration value="ZWL"/>
20732    </xs:restriction>
20733  </xs:simpleType>
20734  <xs:simpleType name="AddressDeviceType">
20735    <xs:restriction base="xs:string">
20736      <xs:pattern value="d:_(i:[1-9][0-9]*|n:[a-zA-Z0-9-]+)_([^\p{Cc}\p{Cf}\p{Z}])+"/>
20737      <xs:maxLength value="256"/>
20738    </xs:restriction>
20739  </xs:simpleType>
20740  <xs:simpleType name="AddressEntityType">
20741    <xs:restriction base="xs:unsignedInt"/>
20742  </xs:simpleType>
20743  <xs:simpleType name="AddressFeatureType">
20744    <xs:restriction base="xs:unsignedInt"/>
20745  </xs:simpleType>
20746  <xs:element name="device" type="ns_p:AddressDeviceType"/>
20747  <xs:element name="entity" type="ns_p:AddressEntityType"/>
20748  <xs:element name="feature" type="ns_p:AddressFeatureType"/>
20749  <xs:complexType name="DeviceAddressType">
20750    <xs:sequence>
20751      <xs:element minOccurs="0" ref="ns_p:device"/>
20752    </xs:sequence>
20753  </xs:complexType>
20754  <xs:complexType name="DeviceAddressElementsType">
20755    <xs:sequence>
20756      <xs:element minOccurs="0" name="device" type="ns_p:ElementTagType"/>
20757    </xs:sequence>
20758  </xs:complexType>
20759  <xs:complexType name="EntityAddressType">
20760    <xs:complexContent>
20761      <xs:extension base="ns_p:DeviceAddressType">
20762        <xs:sequence>
20763          <xs:element minOccurs="0" ref="ns_p:entity" maxOccurs="unbounded"/>
20764        </xs:sequence>
20765      </xs:extension>
20766    </xs:complexContent>
20767  </xs:complexType>
20768  <xs:complexType name="EntityAddressElementsType">
20769    <xs:complexContent>
20770      <xs:extension base="ns_p:DeviceAddressElementsType">
20771        <xs:sequence>
20772          <xs:element minOccurs="0" name="entity" type="ns_p:ElementTagType"/>
20773        </xs:sequence>
20774      </xs:extension>
20775    </xs:complexContent>
20776  </xs:complexType>
20777  <xs:complexType name="FeatureAddressType">
20778    <xs:complexContent>
20779      <xs:extension base="ns_p:EntityAddressType">
20780        <xs:sequence>
20781          <xs:element minOccurs="0" ref="ns_p:feature"/>
20782        </xs:sequence>
20783      </xs:extension>
20784    </xs:complexContent>
20785  </xs:complexType>
20786  <xs:complexType name="FeatureAddressElementsType">
20787    <xs:complexContent>

```

```
20788     <xs:extension base="ns_p:EntityAddressElementsType">
20789         <xs:sequence>
20790             <xs:element minOccurs="0" name="feature" type="ns_p:ElementTagType"/>
20791         </xs:sequence>
20792     </xs:extension>
20793 </xs:complexType>
20794 </xs:complexType>
20795 <xs:simpleType name="ScopeTypeType">
20796     <xs:union memberTypes="ns_p:ScopeTypeEnumType ns_p:EnumExtendType"/>
20797 </xs:simpleType>
20798 <xs:simpleType name="ScopeTypeEnumType">
20799     <xs:restriction base="xs:string">
20800         <xs:enumeration value="ac"/>
20801         <xs:enumeration value="acCosPhiGrid"/>
20802         <xs:enumeration value="acCurrentA"/>
20803         <xs:enumeration value="acCurrentB"/>
20804         <xs:enumeration value="acCurrentC"/>
20805         <xs:enumeration value="acFrequencyGrid"/>
20806         <xs:enumeration value="acPowerA"/>
20807         <xs:enumeration value="acPowerB"/>
20808         <xs:enumeration value="acPowerC"/>
20809         <xs:enumeration value="acPowerLimitPct"/>
20810         <xs:enumeration value="acPowerTotal"/>
20811         <xs:enumeration value="acVoltageA"/>
20812         <xs:enumeration value="acVoltageB"/>
20813         <xs:enumeration value="acVoltageC"/>
20814         <xs:enumeration value="acYieldDay"/>
20815         <xs:enumeration value="acYieldTotal"/>
20816         <xs:enumeration value="dcCurrent"/>
20817         <xs:enumeration value="dcPower"/>
20818         <xs:enumeration value="dcString1"/>
20819         <xs:enumeration value="dcString2"/>
20820         <xs:enumeration value="dcString3"/>
20821         <xs:enumeration value="dcString4"/>
20822         <xs:enumeration value="dcString5"/>
20823         <xs:enumeration value="dcString6"/>
20824         <xs:enumeration value="dcTotal"/>
20825         <xs:enumeration value="dcVoltage"/>
20826         <xs:enumeration value="dhwTemperature"/>
20827         <xs:enumeration value="flowTemperature"/>
20828         <xs:enumeration value="outsideAirTemperature"/>
20829         <xs:enumeration value="returnTemperature"/>
20830         <xs:enumeration value="roomAirTemperature"/>
20831         <xs:enumeration value="charge"/>
20832         <xs:enumeration value="stateOfCharge"/>
20833         <xs:enumeration value="discharge"/>
20834         <xs:enumeration value="gridConsumption"/>
20835         <xs:enumeration value="gridFeedIn"/>
20836         <xs:enumeration value="selfConsumption"/>
20837         <xs:enumeration value="overloadProtection"/>
20838         <xs:enumeration value="acPower"/>
20839         <xs:enumeration value="acEnergy"/>
20840         <xs:enumeration value="acCurrent"/>
20841         <xs:enumeration value="acVoltage"/>
20842         <xs:enumeration value="batteryControl"/>
20843         <xs:enumeration value="simpleIncentiveTable"/>
20844     </xs:restriction>
20845 </xs:simpleType>
20846 <xs:simpleType name="RoleType">
20847     <xs:restriction base="xs:string">
20848         <xs:enumeration value="client"/>
20849         <xs:enumeration value="server"/>
20850         <xs:enumeration value="special"/>
20851     </xs:restriction>
20852 </xs:simpleType>
20853 <xs:simpleType name="FeatureGroupType">
20854     <xs:restriction base="xs:string">
20855         <xs:pattern value="(#[1-9][0-9]*)*/>
20856         <xs:maxLength value="128"/>
20857     </xs:restriction>
20858 </xs:simpleType>
20859 <xs:simpleType name="DeviceTypeType">
20860     <xs:union memberTypes="ns_p:DeviceTypeEnumType ns_p:EnumExtendType"/>
20861 </xs:simpleType>
20862 <xs:simpleType name="DeviceTypeEnumType">
20863     <xs:restriction base="xs:string">
20864         <xs:enumeration value="Dishwasher"/>
20865         <xs:enumeration value="Dryer"/>
```

```
20866         <xs:enumeration value="EnvironmentSensor"/>
20867         <xs:enumeration value="Generic"/>
20868         <xs:enumeration value="HeatGenerationSystem"/>
20869         <xs:enumeration value="HeatSinkSystem"/>
20870         <xs:enumeration value="HeatStorageSystem"/>
20871         <xs:enumeration value="HVACController"/>
20872         <xs:enumeration value="SubMeter"/>
20873         <xs:enumeration value="Washer"/>
20874         <xs:enumeration value="ElectricitySupplySystem"/>
20875         <xs:enumeration value="EnergyManagementSystem"/>
20876         <xs:enumeration value="Inverter"/>
20877         <xs:enumeration value="ChargingStation"/>
20878     </xs:restriction>
20879 </xs:simpleType>
20880 <xs:simpleType name="EntityTypeType">
20881     <xs:union memberTypes="ns_p:EntityTypeEnumType ns_p:EnumExtendType"/>
20882 </xs:simpleType>
20883 <xs:simpleType name="EntityTypeEnumType">
20884     <xs:restriction base="xs:string">
20885         <xs:enumeration value="Battery"/>
20886         <xs:enumeration value="Compressor"/>
20887         <xs:enumeration value="DeviceInformation"/>
20888         <xs:enumeration value="DHWcircuit"/>
20889         <xs:enumeration value="DHWStorage"/>
20890         <xs:enumeration value="Dishwasher"/>
20891         <xs:enumeration value="Dryer"/>
20892         <xs:enumeration value="ElectricalImmersionHeater"/>
20893         <xs:enumeration value="Fan"/>
20894         <xs:enumeration value="GasHeatingAppliance"/>
20895         <xs:enumeration value="Generic"/>
20896         <xs:enumeration value="HeatingBufferStorage"/>
20897         <xs:enumeration value="HeatingCircuit"/>
20898         <xs:enumeration value="HeatingObject"/>
20899         <xs:enumeration value="HeatingZone"/>
20900         <xs:enumeration value="HeatPumpAppliance"/>
20901         <xs:enumeration value="HeatSinkCircuit"/>
20902         <xs:enumeration value="HeatSourceCircuit"/>
20903         <xs:enumeration value="HeatSourceUnit"/>
20904         <xs:enumeration value="HVACController"/>
20905         <xs:enumeration value="HVACRoom"/>
20906         <xs:enumeration value="InstantDHWHeater"/>
20907         <xs:enumeration value="Inverter"/>
20908         <xs:enumeration value="OilHeatingAppliance"/>
20909         <xs:enumeration value="Pump"/>
20910         <xs:enumeration value="RefrigerantCircuit"/>
20911         <xs:enumeration value="SmartEnergyAppliance"/>
20912         <xs:enumeration value="SolarDHWStorage"/>
20913         <xs:enumeration value="SolarThermalCircuit"/>
20914         <xs:enumeration value="SubMeterElectricity"/>
20915         <xs:enumeration value="TemperatureSensor"/>
20916         <xs:enumeration value="Washer"/>
20917         <xs:enumeration value="BatterySystem"/>
20918         <xs:enumeration value="ElectricityGenerationSystem"/>
20919         <xs:enumeration value="ElectricityStorageSystem"/>
20920         <xs:enumeration value="GridConnectionPointOfPremises"/>
20921         <xs:enumeration value="Household"/>
20922         <xs:enumeration value="PVSystem"/>
20923         <xs:enumeration value="EV"/>
20924         <xs:enumeration value="EVSE"/>
20925         <xs:enumeration value="ChargingOutlet"/>
20926         <xs:enumeration value="CEM"/>
20927     </xs:restriction>
20928 </xs:simpleType>
20929 <xs:simpleType name="FeatureTypeType">
20930     <xs:union memberTypes="ns_p:FeatureTypeEnumType ns_p:EnumExtendType"/>
20931 </xs:simpleType>
20932 <xs:simpleType name="FeatureTypeEnumType">
20933     <xs:restriction base="xs:string">
20934         <xs:enumeration value="ActuatorLevel"/>
20935         <xs:enumeration value="ActuatorSwitch"/>
20936         <xs:enumeration value="Alarm"/>
20937         <xs:enumeration value="DataTunneling"/>
20938         <xs:enumeration value="DeviceClassification"/>
20939         <xs:enumeration value="DeviceDiagnosis"/>
20940         <xs:enumeration value="DirectControl"/>
20941         <xs:enumeration value="ElectricalConnection"/>
20942         <xs:enumeration value="Generic"/>
20943         <xs:enumeration value="HVAC"/>
```

```
20944         <xs:enumeration value="LoadControl"/>
20945         <xs:enumeration value="Measurement"/>
20946         <xs:enumeration value="Messaging"/>
20947         <xs:enumeration value="NetworkManagement"/>
20948         <xs:enumeration value="NodeManagement"/>
20949         <xs:enumeration value="OperatingConstraints"/>
20950         <xs:enumeration value="PowerSequences"/>
20951         <xs:enumeration value="Sensing"/>
20952         <xs:enumeration value="Setpoint"/>
20953         <xs:enumeration value="SmartEnergyManagementPs"/>
20954         <xs:enumeration value="TaskManagement"/>
20955         <xs:enumeration value="Threshold"/>
20956         <xs:enumeration value="TimeInformation"/>
20957         <xs:enumeration value="TimeTable"/>
20958         <xs:enumeration value="DeviceConfiguration"/>
20959         <xs:enumeration value="SupplyCondition"/>
20960         <xs:enumeration value="TimeSeries"/>
20961         <xs:enumeration value="TariffInformation"/>
20962         <xs:enumeration value="IncentiveTable"/>
20963         <xs:enumeration value="Bill"/>
20964         <xs:enumeration value="Identification"/>
20965     </xs:restriction>
20966 </xs:simpleType>
20967 <xs:simpleType name="FeatureSpecificUsageType">
20968     <xs:union memberTypes="ns_p:FeatureSpecificUsageEnumType ns_p:EnumExtendType"/>
20969 </xs:simpleType>
20970 <xs:simpleType name="FeatureSpecificUsageEnumType">
20971     <xs:union
20972         memberTypes="ns_p:FeatureDirectControlSpecificUsageEnumType
20973 ns_p:FeatureHvacSpecificUsageEnumType ns_p:FeatureMeasurementSpecificUsageEnumType
20974 ns_p:FeatureSetpointSpecificUsageEnumType
20975 ns_p:FeatureSmartEnergyManagementPsSpecificUsageEnumType"
20976     />
20977 </xs:simpleType>
20978 <xs:simpleType name="FeatureDirectControlSpecificUsageEnumType">
20979     <xs:restriction base="xs:string">
20980         <xs:enumeration value="History"/>
20981         <xs:enumeration value="RealTime"/>
20982     </xs:restriction>
20983 </xs:simpleType>
20984 <xs:simpleType name="FeatureHvacSpecificUsageEnumType">
20985     <xs:restriction base="xs:string">
20986         <xs:enumeration value="OperationMode"/>
20987         <xs:enumeration value="Overrun"/>
20988     </xs:restriction>
20989 </xs:simpleType>
20990 <xs:simpleType name="FeatureMeasurementSpecificUsageEnumType">
20991     <xs:restriction base="xs:string">
20992         <xs:enumeration value="Contact"/>
20993         <xs:enumeration value="Electrical"/>
20994         <xs:enumeration value="Heat"/>
20995         <xs:enumeration value="Level"/>
20996         <xs:enumeration value="Pressure"/>
20997         <xs:enumeration value="Temperature"/>
20998     </xs:restriction>
20999 </xs:simpleType>
21000 <xs:simpleType name="FeatureSetpointSpecificUsageEnumType">
21001     <xs:restriction base="ns_p:FeatureMeasurementSpecificUsageEnumType"/>
21002 </xs:simpleType>
21003 <xs:simpleType name="FeatureSmartEnergyManagementPsSpecificUsageEnumType">
21004     <xs:restriction base="xs:string">
21005         <xs:enumeration value="FixedForecast"/>
21006         <xs:enumeration value="FlexibleChosenForecast"/>
21007         <xs:enumeration value="FlexibleOptionalForecast"/>
21008         <xs:enumeration value="OptionalSequenceBasedImmediateControl"/>
21009     </xs:restriction>
21010 </xs:simpleType>
21011 <xs:simpleType name="FunctionType">
21012     <xs:union memberTypes="ns_p:FunctionEnumType ns_p:EnumExtendType"/>
21013 </xs:simpleType>
21014 <xs:simpleType name="FunctionEnumType">
21015     <xs:restriction base="xs:string">
21016         <xs:enumeration value="actuatorLevelData"/>
21017         <xs:enumeration value="actuatorLevelDescriptionData"/>
21018         <xs:enumeration value="actuatorSwitchData"/>
21019         <xs:enumeration value="actuatorSwitchDescriptionData"/>
21020         <xs:enumeration value="alarmListData"/>
21021         <xs:enumeration value="bindingManagementDeleteCall"/>
```

```
21022 <xs:enumeration value="bindingManagementEntryListData"/>
21023 <xs:enumeration value="bindingManagementRequestCall"/>
21024 <xs:enumeration value="dataTunnelingCall"/>
21025 <xs:enumeration value="deviceClassificationManufacturerData"/>
21026 <xs:enumeration value="deviceClassificationUserData"/>
21027 <xs:enumeration value="deviceDiagnosisHeartbeatData"/>
21028 <xs:enumeration value="deviceDiagnosisServiceData"/>
21029 <xs:enumeration value="deviceDiagnosisStateData"/>
21030 <xs:enumeration value="directControlActivityListData"/>
21031 <xs:enumeration value="directControlDescriptionData"/>
21032 <xs:enumeration value="electricalConnectionDescriptionListData"/>
21033 <xs:enumeration value="electricalConnectionParameterDescriptionListData"/>
21034 <xs:enumeration value="electricalConnectionStateListData"/>
21035 <xs:enumeration value="hvacOperationModeDescriptionListData"/>
21036 <xs:enumeration value="hvacOverrunDescriptionListData"/>
21037 <xs:enumeration value="hvacOverrunListData"/>
21038 <xs:enumeration value="hvacSystemFunctionDescriptionListData"/>
21039 <xs:enumeration value="hvacSystemFunctionListData"/>
21040 <xs:enumeration value="hvacSystemFunctionOperationModeRelationListData"/>
21041 <xs:enumeration value="hvacSystemFunctionPowerSequenceRelationListData"/>
21042 <xs:enumeration value="hvacSystemFunctionSetpointRelationListData"/>
21043 <xs:enumeration value="loadControlEventListData"/>
21044 <xs:enumeration value="loadControlStateListData"/>
21045 <xs:enumeration value="measurementConstraintsListData"/>
21046 <xs:enumeration value="measurementDescriptionListData"/>
21047 <xs:enumeration value="measurementListData"/>
21048 <xs:enumeration value="measurementThresholdRelationListData"/>
21049 <xs:enumeration value="messagingListData"/>
21050 <xs:enumeration value="networkManagementAbortCall"/>
21051 <xs:enumeration value="networkManagementAddNodeCall"/>
21052 <xs:enumeration value="networkManagementDeviceDescriptionListData"/>
21053 <xs:enumeration value="networkManagementDiscoverCall"/>
21054 <xs:enumeration value="networkManagementEntityDescriptionListData"/>
21055 <xs:enumeration value="networkManagementFeatureDescriptionListData"/>
21056 <xs:enumeration value="networkManagementJoiningModeData"/>
21057 <xs:enumeration value="networkManagementModifyNodeCall"/>
21058 <xs:enumeration value="networkManagementProcessStateData"/>
21059 <xs:enumeration value="networkManagementRemoveNodeCall"/>
21060 <xs:enumeration value="networkManagementReportCandidateData"/>
21061 <xs:enumeration value="networkManagementScanNetworkCall"/>
21062 <xs:enumeration value="nodeManagementBindingData"/>
21063 <xs:enumeration value="nodeManagementBindingDeleteCall"/>
21064 <xs:enumeration value="nodeManagementBindingRequestCall"/>
21065 <xs:enumeration value="nodeManagementDestinationListData"/>
21066 <xs:enumeration value="nodeManagementDetailedDiscoveryData"/>
21067 <xs:enumeration value="nodeManagementSubscriptionData"/>
21068 <xs:enumeration value="nodeManagementSubscriptionDeleteCall"/>
21069 <xs:enumeration value="nodeManagementSubscriptionRequestCall"/>
21070 <xs:enumeration value="operatingConstraintsDurationListData"/>
21071 <xs:enumeration value="operatingConstraintsInterruptListData"/>
21072 <xs:enumeration value="operatingConstraintsPowerDescriptionListData"/>
21073 <xs:enumeration value="operatingConstraintsPowerLevelListData"/>
21074 <xs:enumeration value="operatingConstraintsPowerRangeListData"/>
21075 <xs:enumeration value="operatingConstraintsResumeImplicationListData"/>
21076 <xs:enumeration value="powerSequenceAlternativesRelationListData"/>
21077 <xs:enumeration value="powerSequenceDescriptionListData"/>
21078 <xs:enumeration value="powerSequenceNodeScheduleInformationData"/>
21079 <xs:enumeration value="powerSequencePriceCalculationRequestCall"/>
21080 <xs:enumeration value="powerSequencePriceListData"/>
21081 <xs:enumeration value="powerSequenceScheduleConfigurationRequestCall"/>
21082 <xs:enumeration value="powerSequenceScheduleConstraintsListData"/>
21083 <xs:enumeration value="powerSequenceScheduleListData"/>
21084 <xs:enumeration value="powerSequenceSchedulePreferenceListData"/>
21085 <xs:enumeration value="powerSequenceStateListData"/>
21086 <xs:enumeration value="powerTimeSlotScheduleConstraintsListData"/>
21087 <xs:enumeration value="powerTimeSlotScheduleListData"/>
21088 <xs:enumeration value="powerTimeSlotValueListData"/>
21089 <xs:enumeration value="resultData"/>
21090 <xs:enumeration value="sensingDescriptionData"/>
21091 <xs:enumeration value="sensingListData"/>
21092 <xs:enumeration value="setpointConstraintsListData"/>
21093 <xs:enumeration value="setpointDescriptionListData"/>
21094 <xs:enumeration value="setpointListData"/>
21095 <xs:enumeration value="smartEnergyManagementPsConfigurationRequestCall"/>
21096 <xs:enumeration value="smartEnergyManagementPsData"/>
21097 <xs:enumeration value="smartEnergyManagementPsPriceCalculationRequestCall"/>
21098 <xs:enumeration value="smartEnergyManagementPsPriceData"/>
21099 <xs:enumeration value="specificationVersionListData"/>
```

```

21100      <xs:enumeration value="subscriptionManagementDeleteCall"/>
21101      <xs:enumeration value="subscriptionManagementEntryListData"/>
21102      <xs:enumeration value="subscriptionManagementRequestCall"/>
21103      <xs:enumeration value="supplyConditionDescriptionListData"/>
21104      <xs:enumeration value="supplyConditionListData"/>
21105      <xs:enumeration value="supplyConditionThresholdRelationListData"/>
21106      <xs:enumeration value="taskManagementJobDescriptionListData"/>
21107      <xs:enumeration value="taskManagementJobListData"/>
21108      <xs:enumeration value="taskManagementJobRelationListData"/>
21109      <xs:enumeration value="taskManagementOverviewData"/>
21110      <xs:enumeration value="thresholdConstraintsListData"/>
21111      <xs:enumeration value="thresholdDescriptionListData"/>
21112      <xs:enumeration value="thresholdListData"/>
21113      <xs:enumeration value="timeDistributorData"/>
21114      <xs:enumeration value="timeDistributorEnquiryCall"/>
21115      <xs:enumeration value="timeInformationData"/>
21116      <xs:enumeration value="timePrecisionData"/>
21117      <xs:enumeration value="timeTableConstraintsListData"/>
21118      <xs:enumeration value="timeTableDescriptionListData"/>
21119      <xs:enumeration value="timeTableListData"/>
21120      <xs:enumeration value="deviceConfigurationKeyValueConstraintsListData"/>
21121      <xs:enumeration value="deviceConfigurationKeyValueListData"/>
21122      <xs:enumeration value="deviceConfigurationKeyValueDescriptionListData"/>
21123      <xs:enumeration value="loadControlLimitConstraintsListData"/>
21124      <xs:enumeration value="loadControlLimitDescriptionListData"/>
21125      <xs:enumeration value="loadControlLimitListData"/>
21126      <xs:enumeration value="loadControlNodeData"/>
21127      <xs:enumeration value="timeSeriesConstraintsListData"/>
21128      <xs:enumeration value="timeSeriesDescriptionListData"/>
21129      <xs:enumeration value="timeSeriesListData"/>
21130      <xs:enumeration value="tariffOverallConstraintsData"/>
21131      <xs:enumeration value="tariffListData"/>
21132      <xs:enumeration value="tariffBoundaryRelationListData"/>
21133      <xs:enumeration value="tariffTierRelationListData"/>
21134      <xs:enumeration value="tariffDescriptionListData"/>
21135      <xs:enumeration value="tierBoundaryListData"/>
21136      <xs:enumeration value="tierBoundaryDescriptionListData"/>
21137      <xs:enumeration value="commodityListData"/>
21138      <xs:enumeration value="tierListData"/>
21139      <xs:enumeration value="tierIncentiveRelationListData"/>
21140      <xs:enumeration value="tierDescriptionListData"/>
21141      <xs:enumeration value="incentiveListData"/>
21142      <xs:enumeration value="incentiveDescriptionListData"/>
21143      <xs:enumeration value="incentiveTableData"/>
21144      <xs:enumeration value="incentiveTableDescriptionData"/>
21145      <xs:enumeration value="incentiveTableConstraintsData"/>
21146      <xs:enumeration value="electricalConnectionPermittedValueSetListData"/>
21147      <xs:enumeration value="useCaseInformationListData"/>
21148      <xs:enumeration value="nodeManagementUseCaseData"/>
21149      <xs:enumeration value="billConstraintsListData"/>
21150      <xs:enumeration value="billDescriptionListData"/>
21151      <xs:enumeration value="billListData"/>
21152      <xs:enumeration value="identificationListData"/>
21153    </xs:restriction>
21154  </xs:simpleType>
21155  <xs:complexType name="PossibleOperationsClassifierType">
21156    <xs:sequence>
21157      <xs:element minOccurs="0" name="partial" type="ns_p:ElementTagType"/>
21158    </xs:sequence>
21159  </xs:complexType>
21160  <xs:complexType name="PossibleOperationsReadType">
21161    <xs:complexContent>
21162      <xs:extension base="ns_p:PossibleOperationsClassifierType"/>
21163    </xs:complexContent>
21164  </xs:complexType>
21165  <xs:complexType name="PossibleOperationsWriteType">
21166    <xs:complexContent>
21167      <xs:extension base="ns_p:PossibleOperationsClassifierType"/>
21168    </xs:complexContent>
21169  </xs:complexType>
21170  <xs:complexType name="PossibleOperationsType">
21171    <xs:sequence>
21172      <xs:element name="read" minOccurs="0" type="ns_p:PossibleOperationsReadType"/>
21173      <xs:element name="write" minOccurs="0" type="ns_p:PossibleOperationsWriteType"/>
21174    </xs:sequence>
21175  </xs:complexType>
21176  <xs:complexType name="PossibleOperationsElementsType">
21177    <xs:sequence>

```

```

21178         <xs:element name="read" minOccurs="0" type="ns_p:ElementTagType"/>
21179         <xs:element name="write" minOccurs="0" type="ns_p:ElementTagType"/>
21180     </xs:sequence>
21181 </xs:complexType>
21182 <xs:complexType name="FunctionPropertyType">
21183     <xs:sequence>
21184         <xs:element minOccurs="0" name="function" type="ns_p:FunctionType"/>
21185         <xs:element minOccurs="0" name="possibleOperations"
21186 type="ns_p:PossibleOperationsType"/>
21187     </xs:sequence>
21188 </xs:complexType>
21189 <xs:complexType name="FunctionPropertyElementsType">
21190     <xs:sequence>
21191         <xs:element minOccurs="0" name="function" type="ns_p:ElementTagType"/>
21192         <xs:element minOccurs="0" name="possibleOperations"
21193 type="ns_p:PossibleOperationsElementsType"/>
21194     </xs:sequence>
21195 </xs:complexType>
21196 </xs:schema>
21197
21198

```

21199

21200 A.3.4 Datagram

21201 File name: EEBus_SPINE_TS_Datagram.xsd

```

21202
21203 <?xml version="1.0" encoding="UTF-8"?>
21204 <!--
21205     Smart Premises Interoperable Neutral-Message Exchange (SPINE)
21206     Version 1.1.1
21207     2018-12-21
21208     Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
21209     Source: https://www.eebus.org/en/specifications/
21210 -->
21211 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
21212 xmlns:xs="http://www.w3.org/2001/XMLSchema"
21213 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1" blockDefault="#all"
21214 elementFormDefault="qualified">
21215     <xs:annotation>
21216         <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
21217 e.V. All rights reserved.</xs:documentation>
21218     </xs:annotation>
21219     <xs:include schemaLocation="EEBus_SPINE_TS_CommandFrame.xsd"/>
21220     <xs:complexType name="HeaderType">
21221         <xs:sequence>
21222             <xs:element minOccurs="0" name="specificationVersion"
21223 type="ns_p:SpecificationVersionType"/>
21224             <xs:element minOccurs="0" name="addressSource" type="ns_p:FeatureAddressType"/>
21225             <xs:element minOccurs="0" name="addressDestination"
21226 type="ns_p:FeatureAddressType"/>
21227             <xs:element minOccurs="0" name="addressOriginator"
21228 type="ns_p:FeatureAddressType"/>
21229             <xs:element minOccurs="0" name="msgCounter" type="ns_p:MsgCounterType"/>
21230             <xs:element minOccurs="0" name="msgCounterReference" type="ns_p:MsgCounterType"/>
21231             <xs:element minOccurs="0" name="cmdClassifier" type="ns_p:CmdClassifierType"/>
21232             <xs:element minOccurs="0" name="ackRequest" type="xs:boolean"/>
21233             <xs:element minOccurs="0" name="timestamp"
21234 type="ns_p:AbsoluteOrRelativeTimeType"/>
21235         </xs:sequence>
21236     </xs:complexType>
21237     <xs:element name="header" type="ns_p:HeaderType"/>
21238     <xs:complexType name="PayloadType">
21239         <xs:sequence>
21240             <xs:element maxOccurs="unbounded" minOccurs="0" name="cmd" type="ns_p:CmdType"/>
21241         </xs:sequence>
21242     </xs:complexType>
21243     <xs:element name="payload" type="ns_p:PayloadType"/>
21244     <xs:complexType name="DatagramType">
21245         <xs:sequence>
21246             <xs:element ref="ns_p:header"/>
21247             <xs:element ref="ns_p:payload"/>
21248         </xs:sequence>
21249     </xs:complexType>
21250     <xs:element name="datagram" type="ns_p:DatagramType"/>

```


21251 </xs:schema>
 21252
 21253

21254

21255 **A.3.5 Result**

21256 File name: EEBus_SPINE_TS_Result.xsd

21257
 21258 <?xml version="1.0" encoding="UTF-8"?>
 21259 <!--
 21260 Smart Premises Interoperable Neutral-Message Exchange (SPINE)
 21261 Version 1.1.1
 21262 2018-12-21
 21263 Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
 21264 Source: https://www.eebus.org/en/specifications/
 21265 -->
 21266 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
 21267 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 21268 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1"
 21269 blockDefault="#all" elementFormDefault="qualified">
 21270 <xs:annotation>
 21271 <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
 21272 e.V. All rights reserved.</xs:documentation>
 21273 </xs:annotation>
 21274 <xs:include schemaLocation="EEBus_SPINE_TS_CommonDataTypes.xsd"/>
 21275 <xs:simpleType name="ErrorNumberType">
 21276 <xs:restriction base="xs:unsignedInt"/>
 21277 </xs:simpleType>
 21278 <xs:complexType name="ResultDataType">
 21279 <xs:sequence>
 21280 <xs:element minOccurs="0" name="errorNumber" type="ns_p:ErrorNumberType"/>
 21281 <xs:element minOccurs="0" name="description" type="ns_p:DescriptionType"/>
 21282 </xs:sequence>
 21283 </xs:complexType>
 21284 <xs:element name="resultData" type="ns_p:ResultDataType"/>
 21285 </xs:schema>
 21286
 21287
 21288

21289

21290 **A.3.6 Overview example**

21291 For each SPINE class, an additional *_overview.xsd is defined that references the functions defined
 21292 by this class. The following example shows this for the class ActuatorLevel.

21293 File name: EEBus_SPINE_TS_ActuatorLevel_overview.xsd

21294
 21295 <?xml version="1.0" encoding="UTF-8"?>
 21296 <!--
 21297 Smart Premises Interoperable Neutral-Message Exchange (SPINE)
 21298 Version 1.1.1
 21299 2018-12-21
 21300 Copyright (c) 2018 EEBus Initiative e.V. All Rights Reserved.
 21301 Source: https://www.eebus.org/en/specifications/
 21302 -->
 21303 <xs:schema xmlns:ns_p="http://docs.eebus.org/spine/xsd/v1"
 21304 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 21305 targetNamespace="http://docs.eebus.org/spine/xsd/v1" version="1.1.1" blockDefault="#all"
 21306 elementFormDefault="qualified">
 21307 <xs:annotation>
 21308 <xs:documentation>EEBus SPINE Specification schema. Copyright 2018 EEBus Initiative
 21309 e.V. All rights reserved.</xs:documentation>
 21310 </xs:annotation>
 21311 <xs:include schemaLocation="EEBus_SPINE_TS_ActuatorLevel.xsd"/>
 21312 <xs:group name="group_actuatorLevel">
 21313 <xs:sequence>
 21314 <xs:group ref="ns_p:actuatorLevel"/>
 21315 <xs:group ref="ns_p:actuatorLevelDescription"/>
 21316

```
21316         </xs:sequence>
21317     </xs:group>
21318     <xs:group name="actuatorLevel">
21319         <xs:sequence>
21320             <xs:element ref="ns_p:actuatorLevelData"/>
21321         </xs:sequence>
21322     </xs:group>
21323     <xs:group name="actuatorLevelDescription">
21324         <xs:sequence>
21325             <xs:element ref="ns_p:actuatorLevelDescriptionData"/>
21326         </xs:sequence>
21327     </xs:group>
21328 </xs:schema>
21329
21330
```

21331

Annex B - Resource Definitions Overview

B.1 Introduction

All Well Known Identifiers, defined by the EEBus Initiative e.V., are listed in this chapter. Detailed specifications can be found in the referenced sections.

B.2 Device Types

For detailed descriptions, please consider the corresponding device type definitions.

Device Type	Section	Description
ChargingStation	4.1.1	Represents a Charging Station, typically used in the context of electrical vehicle charging.
Dishwasher	4.1.2	A machine that cleans the dishes.
Dryer	4.1.3	A dryer for clothes (laundry) after they were washed.
ElectricitySupplySystem	4.1.4	Represents a set of sources and storages of electric power of a premises.
EnergyManagementSystem	4.1.5	Represents an Energy Management System (EMS).
EnvironmentSensor	4.1.6	Represents an environment sensor appliance.
Generic	4.1.7	Device with no specific content. May be used for developing, testing or evaluating purposes.
HeatGenerationSystem	4.1.8	Represents a heat generation system consisting of one or more heat generation appliances (for example gas boilers, heat pumps, solar thermal systems).
HeatSinkSystem	4.1.9	Represents the end user relevant HVAC system parts (heating circuits, DHW circuits, ventilation system) and system functions (heating, cooling, DHW, ventilation).
HeatStorageSystem	4.1.10	Represents a system of heat storages consisting of one or more heat storage appliances.
HVACController	4.1.11	Represents (as SPINE Device) the central controller appliance of the HVAC System.
Inverter	4.1.12	Represents a power Inverter.
SubMeter	4.1.13	Metering device for households or devices that is not billing relevant.
Washer	4.1.14	A washer for clothes (laundry).

Table 334: Well known device types

B.3 Entity Types

For detailed descriptions and rules, please consider the corresponding entity type definitions.

Entity Type	Section	Description
Battery	4.2.1	Appliance for storing (electrical) energy.
BatterySystem	4.2.2	A system consisting of one or more battery storages and one or more battery inverters.

CEM	4.2.3	Abbreviation for Customer Energy Manager and typically used to represent an Energy Manager within SPINE.
ChargingOutlet	4.2.4	Represents a Charging Outlet of a Charging Station.
Compressor	4.2.5	Represents the compressor of (e.g.) a heat pump appliance.
DeviceInformation	4.2.6	A special entity representing the complete device! All features that are included here hold information about the device and all entities.
DHWCircuit	4.2.7	Represents all elements for domestic hot water supply like cold water line, domestic hot water line, circulation line, tapping points. Should not include DHW storages, these should be represented as Entity Type "DHWStorage".
DHWStorage	4.2.8	Represents a heat storage appliance for storing domestic hot water.
Dishwasher	4.2.9	A functionality that cleans the dishes.
Dryer	4.2.10	A dryer for clothes (laundry) after they were washed.
ElectricalImmersionHeater	4.2.11	Represents an electrical immersion heater (electrical resistance heater) for example as part of a heat pump appliance, a heat source unit or a DHW storage.
ElectricityGenerationSystem	4.2.12	A system that is capable of producing electric power through local generation.
ElectricityStorageSystem	4.2.13	A system that is capable of storing electric power and releasing the majority of it as electric power.
EV	4.2.14	Represents an Electrical Vehicle.
EVSE	4.2.15	Abbreviation for Electrical Vehicle Supply Equipment, typically used to connect an Electrical Vehicle to a Charging Station.
Fan	4.2.16	Represents an air fan, for example of an air water heat pump unit or a ventilation unit.
GasHeatingAppliance	4.2.17	Represents a heating appliance that generates heat (for example for heating or domestic hot water) by burning natural gas.
Generic	4.2.18	Entity with no specific content. May be used for developing, testing or evaluating purposes. Client functionality may be implemented within this entity, too.
GridConnectionPointOfPremises	4.2.19	The electrical connection point between the utility electricity grid and the electric circuit of a premises.
HeatingBufferStorage	4.2.20	Represents a heat storage appliance for storing heating water.
HeatingCircuit	4.2.21	Represents a single heating circuit for space heating, including flow and return lines, optional mixing units (for mixed heating circuits) and the heat distribution: a) directly to one or more heated rooms (see HVACRoom) b) to one or more heating zones (see HeatingZone).

HeatingObject	4.2.22	A more general entity representing functionalities that convert (electrical) energy into heat.
HeatingZone	4.2.23	Represents a heating zone that provides heating energy to one or more heated rooms (see HVACRoom) One or more heating zones can be part of a heating circuit.
HeatPumpAppliance	4.2.24	Represents either a complete heat pump appliance (for example a brine water heat pump, indoor; an air water heat pump, completely indoor or completely outdoor) or the indoor part of a refrigerant split heat pump combination.
HeatSinkCircuit	4.2.25	Represents the circuit of a heat generator that delivers the heat energy to the heat sinks (for example to a heating circuit).
HeatSourceCircuit	4.2.26	Represents the circuit that delivers heat energy from an environmental heat source to a heat pump appliance (for example a brine circuit).
HeatSourceUnit	4.2.27	Represents an additional unit that provides the heat from the environmental heat source to the heat pump appliance (for example an outdoor ventilation unit that transfers the heat energy from the outside air to the heat source circuit of the heat pump appliance).
Household	4.2.28	A more virtual entity representing a complete household.
HVACController	4.2.29	Represents the central controller appliance of the HVAC System (as a SPINE Entity).
HVACRoom	4.2.30	Represents one physical room in a building/house that can be heated (H) / cooled, ventilated (V) or air conditioned (AC).
InstantDHWHeater	4.2.31	Represents an appliance for instant generation of domestic hot water (for example by using heat from a heating buffer storage).
Inverter	4.2.32	Represents an appliance that converts the dc-voltage (e.g. of a PV string) into ac-voltage.
OilHeatingAppliance	4.2.33	Represents a heating appliance that generates heat (for example for heating or domestic hot water) by burning oil.
Pump	4.2.34	Represents a pump to generate a flow rate (for example in a heating circuit).
PVSystem	4.2.35	A system consisting of one or more PV inverters with one or more PV modules. The PVSystem provides aggregated PV related system parts of the house or premises.
RefrigerantCircuit	4.2.36	Represents the Refrigerant circuit of e.g. a heat pump appliance or a fridge or freezer.
SmartEnergyAppliance	4.2.37	Represents an appliance that allows shifting of its load via the SmartEnergyManagementPs concepts.
SolarDHWStorage	4.2.38	Represents a heat storage appliance for storing domestic hot water. The heat energy can be provided especially by solar thermal energy.

SolarThermalCircuit	4.2.39	Represents a single circuit of a solar thermal system including a solar thermal collector field and flow and return pipes to/from the collector field and pumps.
SubMeterElectricity	4.2.40	Electricity metering for households or functionalities that is not billing relevant.
TemperatureSensor	4.2.41	Represents a sensor that can provide a temperature measurement value.
Washer	4.2.42	A washer for clothes (laundry).

Table 335: Well known entity types

B.4 Feature Types

For detailed descriptions and rules, please consider the corresponding feature type definitions.

Feature Type	Section	Description
ActuatorLevel	4.3.1	Standard Feature Type for the Standard Class ActuatorLevel.
ActuatorSwitch	4.3.2	Standard Feature Type for the Standard Class ActuatorSwitch.
Alarm	4.3.3	Standard Feature Type for the Standard Class Alarm.
Bill	4.3.4	Standard Feature Type for the Standard Class Bill.
DataTunneling	4.3.5	Standard Feature Type for the Standard Class DataTunneling.
DeviceClassification	4.3.6	Standard Feature Type for the Standard Class DeviceClassification.
DeviceConfiguration	4.3.7	Standard Feature Type for the Standard Class DeviceConfiguration.
DeviceDiagnosis	4.3.8	Standard Feature Type for the Standard Class DeviceDiagnosis.
DirectControl	4.3.9	Standard Feature Type for the Standard Class DirectControl.
ElectricalConnection	4.3.10	Standard Feature Type for the Standard Class ElectricalConnection.
Generic	4.3.11	Feature with no specific content. May be used for developing, testing or evaluating purposes. Client functionality may be implemented within this feature, too.
HVAC	4.3.12	Standard Feature Type for the Standard Class HVAC.
Identification	4.3.13	Standard Feature Type for the Standard Class Identification.
IncentiveTable	4.3.14	Standard Feature Type for the Complex Class IncentiveTable.
LoadControl	4.3.15	Standard Feature Type for the Standard Class LoadControl.
Measurement	4.3.16	Standard Feature Type for the Standard Class Measurement.
Messaging	4.3.17	Standard Feature Type for the Standard Class Messaging.

NetworkManagement	4.3.18	Standard Feature Type for the Standard Class NetworkManagement.
NodeManagement	4.3.19	Standard Feature Type for the Complex Class NodeManagement.
OperatingConstraints	4.3.20	Standard Feature Type for the Standard Class OperatingConstraints.
PowerSequences	4.3.21	Standard Feature Type for the Standard Class PowerSequences.
Sensing	4.3.22	Standard Feature Type for the Standard Class Sensing.
Setpoint	4.3.23	Standard Feature Type for the Standard Class Setpoint.
SmartEnergyManagementPs	4.3.24	Standard Feature Type for the Complex Class SmartEnergyManagementPs.
SupplyCondition	4.3.25	Standard Feature Type for the Standard Class SupplyCondition.
TariffInformation	4.3.26	Standard Feature Type for the Standard Class TariffInformation.
TaskManagement	4.3.27	Standard Feature Type for the Standard Class TaskManagement.
Threshold	4.3.28	Standard Feature Type for the Standard Class Threshold.
TimeInformation	4.3.29	Standard Feature Type for the Standard Class TimeInformation.
TimeSeries	4.3.30	Standard Feature Type for the Standard Class TimeSeries.
TimeTable	4.3.31	Standard Feature Type for the Standard Class TimeTable.

Table 336: Well known feature types

B.5 Complex Classes

For detailed descriptions and rules, please consider the corresponding complex class definitions.

Complex Class	Section	Description
IncentiveTable	5.2.1	The IncentiveTable data model allows to define tariffs with tiers and different incentives and boundaries.
NodeManagement	5.2.2	All data needed to manage nodes is combined in this complex class. Functions from the standard classes <i>NetworkManagement</i> , <i>BindingManagement</i> , <i>SubscriptionManagement</i> and <i>Version</i> are used.
SmartEnergyManagementPs	5.2.3	Shifting the load of devices in some given constraints is done with this complex class. Functions from the standard classes <i>PowerSequences</i> and <i>OperatingConstraints</i> are used.

Table 337: Well known complex classes

B.6 Standard Classes

For detailed descriptions and rules, please consider the corresponding standard class definitions.

Standard Class	Section	Description
ActuatorLevel	5.3.1	Dimming actuators, actuators with adjustability in stages.
ActuatorSwitch	5.3.2	Switch actuators.
Alarm	5.3.3	Alarms, generated due to exceeding thresholds (see corresponding class) or other events.
Bill	5.3.4	Bills for costs and profits of different purposes (e.g. a charging summary).
BindingManagement	5.3.5	Request / delete bindings or list existing bindings.
DataTunneling	5.3.6	Tunnels any (unspecified) data.
DeviceClassification	5.3.7	Free text information about a device (name, serial, etc.) and some enumerations like device type.
DeviceConfiguration	5.3.8	Used to model configuration data needed for interoperable functionality not fitting any other SPINE class.
DeviceDiagnosis	5.3.9	Information about the state of a device, heartbeat counting and service data.
DirectControl	5.3.10	Notification of instantaneous consumption/generation as well as instantaneous control.
ElectricalConnection	5.3.11	Information about specific parameters used for electrical connections, with relation to Measurement.
HVAC	5.3.12	Monitoring and controlling HVAC system functions (heating, cooling, ventilation, air conditioning, ...) and corresponding operation modes and overrun functionalities.
Identification	5.3.13	The Identification Class can be used to express certain identification values of the parent Entity.
LoadControl	5.3.14	Enabling (partial) (external) management of the load of a household or device by (e.g.) a DSO or some energy manager.
Measurement	5.3.15	General measurements (e.g. temperature, voltage, ...).
Messaging	5.3.16	Text messages (such as error messages, information, warnings, ...).
NetworkManagement	5.3.17	Add devices, remove etc., List of Inventory.
OperatingConstraints	5.3.18	Information on the power/energy constraints and implications.
PowerSequences	5.3.19	PowerSequences can be used to model energy management on a schedule basis.
Sensing	5.3.20	Various sensors, switches, contact sensors (e.g. window contact), smoke detectors, etc.
Setpoint	5.3.21	Setting desired values to control some functionality (with reference to Measurement and TimeTable).
SubscriptionManagement	5.3.22	Request / delete subscriptions or list existing subscriptions.
SupplyCondition	5.3.23	Condition of power supply, limitations, etc.
TariffInformation	5.3.24	The TariffInformation Class provides functions for modelling tariffs.
TaskManagement	5.3.25	General status of device- or entity-wide tasks (triggered by user, internal mechanisms or external configuration).
Threshold	5.3.26	Thresholds for automatic behaviour of a device (may generate alarms for exceeding values).
TimeInformation	5.3.27	Time information (including the approximate synchronisation).
TimeSeries	5.3.28	TimeSeries is a versatile class that allows to express time dependent values as a time series of values.

TimeTable	5.3.29	Providing some amount of tables and slots to be used by other classes (e.g. Setpoint).
UseCaseInformation	5.3.30	To model which Use Cases are supported by a node the UseCaseInformation Class can be used.
Version	5.3.31	SPINE version information.

Table 338: Well known standard classes

B.7 Identifier list

The following table list all identifiers that belong to the identifiers concept as described in section 3.4.

Type Name	Used in elements	Base Type
AbsoluteOrRelativeTimeType	timestamp	<i>Union of xs:duration and xs:dateTime</i>
AddressDeviceType	device	xs:string
AddressEntityType	entity	xs:unsignedInt
AddressFeatureType	feature	xs:unsignedInt
AlarmIdType	alarmId	xs:unsignedInt
AlternativesIdType	alternativesId	xs:unsignedInt
BillCostIdType	costId	xs:unsignedInt
BillIdType	billId	xs:unsignedInt
BillPositionIdType	positionId	xs:unsignedInt
BillValueIdType	valueId	xs:unsignedInt
BindingIdType	bindingId	xs:unsignedInt
CommodityIdType	commodityId	xs:unsignedInt
ConditionIdType	conditionId	xs:unsignedInt
DeviceConfigurationKeyIdType	keyId	xs:unsignedInt
ElectricalConnectionIdType	electricalConnectionId	xs:unsignedInt
ElectricalConnectionParameterIdType	parameterId	xs:unsignedInt
HvacOperationModelIdType	operationModelId	xs:unsignedInt
HvacOverrunIdType	overrunId	xs:unsignedInt
HvacSystemFunctionIdType	systemFunctionId	xs:unsignedInt
IdentificationIdType	identificationId	xs:unsignedInt
IncentiveIdType	incentiveId	xs:unsignedInt
LoadControlEventIdType	eventId	xs:unsignedInt
LoadControlLimitIdType	limitId	xs:unsignedInt
MeasurementIdType	measurementId	xs:unsignedInt
MessagingNumberType	messagingNumber	xs:unsignedInt
PowerSequenceIdType	sequenceId	xs:unsignedInt
PowerTimeSlotNumberType	slotNumber	xs:unsignedInt
SetpointIdType	setpointId	xs:unsignedInt
SubscriptionIdType	subscriptionId	xs:unsignedInt
TariffIdType	tariffId	xs:unsignedInt
TaskManagementJobIdType	jobId	xs:unsignedInt
ThresholdIdType	thresholdId	xs:unsignedInt

TierBoundaryIdType	boundaryId	xs:unsignedInt
TierIdType	tierId	xs:unsignedInt
TimeSeriesIdType	timeSeriesId	xs:unsignedInt
TimeSeriesSlotIdType	timeSeriesSlotId	xs:unsignedInt
TimeSlotIdType	timeSlotId	xs:unsignedInt
TimeTableIdType	timeTableId	xs:unsignedInt

Table 339: Identifier overview