

EEBus UC Technical Specification

EVSE Commissioning and Configuration

Version 1.0.1

Cologne, 2019-07-01

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

Terms of use for publications of EEBus Initiative e.V.**General information**

The specifications, particulars, documents, publications and other information provided by the EEBus Initiative e.V. are solely for general informational purposes. Particularly specifications that have not been submitted to national or international standardisation organisations by EEBus Initiative e.V. (such as DKE/DIN-VDE or IEC/CENELEC/ETSI) are versions that have not yet undergone complete testing and can therefore only be considered as preliminary information. Even versions that have already been published via standardisation organisations can contain errors and will undergo further improvements and updates in future.

Liability

EEBus Initiative e.V. does not assume liability or provide a guarantee for the accuracy, completeness or up-to-date status of any specifications, data, documents, publications or other information provided and particularly for the functionality of any developments based on the above.

Copyright, rights of use and exploitation

The specifications provided are protected by copyright. Parts of the specifications have been submitted to national or international standardisation organisations by EEBus Initiative e.V., such as DKE/DIN-VDE or IEC/CENELEC/ETSI, etc. Furthermore, all rights to use and/or exploit the specifications belong to the EEBus Initiative e.V., Butzweilerhof-Allee 4, 50829 Cologne, Germany and can be used in accordance with the following regulations:

The use of the specifications for informational purposes is allowed. It is therefore permitted to use information evident from the contents of the specifications. In particular, the user is permitted to offer products, developments and/or services based on the specifications.

Any respective use relating to standardisation measures by the user or third parties is prohibited. In fact, the specifications may only be used by EEBus Initiative e.V. for standardisation purposes. The same applies to their use within the framework of alliances and/or cooperations that pursue the aim of determining uniform standards.

Any use not in accordance with the purpose intended by EEBus Initiative e.V. is also prohibited.

Furthermore, it is prohibited to edit, change or falsify the content of the specifications. The dissemination of the specifications in a changed, revised or falsified form is also prohibited. The same applies to the publication of extracts if they distort the literal meaning of the specifications as a whole.

It is prohibited to pass on the specifications to third parties without reference to these rights of use and exploitation.

It is also prohibited to pass on the specifications to third parties without informing them of the authorship or source.

Without the prior consent of EEBus Initiative e.V., all forms of use and exploitation not explicitly stated above are prohibited.

Table of contents

Table of contents.....	3
List of figures	4
List of tables	4
1 Scope of the document	5
1.1 References.....	5
1.1.1 EEBUS documents	5
1.1.2 Normative references.....	5
1.2 Terms and definitions.....	5
1.3 Requirements	6
1.3.1 Requirements wording.....	6
1.3.2 Mapping of High-Level requirements.....	6
2 High-Level description.....	7
2.1 Introduction.....	7
2.2 Actors	7
2.2.1 EVSE	7
2.2.2 CEM	7
2.3 Scenarios	7
2.3.1 Scenario 1 - EVSE sends manufacturer information.....	8
2.3.2 Scenario 2 - EVSE sends error state.....	9
2.4 Dependencies to other Use Cases.....	9
2.5 Assumptions and Prerequisites.....	9
3 Technical SPINE solution	10
3.1 General rules and information	10
3.1.1 Underlying technology documents	10
3.1.2 Use Case Discovery rules.....	10
3.1.3 Rules for "Content of Specialization..." tables and "Content of Function..." tables	11
3.1.4 Rules for "Feature Types and Functions..." tables	17
3.1.5 "Actor ... overview" diagram rules	18
3.1.6 Specializations	19
3.1.7 Order of messages within the sequence diagrams	20
3.1.8 Further information and rules.....	20
3.2 Actors	20
3.2.1 EVSE	20
3.2.2 CEM	24
3.3 Pre-Scenario communication	27
3.3.1 General information	27
3.3.2 Detailed discovery	28
3.3.3 Binding.....	30
3.3.4 Subscription.....	30
3.3.5 Dynamic behaviour.....	31
3.4 Scenarios	31
3.4.1 Scenario 1 - EVSE sends manufacturer information.....	31
3.4.2 Scenario 2 - EVSE sends error state.....	33

List of figures

Figure 1: High-Level Use Case functionality overview	7
Figure 2: Scenario overview	7
Figure 3: Scenario 1 overview	8
Figure 4: Scenario 2 overview	9
Figure 5: Actor overview example.....	19
Figure 6: Actor "EVSE" overview	21
Figure 7: Actor "CEM" overview.....	25
Figure 8: Pre-Scenario communication - Detailed discovery sequence diagram.....	29
Figure 9: Pre-Scenario communication - Binding sequence diagram	30
Figure 10: Pre-Scenario communication - Subscription sequence diagram	31
Figure 11: Scenario 1 - Initial Scenario communication sequence diagram	32
Figure 12: Scenario 1 - Runtime Scenario communication sequence diagram.....	33
Figure 13: Scenario 1 - Initial Scenario communication sequence diagram	34
Figure 14: Scenario 1 - Runtime Scenario communication sequence diagram.....	35

List of tables

Table 1: Scenario implementation requirement for Actors	8
Table 2: Presence indication description	11
Table 3: Example table for cardinality indications	13
Table 4: Presence indication of Feature Types and Functions support	17
Table 5: Feature Types and Functions used within this Use Case by the Actor EVSE	22
Table 6: Content of Function "deviceClassificationManufacturerData" at Actor EVSE	24
Table 7: Content of Function "deviceDiagnosisStateData" at Actor EVSE	24
Table 8: Content of Specialization "DeviceClassification_ManufacturerData" at Actor CEM	27
Table 9: Content of Specialization "DeviceDiagnosis_FailureState" at Actor CEM.....	27
Table 10: Initial Scenario communication content references for Scenario 1	32
Table 11: Runtime Scenario communication content references for Scenario 1	33
Table 12: Initial Scenario communication content references for Scenario 2	34
Table 13: Runtime Scenario communication content references for Scenario 2	35

1 Scope of the document

This document describes the Use Case "EVSE Commissioning and Configuration" (short-name: EVSECC). Chapter 2 specifies the High-Level Use Case. Chapter 3 describes the technical solution for SPINE for this Use Case in detail. Within this document a top-down approach is used to derive the requirements for the technical solution from the High-Level description.

1.1 References

1.1.1 EEBUS documents

[UseCaseBaseSpecification] EEBus_UC_TS_UseCaseBaseSpecification.pdf

[ProtocolSpecification] EEBus_SPINE_TS_ProtocolSpecification.pdf

[ResourceSpecification] EEBus_SPINE_TS_ResourceSpecification.pdf

[SHIP] SHIP_Specification_v1.0.0.pdf

1.1.2 Normative references

[RFC2119] IETF RFC 2119: 1997, Key words for use in RFCs to indicate requirement levels
Please see section 1.3.1 for details.

1.2 Terms and definitions

Actor

An Actor models a role within a Use Case definition (e.g. an energy manager or an electric vehicle).

CEM

Abbreviation for Customer Energy Manager. The CEM is an energy manager located at the home or premises of the user or in a cloud application. The energy manager enables energy-optimized operation of the connected devices by harmonising energy demand and availability.

EV

Electric Vehicle

EVSE

Electric Vehicle Supply Equipment

EVSECC

EVSE Commissioning and Configuration (short name of this Use Case)

Scenario

Part of the Use Case. Splitting a Use Case in Scenarios helps to understand the Use Case more quickly. Some Scenarios are mandatory for a Use Case, whereas others may be recommended or optional.

114 Specialization

115 Reusable data collection for a specific functionality.

116 SPINE

117 **Smart Premises Interoperable Neutral-message Exchange: Technical Specification of EEBus Initiative**
118 **e.V.**

119

120 1.3 Requirements**121 1.3.1 Requirements wording**

122 The following keywords are used:

- 123 - SHALL
- 124 - SHALL NOT
- 125 - SHOULD
- 126 - SHOULD NOT
- 127 - MAY

128 Note: They apply only if written in capital letters.

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

130

131 1.3.2 Mapping of High-Level requirements

132 Within the High-Level Use Case description, the following abbreviation is used:

133 [EVSECC-xyz]

134 e.g.: [EVSECC-007]

135 The abbreviation is used to mark High-Level requirements or rules of this Use Case with a unique
136 number xyz. Those requirements are referenced throughout the technical solution to show how each
137 High-Level requirement is realised in the technical part.

138

2 High-Level description

2.1 Introduction

As the basis for other Use Cases related to the support of EV (electric vehicle) charging, this Use Case specifies the initial setup process between the Actor CEM (the energy manager) and the EVSE (the electric vehicle supply equipment). For most related Use Cases, the Actor EVSE (the charging station) serves as middle box that transmits the necessary information between the CEM and the EV.

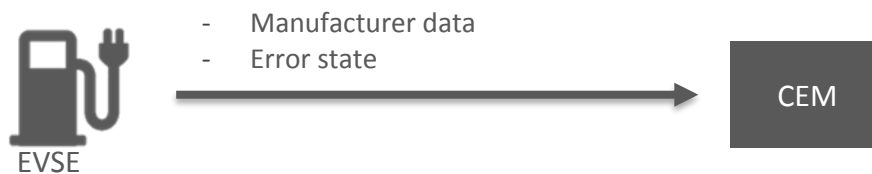


Figure 1: High-Level Use Case functionality overview

The main part of the commissioning and configuration processes is to transmit information from the EVSE to the CEM.

2.2 Actors

2.2.1 EVSE

The Actor EVSE represents the charging station that wants to participate in the local energy management.

2.2.2 CEM

The Actor CEM represents the energy manager that wants to integrate the EVSE into its energy management.

2.3 Scenarios

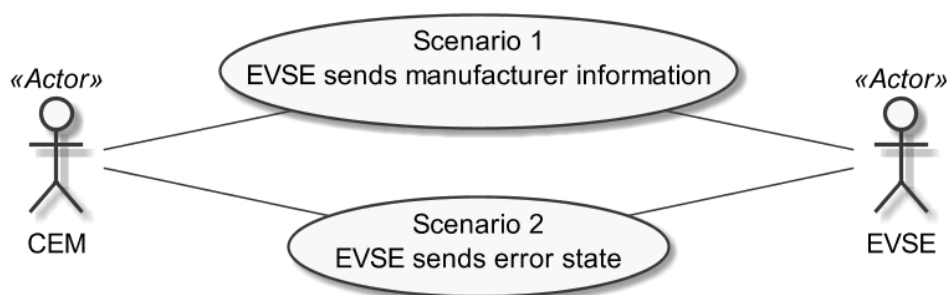


Figure 2: Scenario overview

Scenario number	Scenario name	EVSE	CEM
1	EVSE sends manufacturer information	R	R
2	EVSE sends error state	M	M

Table 1: Scenario implementation requirement for Actors

2.3.1 Scenario 1 - EVSE sends manufacturer information

2.3.1.1 Description

The manufacturer data can hold different information of the corresponding EVSE and the EVSE manufacturer. This can be used to present the corresponding EVSE to a user of the energy management application. E.g. based on the device name a corresponding icon could be used to represent the EVSE.

The following information SHOULD be available:

- device name [EVSECC-010]
- device code [EVSECC-011]
- vendor name [EVSECC-012]
- vendor code [EVSECC-013]
- brand name [EVSECC-014]
- manufacturer label [EVSECC-015]

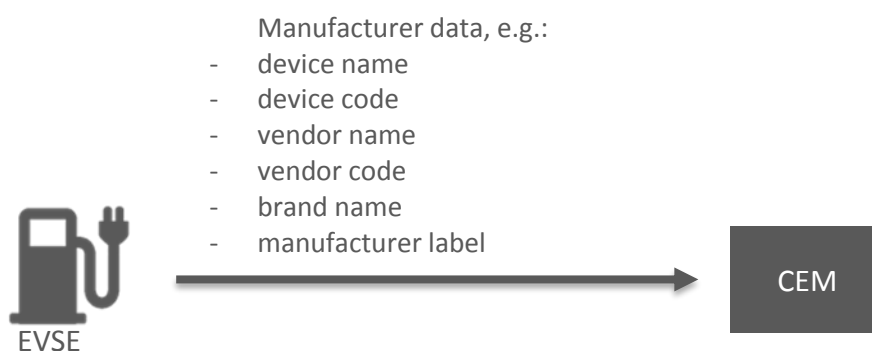


Figure 3: Scenario 1 overview

2.3.1.2 Conditions

Triggering Event:

The Scenario is typically triggered by establishing a communication link between the EVSE and the CEM.

Pre-condition:

The CEM has no manufacturer information of the EVSE.

Post-condition:

The CEM has manufacturer information of the EVSE and e.g. can show this information to the customer.

2.3.2 Scenario 2 - EVSE sends error state**2.3.2.1 Description**

Used to indicate errors of the EVSE to the user. If the EVSE has an error [EVSECC-020], the EV may no longer be able to follow the charging plan correctly and updates from the EV may no longer contain valid data.



Figure 4: Scenario 2 overview

2.3.2.2 Conditions**Triggering Event:**

The Scenario is triggered as soon as the EVSE is connected to the CEM.

Pre-condition:

The CEM has no error state information of the EVSE.

Post-condition:

The CEM has error state information of the EVSE.

2.4 Dependencies to other Use Cases

None.

2.5 Assumptions and Prerequisites

None.

3 Technical SPINE solution

3.1 General rules and information

3.1.1 Underlying technology documents

This technical solution relies on the SPINE Resources Specification version 1.1.1 [ResourceSpecification].

For interoperable connectivity this technical solution relies on:

- SPINE Protocol Specification version 1.1.1 [ProtocolSpecification] as application protocol.
- SHIP Specification version 1.0 [SHIP] as transport protocol.

Further applicable documents:

- EEBUS Use Case Base Specification version 1.0.0 [UseCaseBaseSpecification].

3.1.2 Use Case Discovery rules

The Use Case Discovery SHALL be supported by each Actor and the following rules SHALL apply:

- The string content for the Element "nodeManagementUseCaseData. useCaseInformation. useCaseSupport. useCaseName" within the Use Case Discovery (please refer to [ProtocolSpecification]) SHALL be "evseCommissioningAndConfiguration". The string content SHALL only be defined by this Use Case (regardless of the Use Case version).
- The string content of the Element "nodeManagementUseCaseData. useCaseInformation. actor" within the Use Case Discovery (please refer to [ProtocolSpecification]) SHALL be set to the according value stated within the corresponding Actor's section.
- An Actor A that is implemented to support this Use Case specification SHALL set the Element "nodeManagementUseCaseData. useCaseInformation. useCaseSupport. useCaseVersion" within the Use Case discovery (please refer to [ProtocolSpecification]) to "1.0.1" (for details on the structure of the Use Case version number please refer to [UseCaseBaseSpecification]).
- If an Actor A supports multiple versions of this Use Case with the same major version number, only the highest one SHOULD be set within the Use Case discovery.
- If an Actor A finds a proper counterpart Actor B for this Use Case that supports multiple versions of this Use Case with the same major version number as supported by Actor A, the Actor A SHOULD evaluate from these versions of Actor B only the highest version number.
- If an Actor A supports multiple versions of this Use Case with different major version numbers, for each major version number only the highest version number SHOULD be set within the Use Case discovery.
- If an Actor A finds a proper counterpart Actor B for this Use Case that supports only versions with a major version number not implemented by Actor A, it still might be possible to run the Use Case or parts of the Use Case. Therefore, the Actor A should try to evaluate the Actor B as a valid partner for this Use Case.

3.1.3 Rules for "Content of Specialization..." tables and "Content of Function..." tables

3.1.3.1 General presence indication definitions

Abbreviations for the presence indication of Elements listed in the tables are defined as follows:

Abbreviation	Meaning	Link to requirement keywords
M	Mandatory	SHALL
R	Recommended	SHOULD
O	Optional	MAY

Table 2: Presence indication description

An Actor MAY support Elements that are not listed in the tables. However, another Actor MAY ignore these Elements.

The presence indications "M", "R" and "O" 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.

Note: The indications and the aforementioned rules apply for "complete messages" (so-called "full function exchange", please refer to [ProtocolSpecification]). In contrast, the so-called "restricted function exchange" is designed to permit exchange of specific excerpts of data, i.e. fewer Elements than potentially available from the data owner (partially even not all "mandatory" Elements).

3.1.3.2 Presence indications for "Content of Specialization..." tables

This section only defines rules for the client side.

Elements that are marked with "M" SHALL be supported by the client in case of readable as well as writeable data. This Element may be optional on the server side.

The following applies for readable data that is exchanged in a "read/reply" or "notify" operation:

- "R" means that the data SHOULD be supported by the client. In other words: If the server responds with the according Element, the client SHOULD be able to interpret the according Elements.
- "O" means that the data MAY be supported by the client. In other words: If the server responds with the according Element, the client MAY be able to interpret the according Elements.

The following applies for writeable data that is exchanged in a "write" operation:

- "R" means that the data SHOULD be written by the client.
- "O" means that the data MAY be written by the client.
- "F" means that the data SHALL NOT be written by the client.

The following applies for Elements that are not listed in the Actor section:

- In case of a received "reply" message: The client MAY ignore the Element.
- In case of a "write" operation to be created: The client MAY set the Element but SHALL consider that the server may ignore the Element.

- In case of a received "notify" message: The client MAY ignore the Element.

M, R or O may be combined with the suffix "(event)" to express that a supported Element or value only has to be supported during a certain event and hence does not need to be present at all times. If the event is not active the Element may be omitted or another value may be set. In most cases a High-Level requirement reference for the event is given in the rules column.

3.1.3.3 Presence indications for "Content of Function..." tables

This section only defines rules for the server side.

Elements that are marked with "M" SHALL be supported by the server in case of readable as well as writeable data. In case of writeable data (marked with "M \W") the server does not need to set the Element, because the Element is set only by the client.

The following applies for readable data that is exchanged in a "read/reply" or "notify" operation:

- "R" means that the data SHOULD be provided by the server.
- "O" means that the data MAY be provided by the server.
- "F" means that the data SHALL NOT be provided by the server.

The following applies for writeable data that is exchanged in a "write" operation:

- "R" means that the data SHOULD be supported. In other words: If the client writes the Element, the server SHOULD accept those messages and the contained Elements.
- "O" means that the data MAY be supported. In other words: If the client writes the Element, the server MAY accept those messages and the contained Elements.

The following applies for Elements that are not listed in the Actor section:

- In case of a received "read" request: The according Element MAY be set in the reply.
- In case of a received "write" operation: The server MAY ignore the Element.
- In case of a "notify" operation to be created: The server MAY set the Element.

Note: The server will only accept write operations if the result fulfils the server Function requirements (permitted values, e.g.). Write operations on Elements that are not writeable MAY result in an error message.

M, R or O may be combined with the suffix "(event)" to express that a supported Element or value only has to be supported during a certain event and hence does not need to be present at all times. If the event is not active the Element may be omitted or another value may be set. In most cases a High-Level requirement reference for the event is given in the rules column.

3.1.3.4 Cardinality indications - Permitted number of occurrences

A cardinality indication expresses constraints on the number of occurrences of a given Element or data set. In this section we use "X" as representation for such an Element or data set. Furthermore, "a" and "b" represent constraints. The following rules apply for the occurrence of "X" and its content related to a specific Scenario (see note underneath the list):

1. X
No cardinality indication.
2. X (a..b)
This means "X" SHALL occur at least "a" times and at maximum "b" times.
3. X (a..)
This means "X" SHALL occur at least "a" times and MAY occur more than "a" times.
4. X (..b)
This means "X" SHALL occur at maximum "b" times and MAY occur less than "b" times (even zero occurrences are permissive).

Note: These rules apply only under consideration of presence indications and with regards to the given Scenario or Function definition for this Use Case.

The following table is an example to explain this for two different placements.

Scenario [{...}]: M/R/O [W][C]	Element	Value	[High Level Mapping] Element and value rules
1: O
2: M \W	xFeatureType. xListData. xData. [UC-002] (1..3)		
2: M \W	xId	<g7> [<g8>] [<g9>]	PRIMARY IDENTIFIER of x
2: M \W	timePeriod		...
2: M \W	timePeriod. startTime	<xs:duration>	
2: M \W	xSlot. (1..)		
2: M \W	xSlot. xSlotId		...
2: M \W	xSlot. duration	<xs:duration>	...
2: M \W	qId	<h3>(-><g7>) [<h4>(-><g8>)] [<h5>(-><g9>)]	FOREIGN IDENTIFIER.
...

Table 3: Example table for cardinality indications

The field

xFeatureType. xListData. xData. [UC-002] (1..3)

introduces a data pattern (required Elements and values) for "xData" instances used for Scenario 2. The field itself specifies that such an "xData" instance SHALL occur at least 1 time and at maximum 3 times within "xListData" of Feature Type "xFeatureType". However, this holds only for Scenario 2 and only if such "xData" are required. In this case, they are required, as the left field

2: M \W

denotes that this data set is mandatory for Scenario 2. The "Value" definition

<g7> [<g8>] [<g9>]

of the Element "xId" specifies that this is the reason for the cardinality: There must be at least one "xData" instance and the corresponding "Value" placeholder is "<g7>" (see section 3.1.3.6 for the definition of "Value" placeholders). The second and third instance of "xData" are optional, as the corresponding placeholders "<g8>" and "<g9>" are put in brackets. Of course, the placeholders SHALL then have distinct values.

The "Value" definition of the Element "qId" contains the expression

```
<h3>(-><g7>) [<h4>(-><g8>)] [<h5>(-><g9>)]
```

This means that the placeholder "<h3>" is to be used with "<g7>". Likewise, "<h4>" is associated with "<g8>" and "<h5>" is associated with "<g9>".

Some Scenarios may require the association to two or more placeholders. As an example, we consider an expression

```
<t2>(-><v1>,<k3>)
```

In this case the placeholder "<t2>" is to be used with the pair of "<v1>" and "<k3>".

The field

```
xSlot. (1..)
```

expresses that the Element "xSlot" SHALL occur at least one time within its "xData", but MAY occur more than one time.

The remaining fields do not have an explicit cardinality indication.

3.1.3.5 Writability and changeability indication

In the same column where the presence indications are denoted, a mark is used to distinguish between writeable, changeable or readable Elements:

- Elements that are marked with "\W" are written by a client and SHALL be writeable at the server according to their presence indications. The client is not obliged to read the according data. Received notifications do not need to be evaluated.
- Elements that are marked with "\C" are changed by a client and SHALL be changeable at the server according to their presence indications. The client is not obliged to read the according data. Received notifications do not need to be evaluated.
- Elements that are marked with "\RW" are read and written by a client and SHALL be writeable and provided by the server according to their presence indications. Received notifications SHALL be evaluated according to their presence indications.
- Elements that are marked with "\RC" are read and changed by a client and SHALL be changeable and provided by the server according to their presence indications. Received notifications SHALL be evaluated according to their presence indications.
- Elements that are not marked are only read by a client and SHALL be provided by the server according to their presence indications. Received notifications SHALL be evaluated according to their presence indications.

"Writeable" means that the Element and its value may be written by a client. This includes the possibility to modify (if the Element is already present), create (if the Element is not present yet), and delete the Element. The server SHALL adjust its Function according to the received "write" operation (unless the server cannot accept the "write" operation according to section 3.1.3.3).

"Changeable" means that the Element's value may be changed by a client. If the Element is not present at the resource before, it probably **cannot** be created by the client via the "write" operation. In this case the server MAY decline such a message.

Note: "\W" includes "\C" already.

Note: Depending on the resource a client might need to request a proper binding before the server accepts a "write" operation.

3.1.3.6 Rules for "Value" placeholders

If the "Value" column contains values for identifiers they are always written as placeholder variable (i.e. placeholder for the real value of the Element) in angle brackets, e.g. <x1>. This means all Elements used within a Scenario that have <x1> (e.g.) in the "Value" column SHALL have set the same content of the Element.

A placeholder variable <xY> (e.g. <x1>) for Scenario A is, in general, independent from a placeholder variable <xY> for Scenario B. However, the server SHOULD combine datasets if possible. If there is the requirement that the same value SHALL be used for different stated Scenarios, the according Scenario numbers in column "Scenario" are put in curly brackets (" {... }") for the Element containing the variable. Several curly bracket groups may exist.

Example: An Element with variable <x1> contains in the column "Scenario" the following expression:
{2, 3}, {4, 5}

This means that Scenario 2 and 3 SHALL use the same value for the variable (e.g. 5) as well as Scenario 4 and 5 SHALL use the same value for the variable (e.g. 12). The variable values MAY differ between the two groups ({2, 3} and {4, 5}).

3.1.3.7 Rules for content of "Value" column

For a given Scenario the "Value" column may restrict the permitted content of a Function's Element to one or more particular values. This means that Elements with values deviating from the restriction (i.e. from the permitted values) do not belong to the respective Scenario and need to be considered as if the Element is not set. If more than one particular value is permitted for an Element the values are in a single line each.

If a presence indication is set for the value (in an additional column before the value) the following rules SHALL be applied:

- "M" means that the value SHALL be supported. This means the value needs to be set at a certain point in time (depending on the value rules) or for a certain Element within a list entry.
- "R" means that the value SHOULD be supported.

- "O" means that the value MAY be supported.

If all possible values of a given mandatory Element are optional or recommended and this Element is used for the purpose of the respective Scenario, one of the values SHALL be set. If all possible values of a given optional or recommended Element are optional or recommended, this Element MAY contain also other values, but then this Element SHALL NOT be considered as part of the respective Scenario.

M, R or O may be combined with the suffix "(event)" to express that a supported value only has to be supported during a certain event and hence does not need to be present at all times. If the event is not active another value may be set. In most cases a High-Level requirement reference for the event is given in the rules column.

If no presence indication is set for the value, the following rules SHALL be applied:

- In case of Elements where the server may set or change an Element on its own (see section 3.1.3.5):
 - within the tables in the "Server data - Resources" sections:
 - the server SHALL support at least one of the listed values.
 - within the tables in the "Client data - Specializations" sections:
 - the client SHALL support all listed values.
- In case of Elements that are writable or changeable (see section 3.1.3.5):
 - within the tables in the "Server data - Resources" sections:
 - the server SHALL support all listed values.
 - within the tables in the "Client data - Specializations" sections:
 - the client SHALL support at least one of the listed values.

Depending on the Element, different values may be used during runtime. If this is the case, those rules are described within the value rules.

If a value is placed in parenthesis, the corresponding value is a recommendation. The actual value MAY deviate from this, e.g. "(1024)".

3.1.3.8 General information on how to interpret the "Content of Function..." and "Content of Specialization..." tables

Within the "Client data - Specializations" sections each Specialization is described in an own sub-section with the name "Specialization "<name of the Specialization>" (e.g. "Specialization "Measurement_GridFeedInEnergy"). It contains only one table that includes all Elements needed for this Specialization. The different Functions are mentioned in a continuous row, highlighted with grey background colour. This row contains the following parts:

<Feature Type>. <Function>.[<list entry instance name>.]

The <list entry instance name> is only included if the <Function> is a list-based Function. An example could be:

DeviceConfiguration. deviceConfigurationKeyValueDescriptionListData.
deviceConfigurationKeyValueDescriptionData.

In the following rows, only the names of the Elements are stated, without the prefix described above.

Within the "Server data - Resources" sections each Feature Type is described in an own sub-section with the name "Feature Type "<name of the Feature Type>" (e.g. "Feature Type "Measurement"). It contains sub-sections for each Function named "Function "<name of the Function>" (e.g. "Function "measurementListData"). These sections contain one table with all Elements needed for this resource. The list entries are mentioned in a continuous row, highlighted with grey background colour. This row contains the following parts:

<Feature Type>. <Function>.[<list entry instance name>.]

The <list entry instance name> is only included if the <Function> is a list-based Function. An example could be:

Measurement. measurementDescriptionListData. measurementDescriptionData.

In the following rows, only the names of the Elements are stated, without the prefix described above.

For both kinds of tables, the following applies:

- Parent Elements are marked with a dot at the end of the name:
 <parent Element>.
 E.g.:
 value.
- If there are sub-Elements, they are described in own rows with the name of the parent Element as prefix, separated by a dot and a blank space:
 <parent Element>. <sub-Element>
 E.g.:
 value. number

3.1.4 Rules for "Feature Types and Functions..." tables

3.1.4.1 Presence indications for "Feature Types and Functions..." tables

The following presence indications are used:

Abbreviation	Meaning	Link to requirement keywords
M	Mandatory	SHALL
R	Recommended	SHOULD
O	Optional	MAY

Table 4: Presence indication of Feature Types and Functions support

If at least one Function of a Feature has the presence indication "M", it is mandatory to support the Feature.

3.1.4.2 Rules for "Possible operations" column

Within the "Feature Types and Functions..." tables the column "Possible operations" state whether the Function is read- or writeable (as defined in the detailed discovery mechanism, see [ProtocolSpecification]).

If the "partial" concept (also called "restricted function exchange") SHALL be supported, the following notation is used (separated for read and write access):

read (M). partial (M)

write (M). partial (M)

If the "partial" concept SHOULD be supported, the following notation is used:

read (M). partial (R)

write (M). partial (R)

If the "partial" concept MAY be supported, the following notation is used:

read (M). partial (O)

write (M). partial (O)

The server can decide whether a notification is submitted complete or partial (as described in [ProtocolSpecification]) if not defined differently within this Use Case Specification.

3.1.5 "Actor ... overview" diagram rules

Within the "Actor [...] overview" diagrams in the "Actors" sub-sections the complete functionality of this Use Case is provided, including optional Scenarios. Which Scenarios are optional can be found in Table 1. The Actor MAY have more functionality implemented than needed for this Use Case.

For the following Actor overview example, a brief description of the graphical symbols will be described.

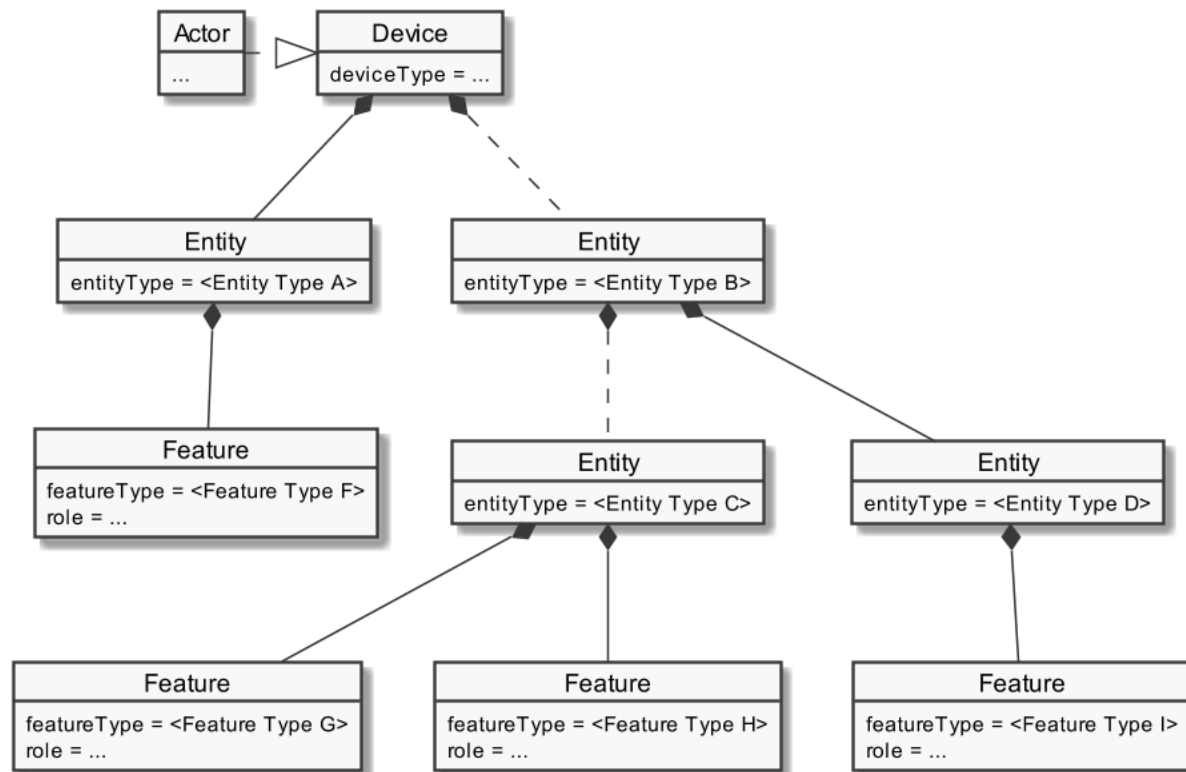


Figure 5: Actor overview example

The solid lines in the figure represent an immediate parent-childhood relation: The Entity with "<Entity Type A>" is a direct child of "Device". The Entity with "<Entity Type D>" is a direct child of the Entity with "<Entity Type B>". All Features are immediate child of the respective Entity.

The dashed lines in the figure express that there MAY be additional Entities between the shown Entities: A vendor's implementation MAY have one or more Entities between "Device" and the Entity with "<Entity Type B>". Likewise, a vendor's implementation MAY have one or more Entities between the Entity with "<Entity Type B>" and the Entity with "<Entity Type D>".

3.1.6 Specializations

Within the "Actors" sub-sections Specializations are referenced. A Specialization describes a dataset necessary to fulfil the specific requirements of a High-Level Use Case and its Scenarios. Often data from multiple different Features and Functions are needed to fulfil the requirements. Therefore, a Specialization defines a dataset that may encompass multiple related Functions from one or more different Features.

As different Use Cases sometimes share similar requirements, Specializations are also important from a re-usability perspective. This approach is used to improve consistency across Use Cases and avoid multiple variances of basically the same dataset. This is especially important in the case when an implementation supports multiple Use Cases. E.g. if a power measurement is necessary in two different Use Cases, both Use Cases could define slightly different datasets. In this case the server as well as the client functionality would have to implement both variances if both Use Cases are supported. This means, depending on the number of Use Cases, two or more datasets need to be

generated, transmitted and stored instead of one. Therefore, already existing Specializations specified within [UseCaseBaseSpecification] are used in this Use Case to avoid such problems.

If a Feature server can provide the data of a Specialization, the data does not necessarily always need to be available at the Feature server. There might be situations where the user deactivates a Use Case. There may also be other reasons why Use Case data cannot be provided currently. Therefore, a client always needs to be subscribed (as described in section 3.3.4) on the corresponding dataset to stay updated.

The SPINE resource description given in the "SPINE resources of the Actor" sections are derived from the Specializations given in the Actor's overview diagram. Please refer to [UseCaseBaseSpecification] for a detailed description of all Specializations.

3.1.7 Order of messages within the sequence diagrams

There are several sequence diagrams in this document describing message flows. The order of the messages SHOULD be kept by the communications partners, but there might be cases where a different order makes sense. The communications partners SHALL be able to handle the Scenario functionalities even if the messages are transmitted in a different order by the other Actor(s). The sequence diagrams can be seen as examples.

3.1.8 Further information and rules

None.

3.2 Actors

3.2.1 EVSE

3.2.1.1 Resource hierarchy

Within the Use Case discovery this Actor SHALL be denoted as "EVSE" in the Element "nodeManagementUseCaseData. useCaseInformation. actor".

The following diagram provides an overview of the Actor EVSE resource hierarchy.

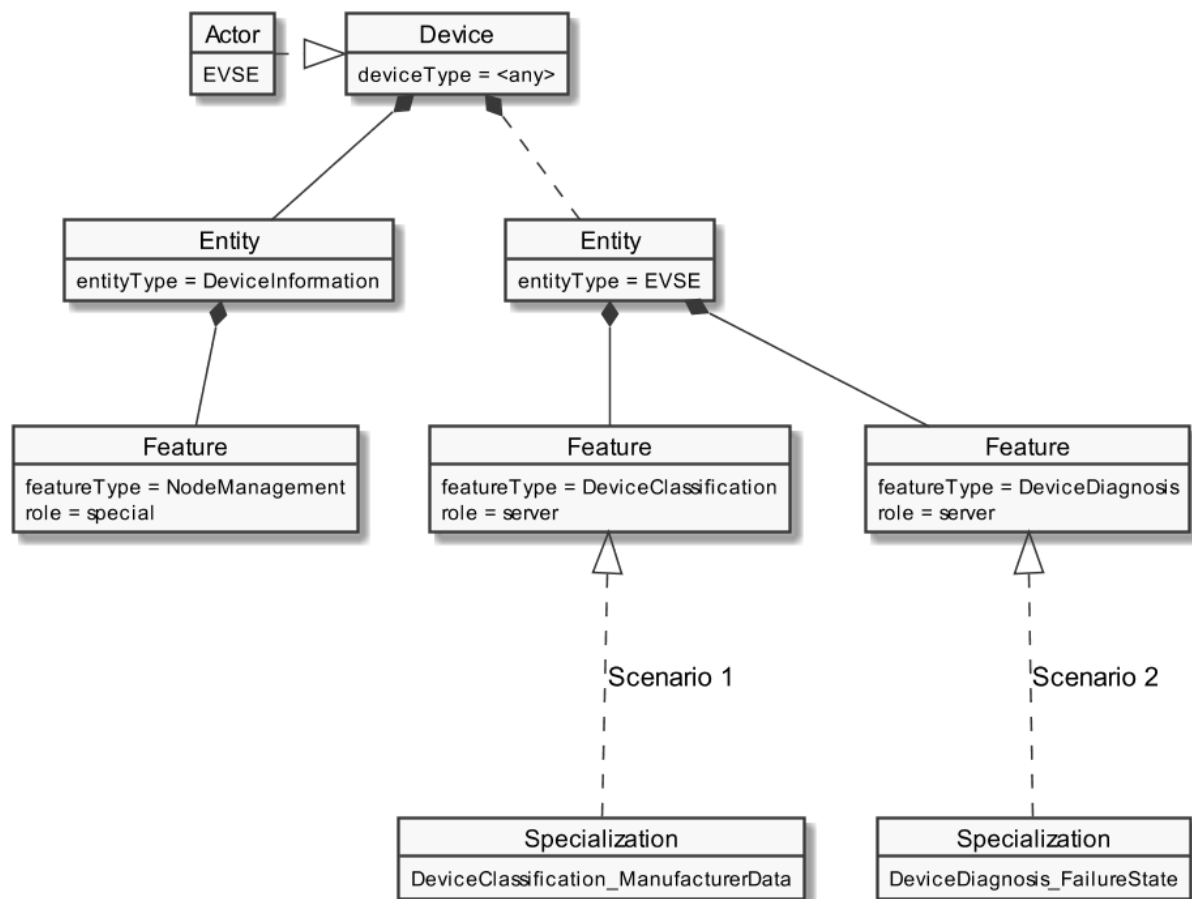


Figure 6: Actor "EVSE" overview

The "Actor ... overview" diagram rules" section describes how to interpret the diagram above. See the "Specializations" section for more information regarding the Specializations given in the diagram above.

Note: The entityType "DeviceInformation" with the featureType "NodeManagement" is required by the SPINE protocol and therefore SHALL be supported. Both types are added in the figure for completeness but are not directly linked to the Use Case.

The Use Case specific data follows behind the entityType "EVSE". The Specializations represent the Scenario specific data that has to be supported for each Scenario and are realized with the according featureTypes.

If a Specialization is connected to a Feature with the role "client", the Actor has a client role for this data. This means the Actor accesses the data set described by the Specialization at a corresponding server Feature. Further details are described in the sub-section "Client data - Specializations".

If a Specialization is connected to a Feature with the role "server", the Actor has the server role for this data. This means the Actor must provide the corresponding data set of the Specialization on its Features. Further details are described in the sub-section "Server data - Resources".

3.2.1.2 Server data - Resources

3.2.1.2.1 Overview

Behind the entityType "EVSE" the Actor EVSE SHALL offer the Feature Types and Functions given in the table below.

Feature Type	Scenario: M/R/O	Function	Possible operations
DeviceClassification	1: M	deviceClassificationManufacturerData	read (M)
DeviceDiagnosis	2: M	deviceDiagnosisStateData	read (M)

Table 5: Feature Types and Functions used within this Use Case by the Actor EVSE

For each of these Feature Types the following rule applies: There SHALL be at maximum one Feature with the Feature Type in the Entity.

Note: As a consequence of the previous rule, an implementation may need to have Feature data from different Scenarios/Specializations or even Use Cases in a given Feature.

The Scenario number shows in which Scenarios the EVSE acts as server and which Feature Types and Functions are relevant in each Scenario.

A detailed definition of the Elements and values that shall be supported in each Function is given in the following sub-sections.

Note: If in the table above "partial" read is not mentioned or is only optional, it still might be mandatory to support partial notifications. The details of "partial" support are described within the Scenario sections.

Note: The presence indications stated above are meant relative to the ones of the according Scenario stated in Table 1. I.e. if a Scenario is optional ("O") and a Feature Type is mandatory ("M") the Feature Type must only be supported if the Scenario is supported, too.

Note: Further Features MAY be implemented on the same Entities, as well as further Functions MAY be implemented in the used Entities.

3.2.1.2.2 Feature Type "DeviceClassification"

3.2.1.2.2.1 Function "deviceClassificationManufacturerData"

Scenario {...}: M/R/O [W][C]	Element	Value	[High Level Mapping] Element and value rules
1: M	DeviceClassification. deviceClassificationManufacturerData.		
1: R	deviceName		[EVSECC-010] 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.
1: R	deviceCode		[EVSECC-011] 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.
1: O	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.
1: O	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.
1: O	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.
1: R	vendorName		[EVSECC-012] 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.
1: R	vendorCode		[EVSECC-013] 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.
1: R	brandName		[EVSECC-014] 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.
1: R	manufacturerLabel		[EVSECC-015] The string-length SHOULD NOT be longer than 256 characters.

1: O	manufacturerDescription		The string-length SHOULD NOT be longer than 4096 characters
------	-------------------------	--	---

Table 6: Content of Function "deviceClassificationManufacturerData" at Actor EVSE

3.2.1.2.3 Feature Type "DeviceDiagnosis"

3.2.1.2.3.1 Function "deviceDiagnosisStateData"

Scenario [{...}]: M/R/O [W][C]	Element	Value		[High Level Mapping] Element and value rules
2: M	DeviceDiagnosis. deviceDiagnosisStateData.			
2: M	operatingState	2: M	"normalOperation"	
		2: M	"failure"	[EVSECC-020]
2: O	lastErrorCode			The string-length SHOULD NOT be longer than 256 characters.

Table 7: Content of Function "deviceDiagnosisStateData" at Actor EVSE

3.2.1.3 Client data - Specializations

As this Actor has only server functionality, no Specializations are described within this section.

3.2.2 CEM

3.2.2.1 Resource hierarchy

Within the Use Case discovery this Actor SHALL be denoted as "CEM" in the Element "nodeManagementUseCaseData. useCaseInformation. actor".

The following diagram provides an overview of the Actor CEM resource hierarchy.

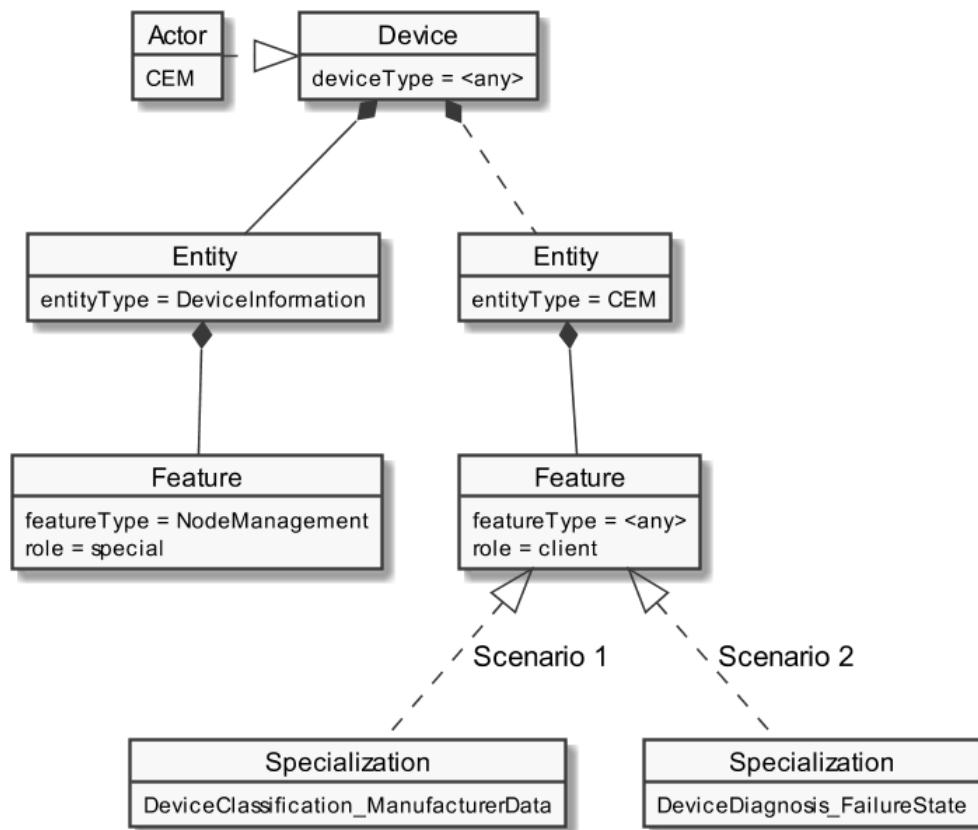


Figure 7: Actor "CEM" overview

The "Actor ... overview" diagram rules" section describes how to interpret the diagram above. See the "Specializations" section for more information regarding the Specializations given in the diagram above.

Note: The entityType "DeviceInformation" with the featureType "NodeManagement" is required by the SPINE protocol and therefore SHALL be supported. Both types are added in the figure for completeness but are not directly linked to the Use Case.

The Use Case specific data follows behind the entityType "CEM". The Specializations represent the Scenario specific data that has to be supported for each Scenario and are realized with the according featureTypes.

If a Specialization is connected to a Feature with the role "client", the Actor has a client role for this data. This means the Actor accesses the data set described by the Specialization at a corresponding server Feature. Further details are described in the sub-section "Client data - Specializations".

If a Specialization is connected to a Feature with the role "server", the Actor has the server role for this data. This means the Actor must provide the corresponding data set of the Specialization on its Features. Further details are described in the sub-section "Server data - Resources".

3.2.2.2 Server data - Resources

As this Actor has only client functionality, no resources are described within this section.

3.2.2.3 Client data - Specializations

3.2.2.3.1 Topic "DeviceClassification"

3.2.2.3.1.1 Specialization "DeviceClassification_ManufacturerData"

Scenario {...}: M/R/O [W][C]	Element	Value	[High Level Mapping] Element and value rules
1: M	DeviceClassification. deviceClassificationManufacturerData.		
1: R	deviceName		[EVSECC-010] 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.
1: R	deviceCode		[EVSECC-011] 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.
1: O	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.
1: O	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.
1: O	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.
1: R	vendorName		[EVSECC-012] 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.
1: R	vendorCode		[EVSECC-013] 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.
1: R	brandName		[EVSECC-014]

			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.
1: R	manufacturerLabel		[EVSECC-015] The string-length SHOULD NOT be longer than 256 characters.
1: O	manufacturerDescription		The string-length SHOULD NOT be longer than 4096 characters

Table 8: Content of Specialization "DeviceClassification_ManufacturerData" at Actor CEM

3.2.2.3.2 Topic "DeviceDiagnosis"

3.2.2.3.2.1 Specialization "DeviceDiagnosis_FailureState"

Scenario [...]: M/R/O [W][C]	Element	Value	[High Level Mapping] Element and value rules
2: M	DeviceDiagnosis. deviceDiagnosisStateData.		
2: M	operatingState	"normalOperation"	
		"failure"	[EVSECC-020]
2: O	lastErrorCode		The string-length SHOULD NOT be longer than 256 characters.

Table 9: Content of Specialization "DeviceDiagnosis_FailureState" at Actor CEM

3.3 Pre-Scenario communication

3.3.1 General information

The Pre-Scenario communication is needed if a client does not know the corresponding addresses on the server or if the required subscriptions or bindings are not active. In this case certain general communication mechanisms SHALL be used within SPINE:

- Detailed discovery: allows to discover resource addresses.
- Binding: allows to bind to resource address, which is frequently necessary to obtain write privileges.
- Subscription: allows to subscribe to resource addresses, which is necessary to receive unsolicited notifications if a resource changes during runtime.

It is possible to combine those steps for multiple Scenarios or also multiple Use Cases:

- E.g. if multiple Scenarios in multiple Use Cases use the same Feature, only one subscription needs to occur.
- E.g. a complete detailed discovery or a subscription to the NodeManagement Feature needs to occur only once for all Use Cases.

Depending on which Entity, Feature and Functions are used within a Scenario the payload of the corresponding messages may slightly differ, but the basic principles and messages used stay the same.

The subsequent messages SHALL be exchanged for those parts that have not already been performed since the current connection is established or if those parts are outdated for another reason (e.g. if the detailed discovery is needed, but the bindings and subscriptions are still active from a previous connection only the detailed discovery messages need to be exchanged). If all Pre-Scenario communication parts are up-to-date, this section MAY be skipped, and the implementation can proceed as described in the corresponding "Scenario communication" sections.

After the connection is re-established (e.g. due to a power loss or a firmware update) the Pre-Scenario communication SHALL be performed as well. There may be circumstances where messages from the Pre-Scenario communication may be exchanged again.

Often the necessary messages of different Scenarios can be combined, so that only one single message is needed instead of multiple messages for the different Scenarios. This also is the case for the Pre-Scenario communication. In most cases only one "read" operation on the detailed discovery is necessary, as well as only one subscription request or binding request is needed for each Feature. Often multiple Scenarios within a Use Case access the same Feature, so only one subscription or binding is necessary.

3.3.2 Detailed discovery

For the functionality where a client already has current detailed discovery information (i.e. independent of this Use Case or any Scenario of it) the remainder of this section SHOULD be skipped.

Otherwise, the following procedure SHALL be performed in the given order:

1. If a client is not subscribed to the primary NodeManagement instance, the client SHALL acquire a subscription according to the figure provided within this sub-section.
2. A client SHALL read the detailed discovery information according to the figure provided within this sub-section. It SHALL keep the received information as far as it concerns mandatory and supported optional Entity Types, Feature Types and Functions of this Use Case that are needed by the client. This means that a client may choose how to store the necessary information. E.g. a client Actor can store the information how to address the necessary Features of the implemented Scenarios but may discard the Entity information.
3. If and as long as a client has a subscription to the detailed discovery information of an Actor and receives proper notifications, it SHALL consider (i.e. integrate into the kept detailed discovery information) the received information as far as it concerns mandatory and supported optional Entity Types, Feature Types and Functions of this Use Case.

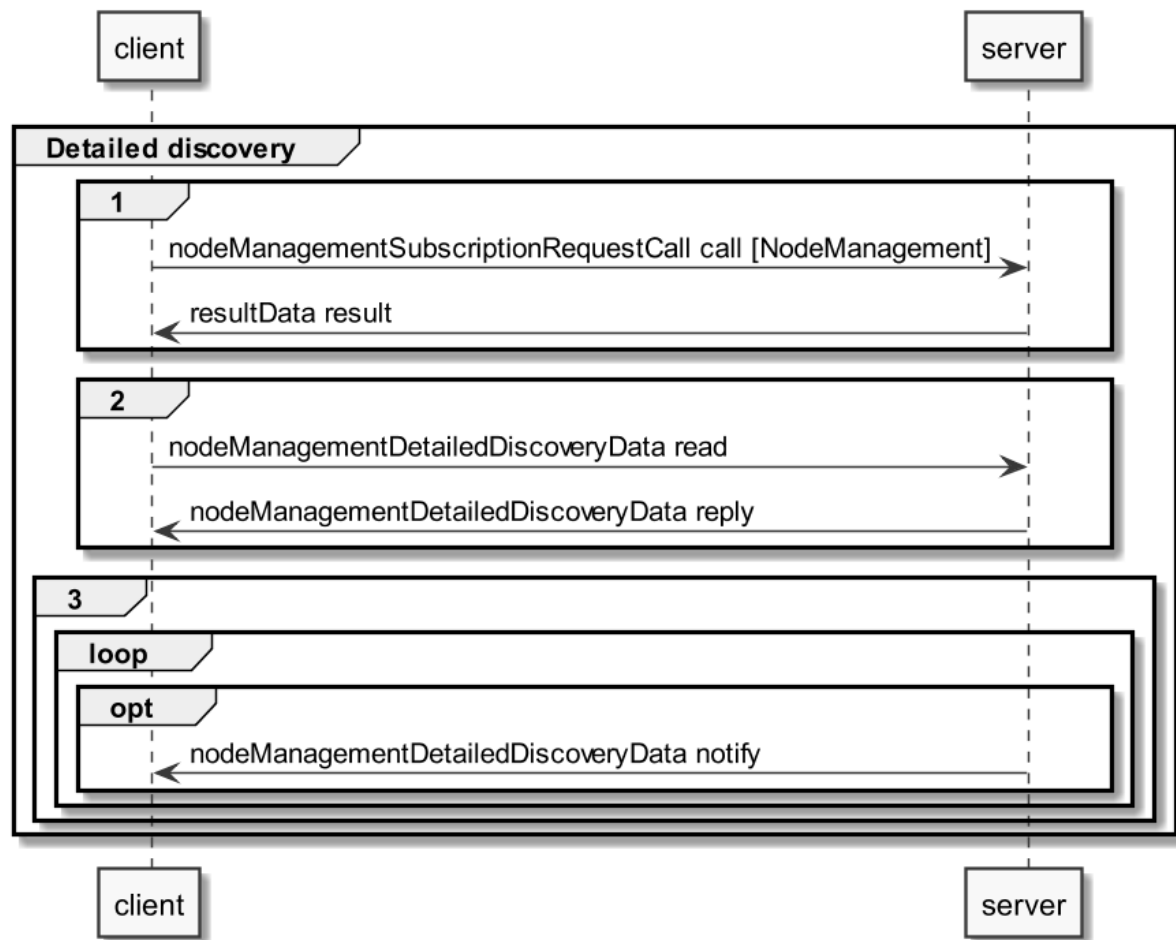


Figure 8: Pre-Scenario communication - Detailed discovery sequence diagram

If the "nodeManagementDetailedDiscoveryData read" fails, the client SHOULD retry to read the detailed discovery information until the "nodeManagementDetailedDiscoveryData reply" message was received successfully.

If all functionality is present at all times: The "nodeManagementDetailedDiscoveryData reply" message contains at least the mandatory Entities and Features given in the "Actor [...] overview" diagrams as well as the used Functions and their "possible operations" described in section 3.2 and its sub-sections.

If functionality is added or removed dynamically: The "nodeManagementDetailedDiscoveryData reply" message does not need to contain all mandatory Entities and Features given in the "Actor [...] overview" diagrams as well as all needed Functions and their "possible operations" described in section 3.2 and its sub-sections. However, as soon as the functionality is available it will be announced via a "nodeManagementDetailedDiscoveryData notify" message.

For the nodeManagementDetailedDiscoveryData read Function it is recommended to use a partial read with separated Selectors that may use one of the following Elements:

- entityType
- featureType

Note: Even with the usage of Selectors Features and Entities that are not relevant for this Use Case may be discovered. However, only Features and Entities that fulfil the hierarchical order as described within the Actors' sections shall be considered for this Use Case.

A "partial" notify SHALL be supported without using Selectors and Elements. Partial "delete" notify SHOULD also be supported with separated Selectors that may use one of the following Elements:

- entityAddress
- featureAddress

3.3.3 Binding

A server SHALL support binding for all Features that contain writeable or changeable data. Before a write on a Function of a Feature occurs, the client SHALL create a binding to the corresponding Feature. For this the nodeManagementBindingRequestCall Function is used as shown in the following sequence diagram:

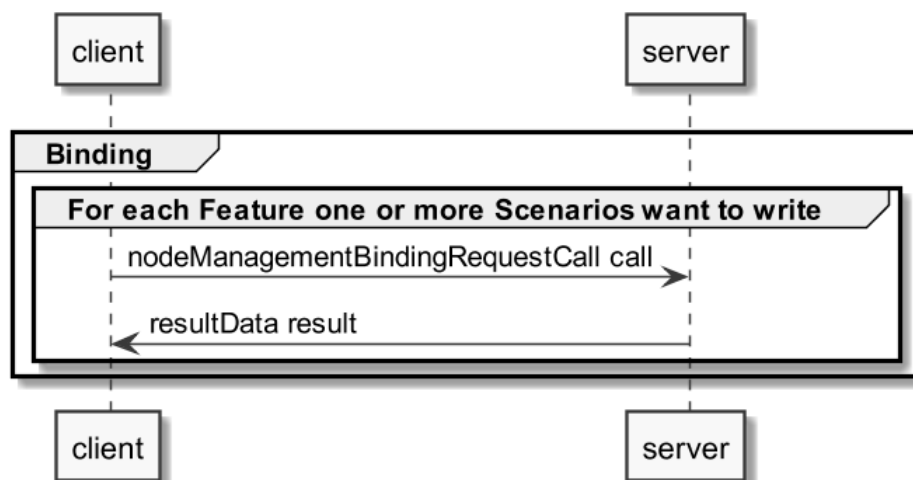


Figure 9: Pre-Scenario communication - Binding sequence diagram

If functionality is added or removed dynamically, binding may not be possible at all times on the required Functions. A client SHALL retry to create a binding again when receiving according updated detailed discovery information.

3.3.4 Subscription

A server SHALL support subscription for all Features that contain readable data that may change during runtime. The client SHALL create a subscription for all Features that the client wants to read. For this the nodeManagementSubscriptionRequestCall Function is used as shown in the following sequence diagram:

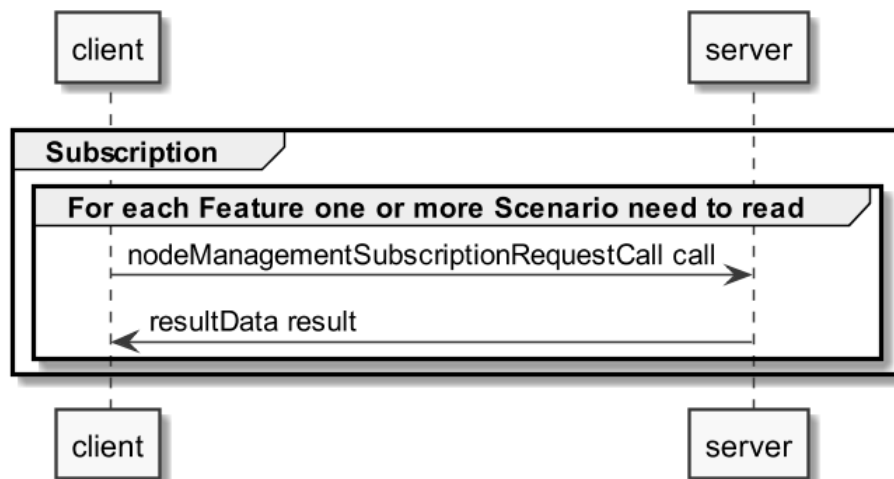


Figure 10: Pre-Scenario communication - Subscription sequence diagram

If the subscription request fails (e.g. because it is not supported by the server or the maximum number of possible subscriptions is reached), the client **SHOULD** read the data periodically (so-called "polling").

If functionality is added or removed dynamically, subscription may not be possible at all times on the required Functions. A client **SHALL** retry its subscription procedure again when receiving according updated detailed discovery information.

3.3.5 Dynamic behaviour

In case Entities or Features are removed, a nodeManagementDetailedDiscoveryData "notify" is transmitted that informs about the deleted Entities and Features. All existing binding or subscription entries on the deleted Features **SHALL** be deleted by each device.

In case Entities or Features are added the Pre-Scenario communication starts with transmitting a nodeManagementDetailedDiscoveryData "notify" that contains the added Entities and Features.

3.4 Scenarios

3.4.1 Scenario 1 - EVSE sends manufacturer information

3.4.1.1 Pre-Scenario communication

1. **Detailed Discovery:** Actors that act as client within this Scenario, need to know the addresses of the server Features used in the Initial Scenario communication. If an address of a particular server Feature is not known, the detailed discovery has to be used, as described in section 3.3.2.
2. **Binding:** Binding **SHOULD NOT** be used for this Scenario.
3. **Subscription:** Actors **SHALL** create a subscription for each server Feature that is relevant for the corresponding Actor within this Scenario, as described in section 3.3.4.

The Initial Scenario communication **SHALL** start at the latest when the required resources on an Actor are known and the necessary binding and subscription procedures have been finished. However, as

soon as an address of a required resource is known, the Initial Scenario communication for this resource MAY start already, even if addresses of other required resources are not known yet.

If required resources are removed and added again, they are re-discovered, and the Initial Scenario communication is triggered again for those resources.

3.4.1.2 Initial Scenario communication

Each time a (re-)connection is established, even if the Pre-Scenario communication phase is skipped, the messages as shown in the following sequence diagram SHALL be exchanged, as the corresponding resources may have changed in the meantime:

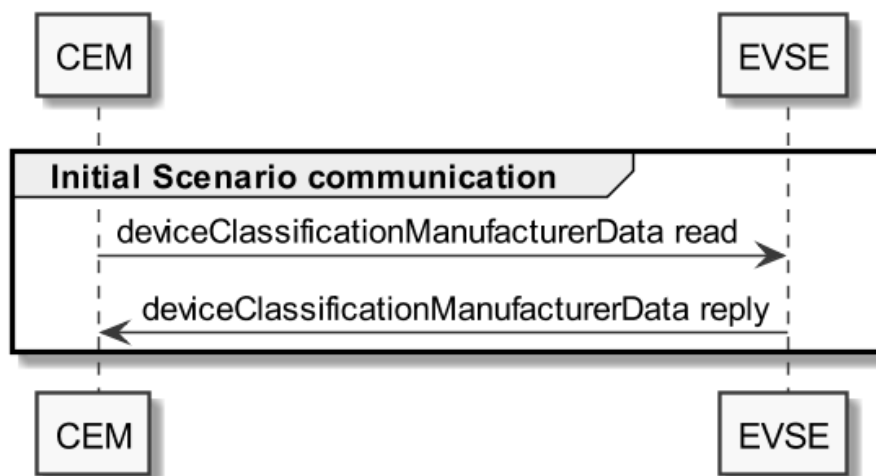


Figure 11: Scenario 1 - Initial Scenario communication sequence diagram

The following table shows where the necessary content of the messages from the sequence diagram is described:

Message name from sequence diagram	Content description in table	Scenario number in table
deviceClassificationManufacturerData reply	Table 6	1

Table 10: Initial Scenario communication content references for Scenario 1

Note: Within the Initial Scenario communication the content required by this Scenario MAY not be provided completely but later on during Runtime Scenario communication.

3.4.1.3 Runtime Scenario communication

Based on the Initial Scenario communication the Runtime Scenario communication provides updates during runtime.

If one of the referenced server Functions' data change, the server SHALL submit the change as shown in the following figure:

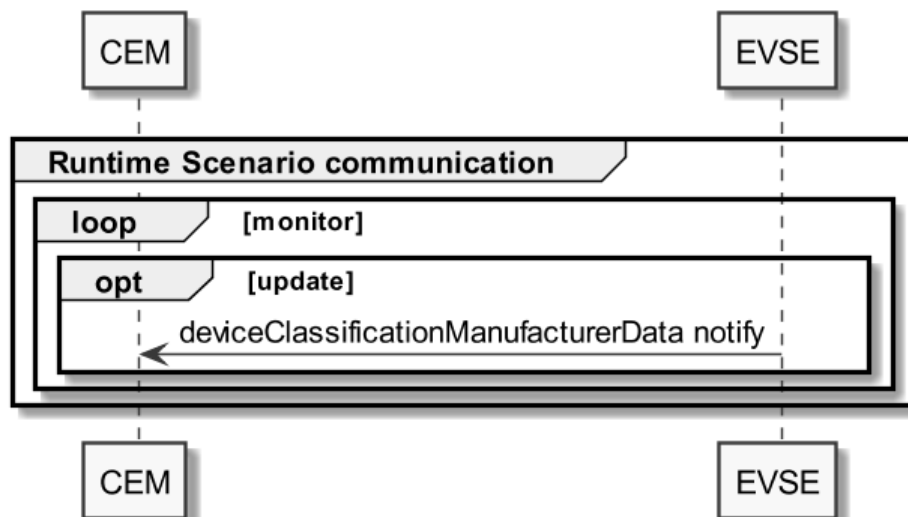


Figure 12: Scenario 1 - Runtime Scenario communication sequence diagram

Note: To interpret partial notification messages correctly the information obtained during the Initial Scenario communication phase is necessary.

Note: A read operation ("polling") on all Functions is possible at any time, e.g. if a notification could not be evaluated.

The following table shows where the necessary content of the messages of the sequence diagram is described:

Message name from sequence diagram	Content description in table	Scenario number in table
deviceClassificationManufacturerData notify	Table 6	1

Table 11: Runtime Scenario communication content references for Scenario 1

3.4.1.4 Additional information

None.

3.4.2 Scenario 2 - EVSE sends error state

3.4.2.1 Pre-Scenario communication

- Detailed Discovery:** Actors that act as client within this Scenario, need to know the addresses of the server Features used in the Initial Scenario communication. If an address of a particular server Feature is not known, the detailed discovery has to be used, as described in section 3.3.2.
- Binding:** Binding SHOULD NOT be used for this Scenario.
- Subscription:** Actors SHALL create a subscription for each server Feature that is relevant for the corresponding Actor within this Scenario, as described in section 3.3.4.

The Initial Scenario communication SHALL start at the latest when the required resources on an Actor are known and the necessary binding and subscription procedures have been finished. However, as soon as an address of a required resource is known, the Initial Scenario communication for this resource MAY start already, even if addresses of other required resources are not known yet.

If required resources are removed and added again, they are re-discovered, and the Initial Scenario communication is triggered again for those resources.

3.4.2.2 Initial Scenario communication

Each time a (re-)connection is established, even if the Pre-Scenario communication phase is skipped, the messages as shown in the following sequence diagram SHALL be exchanged, as the corresponding resources may have changed in the meantime:

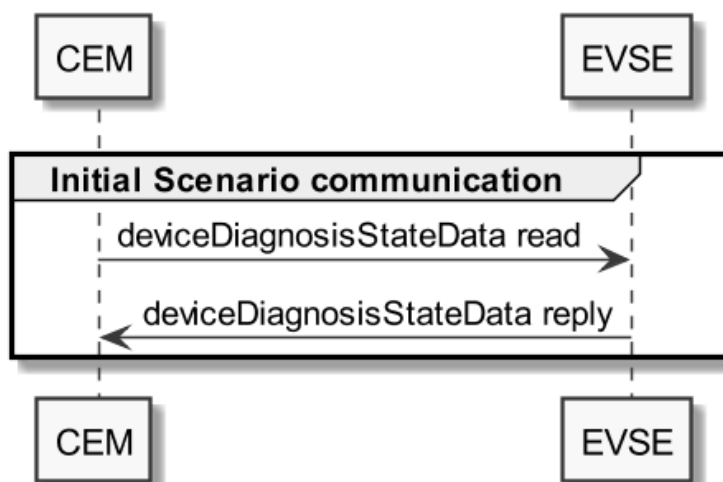


Figure 13: Scenario 1 - Initial Scenario communication sequence diagram

The following table shows where the necessary content of the messages from the sequence diagram is described:

Message name from sequence diagram	Content description in table	Scenario number in table
deviceDiagnosisStateData reply	Table 7	2

Table 12: Initial Scenario communication content references for Scenario 2

Note: Within the Initial Scenario communication the content required by this Scenario MAY not be provided completely but later on during Runtime Scenario communication.

3.4.2.3 Runtime Scenario communication

Based on the Initial Scenario communication the Runtime Scenario communication provides updates during runtime.

If one of the referenced server Functions' data change, the server SHALL submit the change as shown in the following figure:

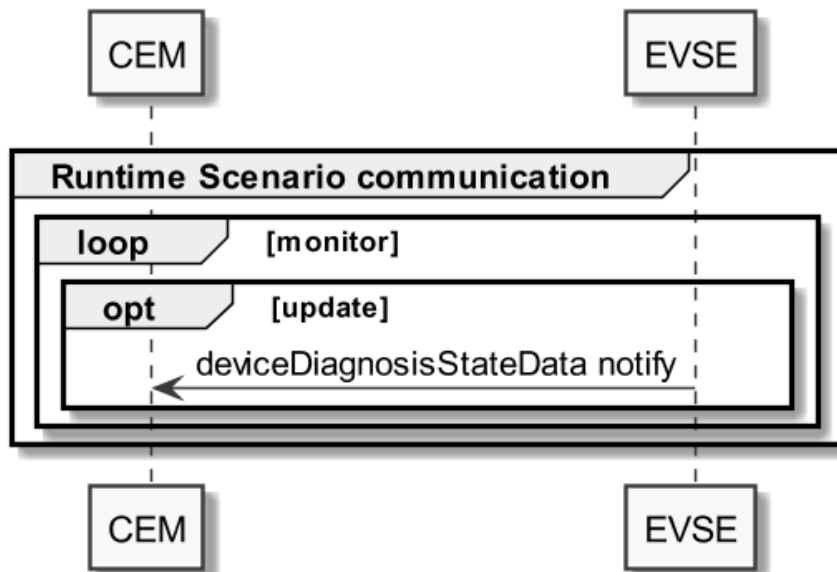


Figure 14: Scenario 1 - Runtime Scenario communication sequence diagram

Note: To interpret partial notification messages correctly the information obtained during the Initial Scenario communication phase is necessary.

Note: A read operation ("polling") on all Functions is possible at any time, e.g. if a notification could not be evaluated.

The following table shows where the necessary content of the messages of the sequence diagram is described:

Message name from sequence diagram	Content description in table	Scenario number in table
deviceDiagnosisStateData notify	Table 7	2

Table 13: Runtime Scenario communication content references for Scenario 2

3.4.2.4 Additional information

None.