DRAFT INTERNATIONAL STANDARD ISO/IEC DIS 15118-2

ISO/IEC TC **22/SC3** Secretariat: **DIN**

Voting begins on **2012-05-18** Voting terminates on **2012-10-18**

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION
INTERNATIONAL ELECTROTECHNICAL COMMISSION • МЕЖДУНАРОДНАЯ ЭЛЕКТРОТЕХНИЧЕСКАЯ КОММИСИЯ • COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

Road vehicles — Vehicle to grid communication interface —

Part 2:
Network and application protocol requirements

Véhicules routiers — Interface de communication entre véhicule et réseau électrique —

Partie 2: Exigences du protocole d'application et du réseau

ICS 43.120

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 15118-2 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronique equipment*.

ISO/IEC 15118 consists of the following parts, under the general title *Road vehicles — Vehicle-to-Grid Communication Interface*:

— *Part 1: General information and use-case definition*

— *Part 2: Network and application protocol requirements*

— *Part 3: Physical layer and Data Link layer requirements*

Introduction

The pending energy crisis and necessity to reduce greenhouse gas emissions has led the vehicle manufacturers to a very significant effort to reduce the energy consumption of their vehicles. They are presently developing vehicles partly or completed propelled by electric energy. Those vehicles will reduce the dependency on oil, improve the global energy efficiency and reduce the total CO_2 emissions for road transportation if the electricity is produced from renewable sources. To charge the batteries of such vehicles, specific charging infra-structure is required.

Much of the standardization work on dimensional and electrical specifications of the charging infrastructure and the vehicle interface is already treated in the relevant ISO or IEC groups. However the question of information transfer between the EV and the EVSE has not been treated sufficiently.

Such communication is necessary for the optimization of energy resources and energy production systems so that vehicles can recharge in the most economic or most energy efficient way. It is also required to develop efficient and convenient billing systems in order to cover the resulting micro-payments. The necessary communication channel may serve in the future to contribute to the stabilization of the electrical grid as well as to support additional information services required to operate electric vehicles efficiently and economically.

Road vehicles — Vehicle-to-Grid Communication Interface — Part 2: Network and application protocol requirements

1 Scope

This International Standard specifies the communication between battery electric vehicles (BEV) or plug-in hybrid electric vehicles (PHEV) and the Electric Vehicle Supply Equipment. The application layer message set defined in this Part of ISO/IEC 15118 is designed to support the energy transfer from an EVSE to an EV. Part 1 contains additional use case elements (Part 1 Use Case Element IDs: F4 and F5) describing the bidirectional energy transfer. The implementation of these use cases requires enhancements of the application layer message set defined herein. The definitions of these additional requirements will be subject of the next revision of this standard.

The purpose of this Part of ISO/IEC 15118 is to detail the communication between an EV (BEV or a PHEV) and an EVSE. Aspects are specified to detect a vehicle in a communication network and enable an Internet Protocol (IP) based communication between EVCC and SECC.

Key

1 Scope of this Part of ISO/IEC DIS 15118-2

2 Message definition considers use cases defined for communication between SECC to SA

Figure 1 — Communication relationship between EVCC SECC, and Secondary Actor

This part defines messages, data model, XML/EXI based data representation format, usage of V2GTP, TLS, TCP and IPv6. In addition the document describes how data link layer services can be accessed from a layer 3 perspective. The Data Link Layer and Physical Layer functionality is described in Part 3 of this standard.

2 Normative references

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61851-1, *Electric vehicle conductive charging system — Part 1: General requirements*

SAE J1772, *SAE Electric Vehicle and Plug in Hybrid Electric Vehicle Conductive Charge Coupler*

IEC 62196, *Plugs, socket-outlets, vehicle connectors and vehicle inlets - Conductive charging of electric vehicles*

DIN 91286, *Electric mobility –Schemes of identifiers for E-Roaming –Contract ID and Electric Vehicle Supply Equipment ID*

W3C EXI 1.0, *Efficient XML Interchange (EXI) Format 1.0, W3C Recommendation (March 2011)*

IETF RFC 768, *User Datagram Protocol (August 1980)*

IETF RFC 793, *Transmission Control Protocol - DARPA Internet Program - Protocol Specification (September 1981)*

IETF RFC 1323, *TCP Extensions for High Performance (Mai 1992)*

IETF RFC 1624, *Computation of the Internet Checksum via Incremental Update (Mai 1994)*

IETF RFC 1981, *Path MTU Discovery for IP version 6 (August 1996)*

IETF RFC 2018, *TCP Selective Acknowledgment Options (October 1996)*

IETF RFC 2460, *Internet Protocol, Version 6 (IPv6) Specification (December 1998)*

IETF RFC 2560, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP (June 1999)*

IETF RFC 3122, *Extensions to IPv6 Neighbor Discovery for Inverse Discovery Specification (June 2001)*

IETF RFC 3315, *Dynamic Host Configuration Protocol for IPv6 (DHCPv6) (July 2003)*

IETF RFC 3484, *Default Address Selection for Internet Protocol version 6 (IPv6) (February 2003)*

IETF RFC 3782, *The NewReno Modification to TCP's Fast Recovery Algorithm (April 2004)*

IETF RFC 4291, *IP Version 6 Addressing Architecture (February 2006)*

IETF RFC 4429, *Optimistic Duplicate Address Detection (DAD) for IPv6 (April 2006)*

IETF RFC 4443, *Internet Control Message Protocol (ICMP v6) for the Internet Protocol version 6 (IPv6) specification (March 2006)*

IETF RFC 4861, *Neighbor Discovery for IP version 6 (IPv6) (September 2007)*

IETF RFC 4862, *IPv6 Stateless Address Autoconfiguration (September 2007)*

IETF RFC 5095, *Deprecation of Type 0 Routing Headers in IPv6 (December 2007)*

IETF RFC 5116, *An Interface and Algorithms for Authenticated Encryption (January 2008)*

IETF RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2 (August 2008)*

IETF RFC 5289, *TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM) (August 2008)*

IETF RFC 5482, *TCP User Timeout Option (March 2009)*

IETF RFC 5681, *TCP Congestion Control (September 2009)*

IETF RFC 5722, *Handling of Overlapping IPv6 Fragments (December 2009)*

IETF RFC 6066, *Transport Layer Security (TLS) Extensions: Extension Definitions (January 2011)*

IETF RFC 6106, *IPv6 Router Advertisement Options for DNS Configuration (November 2010)*

IETF RFC 6298, *Computing TCP's Retransmission Timer (June 2011)*

IETF RFC 6335, *Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transfer Protocol Port Number Registry (August 2011)*

IANA Service&PortRegistry, *Service Name and Transport Protocol Port Number Registry [viewed 2011-01-16], Available from: <http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml>*

3 Terms and definitions

For the purpose of this document, the following terms and definitions apply in addition to the terms and definitions given in Part 1.

3.1
Communication Setup Timer
A Timer monitoring the time from plug-in until the Session Setup message.

3.2
Contract Certificate
Certificate issued to EVCC either by V2G Root CA or by Sub CA, which is used in XML Signatures in application layer so that SECC or Secondary Actor can verify the Contract issued to the EVCC and signatures issued by the EVCC.

3.3
Credentials
anything that provides the basis for confidence, belief, credit, etc.

EXAMPLE Examples include certificates, passwords, user names and so on

3.4
DER/PEM
DER (Distinguished Encoding Rules = ASN-1 encoding rule) is a method for encoding a data object, such as an X.509 certificate, to be digitally signed or to have its signature verified. X.509 certificate files encode in DER are binary files, which can not be used with XML unless they are Base64 encoded. PEM (Privacy Enhanced Mail) Encoding (Base64 encoding) is a commonly used encoding schema for X.509 certificate files. The full specification of DER/PEM is in IETF RFC 1421.

3.5
Global address
IP address with unlimited scope

3.6
Link-local address
IP address with link-only scope that can be used to reach neighboring nodes attached to the same link

3.7
(IP)-Address
IP-layer identifier for an interface or a set of interfaces

3.8
Maximum Transfer Unit (MTU)
maximum size of the Data Link Layer that can be used for the IP Layer

3.9
Message Set
A set of mandatory V2G messages and parameters for the EVCC or SECC covering one or multiple use case elements

3.10
Message Timer
A Timer monitoring the exchange of a Request-Response-Pair.

3.11
Network segment
collection of devices that can exchange data on Data Link Layer level directly via Data Link Addresses

EXAMPLE Ethernet: all devices which can see each other via MAC adresses.

3.12
node
a device that implements IPv6

3.13
Performance Time
A non-functional timing requirement defining the time a V2G Entity shall not exceed when executing or processing certain functionality. This is a fixed time value.

3.14
Profile
A group of mandatory and optional Message Sets covering a set of similar charging scenarios for a specific identification means.

3.15
Ready to Charge Timer
A Timer monitoring the time from plug-in until the first Power Delivery message.

3.16
Ready to Charge Time
A device or piece of software used in an implementation for measuring time. Depending on the specific use case a timer is used to trigger certain system events as well.

3.17
Request-Response Message Pair
A request message and the corresponding response message.

3.18
Request-Response Message Sequence
A Sequence of multiple Request-Response Message Pairs.

3.19
SDP Client
A V2G entity that uses the SDP server to get configuration information about the SECC to be able to access the SECC.

3.20
SDP Server
A V2G entity providing configuration information for accessing the SECC.

3.21
SECC Certificate
Certificate issued to SECC either by V2G Root CA or by Sub CA, which is used in TLS so that EVCC can verify the authenticity of EVCC.

3.22
Sequence Timer
A Timer monitoring a Request-Response Message Sequence

3.23
Sub-CA
Subordinate certificate authority who issues SECC certificates and/or Contract certificates on behalf of the V2G Root CA.

NOTE The ability of issuing the certificates are delegated from V2G Root CA, and V2G Root CA can revoke the sub CA at any time.

3.24
Sub CA Certificate
Certificate issued to Sub CA.

3.25
TCP_DATA
Socket/interface for data transfer based on TCP conncetion

3.26
Timeout
A timing requirement defining the time a V2G Entity monitors the communication system for a certain event to occur. If the specified time is exceeded the respective V2G Entity initiates the related error handling. This is a fixed time value.

3.27
Timer
A device or piece of software used in an implementation for measuring time. Depending on the specific use case a timer is used to trigger certain system events as well.

3.28
Trusted Environment
Closed user group (e. g. members of car sharing system) with some pre-distributed token for access to the SECC charging service (e.g. key to home garage, RFID token for car sharing). Trusted environment is something where a person or instance is responsible for. Responsibility lies for example (not limited to) at a person with its home garage, a car sharing operator or a taxi operator.

3.29
V2G Communication Session
association of two specific V2G entities for exchanging V2G messages

3.30
V2G Entity
primary actor participating in the V2G communication using a mandatory or optional transmission protocol defined by this part of ISO/IEC 15118

3.31
V2G Message
message exchanged on application layer (refer to clause 8 Application Layer messages)

3.32
V2GTP Entity
V2G entity supporting the V2G Transfer Protocol

3.33
V2GTP Root CA
Certificate Authority (CA) who issues Contract Certificates and/or SECC Certificates, or who delegates ability to issue such Certificates to Sub CA.

4 Symbols and abbreviated terms

For the purposes of this document, the following abbreviations apply:

BEV	Battery Electric Vehicle
CA	Certificate Authority

CRL	Certificate Revocation List
DH	Diffie Hellman
DER	Distinguished Encoding Rules
ECDSA	Elliptic Curve Digital Signature Algorithm
EV	Electric Vehicle
EVCC	Electric Vehicle Communication Controller
EVSE	Electric Vehicle Supply Equipment
EXI	Efficient XML Interchange
OCSP	Online Certificate Status Protocol
OEM	Original Equipment Manufacturer
NACK	Negative Acknowledgement
PDU	Protocol Data Unit
PEM	Privacy Enhanced Mail
PHEV	Plug-in Hybrid Vehicle
PKI	Public Key Infrastructure
PLC	Power Line Communication
PnC	Plug and Charge
SA	Secondary Actor
SAML	Security Assertions Markup Language
SDP	SECC Discovery Protocol
SDU	Service Data Unit
SECC	Supply Equipment Communication Controller
TCP	Transmission Control Protocol
V2G	Vehicle to Grid Communication
V2G CI	Vehicle-to-Grid Communication Interface
V2GTP	V2G Transfer Protocol
V2GTPPT_EXI	V2G Transfer Protocol Payload Type for EXI messages
UDP	User Datagram Protocol
UML	Unified Modeling Language
XML	Extensible Markup Language

5 Conventions

5.1 Definition of OSI based services

ISO/IEC°15118-2 is based on the conventions discussed in the OSI Service Conventions (refer to ISO 10731) as they apply for the individual layers specified in this document.

This part of ISO/IEC°15118-2 describes requirements applicable to layer 3-7 according to the OSI layered architecture.

5.2 Requirement structure

This document uses a requirement structure i.e. a unique number identifies each individual requirement included in this document. This requirement structure allows for easier requirement tracking and test case specification. The following format is used:

"[V2G"Y"-"XXX"]" requirement text Where:

— **"**V2G**"** represents the ISO/IEC°15118 set of standards,

— Y represents the document part of the ISO/IEC°15118 document set

— XXX represents the individual requirement number and

— "requirement text" includes the actual text of the requirement.

EXAMPLE **[V2G2-000]** This shall be an example requirement.

5.3 Usage of RFC references

When RFCs are referenced all “must/ must not” requirements are mandatory.

[V2G2-001] In this document, if a referenced RFC has been updated by one or several RFC, the update is fully applicable.

[V2G2-002] If an update or part of an update applicable to an RFC referenced herein is not compatible with the original RFC or the implementation described by this standard the update shall not apply.

[V2G2-003] All published Errata, for the ISO/IEC°15118 referenced RFCs, are fully applicable in this standard.

5.4 Notation used for XML schema diagrams

This standard make use of XML as a description format for V2G messages. For details with regards to the XML schema diagram notation used in this document refer to Altova XMLSpy Manual.

Allowing for an easy way to distinguish the types used for the XML schema definitions in this standard following naming conventions apply:

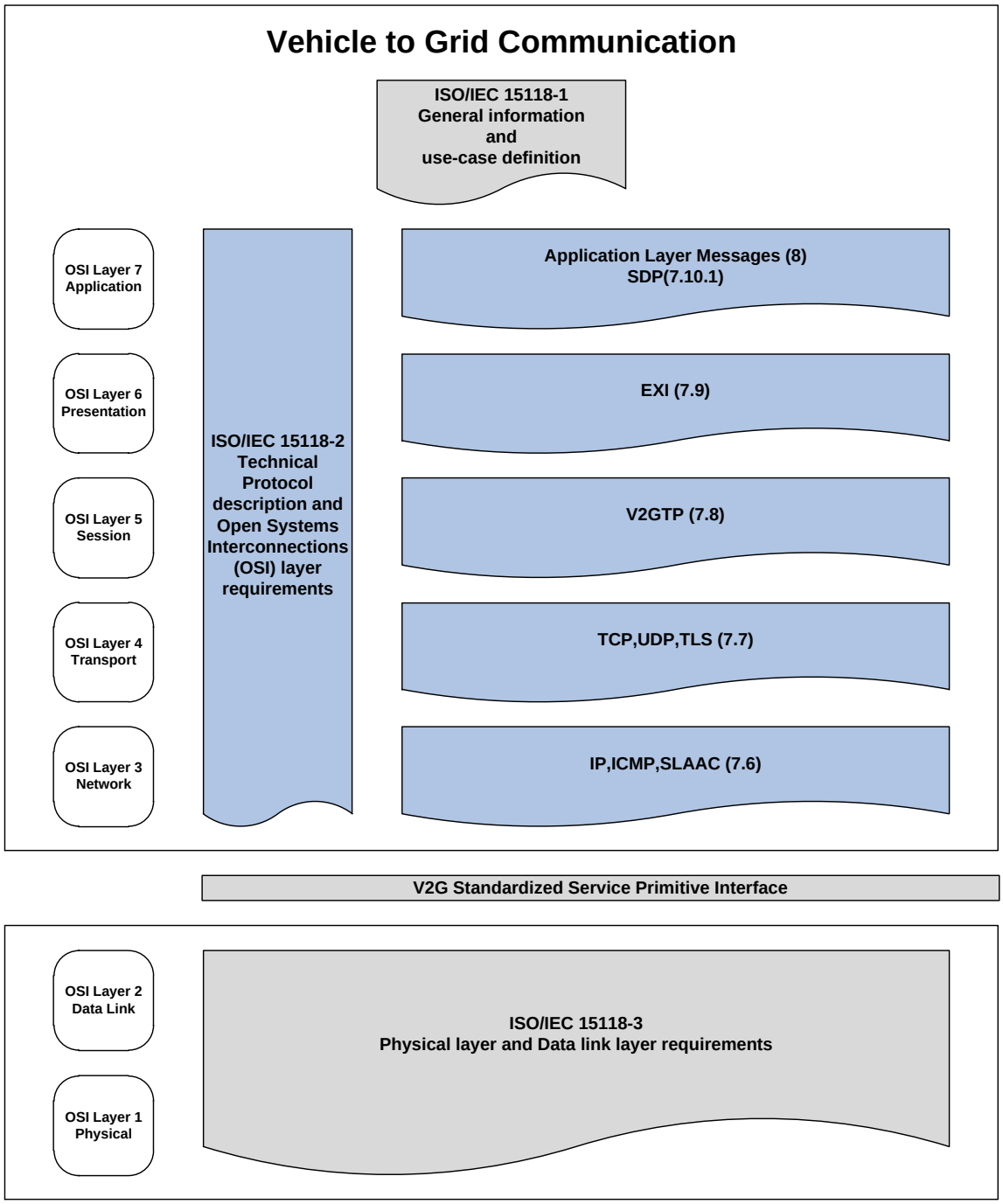
— complex type use capitalized first letters

— simple types use non capitalized first letters

6 Document overview

Figure 2 describes the organization of the different ISO/IEC°15118 documents and the usage of the subclauses , according to the OSI layered architecture.

As indicated by the blue coloured shapes this Part of ISO/IEC°15118 defines requirements applicable to layers 3-7 according to the OSI layered archictecture. Layer 1 and 2 requirements including the V2G Standardized Service Primitive Interface are specified in Part 3 of this standard.



Key

OSI Layers and applicable requirements described in this Part of ISO/IEC°15118

OSI Layers and applicable requirements defined in other Parts of ISO/IEC°15118

Figure 2 — Vehicle to Grid Communication document overview

7 Basic requirements for V2G Communication

7.1 General information

This Part of ISO/IEC 15118 describes the realization of the V2G use cases elements defined by Part 1 of this standard.

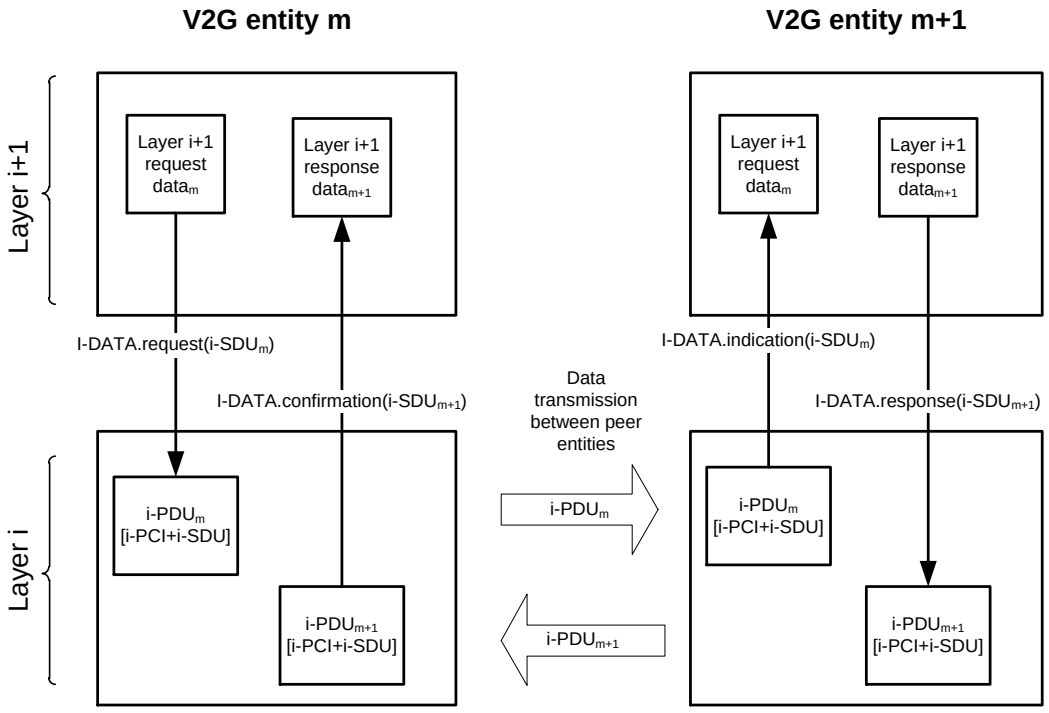
7.2 Service primitive concept of OSI layered architecture

7.2.1 Overview

This subclause explains how the OSI layered architecture is applied for the purpose of this document. It is intended to provide simple means for describing the interfaces between the individual communication protocol layers required by this document and furthermore allows for defining timing requirements more precisely.

Services are specified by describing the service primitives and parameters that characterize a service. This is an abstract definition of services and does not force a particular implementation.

Figure 3 depicts a simplified view of OSI layer interaction sufficient to understand the OSI layered architecture principles for the context of this document.



Key

PDUX: Protocol Data Unit of network entity x
PCI: Protocol Control Information
SDU_X: Service Data Unit of network entity x

Figure 3 — OSI layered architecture principles

When a layer i+1 instance of V2G entity m exchanges data with a layer i+1 instance of V2G entity m+1 each instance uses services of an instance of layer i. A service is defined as a set of service primitives.

7.2.2 Syntax of service primitives

Service primitives are described with the following syntax:

[Initial of layer]-[NAME].[primitive type](parameter list)

— whereas [initial of layer] is one out of the following seven:
[Physical, Data Link, Network, Transport, Session, Presentation, Application]

— whereas [NAME] is the name of the primitive

EXAMPLE Typical examples for [Name] are CONNECT, DISCONNECT, DATA; other names are used in this Part and Part 3 of this standard.

— whereas [primitive type] is one out of the following four:
[request, indication, response, confirmation]

— whereas (parameter list) includes a list of parameters separated by comma the user of the service is supposed to provide when using the respective service primitive; optional parameters are marked with brackets "[..]".

NOTE In this document, the primitive type ".indication" always indicates an event asynchronously to the upper layer.

7.3 Security concept

7.3.1 Call Flows (Flow Charts)

The following two figures (Figure 4 and Figure 5) depict the principal approach for the semi-online and the online case from a security point of view, showing the necessary security services applied as well as an abstract view on the different data necessary for the operation.

The full data flow / sequence charts can be found in subclause 8.8 of this document. In these overview figures only the security relevant information shall be highlighted.

The security concept provides a basic transport based protection mechanism. For certain scenarios, the usage of Transport Layer Security (TLS) for the transport communication between EVCC and SECC is mandated. For some other scenarios, the usage of TLS is optional. Specific messages are protected on application layer (XML-messages), if data has to be protected on the way from or to a secondary actor, or if the protection has to last longer than the existence of the TLS channel. Also the concept is independent from any further protection mechanisms on lower levels than layer 3 in the OSI layer model.

Figure 4 shows an example usecase for a semi-online connection for a plug-n-charge scenario:

In this Plug-and-Charge example, all TCP/IP based communication is protected using a unilateral authenticated TLS channel between the two peers. (Note: TLS is not mandatory for certain Identification modes other than the Plug-and-Charge Identification mode). All communication is terminated at the SECC. The meter reading is cyclically signed by the vehicle to provide an agreement on the amount of electricity delivered. This information may be used for billing if local regulations permit it. The EVSE provides the charging records, containing the signed meter reading to the backend for further processing.

NOTE 1 The communication between SECC and SA in Figure 4 is shown for informational purpose only and not intended to specify a particular message sequence.

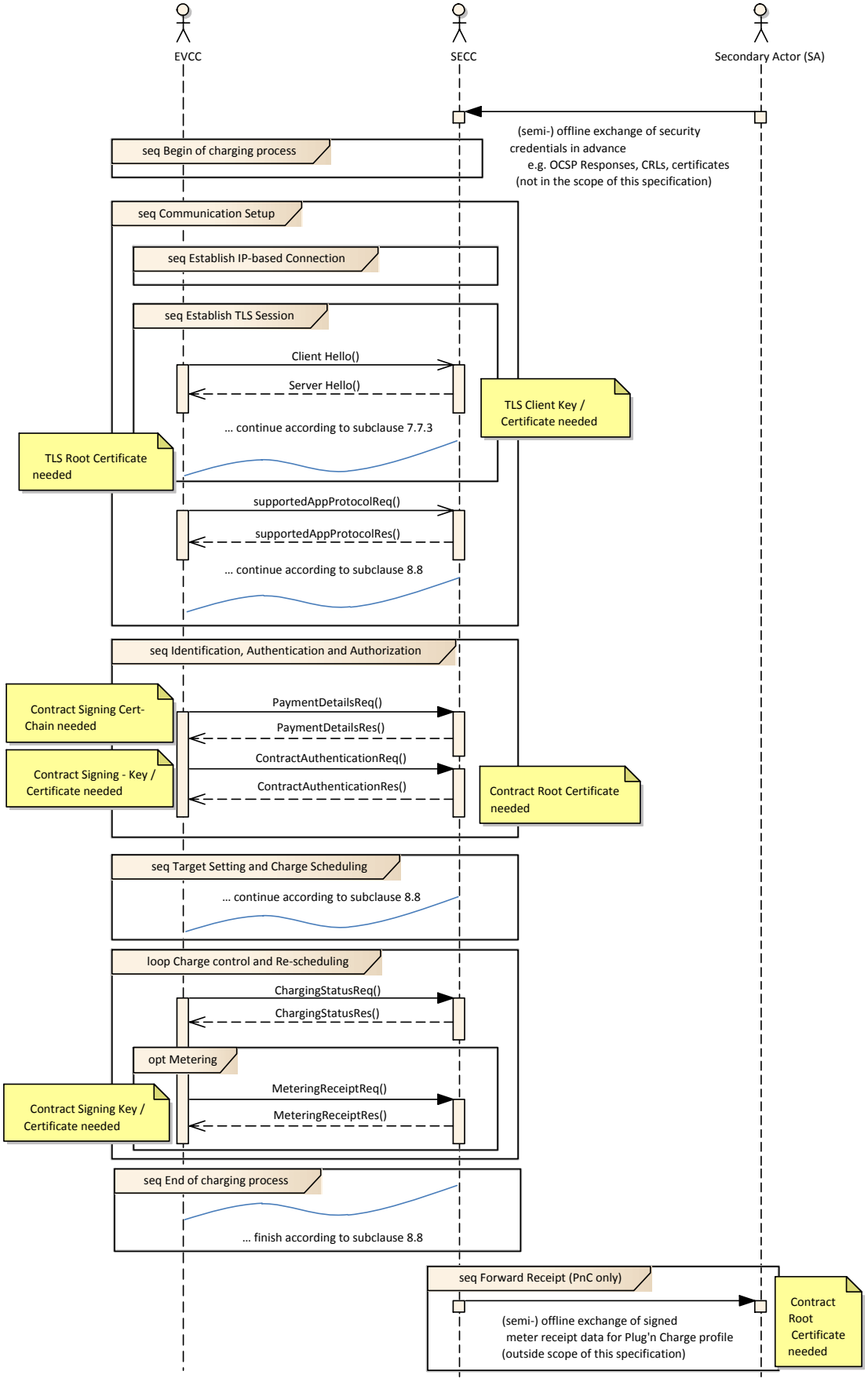


Figure 4 — Example for semi-online communication (1 of 2)

Figure 5 shows an example for an online plug'n charge usecase:

As in the semi-online case, in this Plug-and-Charge example, all TCP/IP based communication is protected using a unilateral authenticated TLS channel between the two peers. (Note: TLS is not mandatory for certain identification modes other than the Plug-and-Charge identification mode). Some of the information provided by the vehicle may need to be sent to the SA for further processing, like the contractID or the vehicles credentials to be able so sign the tarif information. The EVCC calculates a charging profile (refer to subclause 8.4.1.9.2) and sends it to the SECC. The SECC may send it to SA systems like Smart Grid or Demand Clearing House. The further process is similar to the semi-online case with the exception, that the final charging data can be directly submitted to the SA. It is assumed that the SECC will also use a secure transport connection to the SA, although the security of the vehicle communication to the SA is secured on application layer.

NOTE 2 The communication between SECC and SA in Figure 5 is shown for informational purpose only and not intended to specify a particular message sequence.

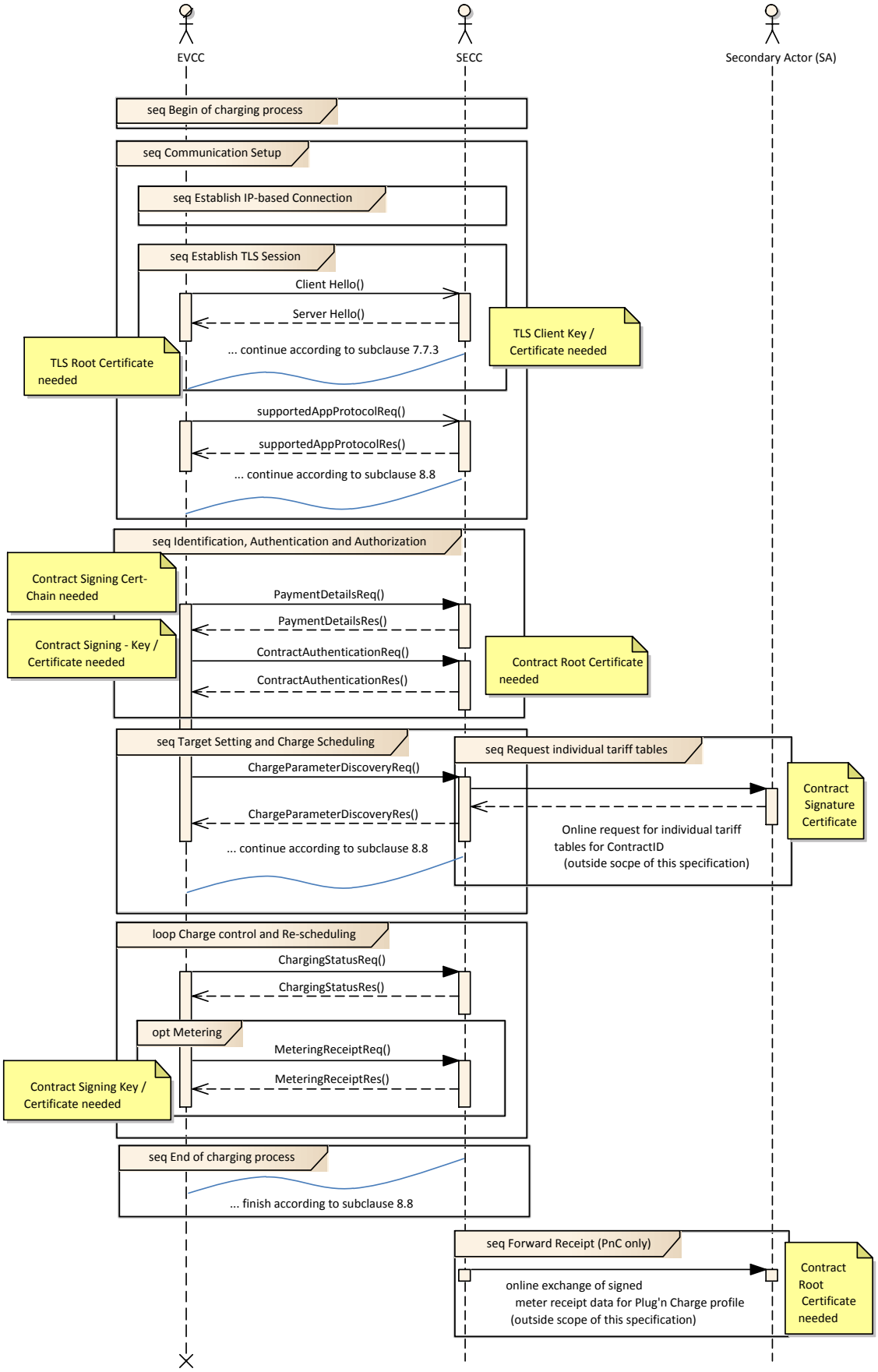


Figure 5 — Example for online communication

There are further usecases where security mechanism are needed on application layer. These are the initial enrollment of contract keys and certificate as well as the update of the certificate. In this case vehicle specific OEM keys are used. Those mechanisms are described in Annex E. An overview on all needed certificates can be found in Annex H.

7.3.2 Certificate and key management

The call flows shown in clause 7.3.1 require the existence of multiple certificates to be applied for the different security layers. There are SECC certificates used in the TLS layer for EVCC to authenticate SECC.There are Contract certificates used in the application layer for SECC and secondary actor to authenticate the contract related to SECC. There are V2G root certificates and possibly sub certificates which certify SECC certificates and Contract certificates.

Separate from above certificates, OEM may have OEM root certificates and OEM provisioning certificates to be used for installing and updating Contract Certificates.

[V2G2-004] Each V2G entity shall use X.509v3 certificates due to the need of extensions for storing EC-parameters. For details please refer to IETF RFC 5280.

Table 1 shows what fields a X.503v3-certificate consists of:

Table 1 — Basic Certificate Fields

Certificate field	Description
Version	Version of certificate (for 15118 shall be v3 = 0x2)
Serial number	Unique number of certificate (within the domain of the issuer)
Signature algorithm	Used signature algorithm
Issuer	Entity, who has issued and signed the certificate
Validity period	Time period, in which the certificate is valid
Subject	Entity, to which the certificate is issued
Public key	Public key corresponding to a private key
Issuer UID	Optional issuer unique identifier
Subject UID	Optional subject unique identifier
Extensions	Optional (see Table 2)
Signature	Signature of the certificate generated by the issuer

NOTE 1 For those not familiar with the various fields please refer to IETF RFC 5280.

Depending on the use case additional information may be included using so called certificate extensions. Table 2 summarizes common certificate extensions.

Table 2 — Certificate extension examples

Certificate extensions	Description
Key usage	Usage of the corresponding private key (e.g. Digital Signature, Non-Repudiation, Key Encipherment, …)
Extended Key Usage	Further specification of key usage using OIDs, e.g.: Server Authentication (1.3.6.1.5.5.7.3.1) Client Authentication (1.3.6.1.5.5.7.3.2) Note, sometimes here also the following values are encoded: Netscape SGC (1.3.6.1.4.1.311.10.3.3) Microsoft SGC (2.16.840.1.113730.4.1) SGC stands for Server Gated Crypto and indicated that the server may also use strong cryptography for the connection with the client's browser. This extension was used at the time of strong crypto export control to enable financial web site to provide appropriate protection of the data transfer.
CRL distribution point	Location to retrieve certificate revocation lists
OCSP	Location to retrieve OCSP response (exists only for SubCA certificates). Refer to IETF RFC 2560 for details.
Authority information	Additional authorization information
Subject alternative name	Alternative names of the entity
Basic constraint = CA	True if the certificate is V2G Root Certificate or SubCA certificate.

NOTE 2 For those not familiar with OIDs, e.g 1.3.6.1.5.5.7.3.1, please refer to: Object Identifier (OID) Repository.

[V2G2-005] Each V2G entity shall support Hash-operation SHA-256 (signature process) according to NIST FIPS PUB 180-3 (for Part 1 Use Case Element ID: F1).

[V2G2-006] For each V2G entity the Signature-operation shall be ECC-based using elliptic curves (secp256r1[SECG notation]) with signature algorithm ECDSA (for Part 1 Use Case Element ID: F1).

[V2G2-007] The Key length for ECC based asymmetric cryptography each V2G entity shall be 256 bit.

NOTE 3 For the time being ECC algorithms are not supported by the W3C XML digital signature specification. It is expected that ECC will be supported for the FDIS of this standard. If final standard won't be available, the latest release candidate will be referenced.

7.3.3 Number of root certificates and root validity, certificate depth and size

[V2G2-008] Each V2G entity shall support at least one V2G root certificate.

NOTE 1 A number of five (5) V2G root certificates is strongly recommented. However, just one is mandatory. Having less than 5 or just 1 certificate brings a risk to the OEM. Having just 1 V2G root certificate allows the car to charge just in that one region. The OEM will need to state in the manual and during offering the car to customers, that this car charges only in its "home" region. If an OEM is afraid, that 5 V2G root certificates are not sufficient to cover the "usage radius" of its cars, OEM is free to provide more root certificate storage locations.

NOTE 2 It is assumed that the V2G root certificates applied on OSI layer 4 are also used on OSI layer 7. Additionally it is assumed that in Europe there will be a single root certificate authority similar to the Trust Center that was established for the EURO5/EU5. It is also assumed that other regions of the world will also manage to have a root certificate authority covering a greater number of actors. This leads to the assumption that 5 root certificates for 5 world-regions are sufficient. If one decides for more space for certificates this doesn't affect compatibility. For the SECC however it is mandatory to store more since there might be 10 root certificates valid concurrently for each world-region. This requirement results in as much as 50 certificates to be stored.

[V2G2-009] The path length constraint of the PKI certificate tree shall be limited to 3.

NOTE 3 The path length constraint defines the number of non selfsigned certificates in a certification path, i.e. there will be up to 3 certificate layers derived from the root certificate.

[V2G2-010] The size of a certificate in DER encoded form shall be not bigger than 800 Bytes. Alternatively certificates may be represented in a PEM document as well (compatibility is granted).

[V2G2-011] The validity period of any V2G root certificate shall be 40 years.

[V2G2-012] At any point in time there shall be a V2G root certificate available which is valid for each V2G root CA at least next 35 years.

NOTE 4 For explanations of certificate validity, number of root certificates, certificate depth and size please refer to Annex E.

7.4 V2G communication states

This subclause describes the basic states of the communication between EVCC and SECC. The timer and timeout values used in this subclause are described in subclause 8.8.

Figure 6 depicts the general communication states of the V2G communication from an EVCC perspective.

[V2G2-014] After data link layer connection is established, the EVCC shall initiate the address assignment mechanism as defined in subclauses 7.6.3.2 and 7.6.3.3.

NOTE 1 In this standard this is described by D-LINK_READY.indication(DLINKSTATUS=Link established). For details refer to Part 3.

[V2G2-015] After the Application Layer requests the start of a Communication Session, the EVCC shall initiate the address assignment mechanism as defined in subclauses 7.6.3.2 and 7.6.3.3.

[V2G2-016] The EVCC shall process the IP address assignment mechanism as defined in subclauses 7.6.3.2 and 7.6.3.3.

[V2G2-017] The EVCC shall stop the IP address assignment mechanism as defined in subclauses 7.6.3.2 and 7.6.3.3 when V2G_EVCC_CommunicationSessionSetup_Timer is equal or larger than V2G_EVCC_CommunicationSessionSetup_TimeOut.

[V2G2-018] After a link-local IP address is assinged, the EVCC shall start the process for discovering the SECC address as defined in subclause 7.10.1.

NOTE 2 In this document this is described by N-IP_Address.indication(N_IP_STATUS = Link Local Address assigned). For details refer to sublclause 7.6.4.

[V2G2-019] The EVCC shall process the SDP according to subclause 7.10.1.

[V2G2-645] Depending on the requested security and transport protocol in the SDP request message and the awaited security and transport protocol sent by SDP server, the EV shall decide on the application of TLS.

[V2G2-020] The EVCC shall stop the SDP when V2G_EVCC_CommunicationSessionSetup_Timer is equal or larger than V2G_EVCC_CommunicationSessionSetup_TimeOut.

[V2G2-021] If the EV decides to apply a secured connection, the EVCC shall establish the TLS connection to the SECC as described in subclause 7.7.3 after the SECC IP address, port and available transport protocol and security options are discovered.

[V2G2-646] If the EV decides to apply an unsecured connection, the EVCC shall not establish a TLS connection and the EVCC shall initiate the V2G Communication Session as defined in clause 8.

NOTE 3 In this document this is described by N-SECC_Address.indication(N_SECC_STATUS = SECC IP-address assigned). For details refer to sublclause 7.10.1.7.

[V2G2-022] The EVCC shall attempt to establish a TLS connection according to 7.7.3.

[V2G2-023] The EVCC shall stop to attempt to establish a TLS connection when V2G_EVCC_CommunicationSessionSetup_Timer is equal or larger than V2G_EVCC_CommunicationSessionSetup_TimeOut.

[V2G2-024] After the TLS connection is established, the EVCC shall initiate the V2G Communication Session as defined in clause 8.

NOTE 4 In this document this is described by N-TLS_Connection.indication (N_SECC_STATUS = Established). For details refer to sublclause 7.7.3.5.

[V2G2-025] The EVCC shall stop the communication and terminate the TLS session after the application layer requests to stop the session.

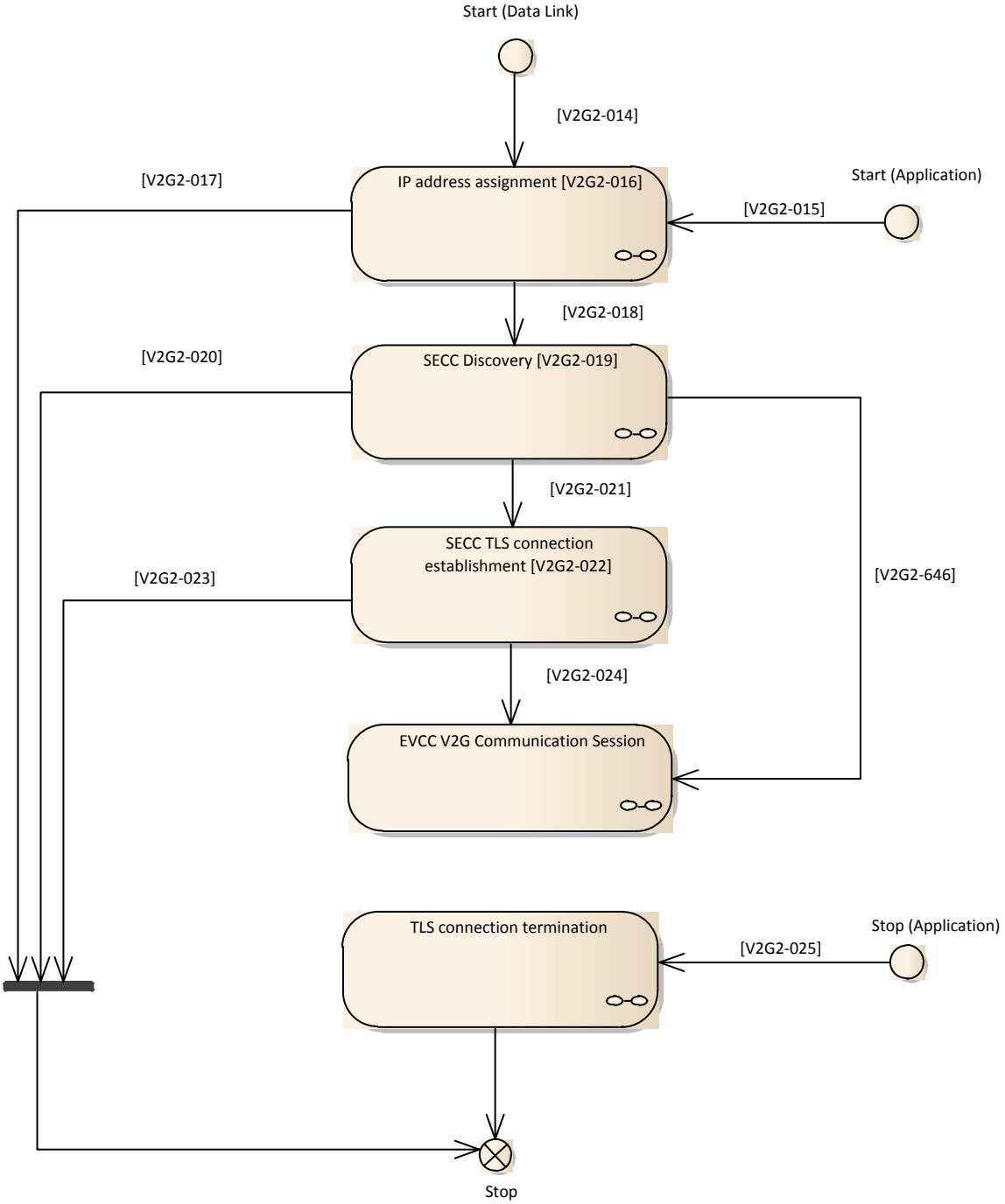


Figure 6 — Overview V2G communication states EVCC

Figure 7 depicts the general communication states of the V2G communication from an SECC perspective.

[V2G2-026] The SECC shall configure an IP address (static or dynamic) by any appropriate mechanism.

[V2G2-027] The SECC discovery service shall be started after an IP address for the SECC is configured.

[V2G2-028] The SECC discovery service shall be updated after an IP address for the SECC is changed.

NOTE 5 It is not required that the SECC discovery service is implemented in the SECC directly. It is also possible to have a separate unit providing the SECC discovery service.

[V2G2-029] The SECC shall stop the IP address assignment mechanism when V2G_SECC_CommunicationSessionSetup_Timer is equal or larger than V2G_SECC_CommunicationSessionSetup_TimeOut.

[V2G2-030] After the IP address is configured, the SECC shall wait for a TLS connection at the IP address and port if TLS (0x00) was indicated in the SDP response message as defined in subclause 7.10.1.5. If a plain TCP connection is opened instead, the SECC shall terminate the connection.

NOTE 6 In this document this is described by N-IP_Address.indication(N_IP_STATUS = Link Local Address assigned) or N-IP_Address.indication(N_IP_STATUS = Global Address assigned). For details refer to sublclause 7.6.4.

[V2G2-647] After the IP address is configured, the SECC shall wait for the initialization of the V2G Communication Session as defined in clause 8, if no transport layer security was indicated (0x10) in the SDP response message.

[V2G2-031] The SECC shall wait unitl the TLS connection is established.

[V2G2-032] The SECC shall stop waiting for establishing the TLS connection when V2G_SECC_CommunicationSessionSetup_Timer is equal or larger than V2G_SECC_CommunicationSessionSetup_TimeOut.

[V2G2-033] After the TLS connection is established, the SECC shall wait for the initialization of the V2G Communication Session as defined in clause 8.

NOTE 7 In this document this is described by N-TLS_Connection.indication (N_SECC_STATUS = Established). For details refer to sublclause 7.7.3.5.

[V2G2-034] The SECC shall stop the communication and terminate the TLS session after the application layer requests to stop the session.

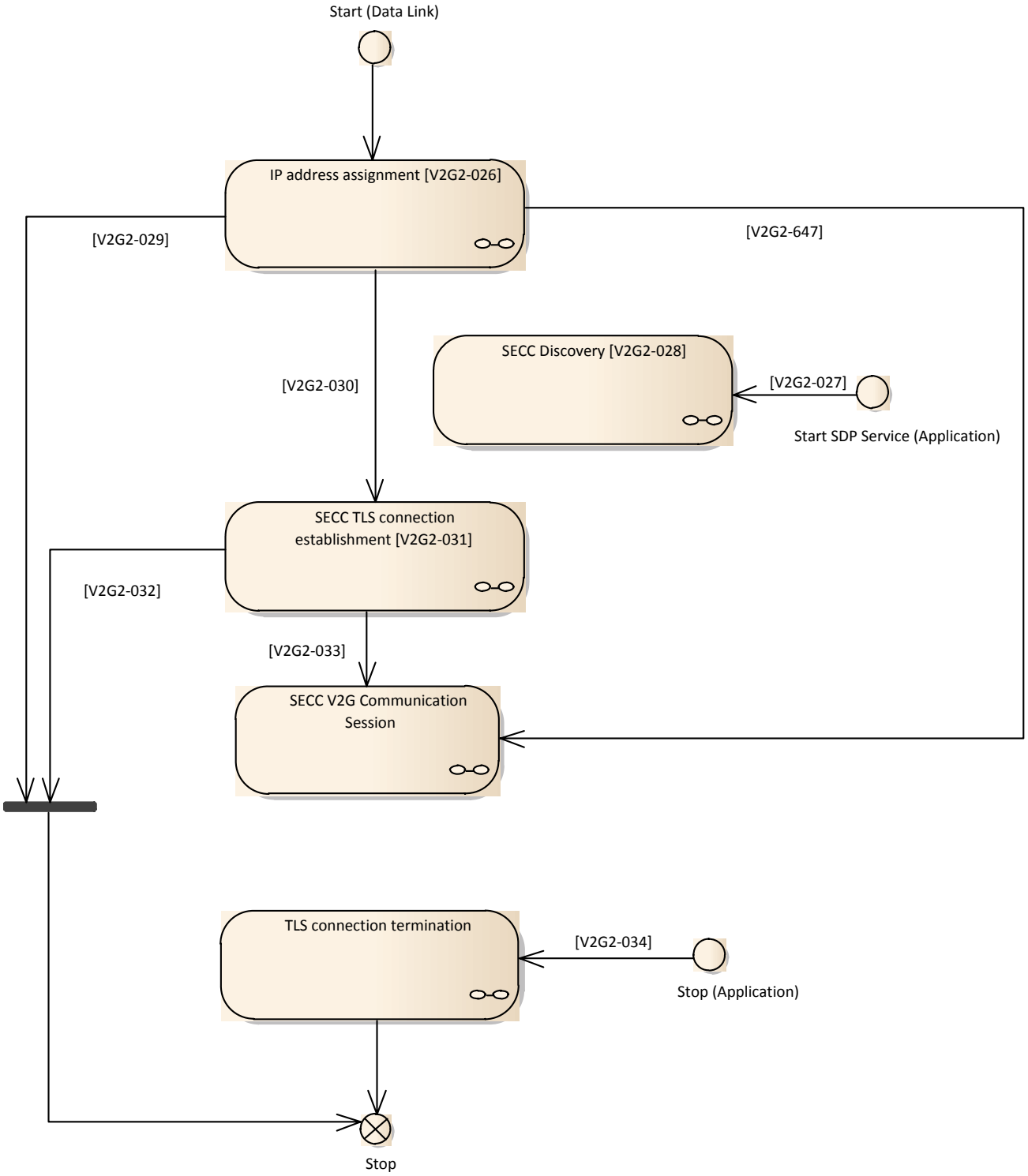


Figure 7 — Overview V2G Communication Session states SECC

7.5 Data Link Layer

The definitions in this document assume the Data Link layer to support the transport of IP packets as defined in the requirements. Part 3 of this standard defines additional details on Data Link Layer to be covered.

[V2G2-035] If the EVCC communicates by PLC, the EVCC shall comply with ISO/IEC 15118-3.

[V2G2-036] If the SECC communicates by PLC, the SECC shall comply with ISO/IEC 15118-3.

7.6 Network Layer

7.6.1 General

The protocol specified in this standard is based on the Internet Protocol standard known as IPv6 (see IETF RFC 2460).

7.6.2 Applicable RFCs and limitations and protocol parameter settings

7.6.2.1 IPv6

[V2G2-037] A V2G entity shall support IPv6 as defined in IETF RFC 2460.

[V2G2-038] While IETF RFC 2460 defines IPsec as mandatory. A V2G entity is not required to implement IPsec.

[V2G2-039] No V2G entity shall implement the RH0 routing header as specified IETF RFC 5095, which updates IETF RFC 2460.

NOTE 1 The IANA allocation guidelines for the routing type field in the IPv6 routing header are described in IETF RFC 5871. It is recommended to adhere to these guidelines.

[V2G2-040] A V2G entity shall implement path MTU discovery according to IETF RFC 1981.

[V2G2-041] A V2G entity shall support handling of overlapping IP fragments according to IETF RFC 5722.

NOTE 2 A V2G entity should comply with the specification in IETF RFC 5220, which extends IETF RFC 2460.

[V2G2-042] When sending an IPv6 packet from EVCC to SECC or from SECC to EVCC no IP fragmentation shall be used.

NOTE 3 The communication between EVCC and Secondary Actors is out of scope of this Part of ISO/IEC 15118 and may or may not use IP fragmentation.

7.6.2.2 Dynamic Host Control Protocol (DHCPV6)

Data-link requirements are described in Part 3 of this standard. The EVCC starts the address assignment triggered by the data-link when a data-link connection is established. This is done according to subclause 7.6.3.2 using SLAAC, which is mandatory according to this standard. DHCPv6 might be implemented as an optional IP configuration method.

Alternatively, if requirements **[V2G2-043]** and **[V2G2-044]** are not implemented by the EVCC and the SECC, an equivalent method might be choosen to retrieve a DNS server IP address e.g. by SLAAC.

[V2G2-043] An EVCC should implement a DHCPv6 client according to IETF RFC 3315.

[V2G2-044] The infrastructure the EV is connected to (EVSE) should implement a DHCPv6 server according to IETF RFC 3315.

7.6.2.3 Neighbor Discovery (ND)

Primary and secondary actors use IPv6 stateless address auto configuration for generating addresses for their interfaces. All interfaces have a link-local address. To ensure unique addresses and to support global addresses the neighbour broadcast protocol is used.

[V2G2-045] The EVCC shall implement ND as defined in IETF RFC 4861.

[V2G2-046] The EVCC shall comply with IETF RFC 4429 allowing assignment of IP addresses before Duplicate Address Detection is finished.

7.6.2.4 Internet Control Message Protocol (ICMP)

The Internet Control Message Protocol (ICMP) is used to send error messages (e.g. a requested service is not available, a host could not be reached etc.).

[V2G2-047] Each V2G entity shall implement ICMPv6 as specified in IETF RFC 4443.

[V2G2-048] Each V2G entity shall implement the ICMP message set defined in Table 3.

[V2G2-049] Each V2G entity shall implement the RFCs referred to in column 'Reference' of Table 3 describing the implementation details for the respective ICMP message type.

Table 3 — Mandatory ICMP message set

ICMP message type	ICMP message name	Reference
1	Destination Unreachable	IETF RFC 4443
2	Packet Too Big	IETF RFC 4443
3	Time Exceeded	IETF RFC 4443
4	Parameter Problem	IETF RFC 4443
128	Echo Request	IETF RFC 4443
129	Echo Reply	IETF RFC 4443
133	Router Solicitation	IETF RFC 4861
134	Router Advertisement	IETF RFC 4861
135	Neighbor Solicitation	IETF RFC 4861
136	Neighbor Advertisement	IETF RFC 4861
137	Redirect Message	IETF RFC 4861
141	Inverse Neighbor Discovery Solicitation Message	IETF RFC 3122
142	Inverse Neighbor Discovery Advertisement Message	IETF RFC 3122

7.6.3 IP Addressing

7.6.3.1 General

This section specifies how an EVCC retrieves valid IP addresses to communicate over an IP-based network. Following addresses are considered for the purpose of this standard:

— Link local IP address of EVCC

— Global IP address of EVCC, if router is present in local link

— IP address of SECC

NOTE An IPv6 host may have multiple IP addresses assigned to one physical network interface e.g. link-local and global address.

7.6.3.2 Stateless auto address configuration (SLAAC)

[V2G2-050] Each V2G entity shall support the configuration of a link-local IPv6 unicast address as specified in IETF RFC 4291.

[V2G2-051] The interface ID of the Link-Local address of a V2G entity shall be generated from its IEEE 48 bit MAC identifier according to the definition in IETF RFC 4291.

[V2G2-052] The EVCC shall support auto configuration of IP6 addresses as described in IETF RFC 4862.

[V2G2-053] If the SECC provides the message sets Certifcate Install, Certificate Update, or Value Added Services as defined in subclause 8.6.2 it shall support IETF RFC 6106 for providing a DNS server address.

7.6.3.3 Address selection

[V2G2-054] If multiple IPv6 addresses are supported, the IPv6 Default Address Selection shall be performed according IETF RFC 3484.

7.6.4 Network Layer service primitive - N-IP_Address.indication

The N-IP_Address.indication notifys about the status of the IP address assignment. Table 4 describes the service primitive and its parameter(s).

Table 4 — N-IP_Address.indication service primitive

Primitive name	N-IP_Address.indication
Entity to support	EVCC, SECC
Parameter Name	**Description**
N_IP_STATUS	- Link local address assigned - Global address assigned - Error

N-IP_Address.indication (N_IP_STATUS= Link local address assigned) indicates the assignment of a local IP address.

N-IP_Address.indication (N_IP_STATUS= Global address assigned) indicates the assignment of a global IP address.

N-IP_Address.indication (N_IP_STATUS= Error) indicates any detected error during IP assignment.

7.7 Transport Layer

7.7.1 Transmission Control Protocol (TCP)

7.7.1.1 Overview

The Transmission Control Protocol (TCP) allows applications of V2G entities to establish a reliable data connection to other entities. To exchange in a reliable way and in-order delivery of sender to receiver data. Additionally TCP provides flow control and congestion control and also provides for various algorithms to handle congestion and influence flow control.

7.7.1.2 Applicable RFCs, limitations and protocol parameter settings

[V2G2-055] Each V2G entity shall implement TCP as specified in IETF RFC 793.

7.7.1.3 TCP Performance and checksum requirements

The following requirements define TCP implementation details relative to congestion control, retransmission, timing, initial window size and Selective Acknowledgement for the purpose of improving the overall performance of TCP.

It is recommended to use the following congestion control and re-transmission algorithms in addition to the standard TCP methods:

[V2G2-057] Each V2G entity should implement TCP congestion control according to IETF RFC 5681.

[V2G2-058] Each V2G entity should implement the NewReno Modification to TCP's Fast Recovery Algorithm according to IETF RFC 3782.

[V2G2-059] Each V2G entity should compute TCP's retransmission timer according to IETF RFC 6298.

[V2G2-060] To increase TCP's performance each V2G entity should implement TCP Extensions for High Performance according to IETF RFC 1323.

[V2G2-061] Each V2G entity should support TCP Selective Acknowledgment Options according to IETF RFC 2018.

[V2G2-062] Each V2G entity should implement the User Timeout Option according to IETF RFC 5482.

[V2G2-063] The urgent pointer for TCP shall not be used by any V2G entity.

It is recommended to use the following checksum algorithm:

[V2G2-064] The checksum fields required in TCP headers should be implemented according to IETF RFC 1624.

7.7.2 User Datagram Protocol (UDP)

7.7.2.1 Overview

The User Datagram Protocol (UDP) is a connectionless protocol. UDP does not provide the reliability and ordering guarantees that TCP does. Packets may arrive out of order or may be lost without notification of the sender or receiver. However, UDP is faster and more efficient for many lightweight or time-sensitive purposes. UDP is located on the Transport Layer of the OSI layered architecture model.

Currently there is no use case utilizing UDP, for which security mechanisms on UDP would be required (refer Part 1 Annex B for details on security use cases).

7.7.2.2 Applicable RFC, limitations and protocol parameter settings

[V2G2-065] Each V2G entities shall implement User Datagram Protocol according to IETF RFC 768.

7.7.3 Transport Layer Security (TLS)

7.7.3.1 Overview

Security on Transport Layer is being provided by using TLS. This allows to establish an authenticated and encrypted (ensures integrity protection and confidentiality protection) channel between the EVCC and the SECC. TLS allows for unilateral or mutual authentication. For ISO/IEC 15118 security it was agreed to use unilateral authentication (the EVCC authenticates the SECC).

7.7.3.2 Applicable RFCs

[V2G2-067] For the considered use cases unilateral authentication with TLS version 1.2 according to IETF RFC 5246 with extensions according to IETF RFC 6066 shall be supported by each V2G entity. The EVCC authenticates the SECC by verifying the SECC certificate (chain) provided from the SECC to the EVCC.

With unilateral authentication the EVCC authenticates the SECC according to [V2G2-067] , which can also be used to protect the cyclic meter reading between the SECC and the EVCC. Application of unilateral authenticated TLS saves the additional digital signature of meter readings on the SECC side due to the availability of a secure session.

NOTE 1 In case of payment at the SECC, the cyclic meter reading is terminated at the SECC and does not need to be sent to a third party. Thus, there is no further signature of the billing relevant data necessary, contrary to the use cases ISO/IEC 15118-1 / C1 and C2 (contract credentials necessary), where billing relevant information is signed at application layer by the EVCC, if local regulations permit it, and forwarded through the SECC to the backend.

The application of unilateral authentication prohibits the SECC to verify the correctness of EVCC. Thus, the SECC may not detect, if the connecting EVCC is authentic. Verification of correctness of the EVCC can be achieved in the application layer as described subclause 7.9.2.

7.7.3.3 Transport Layer Security Usage

[V2G2-068] The SECC shall always act as the TLS server component.

[V2G2-069] Each EVCC shall check the validity of the SECC certificate, when TLS is used.

[V2G2-070] Each EVCC shall check the validity of the sub CA certificate (in its certificate chain) via an OCSP response according to IETF RFC 2560, when TLS is used. The OCSP Response shall be transported as part of the TLS handshake using the extension defined in IETF RFC 6066.

[V2G2-649] SECC should update (cache) the OCSP response at least once a week. One solution for updating might be for example an online connection.

[V2G2-650] Although a valid time in the EVCC is not mandatory, the EVCC should implement a mechanism discarding outdated certificates from the SECC.

NOTE It is out of scope of this standard how errors are handled. It is up to the OEM how these errors are handled.

[V2G2-651] The EVCC shall send a list of V2G Root Certificates it possesses, an extension of type "trusted_ca_keys" in the (extended) client hello as defined in IETF RFC 6066.

7.7.3.4 Transport Layer Security Credentials and Cipher Suites

For transport layer security, EVCC authenticates SECC using SECC certificate. This is being achieved by SECC having a private key corresponding to SECC certificate, and EVCC having V2G Root certificate and verifying the certificate chain from the V2G Root Certificate to the SECC certificating. The validity check of sub CA Certificate in the Certificate chain is performed via the OCSP response received during TLS-handshake (for details please refer to IETF RFC 6066). Mechanism to revoke SECC certificates are not required but instead, it is required to be short term certificates, less than 4 weeks.

As SECC is not aware of which V2G Root certificate EVCC has, among 10 currently valid V2G Root certificates for one region, thus 50 currentlly valid V2G Root certificates worldwide.Therefore it is necessary for EVCC to provide a list of V2G Root certificates that EVCC posses, through TLS handshake. SECC provides a certificate chain of its own SECC certificate whose root certificate is possed by EVCC, together with OCSP response for the sub-CA certificate in the chain. It is strongly recommended that the OCSP response is freshed at least every week.

The communicated message between EVCC and SECC is encrypted using a symmetric key (one-way “key agreement”) negotiated during the TLS key negotiation phase. Therefore the message will not be eavesdropped and both EVCC and SECC can be certain that the opponent is indeed identical thoroughout the session. (There can be no session hijacking.)

[V2G2-071] Each V2G entity shall provide the credentials and security information stated in Table 5, if TLS is used.

Table 5 — TLS authentication

TLS Authentication	Requirements to EVCC	Requirements to SECC
Unilateral (server side)	Availability of root certificates to check the authenticity of the SECC certificate Functional support for OCSP request/response processing to check the validation state of the SECC certificate during TLS handshake	SECC certificate and corresponding private key OCSP response to provide information about the validation state of own SECC certificate The SECC shall cache and store internally OCSP responses for its own chain of sub-cs certificates regulary (at least once per week).

[V2G2-072] Each V2G entity shall support TLS, if TLS is used.

[V2G2-602] The SECC shall support all cipher suites defined in table Table 6, if TLS is used.

[V2G2-603] The EVCC shall support at least one cipher suite as listed in Table 6, if TLS is used.

Additional cipher suites may be supported by any V2G entity.

NOTE The OEM can decide, which cipher suite the EVCC supports, based on the offer which Table 6 gives. One is sufficient allowing for compact implementations. For interoperability, of course, the SECC shall support all cipher suites as listed in Table 6.

Table 6 — Supported cipher suites

Cipher suite	RFC
TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256	IETF RFC 5289
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	IETF RFC 5289
TLS_ECDHE_ECDSA_WITH_AES_128_CCM	IETF RFC 5116
TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8	IETF RFC 5116

7.7.3.5 TLS service primitives

The N-TLS_Connection.indication notifys about the status of the TLS connection. Table 6 describes the service primitive and its parameter(s).

Table 7 — T_TLS_Connection.indication service primitive

Primitive name	N-TLS_Connection.indication
Entity to support	EVCC, SECC
Parameter Name	**Description**
T_TLS_STATUS	- Established - Terminated - Error

N-TLS_STATUS.indication (T_TLS_STATUS=Established) indicates the successful establishment of a TLS connection.

N-TLS_STATUS.indication (T_TLS_STATUS=Terminated) indicates the termination of a TLS connection.

N-TLS_STATUS.indication (T_TLS_STATUS= Error) indicates any detected error related to a TLS connection.

7.8 V2G Transfer Protocol

7.8.1 General Information

The V2G Transfer Protocol (V2GTP) is a compact communication protocol to transfer V2G messages between two V2GTP entities. It mainly consist of a header and payload definition that allows to separate and process V2G messages efficiently. V2GTP is the standard transfer protocol between the EVCC and SECC but may also be used for communication with other V2G Entities that support the V2GTP protocol.

7.8.2 Supported ports

V2GTP is based on TLS+TCP. TLS+TCP uses a pair of IP addresses (source address and destination address) and a pair of port numbers (source port and destination port) to establish and identify a connection for bidirectional exchange of byte streams. The connection is established from the source address and source port to the destination address and destination port. The ports listed in Table 8 are used by V2GTP entities.

Table 8 — Supported TCP ports for V2GTP

Name	Protocol	Port number	Description
V2G_SRC_TCP_DATA	TCP(unicast)	Port number in the range of Dynmaic Ports (49152-65535) as defined in IETF RFC 6335.	V2GTP source port at a Primary Actor (e.g. EVCC) that implements the V2GTP protocol.
V2G_DST_TCP_DATA	TCP (unicast)	Port number at V2GTP entity providing a V2GTP destination port. For a SECC it will be dynamically assigned by the SDP mechanism (7.10.1)	V2GTP destination port at a Primary Actor (e.g. SECC)

For V2GTP entities implementing the V2GTP the following general requirements apply:

[V2G2-073] A V2GTP entity providing a destination port shall support at least one connection on the local port V2G_DST_TCP_DATA as defined in Table 8.

[V2G2-074] A V2GTP entity providing a destination port may support multiple simultaneous connections on the local port V2G_DST_TCP_DATA as defined by Table 8.

[V2G2-075] A V2GTP entity using a source port shall support at least one connection on the local port V2G_SRC_TCP_DATA as defined in Table 8.

[V2G2-076] A V2GTP entity using a source port may support multiple connections on the local port V2G_SRC_TCP_DATA as defined in Table 8.

Especially, for an EVCC and an SECC the following applies:

[V2G2-077] The EVCC shall use a source port V2G_SRC_TCP_DATA as defined in Table 8.

[V2G2-078] The SECC shall provide a destination port V2G_DST_TCP_DATA as defined in Table 8.

[V2G2-079] The EVCC shall support at least one connection for a V2G Communication Session on port V2G_SRC_TCP_DATA.

[V2G2-080] The SECC shall support at least one connection for a V2G Communication Session on port V2G_DST_TCP_DATA.

[V2G2-081] The EVCC shall use the port V2G_DST_TCP_DATA returned in the last SECC Discovery response message (refer to sub-clause 1.2.1) for connecting the SECC

7.8.3 Protocol Data Unit

7.8.3.1 Structure

The V2GTP PDU consists of a header and a body section as shown in Figure 8.

Figure 8 — V2GTP Message structure

The payload contains the application data (e.g. a V2G message). The header separates the payloads (i.e. individual V2GTP messages) within a byte stream and gives information for the payload processing.

The V2GTP message header structure is shown in Figure 9 and described in Table 9. The supported payload types are described in Table 10.

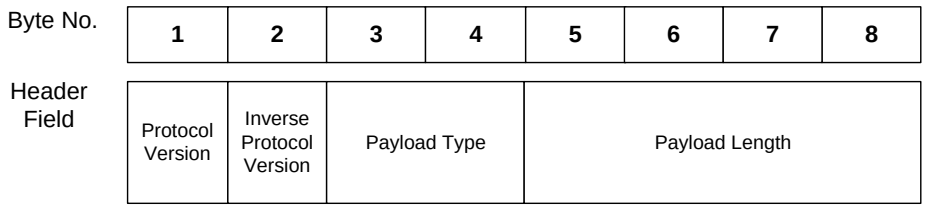


Figure 9 — V2GTP Message header structure

[V2G2-082] A V2GTP entity shall use the header structure as shown in Figure 9.

[V2G2-083] A V2GTP entity shall send the 8 bytes of the V2GTP header in the order as shown in Figure 9.

[V2G2-084] A byte with a lower number shall be sent before a byte with a higher number. The header starts with byte 1 and ends with byte 8.

[V2G2-085] A V2GTP entity shall send the fields “payload type” and “payload length” in big endian format: The most significant byte is sent first the least significant byte is sent last.

Table 9 — Generic V2GTP header structure

Header field	Header field description	Header field values
Protocol Version	Identifies the protocol version of V2GTP messages.	0x01: V2GTP version 1 0x00, 0x02-0xFF: reserved by document
Inverse Protocol Version	Contains the bit-wise inverse value of the protocol version which is used in conjunction with the V2GTP protocol version as a protocol verification pattern to ensure that a correctly formatted V2GTP message is received. Equals the <Protocol_Version> XOR 0xFF	0xFE: V2GTP Version 1
Payload type (GH_PT)	Contains information about how to decode the payload following the V2GTP header.	Refer to Table 10 for a complete list of payload type values.
Payload length (GH_PL)	Contains the length of the V2GTP message payload in bytes (i.e. excluding the generic V2GTP header bytes).	0…4294967295 (= <d>)

Table 10 — Overview on V2GTP payload types

Payload type value	Payload type name	Specified in subclause
0x0000 - 0x8000	Reserved	not applicable
0x8001	EXI encoded V2G Message	7.9.1.2
0x8002 - 0x8FFF	Reserved	not applicable
0x9000	SDP request message	7.10.1.4
0x9001	SDP response message	7.10.1.5
0x9002 - 0x9FFF	Reserved	not applicable
0xA000 - 0xFFFF	Manufacturer specific use	not applicable

[V2G2-086] A V2GTP entity shall use the V2GTP message structure as shown in Figure 8 to send V2G messages as defined in clause 8.

[V2G2-087] A V2GTP entity shall use the definitions as defined in Table 9 and Table 10.

[V2G2-088] For EXI encoded V2G messages (payload 0x8001) a V2GTP entity shall use a separate V2GTP message for each V2G message.

NOTE Requirement [V2G2-088] implies that the payload field can include neither a part of a message nor multiple messages.

7.8.3.2 Header Processing

For the processing of the payload the V2GTP entity has to process the header first. For this, a V2GTP entity that receives a V2GTP message checks the header field step by step. The header processing as defined below is illustrated in Figure 10.

[V2G2-089] A V2GTP entity shall process the V2GTP header as defined in Table 9 before processing the payload as defined in Table 10.

[V2G2-090] A V2GTP entity shall check the protocol version and inverse protocol version fields (synchronization pattern) before any other header fields.

[V2G2-091] A V2GTP entity shall stop the communication session if it detects an error in the version and inverse version field as defined in Table 9.

[V2G2-092] A V2GTP entity shall check the Payload Type after the successful check of the version and inverse version field.

[V2G2-093] A V2GTP entity shall stop the communication session and skip the payload following the header if the payload is not supported.

[V2G2-094] A V2GTP entity shall check the Payload Length after the successful check of the Payload Type.

[V2G2-095] A V2GTP entity shall skip the payload bytes after a header and stop the V2G Communication Session if the payload size can not be processed.

[V2G2-096] If the header processing was successful the V2GTP entity shall process the payload.

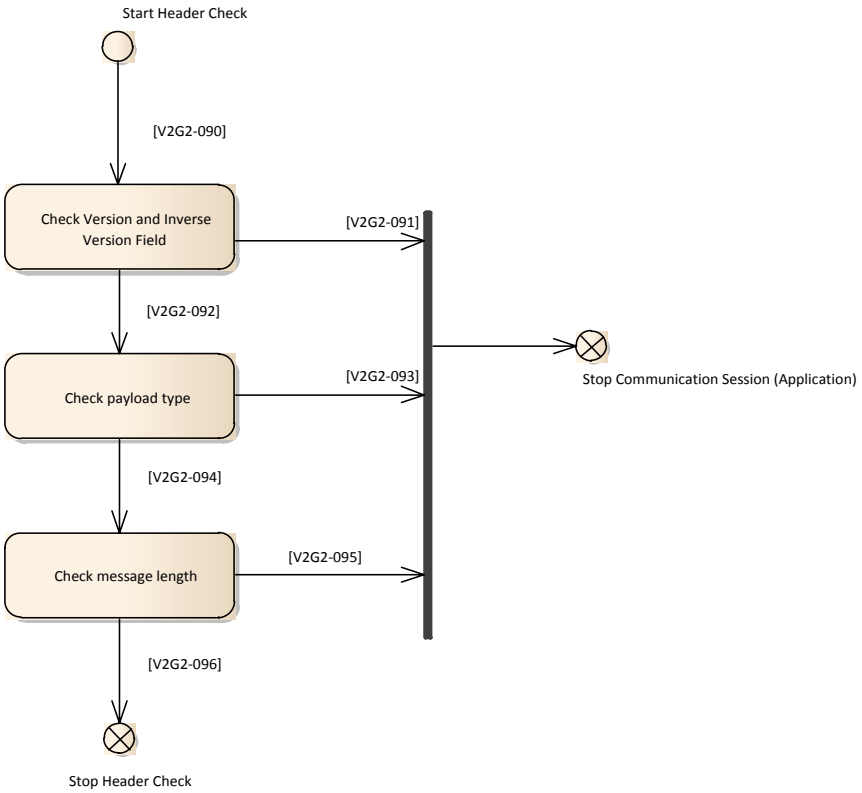


Figure 10 — V2GTP generic header handler

7.9 Presentation Layer

7.9.1 XML and Efficient XML Interchange (EXI)

7.9.1.1 Overview

For the purpose of describing the V2G message set the presentation layer uses the widely adopted XML data representation accordingly the document defines messages (i.e. data structures and data types) based on

XML Schema which allows the type aware use of XML and enables simplified validity evaluation of exchanged messages.

[V2G2-097] When transmitting V2G messages defined in this standard by using XML all V2G entities shall use encoding format accroding to definitions in W3C EXI 1.0.

7.9.1.2 Efficient XML Interchange

The Efficient XML Interchange (EXI) format allows to use and process XML-based messages on a binary level. Thus, the EXI format increases the processing speed of XML-based data as well as reduces the memory usage. Basically, EXI is a W3C recommendation. The EXI format uses a relatively simple grammar driven approach that achieves very efficient encodings for a broad range of use cases. It is not uncommon for EXI messages to be up to 100 times smaller than equivalent XML documents. The EXI specification describes in a predefined process how schema information has to be transformed into EXI grammar. The reason for doing so is that EXI grammar is much simpler to process, compared to XML Schema information. Nevertheless the parsing can be performed in the same accurate way as it is possible in XML.

There are different kinds of coding mechanism with EXI. To meet the demands in ISO/IEC 15118 in terms of efficient processing, less resources usage, message size, and message extendibility schema-aware settings should be selected (see section xxx for the requirement EXI settings for ISO/IEC 15118).

In general, EXI streams can be created in a very efficient way if all encoded information (elements/attributes) are defined by an underlying XML Schema (*Schema-informed Grammars*). Deviant information based on XML Schema knowledge is encoded in a more generic way. The EXI coder encodes the qualified names (namespace and element/attribute name) of such unknown information in a string-based manner. However, simple types of the schema deviations may be still encoded type-aware.

EXI decoders are able decode the efficient EXI streams by using the same underlying XML Schema which was used for the encoding process. Schema deviations are recognized in the EXI stream. These deviations (unknown elements or attributes) can be either processed or skipped.

Figure 11 summarizes the Efficient XML Interchange concept for the ISO/IEC 15118 domain. Due to the high limited resource restriction the EVCC may only be able to handle the XML-based data using a corresponding data structure representation. Such data structures can be used to serialize or deserialize ISO/IEC 15118 application messages. Meanwhile, the SECC may be able to handle the data as data structure and/or the more resource intensive Document Object Style (DOM) or in a traditional plain-text XML variant.

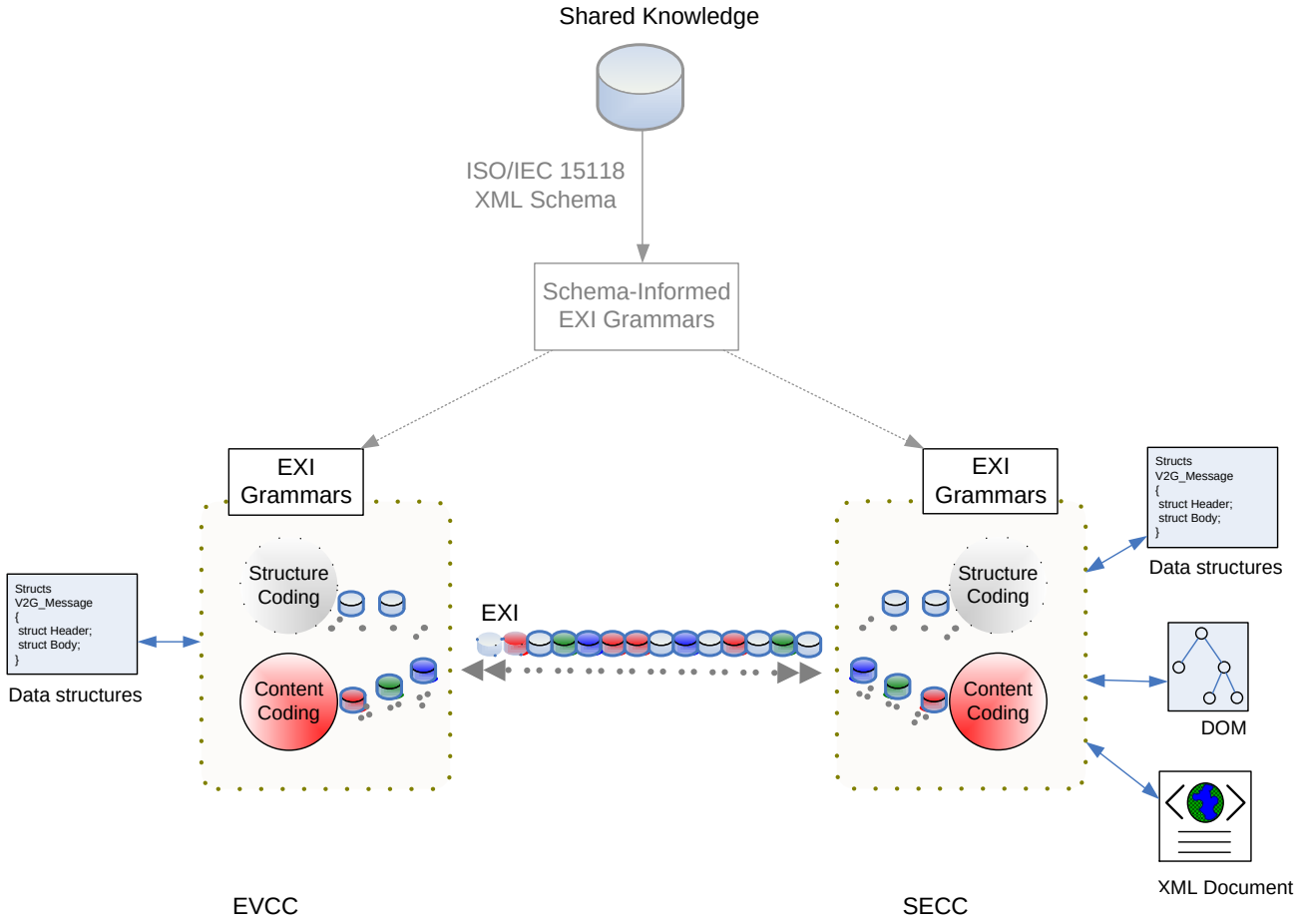


Figure 11 — Basic concept of EXI applied to V2G communication

7.9.1.3 EXI Settings for application layer messages

The following EXI settings are used for the EXI-based V2G communication.

[V2G2-098] The XML Schema with the namespace "urn:iso:15118:2:2010:MsgDef" that represents this ISO/IEC 15118 version 1.0 (major version "1", minor version "0") shall be used for encoding and decoding EXI streams.

[V2G2-099] The EXI coder for encoding and decoding of the ISO/IEC 15118 communication shall use the default EXI coding options according to W3C EXI 1.0 subclause "EXI Options" with the exception of the options listed in Table 11.

Table 11 — EXI option settings

EXI Option	Description	ISO/IEC 15118 value
schemaID	Identify the schema information, if any, used to encode the EXI body.	Schema ID that is provided by the SECC (also see application layer protocol handshake in sub-clause 8.2)
valuePartitionCapacity	Specifies the total capacity of value partitions in a string table	0

[V2G2-100] The EXI header (refer to W3C EXI 1.0 clause "Header") shall be used in a way that fulfils the ISO/IEC 15118 needs. That means, the optional EXI Cookie ($EXI) shall never be used and the

Presence Bit for EXI Options shall be always set to 0 (=false). As a consequence the optional EXI Options shall never be part of an ISO/IEC 15118 message. Each EXI implementation (on EVCC or SECC side) shall discard messages that do not respect the ISO/IEC 15118 EXI header options.

[V2G2-101] An element/attribute which is not defined in “urn:iso:15118:2:2010:MsgDef” namespace shall be encoded and decoded as schema deviation case according to W3C EXI 1.0 subclause "Adding Productions when Strict is False"

[V2G2-600] The EXI coder for encoding and decoding of the ISO/IEC 15118 communication shall use the EXI profile settings (refer to W3C EXI Profile) according to Table 12.

Table 12 — EXI profile settings

EXI profile parameter	Description	ISO/IEC 15118 value
maximumNumberOfNamePartitionCharacters	Defines the maximum number of characters that can be inserted in the name partitions entries. Name partitions entries consist in all URI, and local name partition entries inserted after the partitions pre-population.	100
maximumNumberOfNamePartitionEntries	This is the maximum number of name partitions entries. Name partitions entries consist in all URI, and local-name partition entries inserted after the partitions pre-population.	10

[V2G2-102] A simple type/value of an element/attribute which is not defined in the “urn:iso:15118:2:2010:MsgDef” namespace shall be encoded and decoded type-aware.

7.9.2 XML Security

XML Security refers to a set of technologies that enable integrity, confidentiality and authentication of XML-based messages. XML Encryption and XML Signature are two W3C recommendations that address the requirements of some data fragments (e.g. metering information) of the XML-based V2G. XML Encryption defines a mechanism by which messages and message parts can be encrypted with a key to provide confidentiality. XML Signature defines a mechanism by which messages and message parts can be digitally signed to provide integrity, to ensure that the data is not tampered with, and authentication, to verify the identity of the message producer. If integrity and confidentiality required at the same time, these technologies can be combined to produce messages with encrypted parts and signed parts. For a basic overview on encryption and signature please refer also to Annex K.

7.9.2.1 Application layer credentials and cipher suites

Credentials to be applied on application layer must be suitable for the targeted XML security. Here, XML signature is chosen to protect billing relevant information between EVCC, SECC and/or SA. Moreover, XML Encryption is being proposed providing a confidentiality protected way for information provisioning to the EVCC without having intermediaries access this information. Both approaches require asymmetric key material. For XML security it is proposed to use ContractCertifcates and corresponding private keys for the intended services XML Signature and XML Encryption. It is assumed that an EVCC doesn’t encrypt and a SECC doesn’t sign data. The credentials for EVCC signing is provided by Contract Certificate and credentials for EVCC receiving encrypted data is provided by ECDH key exchange as described in Annex I.

[V2G2-103] The maximum lifetime of the ContractCertificate shall be no longer than 2 years.

[V2G2-104] The minimum lifetime of the certificate used for XML signature and providing mechanisms for encryption shall be 4 weeks. If the contract lifetime is less, the certificate validity period shall be mapped to the contract lifetime.

NOTE 1 For explanation of certificate validity period please refer to Annex E

NOTE 2 If the certificate is not used for charging it might be used for provisioning. A provisioning certificate enables an EVCC to request a valid contract certificate for plug and charge.

7.9.2.2 Contract Certificates as XML signature credentials

Contract certificate is bound to a ContractID and used in XML signature to authorize the vehicle for charging. The contract certificate can be verified even if the SECC is offline.The contract binding is handled as follows:

[V2G2-108] The contractID shall be encoded in the subject of the certificate.

7.9.2.3 XML Encryption Credentials

The XML Encryption credential is the same certificate as used for XML Signature. The difference is that the elliptic curve parameter in the certificate are rather used in ECDH as in ECDSA (see also Annex I).

[V2G2-114] Each V2G entity shall support Encryption-operation ECC-based using elliptic curves defined over finite prime fields with Encryption algorithm ECDH. The public parameters g and p for generating the secret parameters x and y are derived from the ECDSA parameter set. Please refer also to Annex I.

[V2G2-115] The parameters used for Diffie Hellman Key exchange shall correspond to the cryptographic strength of the used ECDSA and symmetric cryptographic mechanisms.

[V2G2-116] Each V2G entity shall use AES128 for encryption (session key).

7.9.2.4 XML Security specifics for 'PnC' Message Set(s)

7.9.2.4.1 XML Data Structures for Application Layer Security

Security on Application Layer is provided using signature and encryption of messages. Information targeted for SA services is exchanged using XML data structures. Consequently, this information can be protected end-to-end using XML Security.

Typcial information exchanged between EVCC and SECC:

In case tariff information is encrypted by the SA, it uses the EVs credentials. This requires an information exchange before the actual tariff information is being sent to provide the SECC and then the SA with the EV credentials. This approach is only possible for online connections.

Signed meter reading approval from the vehicle used for online and semi-online connections. The meter readings from the SECC are sent via the TLS protected tunnel. They may include the signature of the electricity meter providing source authentication to protect them additionally. The vehicle in turn signs the meter readings to provide a base for the billing process if local regulations permit it. This approach saves the meter reading signatures on the SECC side. The meter readings are cumulated so that the latest signed meter reading is the base for the billing. Here XML Signatures are used. They ensure integrity protection with the possibility for all intermediate and participating entities to rely on this information.

7.9.2.4.2 XML Signature mechanism

This subclause is intended as on introduction to XML Signatures.

XML Signatures as defined in W3C XMLSig can be applied to arbitrary digital content (data objects) in the same way as digital signatures are calculated. When applying a digital signature to data objects, the data objects are first digested (hashed) and the result is then signed using an asymmetric algorithm like RSA or ECDSA. In the case of XML, the digest is placed in an XML element, together with additional information. This element is then digested and cryptographically signed. This standard uses enveloping XML Signatures according to W3C XMLSig. That means the XML document is included in the XML signature. It is contained within a child element of the XML signature. The signature may comprise only a part of the XML document referenced by an ObjectID.

Figure 12 shows the schema diagram of the XML Signature element included in the V2G message header.

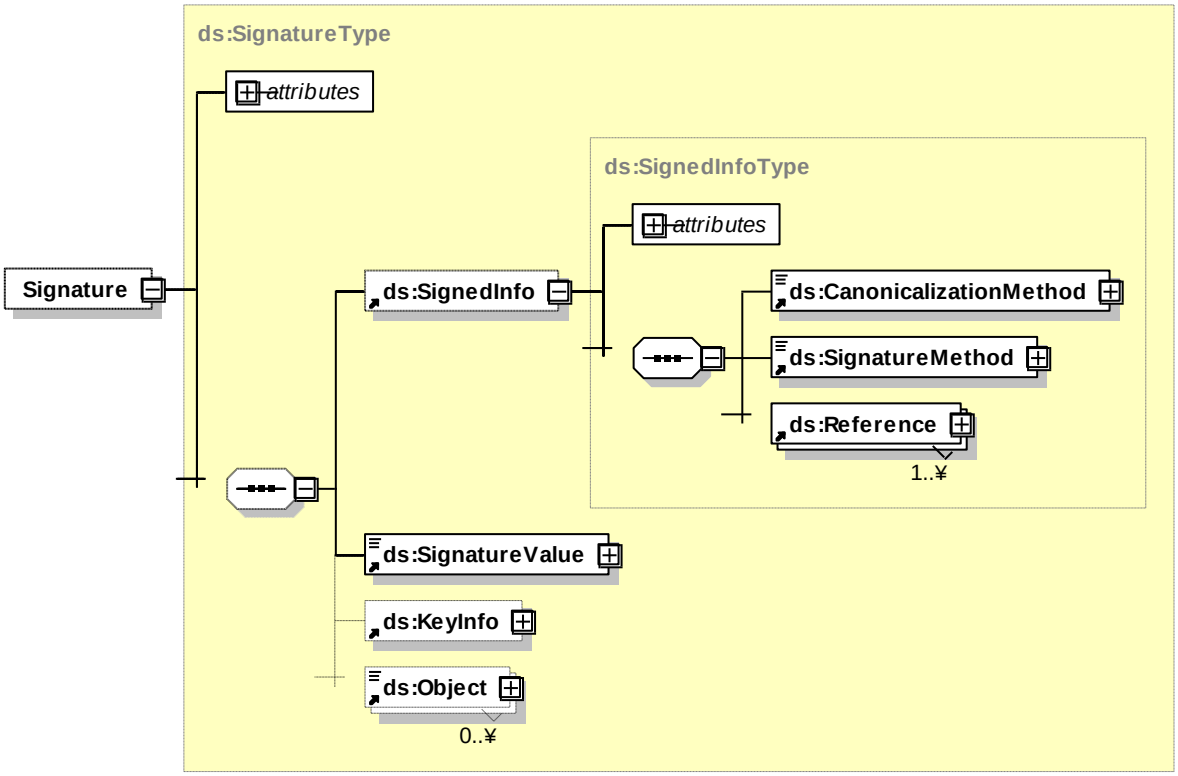


Figure 12 — Schema Diagram – XML Signature

Figure 13 shows the schema diagram of the element Reference included included in the element SignedInfo which is part of the XML Signature depicted in Figure 12.

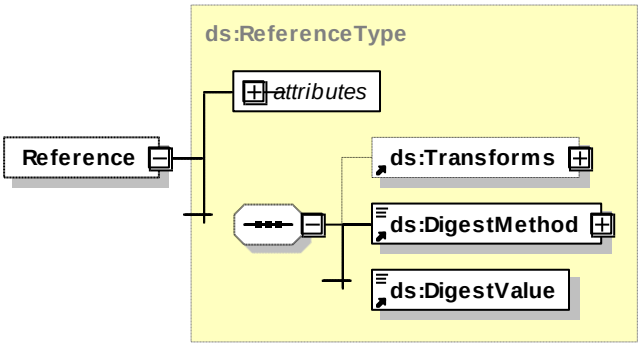


Figure 13 — Schema Diagram – Element Reference included in element SignedInfo

NOTE 1 There are two alternatives:
Alternative 1: Placement in the header of the XML document allows for the provisioning of the signature field in basically every XML message to be exchanged. This requires the specification of the necessary messages and values to be protected, which can be done in a next step. It allows also to add the signature to further messages at a later stage in standardization. This approach may require a profiling to ensure that the right messages are equipped with the digital signatures as needed (used in ISO/IEC 15118 XML signature process).
Alternative 2: Placement in the body of the XML document, providing a more strict definition of dedicated XML messages. This leaves less degrees of freedom but may better ensure interoperability because of less options available (not applicable for ISO/IEC 15118).

NOTE 2 The meter values from the SECC may already been signed. This signature value is treated as informational element and is required for the gauging process. It may optionally be checked by the vehicle. The signature may comprise also the meter ID.

[V2G2-117] Each V2G entity shall support enveloping XML signatures.

[V2G2-119] For XML signature operations, the data which gets signed, shall be the EXI representation of this data.

These messages are assumed to be signed by the Secondary Actor, which can be verified using either V2G root certificate.

7.9.2.4.3 XML Encryption mechanism

XML encryption allows the transport of information in a confidentiality protected way. Base for the XML encryption is the contractID provided by the EVCC during the Service Discovery & Selection phase. For this purpose, we use ContractCertificates. We use the Elliptic Curve Diffie Hellman key exchange protocol as described in Annex I. A message to initiate the key exchange is signed by the EVCC using Contract Certificates.

[V2G2-121] The EVCC shall support calculation of the ECDH secret and the key derivation function for the decryption of encrypted information like private keys.

[V2G2-122] Each V2G entity shall have mechanisms to process ECDH Key exchange. Public parameters are derived from the public ECDSA parameters.

7.9.2.4.4 Application of XML security mechanisms to XML message

In general, two pairs of XML based Security mechanisms are supported:

— Authenticity and Integrity: Signature generation → Signature verification;
XML based signature is applied. The entity, which creates the XML message, signs certain or all fields of an XML message. The receiver verifies the signature.

— Confidentiality: Encryption → Decryption;
XML encryption is applied. The entity, which creates the message, encrypts certain or all fields of an XML message. The receiver decrypts.

The following tables provide an overview of the applied XML security mechanisms.

Table 13 — Overview of applied XML based signatures

XML Message	protected fields	signing entity (sender)	verifying entity (receiver)
ContractAuthenticationReq	message body / all fields	EVCC	SECC
CertificateUpdateReq	message body / all fields	EVCC	Secondary Actor
CertificateUpdateRes	message body / all fields	Secondary Actor	EVCC
CertificateInstallationReq	message body / all fields	EVCC	Secondary Actor
CertificateInstallationRes	message body / all fields	Secondary Actor	EVCC
MeteringReceiptReq	message body / all fields (optionally)	EVCC	SECC
ChargeParameterDiscoveryRes	SalesTariff (optionally)	Secondary Actor	EVCC

Table 14 — Overview of applied XML based encrytion

XML Message	protected fields	encrypting entity (sender)	decrypting entity (receiver)
CertificateUpdateRes	ContractSignatureEncryptedPrivateKey	Secondary Actor	EVCC
CertificateInstallationRes	ContractSignatureEncryptedPrivateKey	Secondary Actor	EVCC

[V2G2-652] Each V2G entity shall be able to generate XML signatures as specified in Table 13.

[V2G2-653] Each V2G entity shall be able to verify XML signatures as specified in Table 13.

[V2G2-654] Each V2G entity shall be able to apply XML encryption as specified in Table 14.

[V2G2-655] Each V2G entity shall be able to decrypt XML encryption as specified in Table 14.

7.10 Application Layer

7.10.1 SECC Discovery Protocol

7.10.1.1 General Information

An EVCC uses the SECC Discover Protocol (SDP) to get the IP address and port number of the SECC. The SDP client sends out SECC Discovery Request messages to the local link (multicast) expecting any SDP server to answer its request with a SECC Discovery Response message containing this information.

After the EVCC received the IP address and the port number of the SECC, it can establish a TLS connection to the SECC (refer to subclause 7.4).

[V2G2-123] An SDP server shall be accessible in the local link.

NOTE As common for internet technologies, SDP server may be implemented on the same physical device as the SECC and may also interface to the same IP address.

7.10.1.2 Supported ports

SDP is a UDP based protocol. The ports listed in Table 15 are used by SDP.

Table 15 — Supported UDP ports for SDP

Name	Protocol	Port number	Description
V2G_UDP_SDP_CLIENT	UDP (unicast)	Port number in the range of Dynamic Ports (49152-65535) as defined in IETF RFC 6335.	SDP client source port at the EVCC
V2G_UDP_SDP_SERVER	UDP (multicast)	15118	SDP server port which accepts UDP packets with a local-link IP multicast destination address.

[V2G2-124] An SDP client shall support the port V2G_UDP_SDP_CLIENT as defined in Table 15 for sending and receiving SDP messages.

[V2G2-125] An SDP server shall support the port V2G_UDP_SDP_SERVER as defined in Table 15 for receiving and sending SDP messages.

NOTE Depending on the implementation of the EVCC the dynamically assigned V2G_UDP_SDP_CLIENT port will be assigned once during or before the first transmission of a UDP packet to a SECC or can be dynamically re-assigned for each individual UDP request message and response. Also depending on whether messages are repeatedly sent, response messages may arrive asynchronously and may not be associated to the exact corresponding request anymore.

[V2G2-126] The SDP client shall be able to handle asynchronously arriving SECC Discovery Response messages.

7.10.1.3 Protocol Data Unit

7.10.1.3.1 Structure

An SDP message is based on the V2GTP message format as defined in subclause 7.8.3.1.

[V2G2-127] An SDP client shall support the definitions in subclause 7.8.3.1.

[V2G2-128] An SDP client shall use a separate UDP packet for each request message.

[V2G2-129] An SDP client shall locate the first byte of the request message header as defined in Figure 9 and Table 9 in the first byte of the UDP packet payload.

[V2G2-130] An SDP server shall support the definitions in subclause 7.8.3.1.

[V2G2-131] An SDP Server shall use a separate UDP packet for each response.

[V2G2-132] An SDP server shall locate the first byte of the response message header as defined in Figure 9 and Table 9 in the first byte of the UDP packet payload.

7.10.1.3.2 Header Processing

An SDP header processing is based on the V2GTP message header processing as defined in subclause 7.8.3.2.

[V2G2-133] An SDP client shall apply to the header processing as defined in subclause 7.8.3.2.

[V2G2-134] An SDP server shall apply to the header processing as defined in subclause 7.8.3.2.

7.10.1.4 SECC Discovery Request Message

The SDP client uses the SECC Discovery Request message to request the IP address and the port number of the SECC.

[V2G2-135] Only SDP client shall send SECC Discovery Request messages.

[V2G2-136] An SDP client shall send SECC Discovery Request messages with the source IP address on which it expects the SECC Discovery Response message.

[V2G2-137] An SDP client shall send SDP request messages to destination port V2G_UDP_SDP_SERVER as defined in Table 15.

[V2G2-138] An SDP client shall send SDP request messages with source port V2G_UDP_SDP_CLIENT as defined in Table 15 on which it expects the SECC Discovery Response message.

[V2G2-139] An SDP client shall send SECC Discovery Request message to the destination local-link multicast address (FF02::1) as defined in IETF RFC 4291.

[V2G2-140] The SDP client shall send the SECC Discovery Request message with payload type value 0x9000 as defined in Table 10.

[V2G2-141] The SDP client shall send the SECC Discovery Request message with the payload length 2.

[V2G2-142] The SDP client shall send the SECC Discovery Request message with the payload as defined in Figure 14.

[V2G2-622] An SDP client shall send the payload in the order as shown in Figure 14. A byte with a lower number shall be sent before a byte with a higher number. The payload starts with byte 1 and ends with byte 2.

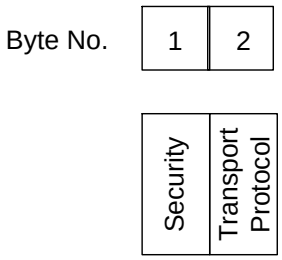


Figure 14 — SECC Discovery request message payload

[V2G2-623] An SDP client shall use the encoding for the requested security option and the requested transport protocol as defined in Table 16.

Table 16 — SDP security and protocol option encoding

	Security	Transport protocol
Byte no. SDP request message	1	2
Byte no. SDP response message	19	20
Applicable values	0x00 = secured with TLS 0x01-0x0F = reserved 0x10 = No transport layer security 0x11-0xFF = reserved	0x00= TCP 0x01-0x0F = reserved 0x10 = reserved for UDP 0x11-0xFF = reserved

[V2G2-624] An SDP server shall use the encoding for the requested security option and the requested transport protocol as defined in Table 16 to define the supported transmission security and transport protocol for the port provided in the same payload as the security and transport protocol bytes.

7.10.1.5 SECC Discovery Response Message

The SDP server uses the SECC Response message to response to an SECC Discovery Request message and provide the IP-address and the port of the SECC to the client.

[V2G2-143] The SDP server shall be able to extract the source IP address and source port of a received UDP packet (client IP address and port number) and send a UDP packet to the identified IP address and port number.

[V2G2-144] An SDP server shall reply to any SECC Discovery Request messages with an SECC Discovery Response Message.

NOTE This requirement ensures that an SDP server serving multiple clients can be reached at any time. This supports charging of multiple EVs at an EVSE with a single SECC.

[V2G2-145] An SDP client shall not reply to any SECC Discovery Request message.

[V2G2-146] An SDP server shall only send response messages after an SECC Discovery Request message has been received.

[V2G2-147] An SDP server shall send an SECC Discovery Response messages as fast as possible after an SECC Discovery Request message has been received.

[V2G2-148] If an SDP server has multiple IP addresses, the SDP server shall send an SECC Discovery Response message with the source IP address on which the SDP server received the SECC Request message.

[V2G2-149] An SDP server shall send an SDP response with source port V2G_UDP_SDP_SERVER as defined in Table 15.

[V2G2-150] An SDP server shall send an SECC Discovery Response message to the SDP client which sent the SECC Discovery Request message.

[V2G2-151] An SDP server shall send an SECC Discovery Response message to the port of the SDP client which sent the SECC Discovery Request message.

[V2G2-152] An SDP server shall send the SECC Discovery Response message with the payload type value 0x9001 as defined in Table 10.

[V2G2-153] An SDP server shall send the SECC Discovery Response message with payload length 18.

[V2G2-154] An SDP server shall send the SECC Discovery Response message with the payload as defined in Figure 15.

[V2G2-155] An SDP server shall send the payload in the order as shown in Figure 15. A byte with a lower number shall be sent before a byte with a higher number. The payload starts with byte 1 and ends with byte 18.

[V2G2-156] An SDP server shall send the fields “SECC IP Address” and “SECC Port” in big endian format: The most significant byte is sent first the least significant byte is sent last.

NOTE 1 The mechanism used by the SDP server to determine its own IP address is out of the scope of this standard.

NOTE 2 The source IP address and the source port of a received UDP packet is usually provided by the TCP/IP stack.

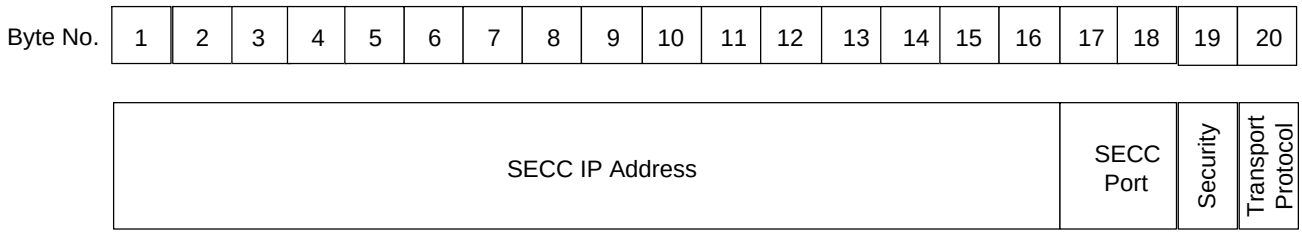


Figure 15 — SECC Discovery response message payload

7.10.1.6 Timing and Error Handling

The process of SECC discovery is based on the application time out definitions as described in 7.4. This subclause describes additional timing and error handling for the SECC Discovery Protocol.

[V2G2-157] The SDP client shall count the number of SECC Discovery Request messages until a valid SECC Discovery Response message has been received.

[V2G2-158] The SDP client shall reset the counter for sent SECC Discovery Request messages after a valid SECC Discovery Response message has been received.

[V2G2-159] After sending an SECC Discovery Request message, the SDP client shall wait for an SECC Discovery Response message messages for at least 250ms.

[V2G2-160] After unsuccessfully waiting for an SECC Discovery Response message the SDP client shall send a new SECC Discovery Request message and increment the counter for sent SECC Discovery Response messages.

[V2G2-161] If the SDP client has not received any SECC Discovery Response message after sending in maximum 5 consecutive SECC Discovery Request messages it shall stop the SECC Discovery.

[V2G2-162] After stopping the SECC Discovery, the SDP client should go to the same state as defined for an application (refer to [V2G2-020] and Figure 6).

7.10.1.7 Protocol and Security Options Handling

For SDP request and response handling, the EVCC and SECC shall implement the following requirements:

[V2G2-625] If the EVCC sends an SDP request with a protocol option code for TCP (as defined in Table 16), the SECC shall send an SDP response with protocol option code TCP.

[V2G2-626] If the EVCC sends an SDP request with a protocol option code for TLS (as defined in Table 16), and the SECC supports TLS, the SECC shall send an SDP response with protocol option code TLS.

[V2G2-627] If the EVCC sends an SDP request with a protocol option code for TLS (as defined in Table 16), and the SECC does not support TLS, the SECC shall send an SDP response with protocol option code TCP.

For TCP/TLS establishment, the following requirements apply:

[V2G2-628] Depending on the use case and security requirements of the EVCC, the EVCC shall either use the protocol as indicated in the SECC response or stop communication establishment.

[V2G2-629] If the EVCC tries to communicate with a transport protocol as defined in Table 16 that differs from the protocol options response in the SDP response from the SECC, the SECC shall not accept this communication.

7.10.1.8 Support and Application of TLS

[V2G2-630] Support of TLS is mandatory for the EVCC for all identification modes except for identification mode "EIM" with Message Set EIM charging AC and Message Set EIM charging DC, in both cases excluding Message Set VAS as defined in subclause 8.6.

[V2G2-631] Support of TLS is mandatory for the SECC for all identification modes except for identification mode "EIM" in a trusted environment with Message Set EIM charging AC excluding Message Set VAS as defined in subclause 8.6.

Based on the transport layer setup of either TCP or TLS, the following restrictions apply:

[V2G2-632] If a V2G Communication Session without TLS is used, the SECC shall only provide the EIM identification mode and the Message Sets EIM charging AC or EIM charging DC by indicating "ExternalPayment" in the parameter PaymentOptions of the ServiceDiscoveryRes message as defined in subclause 8.4.1.3.3.

[V2G2-633] If a V2G Communication Session without TLS is used, the EVCC shall only accept the EIM identification mode with message sets EIM charging AC or EIM charging DC, no matter if the SECC offers also other identification modes and message sets.

[V2G2-634] If a V2G Communication Session without TLS is used, the SECC shall not provide the PnC Message Sets.

[V2G2-635] If a V2G Communication Session without TLS is used, the EVCC shall not apply the PnC Message Sets.

[V2G2-636] If a V2G Communication Session without TLS is used, the SECC shall not provide the Message Set VAS.

[V2G2-637] If a V2G Communication Session without TLS is used, the EVCC shall not apply the Message Set VAS.

[V2G2-638] Security measures for value added services enabled by the Message Set VAS (offered in ServiceDiscoveryRes) are out of scope of this specification.

[V2G2-639] If TLS is not applied both sides have to ensure that a identification mode change from EIM to another identification mode and a message set change from EIM AC or EIM DC to another message set in the current charging session is not possible (avoiding security downgrade of the changed to session).

[V2G2-640] If TLS is applied both sides shall ensure that switching off TLS shall result in ending the charging session (avoiding security downgrade of the actual session).

The support, usage and restrictions regarding TLS are illustrated in Table 17 and Table 18.

Table 17 — TLS implementation / support for EIM identification modes

	Other Environment (not trusted)				**Trusted Environment**			
Allowed message sets [a]	**DC EIM**	**AC EIM**	**DC EIM, VAS**	**AC EIM, VAS**	**DC EIM**	**AC EIM**	**DC EIM, VAS**	**AC EIM, VAS**
EVCC TLS support	optional	optional	mandatory	mandatory	optional	optional	mandatory	mandatory
SECC TLS support	mandatory	mandatory	mandatory	mandatory	mandatory	optional	mandatory	mandatory
TLS usage negotiation using SDP [b]	EV decides, EVSE shall accept EVs decision	EV decides, EVSE shall accept EVs decision	EV shall use TLS, EVSE shall reject, if EV uses no TLS	EV shall use TLS, EVSE shall reject, if EV uses no TLS	EV decides, EVSE shall accept EVs decision	refer to Table 18	EV shall use TLS, EVSE shall reject, if EV uses no TLS	EV shall use TLS, EVSE shall reject, if EV uses no TLS

[a] This refers to messages as defined in this standard. In case of VAS, of course any other connections (which are not part of this standard) are possible

[b] Rejecting means to stop communication

NOTE 1 Not supporting TLS in the SECC might lead in general to aborted charging sessions with particular EVs as it is in the responsibility of the EV to accept sessions without TLS.

Table 18 — SDP handshake with AC EIM identification mode in trusted enviroment

EVCC TLS support	SECC TLS support	SDP request by EVCC	SDP response SECC
EVCC has TLS support	SECC has TLS support	EVCCsignals TLS	SECC shall respond TLS
		EVCC signals TCP	SECC shall respond TCP
	SECC has no TLS support	EVCC signals TLS	SECC shall reject TLS and propose TCP instead via SDP. The EV may now decide, if it wants to establish TCP, or otherwise abort communication.
		EVCC signals TCP	SECC shall respond TCP
EVCC has no TLS support	SECC has TLS support	EVCC signals TCP	SECC shall respond TCP
	SECC has no TLS support	EVCC signals TCP	SECC shall respond TCP

NOTE 2 For the message set "EIM AC" and "EIM DC" it is in the responsibility of the EV to decide on not using TLS and accepting the risk of an unsecured connection.

NOTE 3 If TLS is applied, it is always used with unilateral (EVSE-side) authentication. If TLS is not used, there is no EVSE side authentication towards the EV as well as further protection of the communication channel on transport layer.

Any functional safety related risks occurring through overvoltage and overcurrent are to be handled according requirements listed below:

[V2G2-641] The EV shall support the safety measures described in ISO 17409 to protect against overvoltage and over current when using the AC EIM identification mode.

[V2G2-642] The EV shall support the safety measures described in ISO 17409 to protect against overvoltage and over current when using the DC EIM identification mode.

[V2G2-643] The EVSE shall support the safety measures described in IEC 61851-1 and IEC 61851-22 to protect against overvoltage and over current when using the AC EIM identification mode

[V2G2-644] The EVSE shall support the safety measures described in IEC61851-1 and IEC 61851-23 to protect against overvoltage and over current when using the DC EIM identification mode.

7.10.1.9 SECC Discovery service primitives

The N-SECC_Address.indication notifys about the status of the SECC IP address discovery. Table 19 describes the service primitive and its parameter(s).

Table 19 — N-SECC_Address.indication service primitive

Primitive name	N-SECC_Address.indication
Entity to support	EVCC
Parameter Name	**Description**
N_SECC_STATUS	- SECC IP -address discovered - Error

N-SECC_Address.indication (N_SECC_STATUS = SECC IP-address discovered) indicates that SDP returned a local or globael IP-address for the SECC.

SECC_Address.indication (N_SECC_STATUS = Error) indicates any error during SDP as defined in subclause 7.10.1.

[V2G2-163] If SDP returns a global SECC IP-address, the EVCC shall not indicate the discovered SECC IP-address before the EVCC has configured a global IP address as defined in subclause 7.6.3.2 and 7.6.3.3.

[V2G2-164] If the SDP returns a global SECC IP-address, the EVCC shall indicate the discovered SECC IP-address after a global address assignment as defined in 7.6.3.2 and 7.6.3.3 is indicated.

7.10.2 Vehicle to Grid application layer messages

The Vehicle to Grid application layer message definitions describe the client-server based message exchange between EVCC and SECCs for the purpose of initializing and configuring the charge process of an EV. The message set is designed to cover the use cases defined in Part 1 of this standard. The messages and the required message flow (i.e. communication protocol) represent the application layer according to the OSI layered architecture model.

The message set, message flow and the behaviour specific to a certain message are described in clause 8 Application Layer messages.

7.10.3 Application layer service primitives

7.10.3.1 A-Data.confirmation

The A-Data.confirmation notifys about the status V2G messaging. Table 20 describes the service primitive and its parameter(s).

Table 20 — A-Data.confirmation service primitive

Primitive name	A-Data.confirmation
Entity to support	EVCC
Parameter Name	**Description**
A_Msg	- Session Setup - Service Discovery - Service Detail - Service and Payment Selection - Payment Details - Contract Authentication - Charge Parameter Discovery - Power Delivery - Charging Status - Metering Receipt - Certificate update - Certificate installation - Cable Check - Pre Charging - Current Demand - Welding Detection - Session Stop

A-Data.confirmation (A_Msg= “message name”) indicates the successful receiption of a response message for the V2G message that is given by A_Msg.

EXAMPLE A-Data.confirmation (A_Msg= Session Setup) indicates the successful receiption of a SessionSetupRes Message as defined in subclause 8.4.1.2.3.

7.10.3.2 A-Data.response

The A-Data.response notifys about the status V2G messaging. Table 21 describes the service primitive and its parameter(s).

Table 21 — A-Data.response service primitive

Primitive name	A-Data.response
Entity to support	SECC
Parameter Name	**Description**
A_Msg	- Session Setup - Service Discovery - Service Detail - Service and Payment Selection - Payment Details - Contract Authentication - Charge Parameter Discovery - Power Delivery - Charging Status - Metering Receipt - Certificate update - Certificate installation - Cable Check - Pre Charging - Current Demand - Welding Detection - Session Stop

A-Data.response (A_Msg= “message name”) indicates to the lower layer to send out a V2G response message for the V2G message type that is given by A_Msg.

Example: A-Data.response (A_Msg= Session Setup) initiates the sending of the SesstionSetupRes message as defined in subclause 8.4.1.2.3.

8 Application Layer messages

8.1 General information and definitions

A V2G message uses the EXI-based Presentation Layer as described in 7.9.1. The communication between EVCC and SECC at application layer level is based on a client/server architecture. The EVCC always acts as a client (service requester) during the entire charging process, whereas the SECC always acts as a server (service responder). Hence the EVCC always initates communication by sending a request message to the SECC which then returns the corresponding response message. All messages exchanged between EVCC and SECC are described with their syntax and their semantics in subclauses 8.2, 8.3, 8.4. and 8.5. The entire XML Schema definition describing both V2G message set is included in Annex C.

Subclause 8.6 defines the V2G message and respective message elements required to be supported for a certain set of use case elements described in Part 1 of this standard.

Subclause 8.7 define message timing and error handling for the V2G communication message exchange.

Examples for typical message sequences are shown in subclause 8.8.

V2G communication consists of two different message sets:

— V2G application layer protocol handshake messages (refer to 8.2)

— V2G application layer messages (refer to 8.3)

8.2 Protocol handshake definition

8.2.1 Handshake sequence

[V2G2-165] Before starting the application layer message exchange, an appropriate application layer protocol including its version shall be negotiated between the EVCC and the SECC.

In order to negotiate the protocol between the EVCC and the SECC the following application layer protocol handshake is performed.

[V2G2-166] The EVCC shall initiate the handshake sending a supportedAppProtocolReq message as depicted in Figure 16 to the SECC. This request message provides a list of charging protocols supported by the EVCC.

[V2G2-167] Each entry in the list of supported EVCC protocols shall include the ProtocolNamespace, the VersionNumberMajor and VersionNumberMinor, the SchemaID dynamically assigned by the EVCC and the Priority of the protocol entry. The Priority in the EVCC request message enables the EVCC to announce the preferred application layer protocol where Pritory equal to 1 indicates the highest priority and Priority equal to 20 indicates the lowest priority. The number of protocols included in the request message is limited to 20.

[V2G2-168] The SECC shall respond with the supportedAppProtocolRes message as depicted in Figure 17 indicating the protocol to be used for the subsequent message exchange by both, the EVCC and the SECC.

[V2G2-169] The response message shall include a ResponseCode and the SchemaId of the protocol/schema which is agreed as application protocol for the following communication session. Thereby, the SECC shall select from its own list of supported protocols the protocol with highest Priority indicated by the EVCC.

[V2G2-170] The SECC shall confirm (positively respond) an EVCC supported protocol even if the values of the VersionNumberMinor in EVCC request message does not match with the VersionNumberMinor of an SECC supported protocol where the VersionNumberMajor matches.

NOTE A higher value in the VersionNumberMinor inidicates that (in comparison to a lower value) additional data elements will be transmitted from either the EVCC or SECC. Implementations only supporting the lower VersionNumberMinor value may not be able to process the data and may have to ignore this data, hower a difference in the VersionNumberMinor value between EVCC and SECC does not lead to an incompatibility. Refer to subclause 8.2.4 showing examples for successful protocol negotiation.

[V2G2-171] All additional data element defined by the respective minor version shall be encoded as schema deviated case by the EXI coder (see also EXI option settings in subclause 7.9.1.3).

[V2G2-172] Usually it is expected that the SECC is able to support the relevant application layer protocols indicated by the EVCC. However when none of the application layer protocols included the list received from the EVCC is supported by the SECC, the ResponseCode in the response message shall be equal to Failed_NoNegotiation indicating that the protocol negotiation was not successful. In this error scenario the response message shall not include a SchemaID.

[V2G2-173] If no successful protocol negotiation can be achieved the EVCC shall not initialize a communication session.

[V2G2-174] This protocol handshake between EVCC and SECC shall be peformed prior the actual V2G application layer message exchange. Only the message set defined in the agreed protocol shall be used in the V2G message flow except for minor deviations.

8.2.2 Message definition supportedAppProtocolReq and supportedAppProtocolRes

[V2G2-175] The SECC and EVCC shall implement the message and message elements as in defined Figure 16.

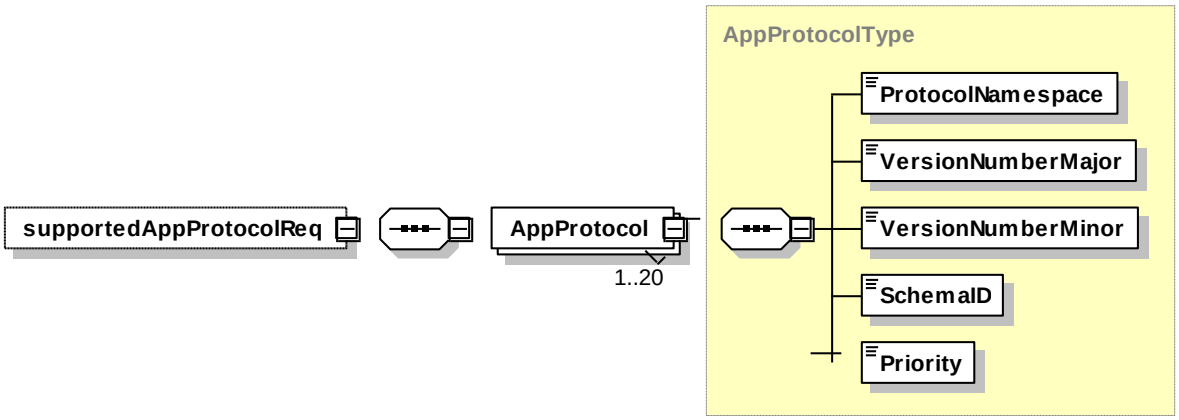


Figure 16 — Schema Diagram – supportedAppProtocolReq

[V2G2-176] The SECC and EVCC shall implement the message and message elements as defined in Figure 17.

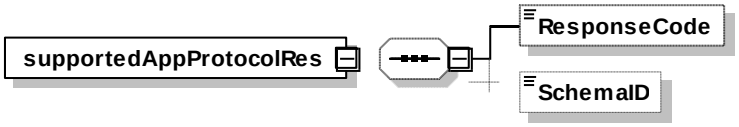


Figure 17 — Schema Diagram – supportedAppProtocolRes

NOTE Refer to Annex C.2 for the XML schema code.

8.2.3 Semantics decription supportedAppProtocol messages

[V2G2-178] The message elements of the messages defined in Figure 16 and Figure 17 shall be used as defined in Table 22.

Table 22 — Semantics and type definition for supportedAppProtocol message elements

Element/Attribute Name	Type	Semantics
AppProtocol	complexType: includes the message elements defined in this table	This message element is used by the EVCC for transmitting the list of supported protcols. Each protocol with a particular version supported by the EVCC is represented by one AppProtocol entry in the request message (maximum number of entries: 20)
ProtocolNamespace	simpleType: protocolNamespaceType string (max length: 100) refer to Annex C.2 for the type definition	This message element is used by the EVCC to uniquely identify the Namespace URI of a specific protocol supported by the EVCC, i.e. this is the protocol name of the related protocol.
VersionNumberMajor	simpleType unsignedInt refer to Annex C.2 for the type definition	This message element is used by the EVCC to indicate the major version number of the protocol indicated in the message element ProtocolNamespace.
VersionNumberMinor	simpleType unsignedInt refer to Annex C.2 for the type definition	This message element is used by the EVCC to indicate the minor version number of the protocol indicated in the message element ProtocolNamespace.
SchemaID	simpleType: unsignedByte refer to Annex C.2 for the type definition	This message element is used by the EVCC to indicate the schemaID assigned by the EVCC to the protocol indicated in the message element ProtocolNamespace, VersionNumberMajor and VersionNumberMinor. This message element is used by the SECC to reference one of the EVCC supported protocols received in the request message. This identifier allows also for referring to a particular protocol later on in the communication process (EXI Option schemaID).
Priority	simpleType: priorityType unsignedByte (range 1..20) refer to Annex C.2 for the type definition	This message element is used by the EVCC for indicating the protocol priority of a specific protocol allowing the SECC to select a protocol based on priorities.
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.2 for the type definition	This message element is used by the SECC for indicating wether the list of protocols received from the EVCC includes at least one protocol matching with the protocols supported by the SECC.

8.2.4 Message Examples

8.2.4.1 Protocol prioritization

V2G message example 1 and V2G message example 2 illustrate the exchange of suppAppProtocol messages between the EVCC and the SECC. In the request message, the EVCC sends a prioritized list of supported application layer protocols (15118:2:2010 with version 2.0, 15118:2:2010 with version 1.0) to the SECC. In the response message the SECC confirms protocol 15118:2:2010 with version 2.0 using a ResponseCode equal to 'OK_SuccessfulNegotiation' and a schemaID equal to ten (10).

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:supportedAppProtocolReq xmlns:ns0="urn:iso:15118:2:2010:AppProtocol"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <AppProtocol>
      <ProtocolNamespace>urn:iso:15118:2:2010:MsgDef</ProtocolNamespace>
      <VersionNumberMajor>2</VersionNumberMajor>
```

```
      <VersionNumberMinor>0</VersionNumberMinor>
      <SchemaID>10</SchemaID>
      <Priority>1</Priority>
   </AppProtocol>
   <AppProtocol>
      <ProtocolNamespace>urn:iso:15118:2:2010:MsgDef</ProtocolNamespace>
      <VersionNumberMajor>1</VersionNumberMajor>
      <VersionNumberMinor>0</VersionNumberMinor>
      <SchemaID>20</SchemaID>
      <Priority>5</Priority>
   </AppProtocol>
</ns0:supportedAppProtocolReq>
```

V2G message example 1 – supportedAppProtocolReq: protocol prioritization

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:supportedAppProtocolRes xmlns:ns0="urn:iso:15118:2:2010:AppProtocol"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <ResponseCode>OK_SuccessfulNegotiation</ResponseCode>
   <SchemaID>10</SchemaID>
</ns0:supportedAppProtocolRes>
```

V2G message example 2 – supportedAppProtocolRes: protocol prioritization

8.2.4.2 Minor Deviation

V2G message example 3 and V2G message example 4 illustrate the exchange of suppAppProtocol messages between the EVCC and the SECC. In the request message, the EVCC sends just one supported application layer protocol (15118:2:2014 with version 1.0) to the SECC. The SECC support protocol version 1.1 only. In the response message the SECC confirms protocol 15118:2:2010 with VersionNumberMajor equal to one (1) using a schemaID equal to one (1). However, the ResponseCode is equal to OK_SuccessfulNegotiationWithMinorDeviation signalling that a minor version deviation applies. The EVCC may now expect message elements which aren't know to the EVCC but can be ignored.

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:supportedAppProtocolReq xmlns:ns0="urn:iso:15118:2:2010:AppProtocol"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <AppProtocol>
      <ProtocolNamespace>urn:iso:15118:2:2010:MsgDef</ProtocolNamespace>
      <VersionNumberMajor>1</VersionNumberMajor>
      <VersionNumberMinor>0</VersionNumberMinor>
      <SchemaID>1</SchemaID>
      <Priority>1</Priority>
   </AppProtocol>
</ns0:supportedAppProtocolReq>
```

V2G message example 3 – supportedAppProtocolReq: deviation in minor version

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:supportedAppProtocolRes xmlns:ns0="urn:iso:15118:2:2010:AppProtocol"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <ResponseCode>OK_SuccessfulNegotiationWithMinorDeviation</ResponseCode>
   <SchemaID>1</SchemaID>
</ns0:supportedAppProtocolRes>
```

V2G message example 4 – supportedAppProtocolRes: deviation in minor version

8.3 V2G Message Definition

8.3.1 Overview

Sub-clause 8.3 describes the messages of the V2G Messages and their contents. It is structured into the following 3 sub-clauses:

— V2G Message definition (refer to subclause 8.3.2)

— V2G Message header definition (refer to subclause 8.3.3)

— V2G Message body definition (refer to subclause 8.3.4)

NOTE Refer to Annex C for the XML schema code.

The ISO/IEC 15118 application layer message set is signalized by the XML schema namespace "urn:iso:15118:2:2010:MsgDef". Refer to the XML schema definition in Annex C for details relative to subnamespace definitions used for the message definition.

8.3.2 Message definition

Figure 18 shows the schema definition of the V2G application layer message.

[V2G2-179] The EVCC and the SECC shall implement the V2G message structure as defined in Figure 18.

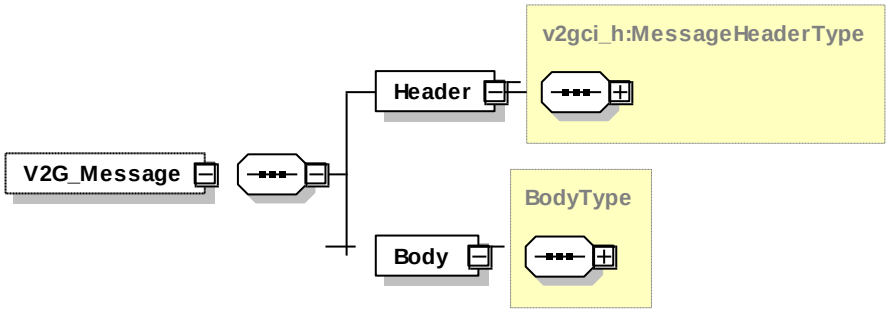


Figure 18 — Schema Diagram – V2G message

[V2G2-180] The message elements of this message shall be used as defined in Table 23.

Table 23 — Semantics and type definition for a V2G message

Element Name	Type	Semantics
V2G_Message	complexType includes the message elements defined in this table	Root element that identifies this XML document as a V2G message. It contains two child elements, a Header and Body element.
Header	complexType: MessageHeaderType refer to subclause 8.3.3	This element contains the content of the message header. It includes generic information for protocol flow and is not directly related to the semantics of each particular message defined in 8.4.
Body	complexType: BodyType refer to subclause 8.3.4	This element contains the content of the message body. The message body provides the actual semantics of each message defined in section in 8.4.

V2G message example 5 shows an instance of a SessionSetupReq message. The header contains a SessionID equal to zero (0) because a new communication session is about to being stated. The body contains the message specific content. In this case the message contains the message element EVCCID.

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_b="urn:iso:15118:2:2010:MsgBody"
 xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
 xmlns:v2gci_d="urn:iso:15118:2:2010:MsgDef"
 xmlns:v2gci_t="urn:iso:15118:2:2010:MsgDataTypes"
```

```
xmlns:v2gci_h="urn:iso:15118:2:2010:MsgHeader">
  <v2gci_d:Header>
    <v2gci_h:SessionInformation>
      <v2gci_t:SessionID>0000000000000000</v2gci_t:SessionID>
    </v2gci_h:SessionInformation>
  </v2gci_d:Header>
  <v2gci_d:Body>
    <v2gci_b:SessionSetupReq>
      <v2gci_b:EVCCID>000000000000000F</v2gci_b:EVCCID>
    </v2gci_b:SessionSetupReq>
  </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 5 - Example for a SessionSetupReq message

8.3.3 Message Header Definition

The message header contains general information that is included in all messages. Figure 19 shows the schema definition of the V2G message header.

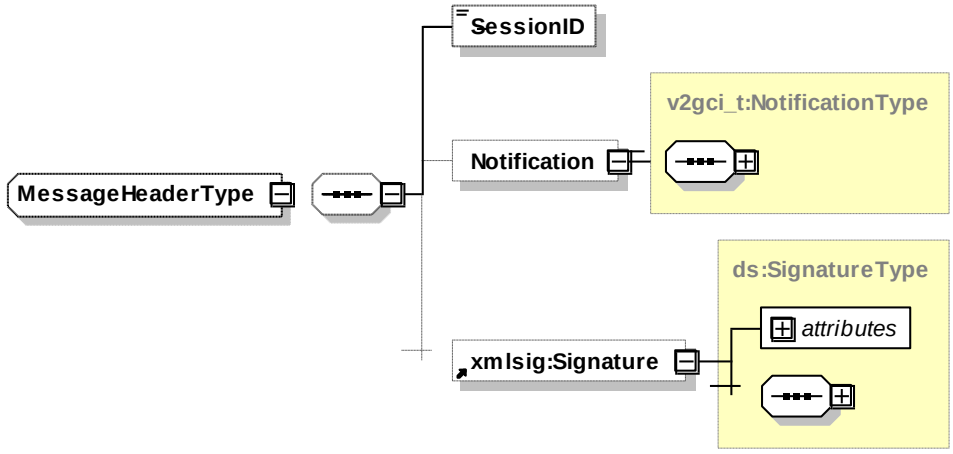


Figure 19 — Schema Diagram – Message header

[V2G2-181] The message elements of the message header shall be used as defined in Table 24.

Table 24 — Semantics and type definition for a V2G message header

Element Name	Type	Semantics
SessionID	simpleType: SessionIDType: hexBinary (max length: 8)	This message element is used by EVCC and SECC for uniquely identifying a V2G communication session. Refer to 8.4.1.2.1 for requirements relative to this message element.
Notification	complexType: NotificationType, see subclause 8.5.2.8	Optional Element: This element is used by the SECC for transmitting additional fault information when an error occurred on the SECC side.
xml:Signature	separate namespace: "http://www.w3.org/2000/09/xmldsig#"	Optional Element: This element is used if a certain V2G message requires to be singed.

[V2G2-182] Each V2G message containing signed elements shall include the xml:Signature element in the header to be able to transmit the signature attached to signed body message elements of the respective message.

8.3.4 Message Body Definition

The message body contains information details related to a specific message. Figure 20 shows the schema definition of the V2G message body. The messages described in the following section are derived from BodyBaseType, which represents the abstract message content (refer to subclause 8.3.2). The different application messages are defined by the BodyElement and described in detail in subclause 8.4.

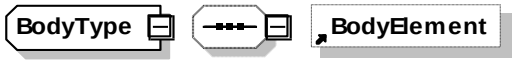


Figure 20 — Schema Diagram – Message body

[V2G2-183] The BodyElement shall be used as defined in Table 25.

Table 25 — Semantics and type definition for a V2G message body

Element Name	Type	Semantics
BodyElement	complexType: BodyBaseType, see subclause 8.4	BodyElement is a head element of a substitution group and does not appear itself in an instance of a message. Instead one of the body elements defined in the substitution group in section 8.4 is instantiated.

8.4 BodyElement Definitions

8.4.1 Common Messages

8.4.1.1 Overview

Messages defined as common messages can be applied to the message sequence in any charging mode defined in ISO/IEC 15118.

8.4.1.2 Session Setup

8.4.1.2.1 General

After a connection has been established between the EVCC and the SECC the V2G communication session is established on application layer (refer to subclause 7.4).

[V2G2-184] A charging session shall be identified by a SessionID.

[V2G2-185] The SessionID shall not change during a V2G communication session.

[V2G2-186] The EVCC shall set SessionID equal to zero (0) when initiating a new V2G communication session.

[V2G2-187] If the EVCC resumes a V2G communication session, the SessionID in SessionSetup Req header shall contain the SessionID value used during the preceding V2G communication session.

8.4.1.2.2 Session Setup Request

By using the SessionSetupReq message the EVCC establishes a V2G communication session.

[V2G2-188] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according to Figure 21.

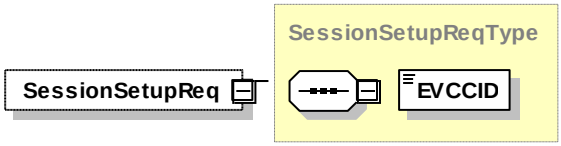


Figure 21 — Schema Diagram – SessionSetupReq

[V2G2-189] The message elements of this message shall be used as defined in Table 26.

Table 26 — Semantics and type definition for SessionSetupReq

Element Name	Type	Semantics
EVCCID	simpleType: evccIDType hexBinary (max length: 8) refer to Annex C.6 for the type definition	Specifies the EV's identification in a readable format. It contains the MAC address of the EVCC.

8.4.1.2.3 Session Setup Response

By using the SessionSetupRes the SECC responds to an SessionSetupReq. With the SessionSetup Res the SECC notifies the EVCC with an enclosed response code, whether establishing a new session or joining a previous Communication Session was successful or not.

[V2G2-190] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according to Figure 22.

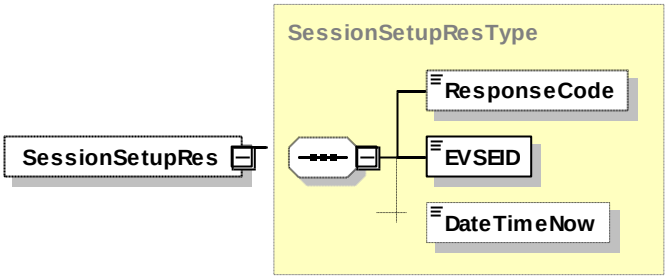


Figure 22 — Schema Diagram – SessionSetupRes

[V2G2-191] The message elements of this message shall be used as defined in Table 27

Table 27 — Semantics and type definition for SessionSetupRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
EVSEID	simpleType: evseIDType hexBinary (max length:32)	Any ID that uniquely identifies the EVSE. The format of this message element is defined in DIN 91286. If an SECC cannot provide such ID data, the value of the EVSEID is set to zero (00hex).
DateTimeNow	simpleType long refer to Annex C.6 for the type definition	Optional: Timestamp of the current SECC time using to the Unix Time Stamp format. This message element Is used by the EVCC to check the validity of the certificates for contract based charging and as external time reference. Based on this information the EVCC might implement a strategy when certificate updates are required. Using this message elment avoids that the time base of the EVCC and SECC need to be synchronized.

[V2G2-192] The SECC and the EVCC shall use the format for EVSEID as defined in DIN 91286.

8.4.1.3 Service Discovery

8.4.1.3.1 Service Discovery Handling

The Service Discovery enables the EVCC to find all services provided by the SECC. This document only describes relevant aspects of the interface between EVCC and SECC with regards to charging the EV. Nevertheless the basis for discovery of future added value services is already considered and offers means for extensibility. Therefore the Service Discovery differentiates between various service types and scopes.

8.4.1.3.2 Service Discovery Request

By sending the ServiceDiescoveryReq message the EVCC triggers the SECC to send information about all services offered by the SECC. Furthermore, the EVCC can limit for particular services by using the service scope and service type elements.

[V2G2-193] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according to Figure 23.

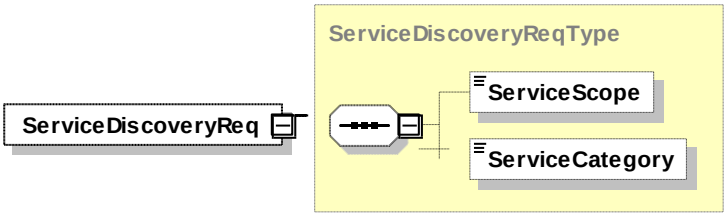


Figure 23 — Schema Diagram – ServiceDiscoveryReq

[V2G2-194] The message elements of this message shall be used as defined in Table 28.

Table 28 — Semantics and type definition for ServiceDiscoveryReq

Element Name	Type	Semantic
ServiceScope	simpleType: serviceScopeType string (max length: 32) refer to Annex C.6 for the type definition	Optional Element: Defines the scope of the Service Discovery. A scope is defined by a unique URI which corresponds to a service provider (e.g. mobility provider, value added service provider etc.). The Service Discovery can respect multiple scopes in one request. By applying a scope to the service discovery the resulting list of services returned in the ServiceDiscoveryRes can be limited to a certain scope, thus enables pre-filtering. The SECC always returns all supported services for alle scopes if no specific ServiceScope has been indicated in request message.
ServiceCategory	simpleType: serviceCategoryType enumeration refer to Annex C.6 for the type definition	Optional Element: Defines the service category for the Service Discovery (e.g. EV charging, internet access etc.). By applying a category to the Service Discovery the resulting list of services returned in the ServiceDiscoveryRes can be limited to a certain category of services, thus enables pre-filtering. The SECC always returns all supported services for all categories if no specific category has been indicated in request message by using the ServicCategory message element.

8.4.1.3.3 Service Discovery Response

After receiving the ServiceDisoveryReq message of the EVCC the SECC sends the ServiceDiscoveryRes message. In case of a successful service discovery, the response lists all available services of the SECC for the defined criteria. In case the service discovery failed the service list is empty and the response code indicates potential reasons.

[V2G2-195] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according to Figure 24.

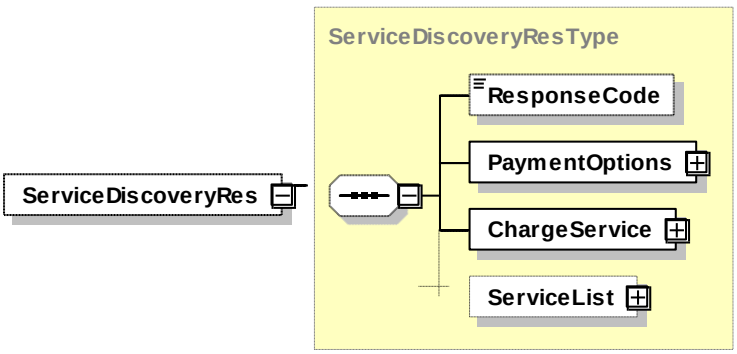


Figure 24 — Schema Diagram – ServiceDiscoveryRes

[V2G2-196] The message elements of this message shall be used as defined in Table 29.

Table 29 — Semantics and type definition for ServiceDiscoveryRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC..
PaymentOptions	complexType: paymentOptionsType refer to 8.5.2.9	This element includes the list of payment options an SECC offers to the EVCC indicating what method could be choosen to pay for the services. The EVCC can only select one payment method for all services used by the EVCC.
ChargeService	complexType: ServiceChargeType refer to subclause 8.5.2.4	Available charging services supported by the EVSE.
ServiceList	complexType: ServiceTagListType Refer to subclause 8.5.2.2	Optional: A list containing information on all other services than charging services offered by the EVSE. The returned service list is a filtered list based on the ServiceScope and ServiceType indicated in the ServiceDiscoveryReq message.

8.4.1.4 Service Detail

8.4.1.4.1 Service Detail Request

By sending the ServiceDetailReq message the EVCC requests the SECC to send specific additional information about services offered by the EVSE.

[V2G2-197] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according to Figure 25.

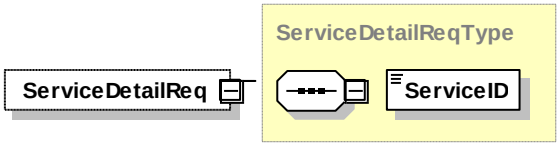


Figure 25 — Schema Diagram – ServiceDetailReq

[V2G2-198] The message elements of this message shall be used as defined in Table 30.

Table 30 — Semantics and type definition for ServiceDetailReq

Element Name	Type	Semantic

ServiceID	simpleType: serviceIDType unsignedShort refer to Annex C.6 for the type definition	This element identfies a service which has been offered by the SECC in the ServiceDiscoveryRes message.

8.4.1.4.2 Service Detail Response

After receiving the ServiceDetailReq message of an EVCC the SECC sends the ServiceDetailRes message and provides details about services.

[V2G2-199] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 26.

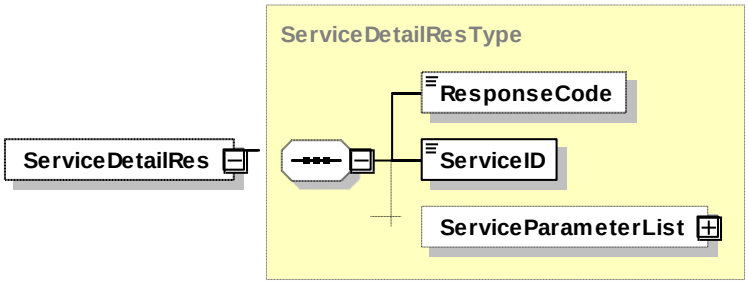


Figure 26 — Schema Diagram – ServiceDetailRes

[V2G2-200] The message elements of this message shall be used as defined in Table 31.

Table 31 — Semantics and type definition for ServiceDetailRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
ServiceID	simpleType: serviceIDType unsignedShort refer to Annex C.6 for the type definition	This element identfies a service which has been offered by the SECC in the ServiceDiscoveryRes message.
ServiceParameterList	complexType: ServiceParameterListType refer to subclause 8.5.2.21	Includes the list of parameters for a specific serviceID received from the SECC in the ServiceDiscoveryRes message.

8.4.1.5 Service and Payment Selection

8.4.1.5.1 Service and Payment Selection Handling

Based on the provided services and the corresponding payment options by the SECC this message pair allows the transmission of the selected PaymentOption, SelectedServices and related ParameterSets. Depending on the selected payment additional messages (PaymentDetails message pair) are exchanged.

8.4.1.5.2 Service and Payment Selection Request

This request message transports the information on the selected services and on how the all the selected services are paid.

[V2G2-201] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 27.

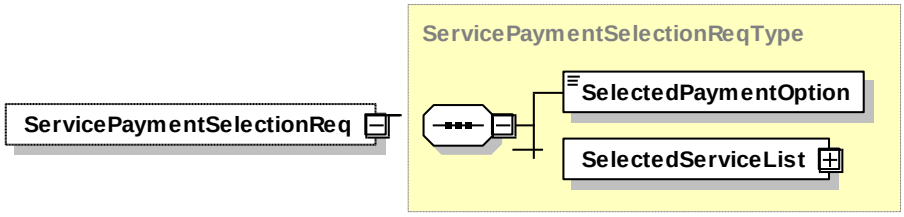


Figure 27 — Schema Diagram – ServicePaymentSelectionReq

[V2G2-202] The message elements of this message shall be used as defined in Table 32.

Table 32 — Semantics and type definition for ServicePaymentSelectionReq

Element Name	Type	Semantics
SelectedPaymentOption	simpleType: paymentOptionType enumeration refer to Annex C.6 for the type definition	This element is used for indicating the payment type selected for the use of all selected services in the selectedServiceList.
SelectedServiceList	complexType: SelectedServiceListType refer to subclause 8.5.2.24	List contains all selected ServiceIDs and the optional parameterSetID for applicable for the respective serviceID.

8.4.1.5.3 Service and Payment Selection Response

With this message the SECC informs the EVCC whether the selected services and payment option were accepted. Depending on the selected payment additional messages (PaymentDetails message pair) are exchanged.

[V2G2-203] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 28.

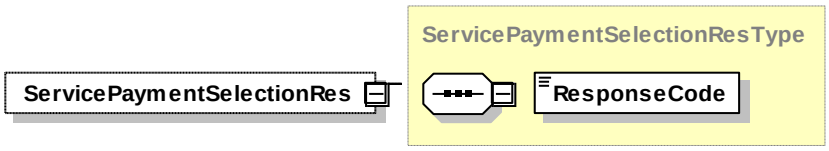


Figure 28 — Schema Diagram – ServicePaymentSelectionRes

[V2G2-204] The message elements of this message shall be used as defined in Table 33.

Table 33 — Semantics and type definition for ServicePaymentSelectionRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.

8.4.1.6 Payment Details

8.4.1.6.1 Payment Details Handling

The payment details message pattern is only used when some particular payment details have to be exchanged (e.g. in case of contract based charging a contract identifier would be necessary).

8.4.1.6.2 Payment Details Request

With the Payment Details Request the EVCC provides the payment details in case the selected payment was “Contract”. By sending the signature certificate chain and the contract id, the EVCC requests the SECC to send a challenge.

[V2G2-205] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 29.

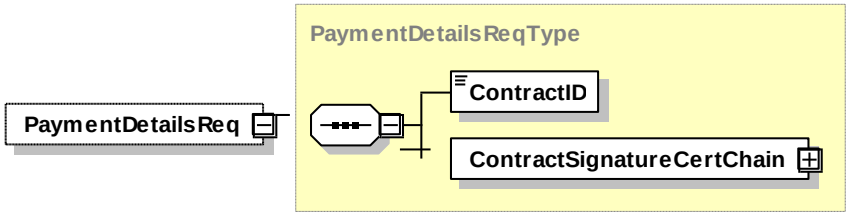


Figure 29 — Schema Diagram – PaymentDetailsReq

[V2G2-206] The message elements of this message shall be used as defined in Table 34.

Table 34 — Semantics and type definition for PaymentDetailsReq

Element Name	Type	Semantics
ContractID	simpleType: contractIDType string (max length 24) refer to Annex C.6 for the type definition	This element identifies the charging contract. The format is defined in DIN 91286.
ContractSignatureCertChain	complexType: CertificateChainType refer to subclause 8.5.2.5	This element contains the Contract Certificate and optional SubCertificates

[V2G2-207] The SECC and the EVCC shall use the format for ContractID as defined in DIN 91286.

8.4.1.6.3 Payment Details Response

With the Payment Details Response the SECC informs the EVCC whether the previously provided payment details were accepted or not. The SECC sends also a challenge in the form of a random number, which has to be signed by the EVCC.

[V2G2-208] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 30.

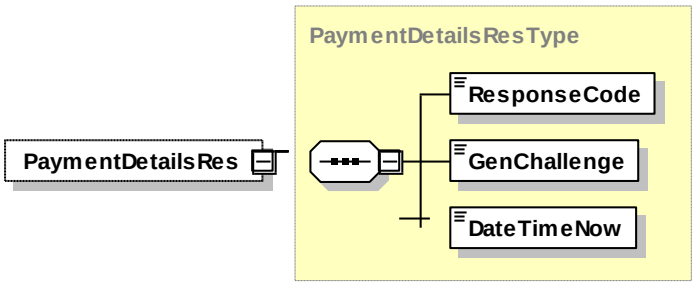


Figure 30 — Schema Diagram – PaymentDetailsRes

[V2G2-209] The message elements of this message shall be used as defined in Table 35.

Table 35 — Semantics and type definition for PaymentDetailsRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
GenChallenge	simpleType genChallengeType string refer to Annex C.5 for the type definition	The challenge sent by the SECC. This element contains the generated random number. Any encoding may be used to encode the random number into a string. The entropy of the random number shall be 128 bit minimum.
DateTimeNow	simpleType long refer to Annex C.5 for the type definition	Timestamp of the current SECC time using to the Unix Time Stamp format. This message element is used by the EVCC to check the validity of the certificates for contract based charging and as external time reference. Based on this information the EVCC might implement a strategy when certificate updates are required. Using this message element avoids that the time base of the EVCC and SECC need to be synchronized.

8.4.1.7 Contract Authentication

8.4.1.7.1 Contract Authentication Request

After receiving the generated challenge from the SECC, EVCC sends another request containing the signature of the challenge and the challenge itself.

[V2G2-210] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 31.

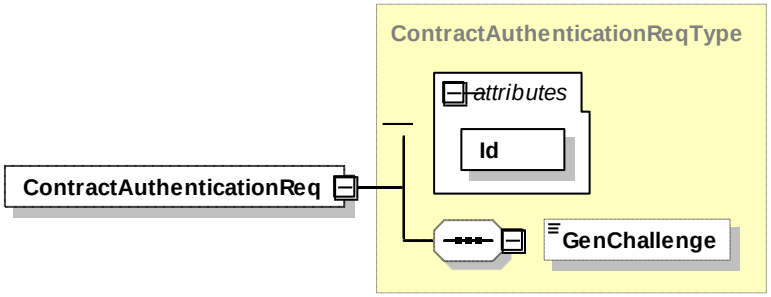


Figure 31 — Schema Diagram – ContractAuthenticationReq

[V2G2-211] The message elements of this message shall be used as defined in Table 36.

Table 36 — Semantics and type definition for ContractAuthenticationReq

Element Name	Type	Semantics
Id	simpleType IDREF refer to Annex C.5 for the type definition	This element is used for referencing the message body in the message header when a signature needs to be applied.
GenChallenge	simpleType genChallengeType string refer to Annex C.5 for the type definition	Optional: The challenge sent by the SECC in PaymentDetailsRes message. This element contains the generated random number. Any encoding may be used to encode the random number into a string. The entropy of the random number shall be 128 bit minimum.

8.4.1.7.2 Contract Authentication Response

Finally, the SECC verifies the certificate and the challenge signature and sends the corresponding authentication answer including the signature of the challenge and the challenge itself.

[V2G2-212] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 32.

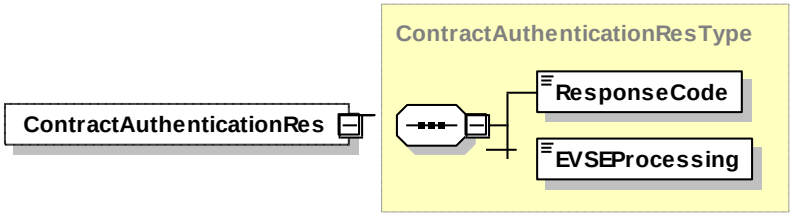


Figure 32 — Schema Diagram – ContractAuthenticationRes

[V2G2-213] The message elements of this message shall be used as defined in Table 37.

Table 37 — Semantics and type definition for ContractAuthenticationRes

Element Name	Type	Semantics
EVSEProcessing	simpleType: EVSEProcessingType enumeration refer to Annex C.6 for the type definition	Parameter indicating that the EVSE has finished the processing that was initiated after the ContractAuthenticationReq or if the EVSE is still processing at the time, the response message was sent.
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.

NOTE By using the EVSEProcessing parameter, the EVSE can indicate to the EVCC that the processing is not finished but a response message has to be sent to fulfil the timeout and performance requirements defined in subclause 8.7.2. This allows to stop the communication session while fulfilling the performance and timeout requirements.

8.4.1.8 Charge Parameter Discovery

8.4.1.8.1 Charge Parameter Discovery Handling

After being authorized for charging at the EVSE (SECC) the EVCC and the SECC negotiate the charging parameters with the Charge Parameter Discovery Request/Response message pair.

Concepts treated for an optimal supply of energy that corresponds to the customer needs :

The charging parameters negotiation that proceeds the delivery of energy or may be engaged during the energy delivery phase is destined to ensure that the client will be satisfied while at the same time ensuring that the energy will effectively be available and fall within the capacity of power supply grid at the local level (private network) and at the regional level (public network). This required negotiation will become more and more necessary as the number of EVs increases, as well as local renewable volatile production.

Initially, before the onset of the energy supply, the EV will negotiate with charging spot operator, and third party actors indirectly, to fit to the known or predicted available electric power. The available energy may evolve once the charging has started due to sudden lack of power source, or increase in demand of other consumptions (e.g. other EV arrival). New negotiation must be allowed to cope with difficulties encountered, locally or regionaly.

Such re-negotiation must be included within the communication protocol between EVSE and EV during the charging period. Upper level systems, to be designed in the future, using this protocol, will manage to combine these functions in order to reach the optimum between satisfaction of end user needs and other constraints.

8.4.1.8.2 Charge Parameter Discovery Request

By sending the Charge Parameter Discovery Request message the EVCC provides its charging parameters to the SECC. This message provides status information about the EV and additional charging parameters, like estimated energy amounts for recharge and the point in time for the end of charge.

[V2G2-214] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 33.

[V2G2-215] If there is no DepatureTime information available the EVCC shall not include this message element in the Charge Parameter Discovery Request message.

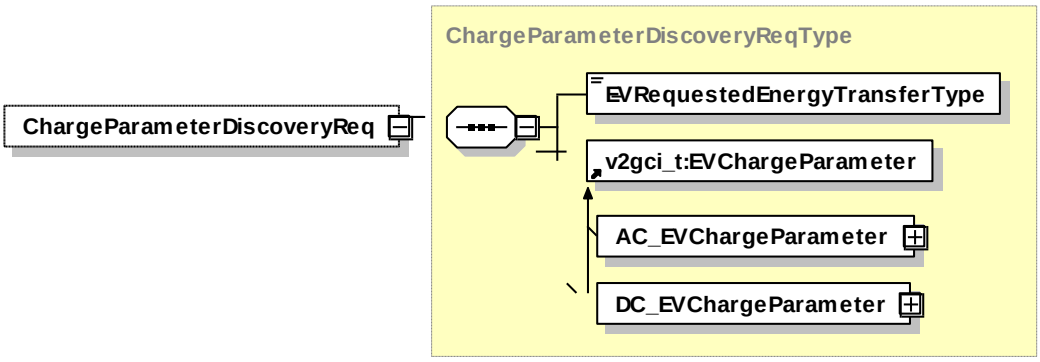


Figure 33 — Schema Diagram – ChargeParameterDiscoveryReq

[V2G2-216] The message elements of this message shall be used as defined in Table 38.

Table 38 — Semantics and type definition for ChargeParameterDiscoveryReq

Element Name	Type	Semantics
EVRequestedEnergyTransfer	simpleType: EVRequestedEnergyTransferType enumeration refer to Annex C.6 for the type definition and the Table 39.	Selected energy transfer for charging that is requested by the EVCC. Refer to Table 39 for details.
AC_EVChargeParameter	complexType: AC_EVChargeParameterType substitutes abstract type EV_ChargeParameterType refer to subclause 8.5.3.2	This element is used the by the EVCC for initating the target setting process for AC charging.
DC_EVChargeParameter	complexType: DC_EVChargeParameterType substitutes abstract type EV_ChargeParameterType refer to subclause 8.5.4.3	This element is used the by the EVCC for initating the target setting process for DC charging.

The definition of EVRequestedEnergyTransferType supports the connectors according to IEC 62196 and SAE J1772 Combo1 (Hybrid). Based on the supported connectors the EVCC can choose the charging services as defined in Table 39.

[V2G2-217] The EVCC shall use the EVRequestedEnergyTransferType as described in Table 39.

Table 39 — Semantics for EVRequestedEnergyTransferType

EnergyTransferType	Offered charging service
AC_single_phase_core	EVCC requests **AC single phase** charging according to IEC 62196.
AC_three_phase_core	EVCC requests **AC three phase** charging according to IEC 62196.
DC_core	EVCC requests **DC** charging according to IEC 62196 on the **core pins**.
DC_extended	EVCC requests **DC** charging using **combo 1** or **combo2** connector on **extended pins**.
DC_combo_core	EVCC requests **DC** charging using **combo 1** or **combo2** connector on the **core pins**.
DC_unique	EVCC requests **DC** charging using a **dedicated DC coupler** (e.g. Chademo).

NOTE The EVSESupportedEnergyTransferType may provide multiple options for charge services. Depending on the options the EVCC has to select only a subset of the offered option. For example, if the EVSE offers AC_single_DC_core, the EVCC has to select either AC_single or DC_core because both options can technically not be supported at the same time (refer to EVRequestedEnergyTransferType, to Annex C.6).

8.4.1.8.3 Charge Parameter Discovery Response

With the Charge Parameter Discovery Response message the SECC provides applicable charge parameters from the grid's perspective. Next to general charge parameters of the EVSE this optionally includes further information on cost over time, cost over demand, cost over consumption or a combination of these. The term cost refers to any kind of cost (see subclause 8.5.2.20) specified in this version of the standard and is not limited to monetary costs. Based on this cost information the EV may optimize its charging schedule for the requested amount of energy.

[V2G2-218] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according to Figure 34.

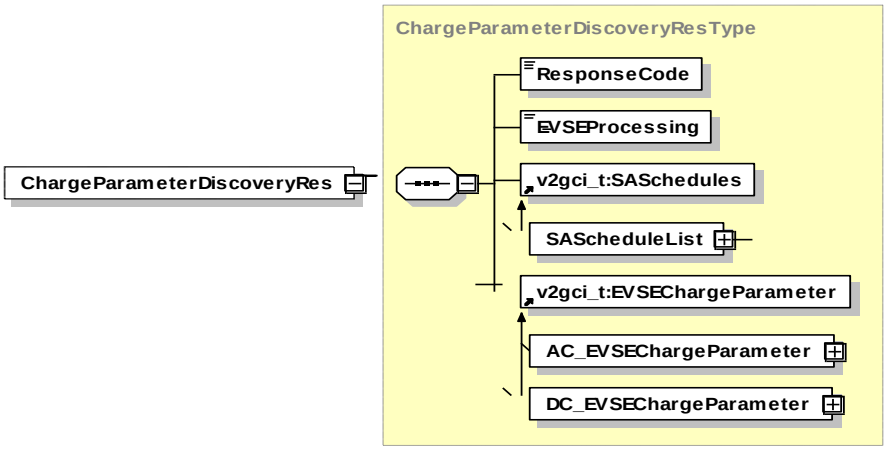


Figure 34 — Schema Diagram – ChargeParameterDiscoveryRes

[V2G2-219] All sales tariffs provided in the SASchedules element shall be originating from the same SA, which may be indicated by the EnergyProvider element.

[V2G2-220] The message elements of this message shall be used as defined in Table 40.

Table 40 — Semantics and type definition for ChargeParameterDiscoveryRes

Element Name	Type	Semantics
EVSEProcessing	simpleType: EVSEProcessingType enumeration refer to Annex C.6 for the type definition	Parameter indicating that the EVSE has finished the processing that was initiated after the ContractAuthenticationReq or if the EVSE is still processing at the time, the response message was sent.
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
SAScheduleList	complexType: SAScheduleListType substitutes abstract type SASchedulesType refer to subclause 8.5.2.12	Includes several tuples of schedules from secondary actors
AC_EVSEChargeParameter	complexType: AC_EVSEChargeParameterType substitutes abstract type EVSE_ChargeParameterType refer to subclause 8.5.3.3	This element is used the by the SECC for initating the target setting process for AC charging.
DC_EVSEChargeParameter	complexType: DC_EVSEChargeParameterType substitutes abstract type EVSE_ChargeParameterType refer to subclause 8.5.4.4	This element is used the by the SECC for initating the target setting process for DC charging.

NOTE By using the EVSEProcessing parameter, the EVSE can indicate to the EVCC that the processing is not finished but a response message has to be sent to fulfil the timeout and performance requirements defined in subclause 8.7.2. This allows to stop the communication session while fulfilling the performance and timeout requirements.

8.4.1.9 Power Delivery

8.4.1.9.1 Power Delivery Handling

The Power Delivery message exchange marks the point in time when the EVSE provides voltage to its output socket and the EV can start to recharge its battery.

8.4.1.9.2 Power Delivery Request

By sending the Power Delivery Request the EVCC requests the SECC to switch power on and transmits the charging identification mode it will follow during the charging process.

NOTE The point in time this message is sent does not necessarily correlate with the start of the charging process. The EV may decide on the basis of its schedule when the charge process starts.

[V2G2-221] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 35.

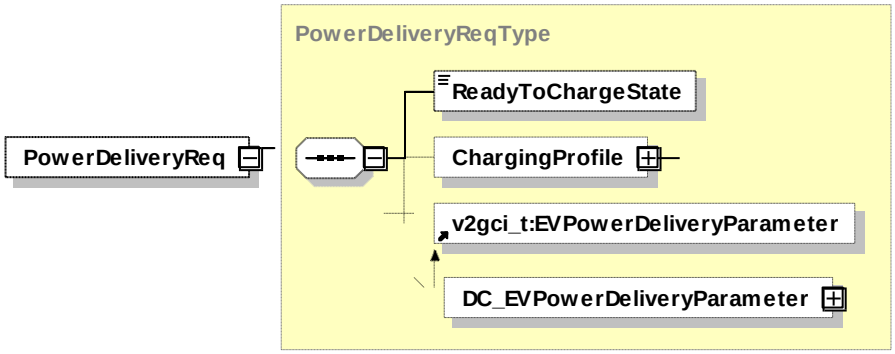


Figure 35 — Schema Diagram – PowerDeliveryReq

[V2G2-222] The message elements of this message shall be used as defined in Table 41.

Table 41 — Semantics and type definition for PowerDeliveryReq

Element Name	Type	Semantics
ReadyToChargeState	simpleType: boolean refer to Annex C.6 for the type definition	This message element is used to request the EVSE to fulfill all conditions that the energy transfer can start as soon as the EV onboard system begins to retrieve energy without any further action to be taken (i.e. the EVSE is requested to close its contactors successfully). If ReadyToCharge is equal to TRUE the EVSE is requested to prepare the energy flow for an immediate start, if ReadyToCharge is equal to FALSE the EVSE is requested to stop the energy flow.
ChargingProfile	complexType: ChargingProfileType refer to subclause 8.5.2.10	Optional Element: Allows an EV to reserve a specific charging profile for the current charging session (i.e. maximum amount of power drawn over time).
DC_EVPowerDeliveryParameter	complexType: DC_EVPowerDeliveryParameterType substitutes abstract type EVPowerDeliveryParameter refer to subclause 8.5.4.5	Optional Element: This element is used the by the EVCC for transmitting the parameters for power delivery

8.4.1.9.3 Power Delivery Response

After receiving the Power Delivery Request message of the EVCC the SECC sends the Power Delivery Response message including information if power will be available.

[V2G2-223] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 36.

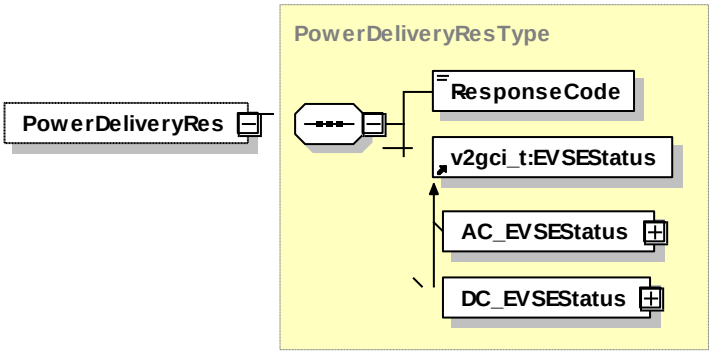


Figure 36 — Schema Diagram – PowerDeliveryRes

[V2G2-224] The SECC shall always accept the ChargingProfile of the EVCC (see subclause 8.5.2.10) if it does not exceed the PMax values of all PMaxScheduleEntry elements (see subclause 8.5.2.15) according to the chosen SAScheduleTuple element (see subclause 8.5.2.13) in the Charge Parameter Discovery Response message (see subclause 8.4.1.8.3).

[V2G2-225] The SECC shall send the negative response code FAILED_ChargingProfileInvalid in the PowerDelivery response message if the EVCC sends a ChargingProfile (see subclause 8.5.2.10) which is not adhering to the PMax values of all PMaxScheduleEntry elements (see subclause 8.5.2.15) according to the chosen SAScheduleTuple element (see subclause 8.5.2.13) in the Charge Parameter Discovery Response message (see subclause 8.4.1.8.3).

[V2G2-226] The message elements of this message shall be used as defined in Table 42.

Table 42 — Semantics and type definition for PowerDeliveryRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
AC_EVSEStatus	complexType: AC_EVSEStatusType substitutes abstract type EVSE_StatusType refer to subclause 8.5.3.1	This element is used the by the SECC for indicating the SECC status.
DC_EVSEStatus	complexType: AC_EVSEStatusType substitutes abstract type EVSE_StatusType refer to subclause 8.5.4.1	This element is used the by the SECC for indicating the SECC status.

8.4.1.10 Certificate Update

8.4.1.10.1 Certificate Update handling

Updating the certificate of the EVCC is required when these certificate is still valid but are about to expire. In this use case the EVCC requests new certificate from the SECC to be installed into the EVCC. This procedure may happen at any time but typically during charging or later than charging. Especially the response may arrive after charging is completed since the certificate may have to be requested from a secondary actor and they may have to be created by this secondary actor.

[V2G2-227] The EVCC shall request the required certificate at the SECC, provided the EVSE has online capabilities.

NOTE 1 If the SECC is not able to deliver the requested certificate an appropriate error handling has to be performed. If an EVSE is not able to support this function in general, it should be marked for instance as “offline EVSE” (e.g. by an label at the EVSE).

NOTE 2 It is only allowed to use the Certificate Update (as described here) if the ContractSignatureCert is not corrupted; i.e. they must be fully valid to perform this procedure. Otherwise, the same procedure has to be used as for the initial storage of the contract certificate in the EVCC. One possibility to do this is using the message types Certificate Installation Request and Certificate Installation Response.

8.4.1.10.2 Certificate Update Request

By sending the Certificate Update Request the vehicle requests the SECC to deliver new certificate that belong to the currently valid contract of the vehicle.

[V2G2-228] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 37.

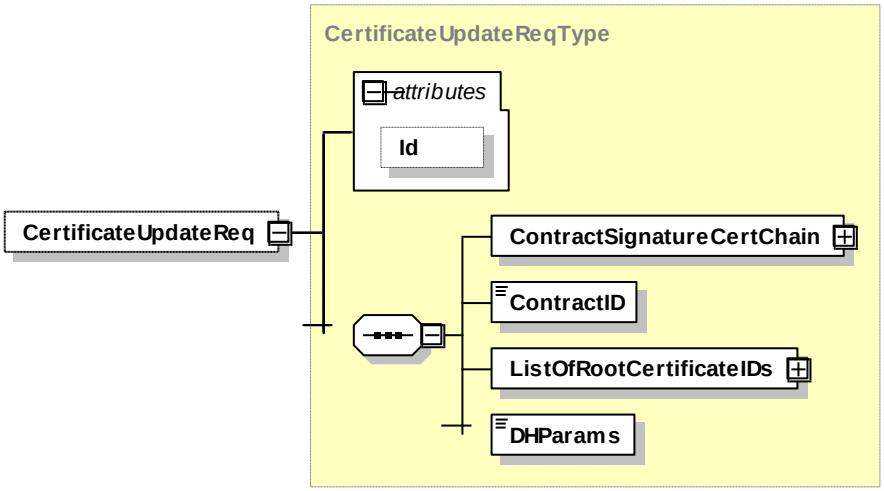


Figure 37 — Schema Diagram –CertificateUpdateReq

[V2G2-229] The message elements of this message shall be used as defined in Table 43.

Table 43 — Semantics and type definition for CertificateUpdateReq

Element Name	Type	Semantics
Id	simpleType IDREF refer to Annex C.6 for the type definition	This element is used for referencing the message body in the message header when a signature needs to be applied.
ContractSignatureCertChain	complexType: CertificateCertChainType refer to subclause 8.5.2.5	Contains the currently available signature certificate including the certificate chain in the EVCC. The SECC uses this certificate(chain) to check the message signature included in the header of the message. The complete chain is transmitted to allow stand-alone verification of the signature.
ContractID	simpleType: contractIDType string (max length 24) refer to Annex C.6 for the type definition	This element identifies the charging contract.

ListOfRootCertificateIDs	complexType: ListOfRootCertificateIDsType refer to subclause 8.5.2.27	This list contains the Certificate IDs of all Root Certificates currently installed in the EVCC.
DHParams	simpleType: dHParamsType base64Binary (max. length 256) refer to Annex C.6 for the type definition	Diffie Hellman parameters from the EVCC to SA for generating a common session key in order to encrypt the Contract Signature Private Key

8.4.1.10.3 Certificate Update Response

After receiving the Certificate Update Request of the EVCC the SECC retreives the requested certificate from the SA. It therefore needs to establish an online connection. Then the SECC sends the Certificate Update Response including the new certificate to the EVCC. Finally the EVCC installs this certificate.

[V2G2-230] The EVCC shall possess a certificate and corresponding private key usable for signature and key agreement to support the decryption of encrypted information.

[V2G2-231] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 38.

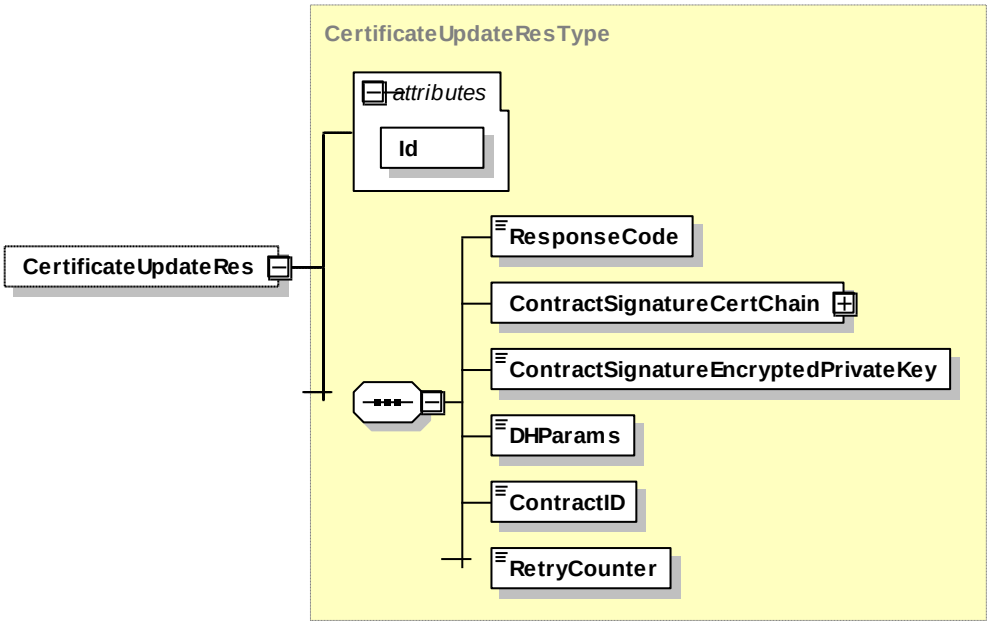


Figure 38 — Schema Diagram – CertificateUpdateRes

[V2G2-232] The message elements of this message shall be used as defined in Table 44.

Table 44 — Semantics and type definition for CertificateUpdateRes

Element Name	Type	Semantics
Id	simpleType IDREF refer to Annex C.6 for the type definition	This element is used for referencing the message body in the message header when a signature needs to be applied.

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
ContractSignatureCertChain	complexType: CertificateCertChainType refer to subclause 8.5.2.5	The new certificate chain for signature purposes that has to be installed in the EVCC.
ContractSignatureEncryptedPrivateKey	simpleType: privateKeyType base64Binary (max. length 128) refer to Annex C.6 for the type definition	The private key that belongs to the new certificate for signature purposes. It has to be installed in the EVCC as well. This is secret data that has to be encrypted using the old contract certificate of the EVCC (based on ContractSignatureCertChain contained in message Certificate Update Request) and using the calculated DH secret for encryption as described in Annex I.
DHParams	simpleType: dHParamsType base64Binary (max. length 256) refer to Annex C.6 for the type definition	Diffie Hellman parameters from the SA for generating the session key at the EVCC in order to encrypt the Contract Signature Private Key.
ContractID	simpleType: contractIDType string (max length 24) refer to Annex C.6 for the type definition	This element identifies the charging contract.
RetryCounter	simpleType short refer to Annex C.6 for the type definition	If the ResponseCode was “FAILED_NoCertificateAvailable” or “FAILED_ContractCanceled”, this field contains information, when the EVCC should try to get the new Certificate again. The following entries are possible: x > 0: after “x” days 0: immediately (at next charging) -1: never

[V2G2-233] The SECC shall request the required certificate at a secondary actor if it has online capabilities.

NOTE 1 If the SECC is not able to deliver the requested certificate an appropriate error handling has to be performed. If an EVSE is not able to support this function in general, it should be marked for instance as “offline EVSE” (e.g. by an label at the EVSE).

NOTE 2 It is only allowed to use the Certificate Update (as described here) if the ContractSignatureCert was not corrupted; i.e. they must be fully valid to perform this procedure. Otherwise, the same procedure has to be used as for the initial storage of the contract certificate in the EVCC. One possibility to do this is using the message types Certificate Installation Request and Certificate Installation Response.

8.4.1.11 Certificate Installation

8.4.1.11.1 Certificate Installation handling

Installing the certificate into the EVCC is required if it currently does not possess a valid certificate; e.g. because they are expired, revoked, or do not exist at all. In this use case the EVCC requests the certificate from the SECC to be installed into the EVCC. This procedure typically happens before charging since charging with authorisation can only be started if a valid certificate is available in the EVCC. The SECC may

have to request the certificate from a SA and they may have to be created by this SA. After installation of this certificate, the charging process at the EVSE the EV it is connected to may start.

NOTE 1 It is the decision of the OEM whether an OEMProvisioningCert is used. If the OEM decides to do so, the procedure described here may be used for the installation of the contract certificate. If no OEMProvisioningCert is available in the EVCC, this procedure can not be used and the initial contract certificate has to be installed into the EVCC by other means. Other mechanisms than using an OEMProvisioningCert for installing the contract certificate are out of scope of this specification. For details on OEM Provisioning Certificates please refer to Annex J.

[V2G2-234] The SECC shall request the required certificate at a secondary actor if it has online capabilities.

NOTE 2 If the SECC is not able to deliver the requested certificate an appropriate error handling has to be performed. If an EVSE is not able to support this function in general, it should be marked for instance as “offline EVSE” (e.g. by a label at the EVSE). Since the EVCC currently does not possess a valid certificate assigned with its contract, it is not able to use plug and charge until certificate installation is performed successfully.

8.4.1.11.2 Certificate Installation Request

By sending the Certificate Installation Request the vehicle requests the SECC to deliver the certificate that belong to the currently valid contract of the vehicle.

that belong to the currently valid contract of the vehicle.

[V2G2-235] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 39.

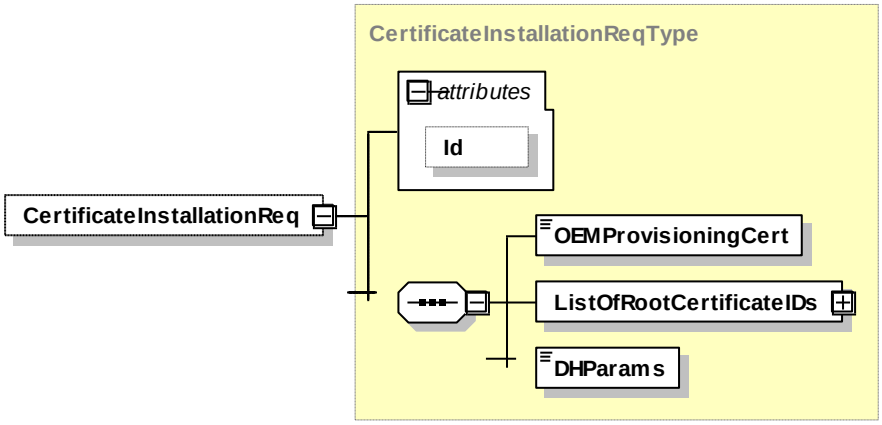


Figure 39 — Schema Diagram – CertificateInstallationReq

[V2G2-236] The message elements of this message shall be used as defined in Table 45.

Table 45 — Semantics and type definition for CertificateInstallationReq

Element Name	Type	Semantics
Id	simpleType IDREF refer to Annex C.6 for the type definition	This element is used for referencing the message body in the message header when a signature needs to be applied.

Element Name	Type	Semantics
OEMProvisioningCert	simpleType certificateType base64Binary (max. length 1200) refer to Annex C.6 for the type definition	An EV specific certificate that was earlier installed in the EVCC typically by an OEM. The ID of this certificate together with the information stored in at the SA (contract partner) are used to identify the currently valid contract of the EV. The certificate ID is given to the SA by the customer using a different communication channel. The certificate itself (i.e. its public key) is used to encrypt data elements in the Certificate Installation Response later on. Please refer also to Annex E and Annex J.
ListOfRootCertificateIDs	complexType: ListOfRootCertificateIDsType refer to subclause 8.5.2.27	This list contains the Certificate IDs of all Root Certificates currently installed in the EVCC.
DHParams	simpleType: dHParamsType base64Binary (max. length 256) refer to Annex C.6 for the type definition	Diffie Hellman parameters from the EVCC to SA for generating a common session key order to encrypt the Contract Signature Private Key

8.4.1.11.3 Certificate Installation Response

After receiving the Certificate Installation Request from the EVCC, the SECC sends the Certificate Installation Response including the requested certificate. Then the EVCC installs this certificate.

There is only one certificate delivered to the EVCC: the one for signing messages. It belongs to the currently valid contract of the EV.

[V2G2-237] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 40.

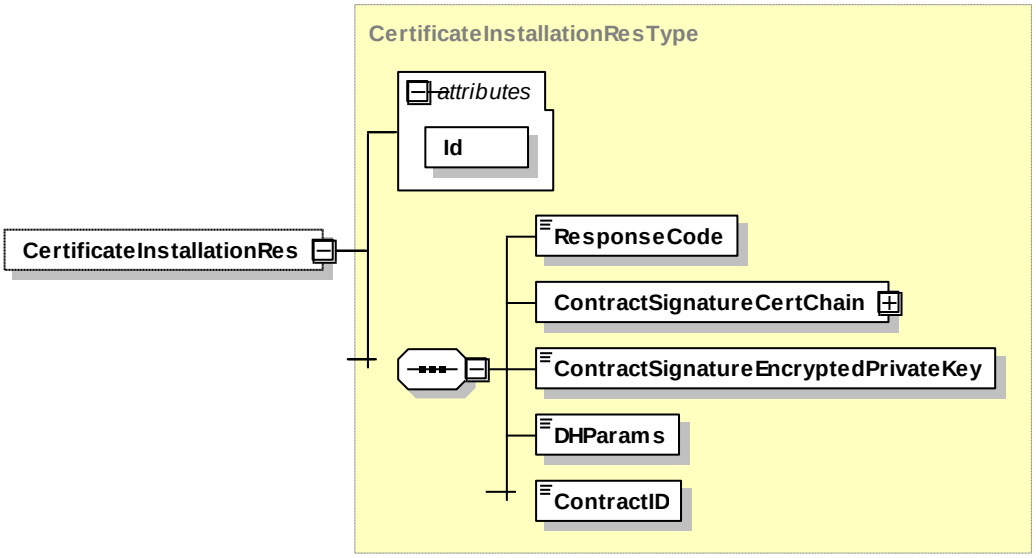


Figure 40 — Schema Diagram – CertificateInstallationRes

[V2G2-238] The message elements of this message shall be used as defined in Table 46.

Table 46 —Semantics and type definition for CertificateInstallationRes

Element Name	Type	Semantics
Id	simpleType IDREF refer to Annex C.6 for the type definition	This element is used for referencing the message body in the message header when a signature needs to be applied.
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
ContractSignatureCertChain	complexType: CertificateCertChainType refer to subclause 8.5.2.5	The new certificate chain for signature purposes that has to be installed in the EVCC.
ContractSignatureEncryptedPrivateKey	simpleType: privateKeyType base64Binary (max. length 128) refer to Annex C.6 for the type definition	The private key that belongs to the new certificate for signature purposes. It has to be installed in the EVCC as well. This is secret data and therefore has to be encrypted using the old contract certificate of the EVCC (based on the ContractSignatureCertChain contained in message Certificate Installation Request) and using the calculated DH secret for encryption as described in Annex I.
DHParams	simpleType: dHParamsType base64Binary (max. length 256) refer to Annex C.6 for the type definition	Diffie Hellman parameters from the SA for generating the session key at the EVCC in order to encrypt the Contract Signature Private Key.
ContractID	simpleType: contractIDType string (max length 24) refer to Annex C.6 for the type definition	This element identifies the charging contract.

8.4.1.11.4 Offline Certificate Installation

As an alternative to the procedure described in this subclause, a Contract Certificate may have to be transmitted to the vehicle without using the charge protocol. This may be for instance necessary if the infrastructure, the secondary actor or the vehicle does not support the Certificate Installation Request / Response. In such cases the Contract Certificate that has to be installed is transmitted to the customer (e.g. per postal mail, electronic mail) and then to the EV (e.g. using the diagnosis interface of the vehicle or an internet access to the vehicle). That means, all these transmissions occur offline and not via the charge protocol. The file (given to the customer) contains the certificate chain and the corresponding private key (similary to CertificateInstallationRes). In order to avoid incompatible file formats and the necessity for the realization of transformation algorithms, a uniform file format shall be used by all secondary actors when distributing contract certificate files:

[V2G2-648] Whenever a secondary actor distributes a file that contains a contract certificate, certificate chain and private signature key, it shall be provided in a PKCS#12 format.

NOTE It is the decision of the secondary actor whether it encrypts the data contained in this file. If the data is encrypted, a password has to be delivered to the customer additionally. For the password transmission a sufficiently secure channel has to be used. Alternatively, the data contained in the PKCS#12-file may not be encrypted but transmitted to the customer via a secure channel (e.g. delivered personally).

8.4.1.12 Session Stop

8.4.1.12.1 Session Stop Handling

This V2G message pair shall be used for terminating a V2G communication session initated by preceding SessionSetupReq message.

8.4.1.12.2 Session Stop Request

By sending the Session Stop Request the EVCC requests termination of the charging process.

[V2G2-239] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 41.

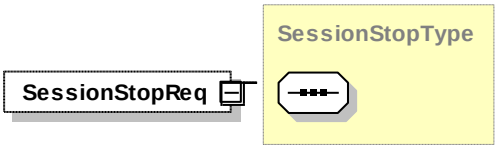


Figure 41 — Schema Diagram – SessionStopReq

8.4.1.12.3 Session Stop Response

After receiving the Session Stop Request of the EVCC the SECC sends the Session Stop Response informing the EVCC if terminating the charging process was successful.

[V2G2-240] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 42.

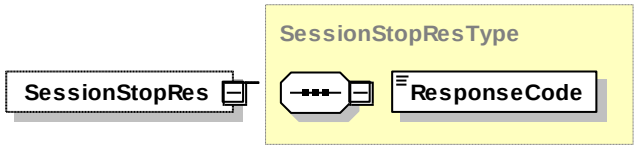


Figure 42 — Schema Diagram – SessionStopRes

[V2G2-241] The message elements of this message shall be used as defined in Table 47.

Table 47 — Semantics and type definition for SessionStopRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.

8.4.2 AC-Messages

8.4.2.1 Overview

Messages defined as AC-Messages belong to the AC Message Set(s).

8.4.2.2 Charging Status

8.4.2.2.1 Charging Status Handling

The Charging Status message pair provides sanity checks on the meter readings provided by the EVSE. On the basis of the iteratively exchanged Charging Status messages the EV has means to check and validate the power drawn from the EVSE. Also, it allows the SECC requesting the EVCC to sign the meter info record included in the charging status response message by using the meter receipt message pair. Usage of this signed meter information for billing purpose may be subject of regulation in certain countries. In addition the request message is used to keep a communication session alive (for session handling refer to subclause 8.7.2).

8.4.2.2.2 Charging Status Request

[V2G2-242] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 43.

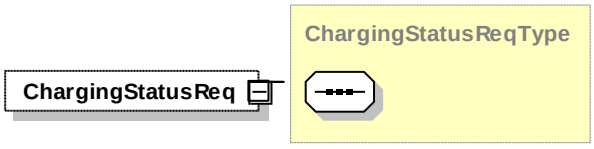


Figure 43 — Schema Diagram – ChargingStatusReq

8.4.2.2.3 Charging Status Response

After receiving the Charging Status Request from the EVCC, the SECC sends the Charging Status Response. In case of a successful Metering Status Request, the response provides the current meter readings from the smart meter installed in the EVSE. In case the Meter Status Request failed no meter reading are provided. The failure is indicated by the response code.

[V2G2-243] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 44.

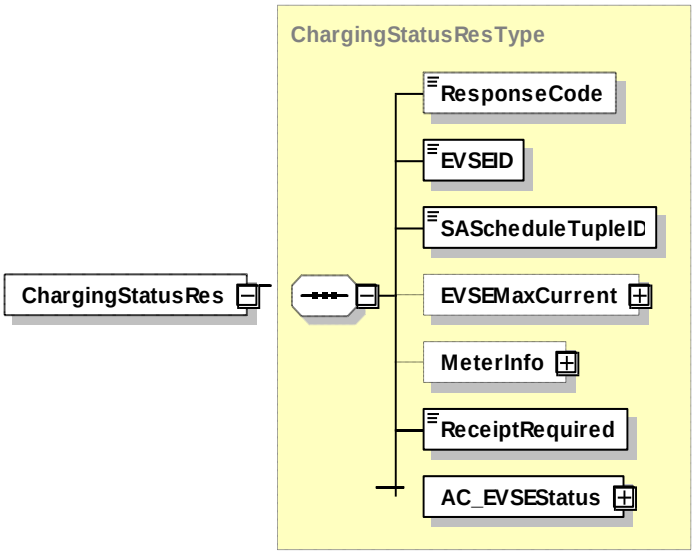


Figure 44 — Schema Diagram – ChargingStatusRes

[V2G2-244] The message elements of this message shall be used as defined in Table 48.

Table 48 — Semantics and type definition for ChargingStatusRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
EVSEID	simpleType: evseIDType hexBinary (max length:32) refer to Annex C.6 for the type definition	Any ID that uniquely identifies the EVSE. The format of this message element is defined in DIN 91286. If an SECC cannot provide such ID data, the value of the EVSEID is set to zero (00hex).
SAScheduleTupleID	simpleType: SAIDType short refer to Annex C.6 for the type definition	Optional: Unique identifier within a charging session for a SAScheduleTuple element. This is used by the SECC to indicate which Tariff applies for the energy currently transferred.
EVSEMaxCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: This element is used by the SECC to indicate the maximum line current the EV can draw. This element is not included in the message if any AC PnC message set has been selected.
MeterInfo	complexType MeterInfoType refer to subclause 8.5.2.6	Optional: Includes the meterInfo record containing the latest meter reading and other meter relevant data.
ReceiptRequired	simpleType boolean refer to Annex C.6 for the type definition	This element is used by the SECC to indicate that the EVCC is required to sent a MeteringReceiptReq message for the purpose of signing the meter info record. If ReceiptRequired is equal to True, the EVCC is required to send a MeteringReceiptReq message including the a signature. If ReceiptRequired is equal to Fals the EVCC is not required to send a MeteringReceiptReq message.
AC_EVSEStatus	complexType: AC_EVSEStatusType substitutes abstract type EVSE_StatusType refer to subclause 8.5.3.1	This element is used the by the SECC for indicating the SECC status.

8.4.2.3 Metering Receipt

8.4.2.3.1 Metering Receipt Handling

By exchanging a Metering Receipt the EVCC acknowledges that the metering information from the EVSE sent by the SECC previously with the Metering Status Response.

8.4.2.3.2 Metering Receipt Request

When sending the Metering Receipt Request the EVCC confirms that the data elements MeterInfo record, SessionID and the SAScheduleTupleID have been received from the SECC. This confirmation is implemented by applying a signature to the message body the MeteringReceiptReq message.

[V2G2-245] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 45.

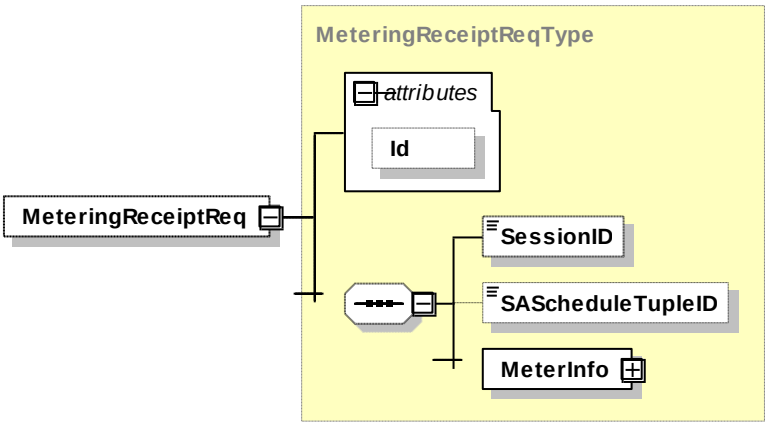


Figure 45 — Schema Diagram – MeteringReceiptReq

[V2G2-246] The message elements of this message shall be used as defined in Table 49.

Table 49 — Semantics and type definition for MetertingReceiptReq

Element Name	Type	Semantic
Id	simpleType IDREF refer to Annex C.6 for the type definition	This element is used for referencing the message body in the message header when a signature needs to be applied.
SessionID	simpleType: SessionIDType: hexBinary (max length: 8)	This message element is used by EVCC and SECC for uniquely identifying a V2G communication session. This element is identical with the one included in the message header. It is placed in the body in addition to be able to apply a signature to it (complete body is signed).
SAScheduleTupleID	simpleType: SAIDType short refer to Annex C.6 for the type definition	Optional: Unique identifier within a charging session for a SAScheduleTuple element. This element is just an echo of the value received in the ChargingStatusRes message from the SECC.
MeterInfo	complexType MeterInfoType refer to subclause 8.5.2.6.	If the SECC indicated in the ChargingStatusRes that a MeteringReceiptReq is required this message element is the echo of the MeterInfo record received in the ChargingStatusRes from the SECC.

8.4.2.3.3 Metering Receipt Response

After receiving the Metering Receipt Request from the EVCC the SECC sends the Metering Receipt Response informing the EVCC whether the receipt was successfully received and accepted by the SECC.

[V2G2-247] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 46.

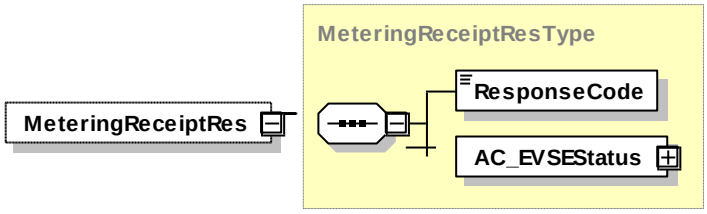


Figure 46 — Schema Diagram – MeteringReceiptRes

[V2G2-248] The message elements of this message shall be used as defined in Table 50.

Table 50 — Semantics and type definition for MeteringReceiptRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
AC_EVSEStatus	complexType: AC_EVSEStatusType substitutes abstract type EVSE_StatusType refer to subclause 8.5.3.1	This element is used the by the SECC for indicating the SECC status.

8.4.3 DC-Messages

8.4.3.1 Overview

Messages defined as DC-Messages belong to the DC Message Set(s) (for details refer to 8.6.2.3)

8.4.3.2 Cable Check

8.4.3.2.1 Cable Check Handling

For a safe DC charging a cable check must be performed.

8.4.3.2.2 Cable Check Request

The cable check request asks for the cable check status of the EVSE and e.g. tells the EVSE if the connector is locked on EV side and if the EV is ready to charge.

[V2G2-249] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 47.

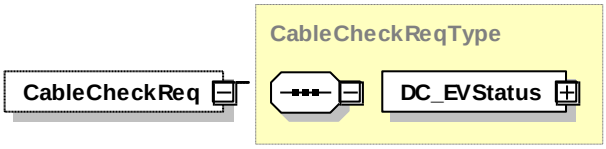


Figure 47 — Schema Diagram – CableCheckReq

[V2G2-250] The message elements of this message shall be used as defined in Table 51.

Table 51 — Semantics and type definition for CableCheckReq

Element Name	Type	Semantics
DC_EVStatus	complexType: DC_EVStatusType refer to subclause 8.5.4.2	Current status of the EV

8.4.3.2.3 Cable Check Response

After receiving the Cable Check Request from the EVCC the SECC sends the Cable Check Response informing the EVCC about result of cable check and EVSE status.

[V2G2-251] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 48.

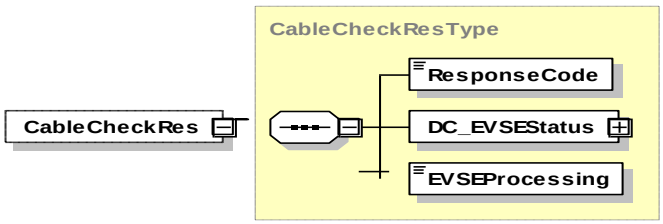


Figure 48 — Schema Diagram – CableCheckRes

[V2G2-252] The message elements of this message shall be used as defined in Table 52.

Table 52 — Semantics and type definition for CableCheckRes

Element Name	Type	Semantics
EVSEProcessing	simpleType: EVSEProcessingType enumeration refer to Annex C.6 for the type definition	Parameter indicating that the EVSE has finished the processing that was initiated after the ContractAuthenticationReq or if the EVSE is still processing at the time, the response message was sent.
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
DC_EVSEStatus	complexType: DC_EVSEStatusType refer to subclause 8.5.4.1	Current status of the EVSE

NOTE By using the EVSEProcessing parameter, the EVSE can indicate to the EVCC that the processing is not finished but a response message has to be sent to fulfil the timeout and performance requirements defined in subclause 8.7.2. This allows to stop the communication session while fulfilling the performance and timeout requirements.

8.4.3.3 Pre Charge

8.4.3.3.1 Pre Charge Handling

Pre charge is used for adjusting the EVSE output voltage to the EV battery voltage.

8.4.3.3.2 Pre Charge Request

The Pre Charge Request is used to start the Pre Charge process from EV side.

[V2G2-253] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 49.

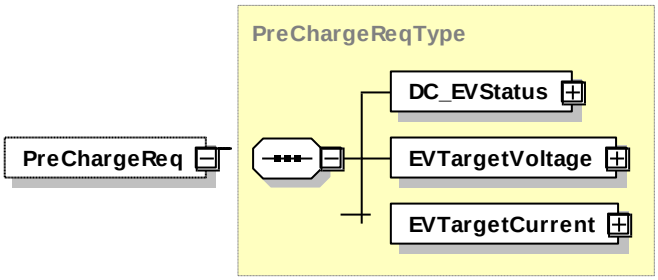


Figure 49 — Schema Diagram – PreChargeReq

[V2G2-254] The message elements of this message shall be used as defined in Table 53.

Table 53 — Semantics and type definition for PreChargeReq

Element Name	Type	Semantics
DC_EVStatus	complexType: DC_EVStatusType refer to subclause 8.5.4.2	Current status of the EV
EVTargetVoltage	complexType PhysicalValueType refer to subclause 8.5.2.7	Target Voltage requested by EV
EVTargetCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	Current demanded by EV

8.4.3.3.3 Pre Charge Response

After receiving the Pre Charge Request from the EVCC the SECC sends the Pre Charge Response informing the EV about the EVSE status and the present EVSE output voltage.

[V2G2-255] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 50.

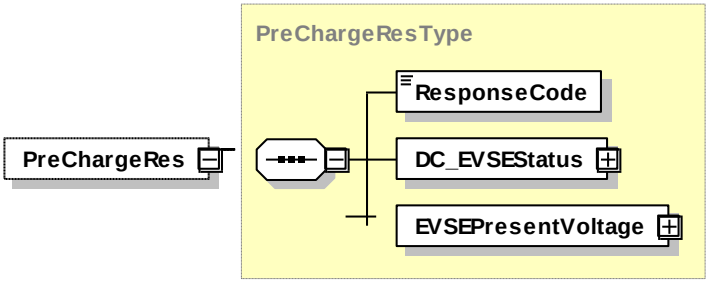


Figure 50 — Schema Diagram – PreChargeRes

[V2G2-256] The message elements of this message shall be used as defined in Table 54.

Table 54 — Semantics and type definition for PreChargeRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
DC_EVSEStatus	complexType: DC_EVSEStatusType refer to subclause 8.5.4.1	Current status of the EVSE
EVSEPresentVoltage	complexType PhysicalValueType refer to subclause 8.5.2.7	Present voltage of EVSE, refers to SAE Voltage Output

8.4.3.4 Current Demand

8.4.3.4.1 Current Demand Handling

For DC charging control cyclic exchange of the requested current from EV side is necessary. Also the target voltage and the difference in current and voltages is transferred.

8.4.3.4.2 Current Demand Request

By sending the Current Demand Request the EV requests a certain current from the EVSE. Also the target voltage, current and voltage difference are transferred.

[V2G2-257] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 51.

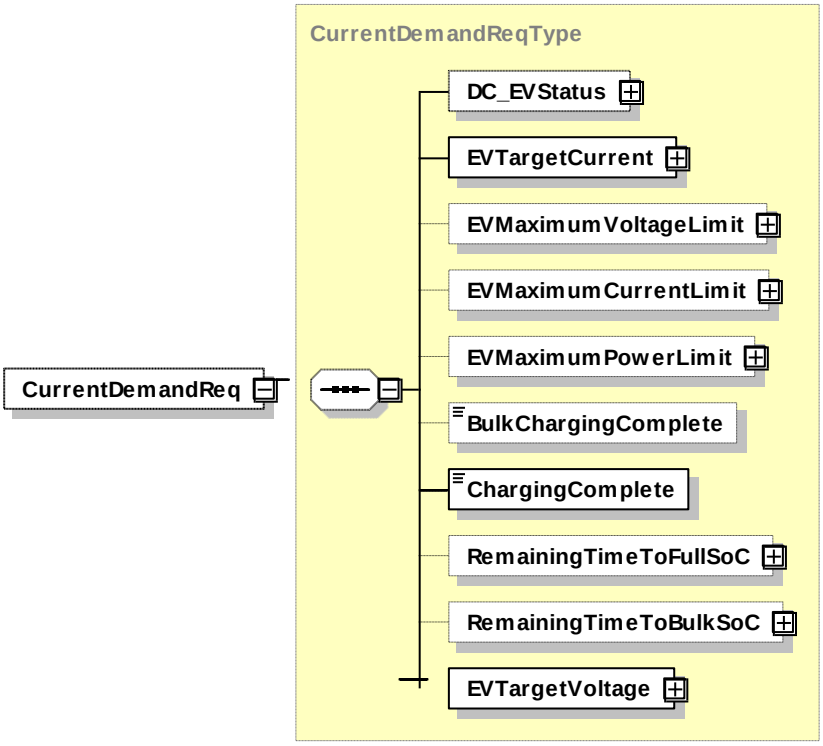


Figure 51 — Schema Diagram – CurrentDemandReq

[V2G2-258] The message elements of this message shall be used as defined in Table 55.

Table 55 — Semantics and type definition for CurrentDemandReq

Element Name	Type	Semantics
DC_EVStatus	complexType: DC_EVStatusType refer to subclause 8.5.4.2	Current status of the EV
EVTargetCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	Instantaneous current requested by the EV
EVMaximumVoltageLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Maximum voltage allowed by the EV
EVMaximumCurrentLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Maximum current allowed by the EV
EVMaximumPowerLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Maximum power allowed by the EV
BulkChargingComplete	simpleType boolean refer to Annex C.6 for the type definition	Optional: If set to TRUE, the EV indicates that bulk charge (approx. 80% SOC) is complete.
ChargingComplete	simpleType boolean refer to Annex C.6 for the type definition	If set to TRUE, the EV indicates that full charge (100% SOC) is complete.
RemainingTimeToFullSoC	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Estimated or calculated time until full charge (100% SOC) is complete

Element Name	Type	Semantics
RemainingTimeToBulkSoC	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Estimated or calculated time until bulk charge (approx. 80% SOC) is complete
EVTargetVoltage	complexType PhysicalValueType refer to subclause 8.5.2.7	Target voltage requested by the EV.

8.4.3.4.3 Current Demand Response

After receiving the Current Demand Request from the EVCC the SECC sends the Current Demand Response informing the EV about the EVSE status and the present EVSE output voltage and current.

[V2G2-259] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 52.

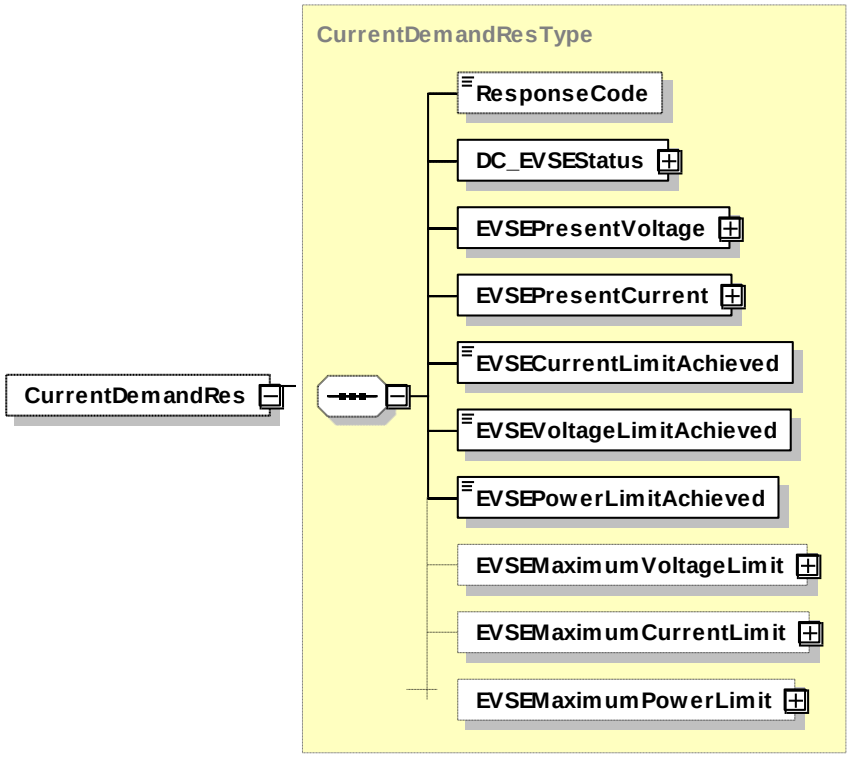


Figure 52 — Schema Diagram – CurrentDemandRes

[V2G2-260] The message elements of this message shall be used as defined in Table 56.

Table 56 — Semantics and type definition for CurrentDemandRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.

DC_EVSEStatus	complexType: DC_EVSEStatusType refer to subclause 8.5.4.1	Current status of the EVSE
EVSEPresentVoltage	complexType PhysicalValueType refer to subclause 8.5.2.7	Present output voltage of the EVSE. Refer to SAE – ISO Mapping Table
EVSEPresentCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	Present output current of the EVSE. Refer to SAE – ISO Mapping Table
EVSECurrentLimitAchieved	simpleType boolean refer to Annex C.6 for the type definition	If set to TRUE, the EVSE has reached its current limit.
EVSEVoltageLimitAchieved	simpleType boolean refer to Annex C.6 for the type definition	If set to TRUE, the EVSE has reached its voltage limit
EVSEPowerLimitAchieved	simpleType boolean refer to Annex C.6 for the type definition	If set to TRUE, the EVSE has reached its power limit
EVSEMaximumVoltageLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Maximum voltage the EVSE can deliver
EVSEMaximumCurrentLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Maximum current the EVSE can deliver
EVSEMaximumPowerLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Maximum power the EVSE can deliver

8.4.3.5 Welding Detection

8.4.3.5.1 Welding Detection Handling

Since welded contactors impose safety risks welding detection is required.

8.4.3.5.2 Welding Detection Request

By sending the Welding Detection Request the EV requests welding detection on EVSE side.

[V2G2-261] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 53.

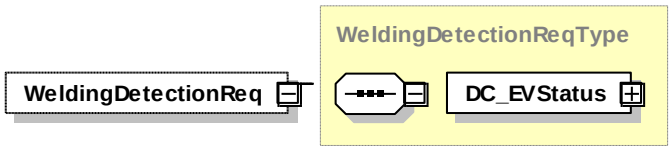


Figure 53 — Schema Diagram – WeldingDetectionReq

[V2G2-262] The message elements of this message shall be used as defined in Table 57.

Table 57 — Semantics and type definition for WeldingDetectionReq

Element Name	Type	Semantics
DC_EVStatus	complexType: DC_EVStatusType refer to subclause 8.5.4.2	Current status of the EV

8.4.3.5.3 Welding Detection Response

After receiving the Welding Detection Request from the EVCC, the SECC sends the Welding Detection Response informing the EV about the EVSE status and the present EVSE output voltage.

[V2G2-263] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement the mandatory messages and message elements as defined Table 98 and according Figure 54.

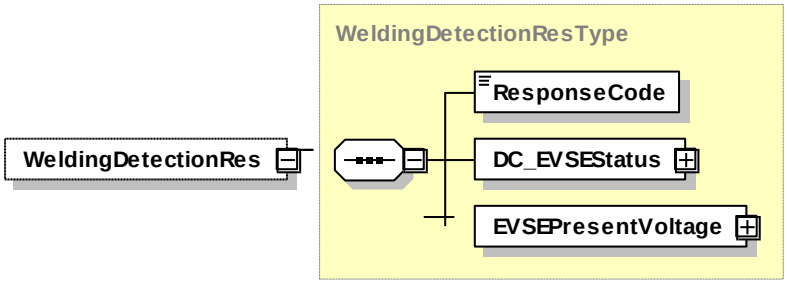


Figure 54 — Schema Diagram – WeldingDetectionRes

[V2G2-264] The message elements of this message shall be used as defined in Table 58.

Table 58 — Semantics and type definition for WeldingDetectionRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	Response Code indicating the acknowledgment status of any of the V2G messages received by the SECC.
DC_EVSEStatus	complexType: DC_EVSEStatusType refer to subclause 8.5.4.1	Current status of the EVSE
EVSEPresentVoltage	complexType PhysicalValueType refer to subclause 8.5.2.7	Present voltage of EVSE, refers to SAE Voltage Output

8.5 Complex Data Types

8.5.1 Overview

In this section complex data types are defined which are used in the messages. Complex data types are composed of several elements which themselves are based on simple data types.

8.5.2 Common

8.5.2.1 ServiceTagType

This Type repersents a tag for a specific service. It gives a short definition and identification of a specific service.

[V2G2-265] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 55.

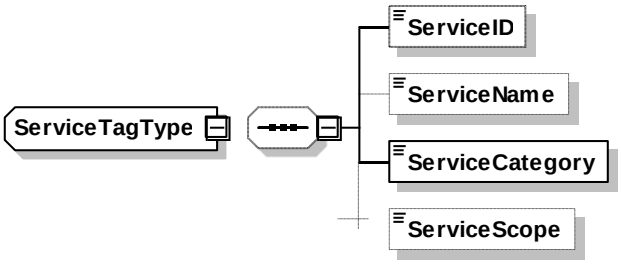


Figure 55 — Schema Diagram – ServiceTagType

[V2G2-266] The message element shall be used as defined in Table 59.

Table 59 —Semantics and type definition for ServiceTagType

Element Name	Type	Semantics
ServiceID	simpleType: unsignedShort refer to Annex C.6 for the type definition	Unique identifier of the service
ServiceName	simpleType: serviceNameType string (max length: 32) refer to Annex C.6 for the type definition	Optional: Human readable service name
ServiceCategory	simpleType: serviceNameType enummeration refer to Annex C.6 for the type definition	Category of a service, corresponds to the defined services, derived from the base service
ServiceScope	simpleType: serviceScopeType string (max length: 32) refer to Annex C.6 for the type definition	Optional: Additional information about usage of that service

8.5.2.2 ServiceTagListType

[V2G2-267] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 56.

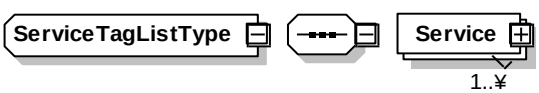


Figure 56 — Schema Diagram – ServiceTagListType

[V2G2-268] The message element shall be used as defined in Table 60.

Table 60 — Semantics and type definition for ServiceChargeType

Element Name	Type	Semantics
Service	complexType: serviceType refer to subclause 8.5.2.3	Contains all information for identifying a service.

8.5.2.3 ServiceType

[V2G2-269] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 57.

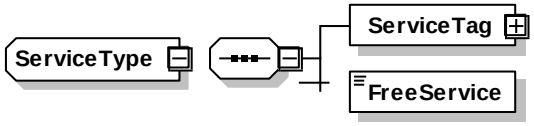


Figure 57 — Schema Diagram – ServiceType

[V2G2-270] The message element shall be used as defined in Table 61.

Table 61 — Semantics and type definition for ServiceType

Element Name	Type	Semantics
ServiceTag	complexType: serviceTagType refer to subclause 8.5.2.1	Includs the data elements to uniquely identify a service.
FreeService	simpleType: boolean refer to Annex C.6 for the type definition	This element is used by the SECC to indicate if a service can be used by the EVCC free of charge or not. If FreeService is equal to true, the EV can use the offered service without payment. If FreeService is equal to false, the service, if used by the EV, will be billed using the payment method negotiated using the payment option message element.

8.5.2.4 ServiceChargeType

EV charging specific service derived from ServiceType (refer to subclause 8.5.2.3) which contains additional information about EVSESSupportedEngeryTransferType(s) offered by the EVSE.

[V2G2-271] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 58.

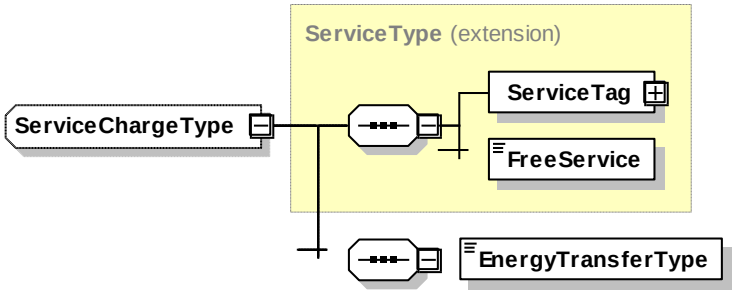


Figure 58 — Schema Diagram – ServiceChargeType

[V2G2-272] The message element shall be used as defined in Table 62.

Table 62 — Semantics and type definition for ServiceChargeType

Element Name	Type	Semantics
EnergyTransferType	simpleType: EVSESupportedEnergyTransferType enumeration refer to Annex C.6 for the type definition	Available charging types or methods, supported by the EVSE.

The definition of the EVSESupportedEnergyTransferType supports the connectors as defined in IEC 62196 and SAE J1772 Combo1 (Hybrid). Based on the supported connectors the SECC can offer charging services as defined in Table 63.

[V2G2-273] The SECC shall use the EVSESupportedEnergyTransferType as described in Table 63.

Table 63 — Semantics for EVSESupportedEnergyTransferType

EnergyTransferType	Offered charging service
AC_single_phase_core	EVSE supports **AC single phase** charging according to IEC 62196.
AC_three_phase_core	EVSE supports **AC three phase** charging according to IEC 62196.
DC_core	EVSE supports **DC** charging according to IEC 62196 on the **core pins**.
DC_extended	EVSE supports **DC** charging using of **combo1** or **combo2** connector on **extended pins**.
DC_combo_core	EVSE supports **DC** charging using of **combo1** or **combo2** connector on **core pins**
DC_dual	EVSE supports **DC** charging using of **combo1** or **combo2** connector on **extended or core pins**
AC_core1p_DC_extended	EVSE supports **AC single phase** charging using **core pins** and **DC charging** using **extended pins** on **combo1** or **combo2** connector
AC_single_DC_core	EVSE supports **single phase AC-** or **DC** charging according to IEC 62196 on the **core pins**.
AC_single_phase_three_phase_core _DC_extended	EVSE supports **single- and three phase AC** charging using **core pins** of the **combo2** connector or **core (single phase)** and **extended pins** (3 phases) on the **combo1** connector.
AC_core3p_DC_extended	EVSE supports **AC three phases** charging using **core pins** and **DC charging** using **extended pins** on **combo2** connector.

NOTE The EVSESupportedEnergyTransferType may provide multiple options for charge services. Depending on the options the EVCC has to select only a subset of the offered options. For example, if the EVSE offers AC_single_DC_core, the EVCC has to select either AC_single or DC_core because both options can technically not be supported at the same time (refer to EVRequestedEnergyTransferType, to Annex C.6).

8.5.2.5 CertificateChainType

This data type stores the client certificate, and all certificates in the chain up to the root. The root certificate is not included in this data type. In the special (but unlikely) case, that a client certificate is directly signed by the root, the “SubCertificates” filed is empty. In all other cases, this field contains an ordered list of sub-CA-certificates to follow the trust-path from the client certificate up to the root.

[V2G2-274] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 59.

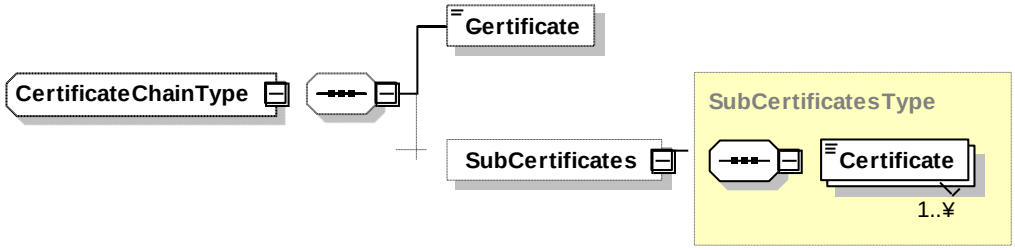


Figure 59 — Schema Diagram – CertificateChainType

[V2G2-275] The message element shall be used as defined in Table 64.

Table 64 — Semantics and type definition for CertificateChainType

Element Name	Type	Semantics
Certificate	simpleType: certificateType base64Binary (max length: 1200) refer to Annex C.6 for the type definition	An x.509v3 Certificate (the “client” certificate)
SubCertificates	complexType: SubCertificatesType refer to subclause 8.5.2.26	Optional: The Chain with all Subcertificates to the Root-Certificate (not including root certificate)

8.5.2.6 MeterInfoType

[V2G2-276] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according Figure 60.

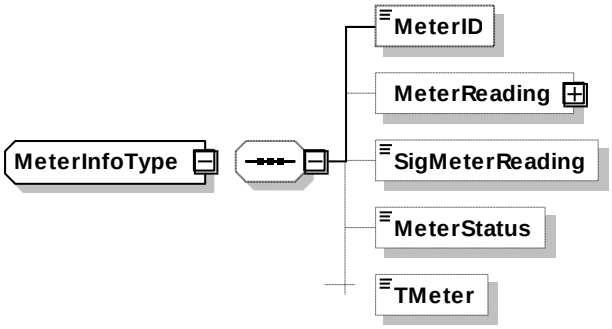


Figure 60 — Schema Diagram – MeterInfoType

[V2G2-277] The message element shall be used as defined in Table 65.

Table 65 — Semantics and type definition for MeterInfoType

Element Name	Type	Semantics
MeterID	simpleType: meterIDType string (max length: 32) refer to Annex C.6 for the type definition	ID of the meter in the EVSE
MeterReading	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional Element: Current meter reading in Watthours from the EVSE
SigMeterReading	simpleType: sigMeterReadingType base64Binary (max length: 32) refer to Annex C.6 for the type definition	Optional Element: Signature of the meter reading This signature is generated by the EVSE meter. It is not verified at the EVCC. It might be used by a SA system for billing purposes if local regulations on metering permit it
MeterStatus	simpleType: meterStatusType short refer to Annex C.6 for the type definition	Optional Element: Current status of the meter
TMeter	simpleType short refer to Annex C.6 for the type definition	Optional Element: Timestamp of the current SECC time using to the Unix Time Stamp format.

8.5.2.7 PhysicalValueType

[V2G2-278] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 61.

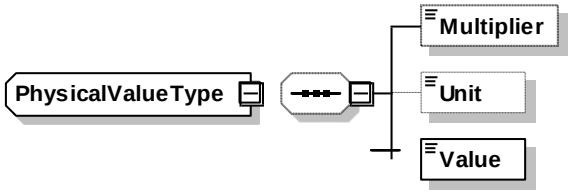


Figure 61 — Schema Diagram – PhysicalValueType

[V2G2-279] The message element shall be used as defined in Table 66.

Table 66 — Semantics and type definition for PhysicalValueType

Element Name	Type	Semantics
Multiplier	simpleType: unitMultiplierType byte (range: -3..+3) refer to Annex C.6 for the type definition	multiplier to be applied to the value included in the message element value
Unit	simpleType: unitSymbolType enumeration refer to Annex C.6 for the type definition	Optional: Unit of the value
Value	simpleType short refer to Annex C.6 for the type definition	Value which has to be multiplied

8.5.2.8 NotificationType

[V2G2-280] This optional message element is included in the header of a V2G message and allows to notify the receiving entity about a parsing error or any other error related to the decoding of a V2G message. The message elment shall be implemented according to Figure 62.

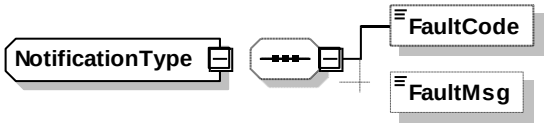


Figure 62 — Schema Diagram – NotificationType

[V2G2-281] The message element shall be used as defined in Table 67.

Table 67 — Semantics and type definition for NotificationType

Element Name	Type	Semantics
FaultCode	simpleType: faulCodeType enumeration refer to Annex C.6 for the type definition	Identifies a parsing error or any other fault related to the decoding of the V2G message.
FaultMsg	simpleType string (max length: 64) refer to Annex C.6 for the type definition	Optional Element: Includes fault related information which may be used for fault analysis.

8.5.2.9 PaymentOptionsType

[V2G2-282] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 63.

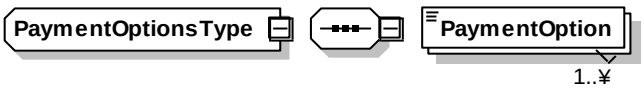


Figure 63 — Schema Diagram – PaymentOptionsType

[V2G2-283] The message element shall be used as defined in Table 68.

Table 68 — Semantics and type definition for PaymentOptionsType

Element Name	Type	Semantics
PaymentOption	simpleType: paymentOptionType enumeration refer to Annex C.6 for the type definition	This type includes the list of payment options an SECC offers to the EVCC indicating what method could be choosen to pay for the services. The EVCC can only select one payment method for all services used by the EVCC.

8.5.2.10 ChargingProfileType

[V2G2-284] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 64.

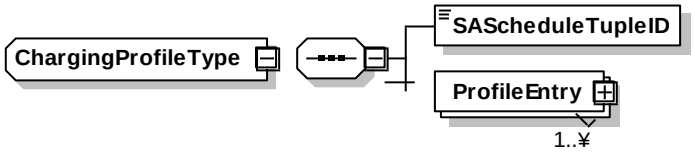


Figure 64 — Schema Diagram – ChargingProfileType

[V2G2-606] The message element shall be used as defined in Table 69.

Table 69 — Semantics and type definition for ChargingProfileType

Element Name	Type	Semantics
SAScheduleTupleID	simpleType: SAIDType short refer to Annex C.6 for the type definition	Unique identifier within a charging session referring to the selected SAScheduleTuple element from the SAScheduleListType (see subclause 8.5.2.12).
ProfileEntry	complexType: ProfileEntryType, refer to subclause 8.5.2.11	Element used for encapsulating an individual charging profile entry of the charge schedule

[V2G2-285] The value of the SAScheduleTupleID element shall be equal to one of the values of the SAScheduleTupleID elements in the list of SAScheduleTuple elements (see subclause 8.5.2.13) provided with the Charge Parameter Discovery Response message (see subclause 8.4.1.8.3).

[V2G2-286] The SAScheduleTupleID element shall identify the selected SAScheduleTuple element (see subclause 8.5.2.13) in the list of SAScheduleTuple elements (see subclause 8.5.2.12) provided in the Charge Parameter Discovery Response message (see subclause 8.4.1.8.3).

[V2G2-287] The number of ProfileEntry elements in the ChargingProfileType shall be limited to 24.

8.5.2.11 ProfileEntryType

[V2G2-288] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 65.

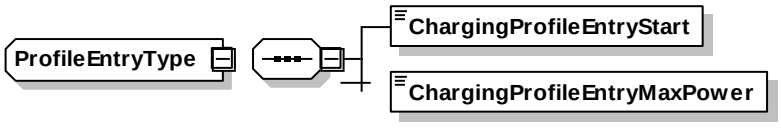


Figure 65 — Schema Diagram – ProfileEntryType

[V2G2-607] The message element shall be used as defined in Table 70.

Table 70 —Semantics and type definition for ProfileEntryType

Element Name	Type	Semantics
ChargingProfileEntryStart	simpleType unsignedInt refer to Annex C.6 for the type definition	Time when chargingProfileEntry starts to be valid. Offset in seconds from NOW.
ChargingProfileEntryMaxPower	simpleType: PMaxType short refer to Annex C.6 for the type definition	Maximum power in Watt consumed by the EV within the current charging profile entry (beginning from ChargingProfileEntryStart)

[V2G2-289] The value of the ChargingProfileEntryStart element shall be defined as point in time when this element of ProfileEntryType starts to be active.

[V2G2-290] The value of the next ChargingProfileEntryStart element in the list of elements of ProfileEntryType (see subclause 8.5.2.11) shall be defined as point in time when this element of ProfileEntryType becomes inactive.

NOTE [V2G2-289] and [V2G2-290] define the period of time an element of ProfileEntryType is active.

[V2G2-291] The last element in the list of elements of type ProfileEntryType is active until the list is updated according to [V2G2-305].

[V2G2-292] The value of the ChargingProfileEntryMaxPower element shall be defined as maximum power in Watts consumed by the EV within the active period of an element of type ProfileEntryType.

[V2G2-293] The values of the ChargingProfileEntryMaxPower element shall be equal to or smaller than the limits in respective elements of the PMaxScheduleType (see subclause 8.5.2.14) provided in the Charge Parameter Discovery Response message.

8.5.2.12 SAScheduleListType

[V2G2-294] The Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 66.

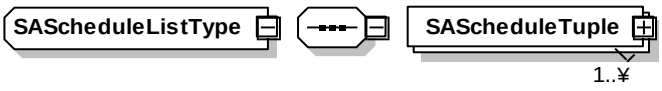


Figure 66 — Schema Diagram – SAScheduleListType

[V2G2-608] The message element shall be used as defined in Table 71.

Table 71 — Semantics and type definition for SAScheduleListType

Element Name	Type	Semantics
SAScheduleTuple	complexType: SAScheduleTupleType refer to subclause 8.5.2.13	Includes several tuples of schedules from secondary actors

[V2G2-295] The number of SAScheduleTuple elements in the SAScheduleListType shall be limited to 3.

[V2G2-296] The EVCC may implement a mechanism to compare different SAScheduleTuple elements in order to optimize the charge schedule considering any given kind of cost according to [V2G2-323] and [V2G2-337].

[V2G2-297] The first SAScheduleTuple element in the SAScheduleListType shall be defined as default SASchedule.

[V2G2-298] If the EVCC is not capable of comparing different SAScheduleTuple elements or comparison fails, the EVCC shall choose the default SAScheduleTuple according to [V2G2-297].

8.5.2.13 SAScheduleTupleType

[V2G2-299] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 67.

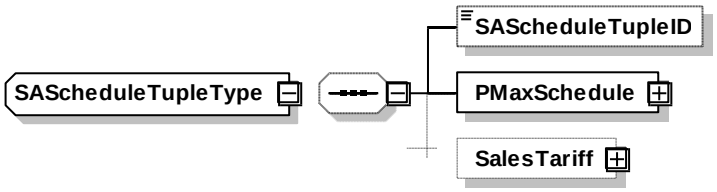


Figure 67 — Schema Diagram – SAScheduleTupleType

[V2G2-609] The message element shall be used as defined in Table 72.

Table 72 — Semantics and type definition for SAScheduleTupleType

Element Name	Type	Semantics
SAScheduleTupleID	simpleType: SAIDType short refer to Annex C.6 for the type definition	Unique identifier within a charging session for a SAScheduleTuple element
PMaxSchedule	complexType: PMaxScheduleType refer to subclause 8.5.2.14	Encapsulating element describing all relevant details for one PMaxSchedule from the Secondary Actor
SalesTariff	complexType: SalesTariffType refer to subclause 8.5.2.16	Optional: Encapsulating element describing all relevant details for one SalesTariff from the Secondary Actor

[V2G2-300] The SAScheduleTupleID element shall be unique within all SAScheduleTuple elements in the SAScheduleListType and uniquely identifies a tuple of PMaxSchedule and SalesTariff elements during the entire charging session.

[V2G2-301] The SECC shall provide a PMaxSchedule element based upon the limits of the local installation if no secondary actor provides a grid schedule.

[V2G2-303] The Secondary Actor(s) shall provide PMaxSchedule (refer to 8.5.2.14) and SalesTariff (see 8.5.2.16) elements covering the period of time indicated by the EVCC in the message element DepatureTime of the Charge Parameter Discovery Request message.

[V2G2-304] If the EVCC did not provide an DepatureTime Target Setting (refer to subclause 8.4.1.8.2 and 8.5.3.2), the Secondary Actor(s) shall provide PMaxSchedule and SalesTariff elements covering at least 24 hours presuming requirement [V2G2-312] and [V2G2-320] are still fulfilled.

[V2G2-305] If the number of SalesTariffEntry elements in the SalesTariff provided by the Secondary Actor is not covering the entire time period until DepatureTime, the Target Setting EAmount (refer to

subclause 8.4.1.8.2 and 8.5.3.2) has not been met and the communication session has not been finished, it is in the responsibility of the EVCC to request a new element of type SAScheduleListType as soon as the last SalesTariffEntry element (entry no. 12) becomes active by sending a new Charge Parameter Discovery Request message.

[V2G2-306] If the number of SalesTariffEntry elements provided by the Secondary Actor is not covering the entire time period until DepatureTime, it is in the responsibility of the EVCC to optimize the schedule based on the available information.

NOTE The algorithm for optimizing the charging profile is out of scope of this standard.

[V2G2-307] The Secondary Actor shall sign the message element SalesTariff.

[V2G2-308] The SECC shall not change the signature attached to the message element SalesTariff when receiving tariff information from the Secondary Actor.

NOTE This presumes that the data structure used by the Secondary Actor for the provisioning of the SalesTariff is identical with the data structure defined in this standard.

[V2G2-309] The SECC shall 'copy' the signature value received from the SA and transmit this value in the header of the Charge Parameter Discovery Response message (see subclause 8.4.1.8.3).

8.5.2.14 PMaxScheduleType

[V2G2-310] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 68.

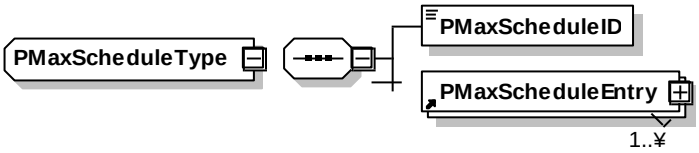


Figure 68 — Schema Diagram – PMaxScheduleType

[V2G2-610] The message element shall be used as defined in Table 73.

Table 73 — Semantics and type definition for PMaxScheduleType

Element Name	Type	Semantics
PMaxScheduleID	simpleType: SAIDType short refer to Annex C.6 for the type definition	Unique identifier for an element of type PMaxScheduleType across a charging session.
PMaxScheduleEntry	complexType: PMaxScheduleEntryType refer to subclause 8.5.2.15	List of PMaxScheduleEntry elements

[V2G2-311] The value of the PMaxScheduleID element shall uniquely identify an element of type PMaxScheduleType during the entire charging session.

[V2G2-312] The number of PMaxScheduleEntry elements in the PMaxScheduleType shall be limited to 12.

8.5.2.15 PMaxScheduleEntryType

[V2G2-313] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 69.

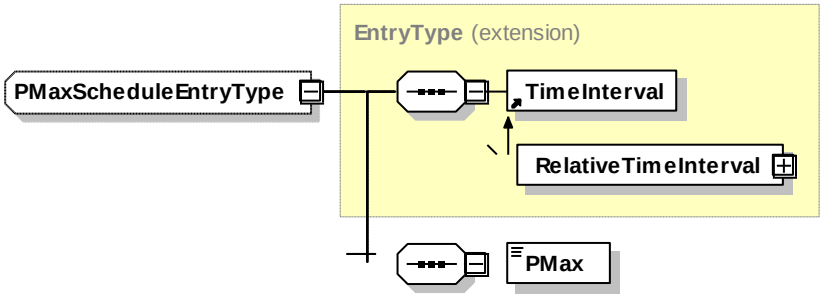


Figure 69 — Schema Diagram – PMaxScheduleEntryType

[V2G2-611] The message element shall be used as defined in Table 74.

Table 74 — Semantics and type definition for PMaxScheduleEntryType

Element Name	Type	Semantics
RelativeTimeInterval	complexType: RelativeTimeInterval substitutes abstract element TimeInterval. refer to subclause 8.5.2.18	Optional: Extends the TimeIntervalType and defines the time interval the PMaxScheduleEntry is valid for based upon relative times.
PMax	simpleType: PMaxType short refer to Annex C.6 for the type definition	Defines maximum amount of power to be drawn from the EVSE outlet the vehicle is connected to.

[V2G2-314] Any extension of the TimeInterval element shall define the active period of time for the respective parent element of type PMaxScheduleEntryType.

[V2G2-315] The PMax element shall define the maximum amount of power to be drawn from the EVSE outlet when the element of type PMaxScheduleEntryType is active.

8.5.2.16 SalesTariffType

The sales tariff table in this standard provides means for optimizing the charge schedule in the EVCC based upon cost-based information. This cost information is provided within the sales tariff information. The term “cost” refers to any given kind of cost defined in this standard (refer to subclause 8.5.2.16). However, this version of the standard does not allow to provide absolute price information or an advice of charge to the EV user. This functionality may be implemented through other means being out of scope of ISO/IEC 15118.

[V2G2-316] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 70.

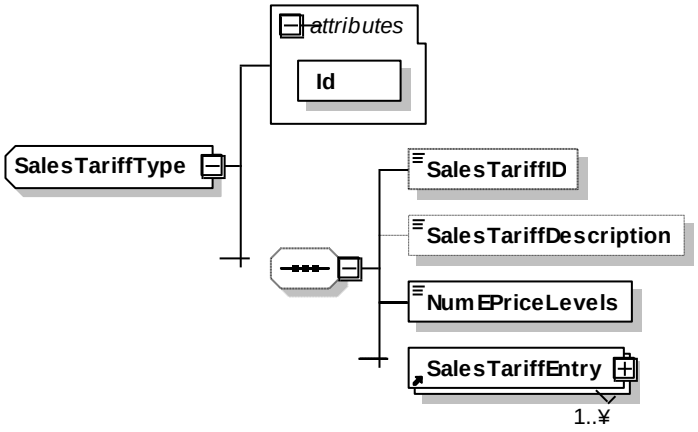


Figure 70 — Schema Diagram – SalesTariffType

[V2G2-612] The message element shall be used as defined in Table 75.

Table 75 — Semantics and type definition for SalesTariffType

Element Name	Type	Semantics
SalesTariffID	simpleType: SAIDType short refer to Annex C.6 for the type definition	SalesTariff identifier used to identify one sales tariff
SalesTariffDescription	simpleType: tariffDescriptionType string (max. length: 32) refer to Annex C.6 for the type definition	Optional: A human readable title/short descirption of the sales tariff e.g. for HMI display purposes
NumEPriceLevels	simpleType unsignedByte refer to Annex C.6 for the type definition	Defines the maximum number of distinct price levels across all provided tariffs (e.g. on-peak, mid-peak, off-peak means three different price levels).
SalesTariffEntry	complexType: SalesTariffEntryType refer to subclause 8.5.2.17	Encapsulating element describing all relevant details for one time interval of the SalesTariff.

[V2G2-317] The value of the SalesTariffID element shall uniquely identify an element of type SalesTariffType during the entire charging session.

[V2G2-318] The NumEPriceLevels element shall be defined as the overall number of distinct EPriceLevels across all SalesTariff elements in the list of SAScheduleTuple.

NOTE This allows cost optimization of the charge process as well as comparison between different tariffs based on the element of type EPriceLevel (see subclause 8.5.2.17).

EXAMPLE Different price levels for on-peak, mid-peak, off-peak intervals shall result in three different price levels (NumEPriceLevels set to 3). A flatrate tariff shall result in only one price level (NumEPriceLevels set to 1).

[V2G2-319] If no price information is available (e.g. payment handled by external means) but an element of type SalesTariffType is still present the value of the NumEPriceLevels element shall be set to 0.

[V2G2-320] The number of SalesTariffEntry elements in the SalesTariffType shall be limited to 12.

8.5.2.17 SalesTariffEntryType

[V2G2-321] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 71.

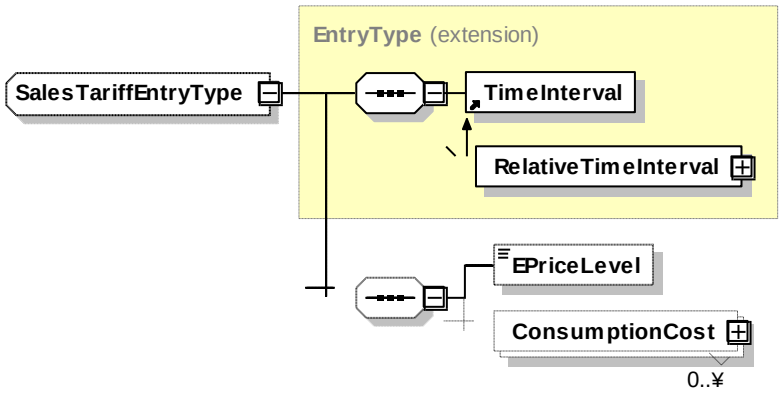


Figure 71 — Schema Diagram – SalesTariffEntryType

[V2G2-613] The message element shall be used as defined in Table 88.

Table 76 — Semantics and type definition for SalesTariffEntryType

Element Name	Type	Semantics
RelativeTimeInterval	complexType: RelativeTimeInterval substitutes abstract element TimeInterval. refer to subclause 8.5.2.18	Extends the TimeIntervalType and defines the time interval the SalesTariffEntry is valid for based upon relative times.
EPriceLevel	simpleType unsignedByte refer to Annex C.6 for the type definition	Defines the price level of this SalesTariffEntry (refering to NumEPriceLevels). Small values for the EPriceLevel represent a cheaper TariffEntry. Large values for the EPriceLevel represent a more expensive TariffEntry.
ConsumptionCost	complexType: SalesTariffEntryType refer to subclause 8.5.2.19	Optional: Defines additional means for further relative price information and/or alternative costs.
NOTE The definition of EPriceLevel allows simple price-based optimization by the EVCC without any ConsumptionCost element.		

[V2G2-322] The TimeInterval element of type IntervalType shall define the active period of time for an element of type SalesTariffEntryType.

[V2G2-323] The EPriceLevel element shall be used to allow for price optimized charge scheduling in addition to power limits.

[V2G2-324] The value of the EPriceLevel element shall be equal to or smaller than NumEPriceLevels in the SalesTariffType (see subclause 8.5.2.16).

[V2G2-325] The EPriceLevel shall adhere to the following rule: The smaller the values for EPriceLevel, the cheaper is the actual price level in the respective Interval. The higher the values for EPriceLevel, the more expensive is the actual price level in the respective Interval.

[V2G2-326] The number of ConsumptionCost elements in the SalesTariffEntryType shall be limited to 3.

8.5.2.18 RelativeTimeIntervalType

[V2G2-327] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 72.

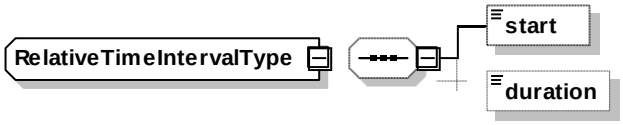


Figure 72 — Schema Diagram – RelativeTimeIntervalType

[V2G2-614] The message element shall be used as defined inTable 77.

Table 77 — Semantics and type definition for RelativeTimeIntervalType

Element Name	Type	Semantics
duration	simpleType unsignedInt refer to Annex C.6 for the type definition	Optional: Duration of the interval, in seconds.
start	simpleType unsignedInt refer to Annex C.6 for the type definition	Start of the interval, in seconds from NOW.

[V2G2-328] The value of the start element shall be defined in seconds from NOW.

[V2G2-329] The value of the start element shall simultaneously define the start time of this interval and the stop time of the previous interval.

[V2G2-330] The value of the duration element shall be defined as period of time in seconds.

[V2G2-331] The duration element shall only be used for the last interval of the SalesTariff (see subclause 8.5.2.16) or PMaxSchedule (see subclause 8.5.2.14).

NOTE It indicates the end of the coverage time of the delivered SalesTariff or PMaxSchedule information.

8.5.2.19 ConsumptionCostType

[V2G2-332] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 73.

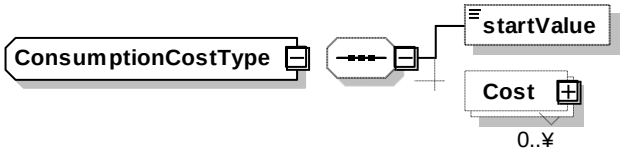


Figure 73 — Schema Diagram – ConsumptionCostType

[V2G2-615] The message element shall be used as defined in Table 78.

Table 78 — Semantics and type definition for ConsumptionCostType

Element Name	Type	Semantics

Cost	complexType: CostType refer to subclause 8.5.2.20	Optional: Encapsulating element describing all relevant cost details for this consumption block in this TariffEntry.
startValue	simpleType unsignedInt refer to Annex C.6 for the type definition	The lowest level of consumption that defines the starting point of this consumption block. The block interval extends to the start of the next interval.

[V2G2-333] Refering to [V2G2-326] the first element in the list of elements of type ConsumptionCostType shall always start with the startValue set to 0.

[V2G2-334] The maximum number of Cost elements in the ConsumptionCostType shall be limited to the number of different enumeration types defined in the costKindType (refer to table Table 80).

8.5.2.20 CostType

[V2G2-335] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 74.

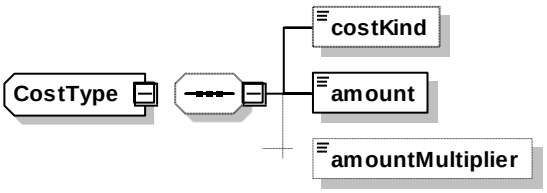


Figure 74 — Schema Diagram – CostType

[V2G2-616] The message element shall be used as defined in Table 79 and Table 80.

Table 79 — Semantics and type definition for CostType

Element Name	Type	Semantics
amount	simpleType unsignedInt refer to Annex C.6 for the type definition	The estimated or actual cost, per unit of measurement (typically kWh)
amountMultiplier	simpleType unitMultiplierType (min value: -3 ; max value: 3) refer to Annex C.6	Optional: Decimal shift operator for amount
costKind	simpleType: CostKindType enumeration refer to Annex C.6 for the type definition	The kind of cost referred to in the message element amount (refer to Table 80 for the semantics description of individual enum values defined for this type).

Table 80 — Semantics for CostKindType

CostKindType	Semantics
relativePricePercentage	Relative price, as percentage relative to a common base between all tariffs.
RenewableGenerationPercentage	Renewable generation, as a percentage of overall generation.
CarbonDioxideEmission	Carbon Dioxide emissions, in grams per unit of measure.

[V2G2-336] The unit of measure for any kind of cost shall be kWh.

[V2G2-337] The applicable costs per unit of measure according to [V2G2-336] shall be defined with the amount element.

[V2G2-338] Within one CostType the amount element shall always refer to its respective costKind element.

[V2G2-339] For relative kinds of costs (refer to Table 80) the amount element shall be defined as percentage relative to the highest cost of the same kind.

[V2G2-340] For absolute kinds of costs (refer to Table 80) the amount element shall be defined as absolute value of the cost.

[V2G2-341] The amountMultiplier shall be defined as the exponent to base 10 for a given amount.

EXAMPLE If the multiplier is 2 and the transmitted value 1, the value to be calculated is 1*10^2=100.

[V2G2-342] For comparison between different tariffs the applicable costs shall either be based upon absolute costs or for relative costs shall be normalized to a common base across all SalesTariff elements. Refer to costKindType as defined in Table 80 for all kinds of costs being defined in this standard.

NOTE If this precondition is not satisfied, the EVCC is not able to compare different tariffs.

8.5.2.21 ServiceParameterListType

[V2G2-343] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 75.

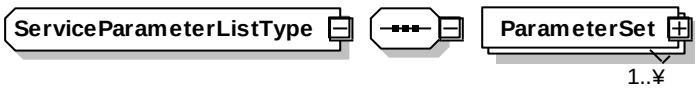


Figure 75 — Schema Diagram – ServiceParameterListType

[V2G2-344] The message element shall be used as defined in Table 81.

Table 81 —Semantics and type definition for ServiceParameterListType

Element Name	Type	Semantics
ParameterSet	complexType: ParameterSetType refer to subclause 8.5.2.22	Defines parameters for a specific serviceID received from the SECC in the ServiceDiscoveryRes message.

8.5.2.22 ParameterSetType

[V2G2-345] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 76.

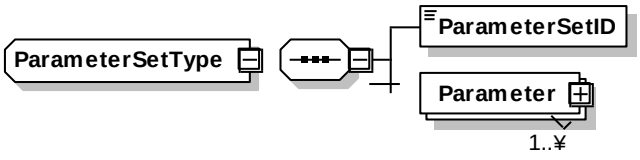


Figure 76 — Schema Diagram – ParameterSetType

[V2G2-346] The message element shall be used as defined in Table 82.

Table 82 —Semantics and type definition for ParameterSetType

Element Name	Type	Semantics
ParameterSetID	simpleType: short refer to Annex C.6 for the type definition	This element is used to select a specific parameter set for a specific ServideID when selection a service using the ServicePaymentSelectionReq message.
Parameter	complexType: ParameterType refer to subclause 8.5.2.23	This element is used by the SECC to indicate which service specific parameters can be selected for a certain service using the ParameterSetID.

8.5.2.23 ParameterType

[V2G2-347] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 77.

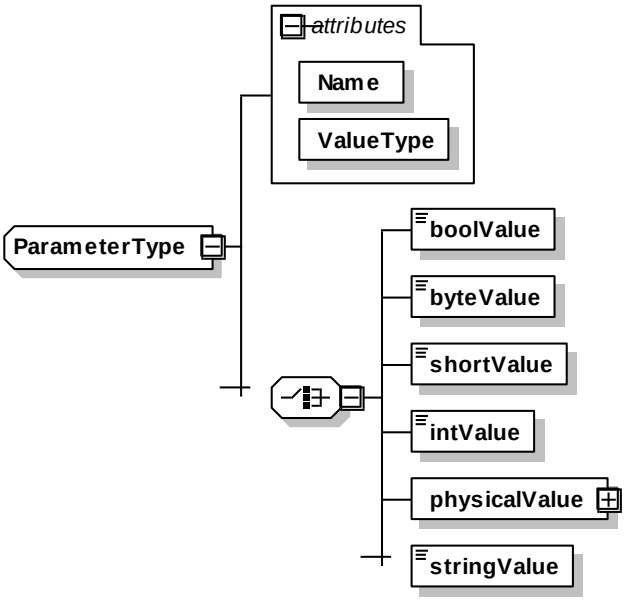


Figure 77 — Schema Diagram – ParameterType

[V2G2-348] The message element shall be used as defined in Table 83.

Table 83 —Semantics and type definition for ParameterType

Element Name	Type	Semantics
Name	simpleType: string refer to Annex C.6 for the type definition	This element is used to indicate the name of the parameter..
ValueType	simpleType: valueType enumeration refer to Annex C.6 for the type definition	This element is used to indicate the value for the parameter indicated by the element Name.

8.5.2.24 SelectedServiceListType

[V2G2-349] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 78.

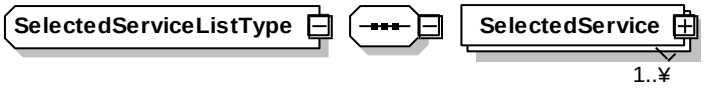


Figure 78 — Schema Diagram – SelectedServiceListType

[V2G2-350] The message element shall be used as defined in Table 84.

Table 84 —Semantics and type definition for SelectedServiceListType

Element Name	Type	Semantics
SelectedService	complexType: SelectedServiceType refer to subclause 8.5.2.25	This element is used to indicate the selected ServiceID and the associated parameterSet.

8.5.2.25 SelectedServiceType

[V2G2-351] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 79.

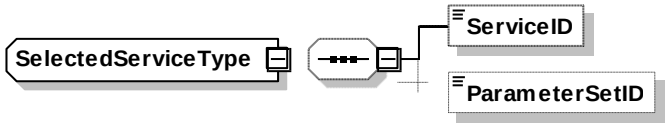


Figure 79 — Schema Diagram – SelectedServiceType

[V2G2-352] The message element shall be used as defined in Table 85.

Table 85 —Semantics and type definition for SelectedServiceType

Element Name	Type	Semantics
ServiceID	simpleType: unsignedShort refer to Annex C.6 for the type definition	Unique identifier of the service
ParameterSetID	simpleType: short refer to Annex C.6 for the type definition	This element is used to select a specific parameter set for a specific ServideID when selection a service using the ServicePaymentSelectionReq message.

8.5.2.26 SubCertificatesType

[V2G2-353] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 80.

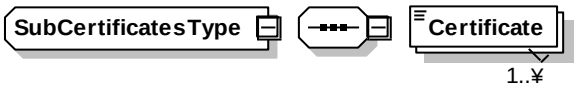


Figure 80 — Schema Diagram – SubCertificatesType

[V2G2-354] The message element shall be used as defined in Table 86.

Table 86 —Semantics and type definition for SubCertificatesType

Element Name	Type	Semantics
Certificate	simpleType: certificateType base64Binary (max length: 1200) refer to Annex C.6 for the type definition	An x.509v3 Certificate (the “client” certificate)

[V2G2-656] The number of Certificates in the SubCertificates shall not exceed 4.

8.5.2.27 ListOfRootCertificateIDsType

[V2G2-355] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 81.

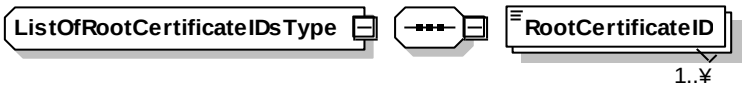


Figure 81 — Schema Diagram – ListOfRootCertificateIDsType

[V2G2-356] The message element shall be used as defined in Table 87.

Table 87 —Semantics and type definition for ListOfRootCertificateIDsType

Element Name	Type	Semantics
RootCertificateID	simpleType rootCertificateIDType string (max length: 40) refer to Annex C.6 for the type definition	This message element uniquely identifies a V2G Root Certificate installed in the EVCC.

[V2G2-357] The number of RootCertificateIDs in the ListOFRootCertificateIDs shall not exceed 20.

8.5.3 AC

8.5.3.1 AC_EVSEStatusType

[V2G2-358] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the SECC and the EVCC shall implement this type as defined in Table 98 and according to

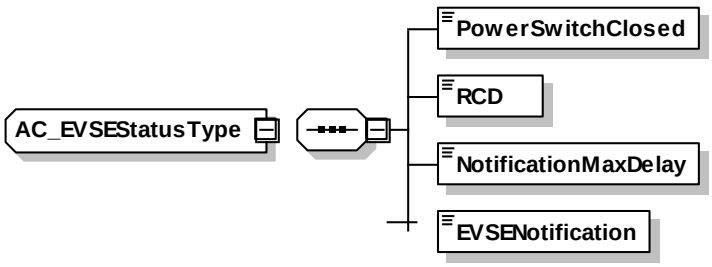


Figure 82.

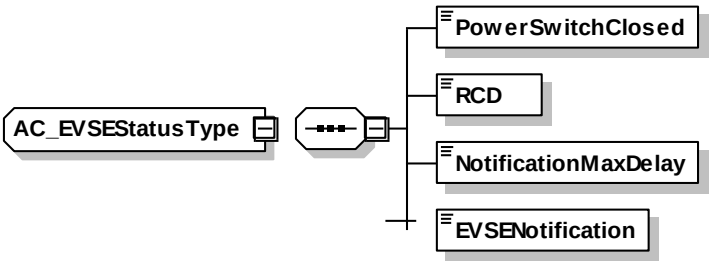


Figure 82 — Schema Diagram – AC_EVSEStatusType

[V2G2-359] The message element shall be used as defined in Table 88.

Table 88 —Semantics and type definition for AC_EVSEStatusType

Element Name	Type	Semantics
PowerSwitchClosed	simpleType boolean refer to Annex C.6 for the type definition	Indicates if the power switch in the EVSE is open or closed.). If PowerSwitchClosed is equal to true, the contactors in the EVSE are closed. If PowerSwitchClosed is equal to fals, the contactors in the EVSE are open. This status flag is for informational purpose only.
RCD	simpleType boolean refer to Annex C.6 for the type definition	Indicates the current status of the Residual Current Device (RCD). If RCD is equal to true, the RCD has detected an error. If RCD is equal to false, the RCD has not detected an error. This status flag is for informational purpose only.
NotificationMaxDelay	simpleType unsignedInt refer to Annex C.6 for the type definition	The SECC uses the NotificationMaxDelay element in the EVSEStatus to indicate the time until it expects the EVCC to react on the action request indicated in EVSENotification. If the target time is not in the future, the EVCC is expected to perform the action immediately.
EVSENotification	simpleType EVSENotificationType enumeration refer to Annex C.6 for the type definition	This value is used by the SECC to influence the behaviour of the EVCC. The EVSENotification contains an action that the SECC wants the EVCC to perform. The requested action is expected by the EVCC until the time provided in NotificationMaxDelay. If the target time is not in the future, the EVCC is expected to perform the action immediately. During normal operation the value of EVSENotification is set to “none”.

NOTE The behaviour of the EVSE and EV after a shutdown (shutdown time unequal to zero) has been performed is not in the scope of this document. This rather depends on the specific system implementation.

EXAMPLE The implementation of a charging session which continues after a shutdown has ouccured could allow 3 retries to establish a charging session before the SECC would switch into a failure or maintenance mode.

8.5.3.2 AC_EVChargeParameterType

[V2G2-360] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 83.

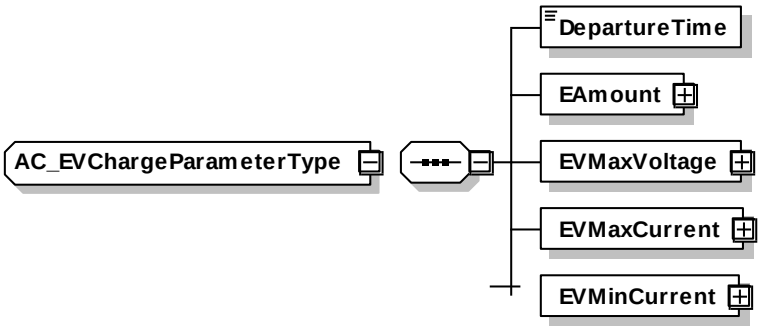


Figure 83 — Schema Diagram – AC_EVChargeParameterType

[V2G2-361] The message element shall be used as defined in Table 89.

Table 89 — Semantics and type definition for AC_EVChargeParameterType

Element Name	Type	Semantics
DepartureTime	simpleType unsignedInt refer to Annex C.6 for the type definition	This element is used to indicate when the vehicle intends to finish the charging process. Offset in seconds from the point in time of sending this message. A value of zero (0) indicates that the charging process shall be finished as fast as possible.
EAmount	complexType PhysicalValueType refer to subclause 8.5.2.7	Amount of Energy required by the EV unil the DepartureTime has been reached or the HV Battery's SOC is at 100%. This might include the amount of energy the EV consumes for other vehicle features than solely charging the HV Battery.
EVMaxVoltage	complexType PhysicalValueType refer to subclause 8.5.2.7	Maximum voltage supported by the EV. This is the voltage measured between one phase and neutral.
EVMaxCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	Maximum current supported by the EV per phase.

Element Name	Type	Semantics
EVMinCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	EVMin Current is used to indicate to the SECC that charging below this minimum is not energy/cost efficient for the EV. It is recommended that the SECC considers this value during the target setting process (e.g. sale tariff table should account for this value). However, if there is physical limitations or limitations indicated by the PWM signal these limitations overwrite the EVMinCurrent the EV indicated. It is implementation specific whether a vehicle chooses not charge if the EVMinCurrent is higher than the physical limitations for efficiency reasons.

8.5.3.3 AC_EVSEChargeParameterType

[V2G2-362] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 84.

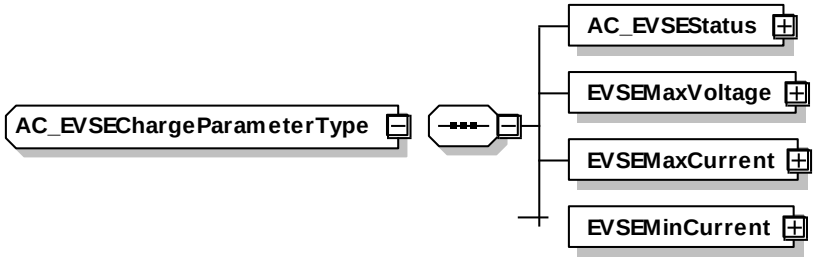


Figure 84 — Schema Diagram – AC_EVSEChargeParameterType

[V2G2-363] The message element shall be used as defined in Table 90.

Table 90 —Semantics and type definition for AC_EVSEChargeParameterType

Element Name	Type	Semantics
AC_EVSEStatus	complexType AC_EVSEStatusType refer to subclause 8.5.3.1.	Current status of the EVSE
EVSEMaxVoltage	complexType PhysicalValueType refer to subclause 8.5.2.7	Line voltage supported by the EVSE. This is the voltage measured between one phase and neutral. If the EVSE supports multiple phase charging the EV might easily calculate the voltage between phases.
EVSEMaxCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	Maximum line current the EVSE can provide considering the power rating of the EVSE including the charge cord limitations.
EVSEMinCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	Minimum line current the EVSE can provide

8.5.4 DC

8.5.4.1 DC_EVSEStatusType

[V2G2-364] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 85.

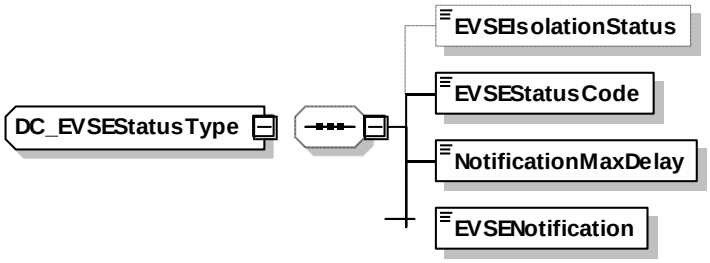


Figure 85 — Schema Diagram – DC_EVSEStatusType

[V2G2-365] The message element shall be used as defined in Table Table 91.

Table 91 — Semantics and type definition for DC_EVSEStatusType

Element Name	Type	Semantics
NotificationMaxDelay	simpleType unsignedInt refer to Annex C.6 for the type definition	The SECC uses the NotificationMaxDelay element in the EVSEStatus to indicate the time until it expects the EVCC to react on the action request indicated in EVSENotification. If the target time is not in the future, the EVCC is expected to perform the action immediately.

Element Name	Type	Semantics
EVSENotification	simpleType EVSENotificationType enumeration refer to Annex C.6 for the type definition	This value is used by the SECC to influence the behaviour of the EVCC. The EVSENotification contains an action that the SECC wants the EVCC to perform. The requested action is expected by the EVCC until the time provided in NotificationMaxDelay. If the target time is not in the future, the EVCC is expected to perform the action immediately. During normal operation the value of EVSENotification is set to “none”.
EVSEIsolationStatus	simpleType: isolationLevelType enumeration refer to Annex C.6 for the type definition	Optional: Indicates the isolation condition (result of the isolation monitoring).
DC_EVSEStatusCode	simpleType: DC_EVSEStatusCodeType enumeration refer to Annex C.6 for the type definition	Indicates if the internal state of the EVSE. Refer to Table 92 for details.

[V2G2-366] The EVCC shall use the EVSEStatusCodes as described in Table 92 when any of the DC Message Set(s) as defined in subclause 8.6.2 has been selected.

Table 92 — Semantics and type definition for EVSEStatusCodeType

Element Name	Semantics
EVSE_NotReady	Not authorized, StandBy, on maintenance, …
EVSE_Ready	Charging procedure is running
EVSE_Shutdown	Charger Shutdown, Customer Initiated Shutdown
EVSE_UtilityInterruptEvent	Utility Interrupt Event, Utility or Equipment operator has requested a temporary reduction in load.
EVSE_IsolationMonitoringActive	After the Charging Station has confirmed HV isolation internally, it will remain in this state until the cable isolation integrity is checked
EVSE_EmergencyShutdown	Charging System Incompatibility, Emergency Shutdown or 'E-Stop' button pressed at charging station.
EVSE_Malfunction	A non-recoverable charger fault has occurred (Isolation Failure, …)
Reserved 8-C	Reserved by ISO/IEC for future use.

8.5.4.2 DC_EVStatusType

[V2G2-367] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 86.

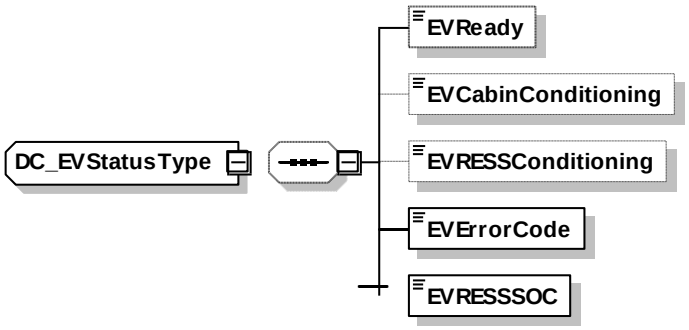


Figure 86 — Schema Diagram – DC_EVStatusType

[V2G2-368] The message element shall be used as defined in Table 93.

Table 93 — Semantics and type definition for DC_EVStatusType

Element Name	Type	Semantics
EVReady	simpleType: boolean refer to Annex C.6 for the type definition	If set to TRUE, the EV is ready to charge.
EVCabinConditioning	simpleType: boolean refer to Annex C.6 for the type definition	Optional: Vehicle Cabin Conditioning, The EV is using energy from the DC supply to heat or cool the passenger compartment.
EVRESSConiditioning	simpleType: boolean refer to Annex C.6 for the type definition	Optional: Vehicle RESS Conditioning, The vehicle is using energy from the DC charger to condition the RESS to a target temperature.
EVErrorCode	simpleType: DC_EVErrorCodeType enumeration refer to Annex C.6 for the type definition	Indicates the EV internal status.Refer to Table 94 for details.
EVRESSSOC	simpleType: percentValueType byte (range: 0-100) refer to Annex C.6 for the type definition	State of charge of the EV's battery (RESS)

[V2G2-369] The EVCC shall use the EVErrorCodes as described in Table 94 when any of the DC Message Set(s) as defined in subclause 8.6.2 has been selected.

Table 94 — Semantics and type definition for EVErrorCodeType

Element Name	Semantics
NO_ERROR	Default value, when EVCC has no Error detected
FAILED_RESSTemperatureInhibit	Battery Temperature Inhibit, Battery too hot/cold to accept charge
FAILED_EVShiftPosition	Vehicle Shift Position, Vehicle is not in Park
FAILED_ChargerConnectorLockFault	Charger Connector Lock Fault, Vehicle has not detected the Charge cord connector locked into the inlet or failure where connector cannot be unlocked from the charging inlet.
FAILED_EVRESSMalfunction	Vehicle RESS Malfunction, Any non-recoverable fault or error condition of the Vehicle RESS.

FAILED_ChargingCurrentdifferential	Charging Current Differential, Indication that vehicle has stopped the charging session after detecting that the charging station is not able to maintain the current request.
FAILED_ChargingVoltageOutOfRange	Charging voltage out of range, Indication that vehicle has stopped the charging session after detecting that the RESS is either under or above normal operating voltage range.
Reserved A-C	Reserved by ISO/IEC for future use.
FAILED_ChargingSystemIncompatibility	Charging System Incompatibility, If the vehicle determines that the charging station is incompatible. This state is Optional, as an alternative, the vehicle can use 0x0 (not ready)
NoData	No Data, Only used when vehicle has not yet determined its operating state.

8.5.4.3 DC_EVChargeParameterType

[V2G2-370] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC and the EVCC shall implement this type as defined in Table 98 and according to Figure 87.

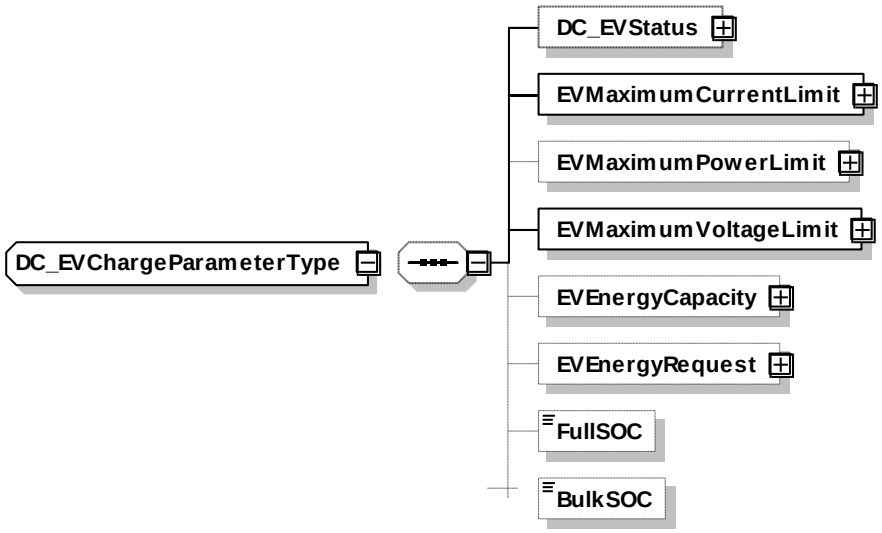


Figure 87 — Schema Diagram – DC_EVChargeParameterType

[V2G2-371] The message element shall be used as defined in Table 95.

Table 95 — Semantics and type definition for DC_EVChargeParameterType

Element Name	Type	Semantics
DC_EVStatus	complexType DC_EVStatusType refer to subclause 8.5.4.2	Current status of the EV
EVMaximumCurrentLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Maximum current supported by the EV
EVMaximumPowerLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Maximum power supported by the EV

Element Name	Type	Semantics
EVMaximumVoltageLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Maximum voltage supported by the EV
EVEnergyCapacity	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Maximum power capacity supported by the EV
EVEnergyRequest	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Amount of energy the EV requests from the EVSE.
FullSOC	simpleType: percentValueType byte (range: 0-100) refer to Annex C.6 for the type definition	Optional: SOC at which the EV considers the battery to be fully charged
BulkSOC	simpleType: percentValueType byte (range: 0-100) refer to Annex C.6 for the type definition	Optional: SOC at which the EV considers a fast charge process to end.

8.5.4.4 DC_EVSEChargeParameterType

[V2G2-372] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement this type as defined in Table 98 and according to Figure 88.

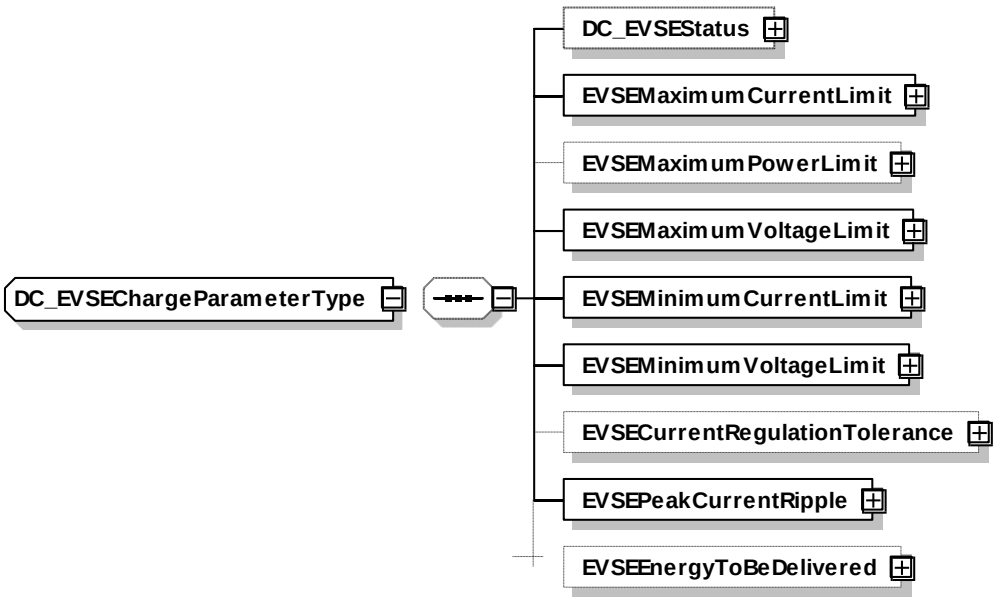


Figure 88 — Schema Diagram – DC_EVSEChargeParameterType

[V2G2-373] The message element shall be used as defined in Table 96.

Table 96 — Semantics and type definition for DC_EVSEChargeParameterType

Element Name	Type	Semantics

Element Name	Type	Semantics
DC_EVSEStatus	complexType DC_EVSEStatusType refer to subclause 8.5.4.1	Current status of the EVSE
EVSEMaximumCurrentLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Maximum current the EVSE can deliver
EVSEMaximumPowerLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Maximum power the EVSE can deliver
EVSEMaximumVoltageLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Maximum voltage the EVSE can deliver
EVSEMinimumCurrentLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Minimum current the EVSE can deliver with the expected accuracy
EVSEMinimumVoltageLimit	complexType PhysicalValueType refer to subclause 8.5.2.7	Minimum voltage the EVSE can deliver with the expected accuracy
EVSECurrentRegulationTolerance	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Absolute magnitude of the regulation tolerance of the EVSA
EVSEPeakCurrentRipple	complexType PhysicalValueType refer to subclause 8.5.2.7	Peak-to-peak magnitude of the current ripple of the EVSE
EVSEEnergyToBeDelivered	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Amount of energy to be delivered by the EVSE

8.5.4.5 DC_EVPowerDeliveryParameterType

[V2G2-374] Depending on the selected Message Set(s) as defined in subclause 8.6.2, the EVCC and the SECC shall implement this type as defined in Table 98 and according to Figure 89.

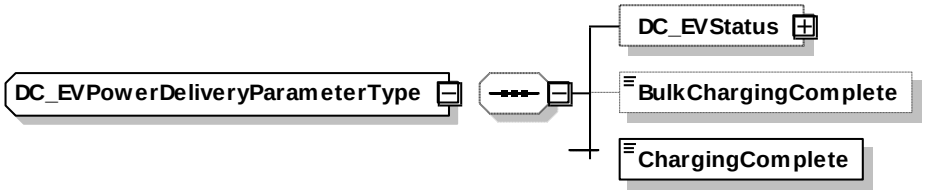


Figure 89 — Schema Diagram – DC_EVPowerDeliveryParameterType

[V2G2-375] The message element shall be used as defined in Table 97.

Table 97 — Semantics and type definition for DC_EVPowerDeliveryParameterType

Element Name	Type	Semantics
DC_EVStatus	complexType DC_EVStatusType refer to subclause 8.5.4.2.	Current status of the EV.
BulkChargingComplete	simpleType: boolean refer to Annex C.6 for the type definition	Optional: If set to TRUE, the EV indicates that bulk charge (approx. 80% SOC) is complete.
ChargingComplete	simpleType: boolean refer to Annex C.6 for the type definition	If set to TRUE, the EV indicates that full charge (100% SOC) is complete.

8.6 Identification modes and message set definitions

8.6.1 Overview

Part 1 of this standard defines use case elements covering different charging scenarios. Depending on a use case and a charging scenario only a subset of the messages and parameters defined in this document have to be supported and transmitted by the EVCC and the SECC, respectively.

This chapter defines the mandatory and optional messages and parameter with respect to charging scenarios using External Identification Means (EIM) and charging scenarios using plug and charge (PnC) for identification. In the following this is references as “identification mode”

Figure 91 shows the relation between the identification modes PnC and EIM with respect to the messages and parameters as defined in this document. While identification mode PnC may use all definitions in this standard EIM uses a subset of messages excluding the definitions for Plug and Charge support.

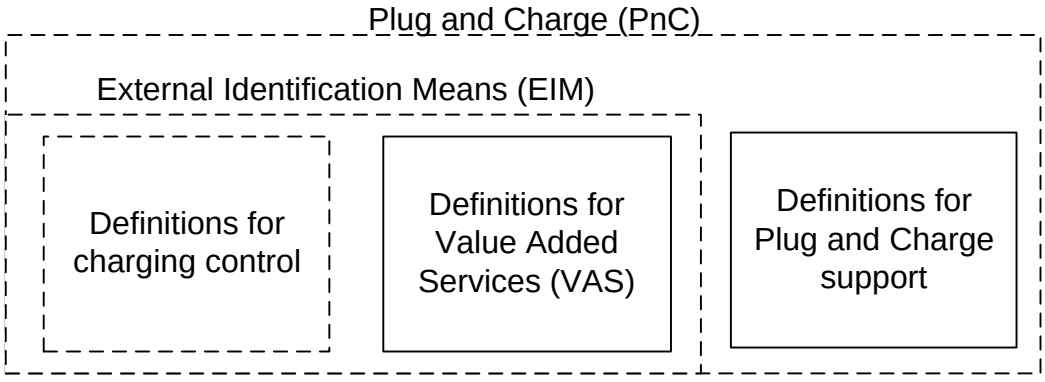


Figure 90 — Relation of identification modes and definitions for messages and parameter

NOTE Because the main difference between the identification modes PnC and EIM is the support of definitions for plug and charge, an ISO/IEC 15118 implementation for a charging scenario with identification mode PnC can also support the same charging scenario with identification mode EIM.

In the following, mandatory messages and parameters covering the use cases and charging scenarios defined in Part 1 are defined. Annex A gives a detailed mapping of the use case elements as defined in Part 1 and the mandatory V2G messages and parameters as defined in the following subclauses.

This document distinguishes between the mandatory/optional support of messages and parameters in the EVCC and the SECC and mandatory/optional definitions for parameters in the XML Schema definitions:

— Mandatory/optional support: The support of mandatory messages and parameters defines the subset of messages and parameters, an EVCC and an SECC must be able to process. This definition ensures a well known set of functionalities and therefore the compatibility with respect to a set of use cases.

— Mandatory/optional parameter definitions in the XML Schema: The definition of mandatory and optional parameters in the XML schema is derived from mandatory/optional support of parameters in the EVCC and the SECC for all use cases. As long as a parameter is mandatory in all use cases the parameter is defined as mandatory in the XML Schema. If a parameter is optional in at least on use case it is defined as optional in the XML schema. This enables the sending of messages with only the required parameters and omit parameters that are not mandatory for a use case.

NOTE The mandatory support of a message or parameter in one V2G entity does not necessarily mean that this message or parameter will be transmitted. E.g. if the support of a parameter is optional in the EVCC and the same parameter is defined to be mandatory in the SECC, the EVCC can choose to send this parameter. The mandatory support of this parameter in the SECC only ensures that it can be assumed that the SECC is able to process the value if sent by the EVCC. If an parameter is defined to be optional on both sides, this parameter can only be used if both sides support the parameter and it is transmitted.

NOTE The focus of the definition of mandatory and optional parameters in the XML schema is to optimize the transmission of V2G messages. In general it is not possible to derive the mandatory and optional parameters that must be supported in an EVCC and an SECC. E.g. if the support of 2 parameters is mandatory in the EVCC and the support in the SECC is only mandatory for one parameters, the SECC can decide to send only one parameter in a response message. For this use case the XML schema defines one parameter optional and the other mandatory. But the XML schema does not allow to decide at which side the support of the two parameters is mandatory or optional for a certain use case.

The XML Schema with all mandatory and optional definitions is described in subclause 8.3 and Annex C in detail. The following subclauses focus on the definition of mandatory and optional messages and parameters to be supported by the EVCC and the SECC Message Sets for the identification modes PnC and EIM.

A Message Set defines mandatory messages and parameters for the EVCC and the SECC covering one or multiple use case elements of Part 1 of this standard. An identification mode combines one or multiple mandatory and optional Message Sets covering a set of similar charging scenarios. An identification mode consist of at least one mandatory Message Set providing the basis for a specific set of charging scenarios. A identification mode may additionally define optional Message Sets which allow to extend the charging scenarios.

Figure 91 shows an overview of the mandatory and optional message sets for the identification modes. The identification mode External Identification Means (EIM) defines Messages and Parameters for charging scenarios with authorization outside the EV as defined in *Part 1*, subclause 7.5 "Identification, authentication and authorisation [D]". The identification mode Plug and Charge (PnC) defines Messages and Parameters for charging scenarios with authorization inside the EV as defined in *Part 1*, subclause 7.5 "Identification, authentication and authorisation [D]".

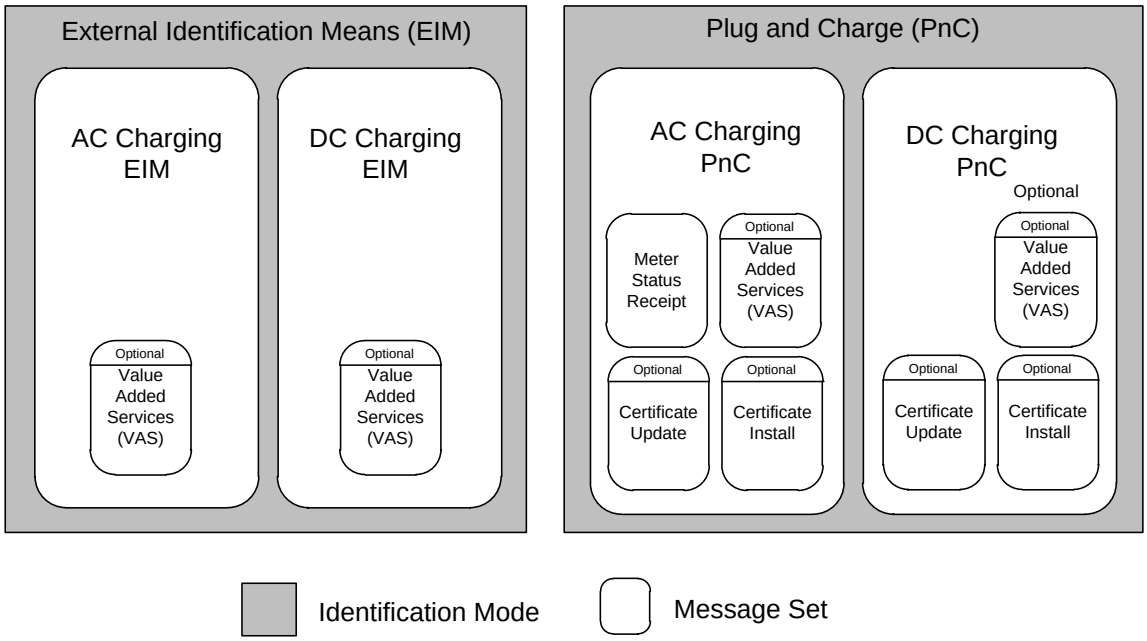


Figure 91 — Overview on Identification modes and Message Sets

8.6.2 Supported Message Sets

8.6.2.1 Overview

The EV manufacturer and the EVSE manufacturer can choose to support either one or multiple Message Sets shown in Figure 91. Table 98 defines all mandatory parts of the Message Sets as shown in Figure 91.

For the EVCC the following applies in general:

[V2G2-659] An EVCC shall support a message or parameter in a Message Set if it is marked with an "M" for the EVCC.

[V2G2-660] An EVCC may support a parameter in a Message Set if it is marked with an "O" for the EVCC.

[V2G2-661] An EVCC shall not support any parameter in a Message Set that is marked with an "-" for the EVCC.

[V2G2-662] An EVCC shall send a parameter if it is marked with an "M" for the EVCC in a request message.

[V2G2-663] An EVCC shall support the processing of a parameter if it is marked with an "M" for the EVCC in a response message.

For the SECC the following applies in general:

[V2G2-664] An SECC shall support a message or parameter in a Message Set if it is marked with an "M" for the SECC.

[V2G2-665] An SECC may support a parameter in a Message Set if it is marked with an "O" for the SECC.

[V2G2-666] An SECC shall not support any parameter in a Message Set that is marked with an "-" for the SECC.

[V2G2-667] An SECC shall send a parameter if it is marked with an "M" for the SECC in a response message.

[V2G2-668] An SECC shall support the processing of a parameter if it is marked with an "M" for the SECC in a request message.

Table 98 — Mandatory messages and message elements of Message Sets

V2G Message							Message Set																	
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
Supported App Protocol Req							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	ProtocolNamespace						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	VersionNumberMajor						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	VersionNumberMinor						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	SchemaID						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	Priority						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
Supported App Protocol Res							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	ResponseCode						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	SchemaID						M	O	M	O	M	O	M	O	-	-	-	-	-	-	-	-
Session Setup Req							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	EVCCID						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
Session Setup Res							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	ResponseCode						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	EVSEID						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	DateTimeNow						O	O	O	O	O	O	O	O	-	-	-	-	-	-	-	-
Service Discovery Req							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	Service Scope						O	O	O	O	O	O	O	O	-	-	-	-	-	-	-	-
	Service Category						O	M	O	M	O	M	O	M	-	-	-	-	-	-	-	-
Service Discovery Res							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	ResponseCode						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	PaymentOption						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
	Charge Service						**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
		ServiceTag					**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
			ServiceID				**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
			ServiceName				O	O	O	O	O	O	O	O	-	-	-	-	-	-	-	-
			ServiceCategory				O	**M**	O	**M**	O	**M**	O	**M**	-	-	-	-	-	-	-	-
			ServiceScope				O	O	O	O	O	O	O	O	-	-	-	-	-	-	-	-
		FreeService					**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
		EnergyTransferType					**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	ServiceList						-	-	-	-	-	-	-	-	**M**	**M**	**M**	**M**	-	-	**M**	**M**
		Service					-	-	-	-	-	-	-	-	**M**	**M**	**M**	**M**	-	-	**M**	**M**
			ServiceTag				-	-	-	-	-	-	-	-	**M**	**M**	**M**	**M**	-	-	**M**	**M**
				ServiceID			-	-	-	-	-	-	-	-	**M**	**M**	**M**	**M**	-	-	**M**	**M**
				Service Name			-	-	-	-	-	-	-	-	O	O	O	O	-	-	O	O
				Service Category			-	-	-	-	-	-	-	-	O	**M**	O	**M**	-	-	O	**M**
				Service Scope			-	-	-	-	-	-	-	-	O	O	O	O	-	-	O	O
			Free Service				-	-	-	-	-	-	-	-	**M**	**M**	**M**	**M**	-	-	**M**	**M**
Service Detail Req							-	-	-	-	-	-	-	-	**M**	**M**	**M**	**M**	-	-	**M**	**M**
	ServiceID						-	-	-	-	-	-	-	-	**M**	**M**	**M**	**M**	-	-	**M**	**M**
Service							-	-	-	-	-	-	-	-	**M**	**M**	**M**	**M**	-	-	**M**	**M**

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
Detail Res	Response Code						-	-	-	-	-	-	-	-	M	M	M	M	-	-	M	M
	ServiceID						-	-	-	-	-	-	-	-	M	M	M	M	-	-	M	M
	ServiceParameterList						-	-	-	-	-	-	-	-	M	M	M	M	-	-	M	O
		ParameterSet					-	-	-	-	-	-	-	-	M	M	M	M	-	-	M	M
			ParameterSetID				-	-	-	-	-	-	-	-	M	M	M	M	-	-	M	M
			Parameter				-	-	-	-	-	-	-	-	M	M	M	M	-	-	M	M
Service Payment Selection Req							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	Selected Payment Option						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	Selected ServiceList						M	M	M	M	M	M	M	M	M	M	M	M	-	-	M	M
		SelectedService					M	M	M	M	M	M	M	M	M	M	M	M	-	-	M	M
			ServiceTag				M	M	M	M	M	M	M	M	M	M	M	M	-	-	M	M
				ServiceID			M	M	M	M	M	M	M	M	M	M	M	M	-	-	M	M
				ParameterSetID			-	-	-	-	-	-	-	-	M	M	M	M	-	-	O	M
Service Payment Selection Res							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	Response Code						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
Payment Details Req							-	-	-	-	M	M	M	M	-	-	-	-	-	-	-	-
	ContractID						-	-	-	-	M	M	M	M	-	-	-	-	-	-	-	-
	ContractSignatureCertChain						-	-	-	-	M	M	M	M	-	-	-	-	-	-	-	-
		Certificate					-	-	-	-	M	M	M	M	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
		SubCertificates					-	-	-	-	O	O	O	O	-	-	-	-	-	-	-	-
			Certificate				-	-	-	-	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
Payment Details Res							-	-	-	-	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	Response Code						-	-	-	-	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	GenChallenge						-	-	-	-	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	DateTimeNow						**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
Contract Authentication Req							**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	GenChallenge						-	-	-	-	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
Contract Authentication Res							**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	EVSEProcessing						**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	ResponseCode						**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
Charge Parameter Discovery Req							**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	EVRequestedEnergyTransferType						**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	AC_EVChargeParameter						**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		DepartureTime					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		Eamount					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		EVMaxVoltage					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		EVMaxCurrent					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		EVMinCurrent					**M**	**M**	-	-	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	DC_EVChargeParameter						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
		DC_EVStatus					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
			EVReady				-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
			EVCabinConditioning				-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
			EVRESSConiditioning				-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
			EVErrorCode				-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
			EVRESSSOC				-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVMaximumCurrentLimit					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVMaximumPowerLimit					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		EVMaximumVoltageLimit					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVEnergyCapacity					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		EVEnergyRequest					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		FullSOC					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		BulkSOC					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
Charge Parameter Discovery Res							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	Response Code						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	EVSEProcessing						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	SASchedule List						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
		SAScheduleTuple					M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
			SAScheduleTupleID				M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
			PMaxSchedule				M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
				PMaxScheduleID			M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
				PMAaxScheduleEntry			M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
					RelativeTimeInterval		M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
						start	M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
						duration	M	O	M	O	M	O	M	O	-	-	-	-	-	-	-	-
					Pmax		M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
			SalesTariff				M	O	M	O	M	O	M	O	-	-	-	-	-	-	-	-
				SalesTariffID			M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
				SalesTariffDescription			O	O	O	O	O	O	O	O	-	-	-	-	-	-	-	-
				NumEPriceLevels			M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
				SalesTariffEntry			M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
					RelativeTimeInterval		M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
						start	M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
						duration	M	O	M	O	M	O	M	O	-	-	-	-	-	-	-	-
					EPriceLevel		M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
					Consumption Cost		M	O	M	O	M	O	M	O	-	-	-	-	-	-	-	-
						Start Value	M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
						Cost	M	O	M	O	M	O	M	O	-	-	-	-	-	-	-	-
	AC_EVSEChargeParameter						M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
		AC_EVSEStatus					M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
			PowerSwitchClosed				M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
			RCD				M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
			NotificationMaxDelay				M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
			EVSENotification				M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
		EVSE MaxVoltage					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		EVSE MaxCurrent					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		EVSE MinCurrent					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
	DC_EVSECharge Parameter						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		DC_EVSEStatus					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
			EVSEIsolationStatus				-	-	**M**	O	-	-	**M**	O	-	-	-	-	-	-	-	-
			EVSEStatusCode				-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
			NotificationMaxDelay				-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
			EVSENotification				-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSEMaximumVoltageLimit					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSEMinimumCurrentLimit					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSEMinimumVoltageLimit					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
		EVSECurrentRegulationTolerance					-	-	O	O	-	-	O	O	-	-	-	-	-	-	-	-
		EVSEPeakCurrentRipple					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSEEnergyToBeDelivered					-	-	O	O	-	-	O	O	-	-	-	-	-	-	-	-
Power Delivery Req							**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	ReadyToChargeState						**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	Charging Profile						O	**M**	O	**M**	O	**M**	O	**M**	-	-	-	-	-	-	-	-
		SAScheduleTupleID					**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
		Profile Entry					**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
			SAScheduleTupleID				**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
			ProfileEntry				**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
				ChargingProfileEntryStart			**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
				ChargingProfileEntryMaxPower			**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	DC_EVPowerDeliveryParameter						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
		DC_EVStatus					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
			EVReady				-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
			EVCabinConditioning				-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
			EVRESSConiditioning				-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
			EVErrorCode				-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
			EVRESSSOC				-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		BulkChargingComplete					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		ChargingComplete					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
Power Delivery Res							**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	Response Code						**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	-	-	-	-	-	-	-	-
	AC_EVSE Status						**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		PowerSwitchClosed					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		RCD					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		NotificationMaxDelay					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
		EVSENotification					**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-	-	-
	DC_EVSEStatus						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
		EVSEIsolationStatus					-	-	**M**	O	-	-	**M**	O	-	-	-	-	-	-	-	-
		EVSE Status Code					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		Notification MaxDelay					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSE Notification					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
Certificate update Req							-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-
	ContractSignatureCertChain						-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-
		Certificate					-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-
		SubCertificates					-	-	-	-	-	-	-	-	O	O	-	-	-	-	-	-
			Certificate				-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-
	ContractID						-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-
	ListOfRootCertificateIDs						-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-
	DHParams														O	**M**						
Certificate update Res							-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-
	Response Code						-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-
	ContractSignatureCertChain						-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-
		Certificate					-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-
		SubCertificates					-	-	-	-	-	-	-	-	O	O	-	-	-	-	-	-
			Certificate				-	-	-	-	-	-	-	-	**M**	**M**	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
	ContractSignatureEncryptedPrivateKey						-	-	-	-	-	-	-	-	M	M	-	-	-	-	-	-
	DHParams						-	-	-	-	-	-	-	-	M	M	-	-	-	-	-	-
	ContractID						-	-	-	-	-	-	-	-	M	M	-	-	-	-	-	-
	RetryCounter						-	-	-	-	-	-	-	-	M	M	-	-	-	-	-	-
Certificate installation Req							-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
	OEMProvisioningCert						-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
	ListOfRootCertificateIDs						-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
		RootCertificateID					-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
	DHParams														-	-	O	M				
Certificate installation Res							-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
	ResponseCode						-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
	ContractSignatureCertChain						-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
		Certificate					-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
		SubCertificates					-	-	-	-	-	-	-	-	-	-	O	O	-	-	-	-
			Certificate				-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
	ContractSignatureEncryptedPrivateKey						-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
	DHParams						-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
	ContractID						-	-	-	-	-	-	-	-	-	-	M	M	-	-	-	-
Session Stop Req							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
Session Stop Res							M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
	Response Code						M	M	M	M	M	M	M	M	-	-	-	-	-	-	-	-
Charging Status Req							M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
Charging Status Res							M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
	Response Code						M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
	EVSEID						M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
	SASchedule TupleID						M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
	EVSEMax Current						M	O	-	-	O	O	-	-	-	-	-	-	-	-	-	-
	MeterInfo						M	O	-	-	M	O	-	-	-	-	-	-	-	-	-	-
	ReceiptRequired						M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
	AC_EVSE Status						M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
		Power Switch Closed					M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
		RCD					M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
		Notification MaxDelay					M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
		EVSE Notification					M	M	-	-	M	M	-	-	-	-	-	-	-	-	-	-
Metering Receipt Req							-	-	-	-	-	-	-	-	-	-	-	-	M	M	-	-
	SessionID						-	-	-	-	-	-	-	-	-	-	-	-	M	M	-	-
	SASchedule TupleID						-	-	-	-	-	-	-	-	-	-	-	-	M	M	-	-
	MeterInfo						-	-	-	-	-	-	-	-	-	-	-	-	M	M	-	-
Metering Receipt Res							-	-	-	-	-	-	-	-	-	-	-	-	M	M	-	-
	Response Code						-	-	-	-	-	-	-	-	-	-	-	-	M	M	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
	AC_EVSE Status						-	-	-	-	-	-	-	-	-	-	-	-	**M**	**M**	-	-
		Power Switch Closed					-	-	-	-	-	-	-	-	-	-	-	-	**M**	**M**	-	-
		RCD					-	-	-	-	-	-	-	-	-	-	-	-	**M**	**M**	-	-
		Notification MaxDelay					-	-	-	-	-	-	-	-	-	-	-	-	**M**	**M**	-	-
		EVSE Notification					-	-	-	-	-	-	-	-	-	-	-	-	**M**	**M**	-	-
Cable Check Req							-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	DC_EVStatus						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVReady					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVCabinConditioning					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		EVRESSConiditioning					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		EVErrorCode					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVRESSSOC					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
Cable Check Res							-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	Response Code						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVSEProcessing						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	DC_EVSEStatus						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSEIsolationStatus					-	-	**M**	O	-	-	**M**	O	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
		EVSE Status Code					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		Notification MaxDelay					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSE Notification					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
PreCharge Req							-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	DC_EVStatus						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVReady					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVCabinConditioning					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		EVRESSConiditioning					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		EVErrorCode					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVRESSSOC					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVTargetVoltage						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVDemandCurrent						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
PreCharge Res							-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	Response Code						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	DC_EVSEStatus						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSEIsolationStatus					-	-	**M**	O	-	-	**M**	O	-	-	-	-	-	-	-	-
		EVSE Status Code					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
		Notification MaxDelay					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSE Notification					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVSEPresentVoltage						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
Current Demand Req							-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	DC_EVStatus						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVReady					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVCabinConditioning					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		EVRESSConiditioning					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		EVErrorCode					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVRESSSOC					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVTargetCurrent						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVMaximumVoltageLimit						-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
	EVMaximumCurrentLimit						-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
	EVMaximumPowerLimit						-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
	BulkChargingComplete						-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
	ChargingComplete						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
	RemainingTimeToFullSoC						-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
	RemainingTimeToBulkSoC						-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
	EVTargetVoltage						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
Current Demand Res							-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	Response Code						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	DC_EVSEStatus						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSEIsolationStatus					-	-	**M**	O	-	-	**M**	O	-	-	-	-	-	-	-	-
		EVSE Status Code					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		Notification MaxDelay					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSE Notification					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVSEPresentVoltage						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVSEPresentCurrent						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVSECurrentLimitAchieved						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVSEVoltageLimitAchieved						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVSEPowerLimitAchieved						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVSEMaximumVoltageLimit						-	-	**M**	O	-	-	**M**	O	-	-	-	-	-	-	-	-

V2G Message							Message Set															
Name	Parameter Level						AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Options: MeteringReceipt		Option: VAS	
	1	2	3	4	5	6	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC
	EVSEMaximumCurrentLimit						-	-	**M**	O	-	-	**M**	O	-	-	-	-	-	-	-	-
	EVSEMaximumPowerLimit						-	-	**M**	O	-	-	**M**	O	-	-	-	-	-	-	-	-
Welding Detection Req							-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	DC_EVStatus						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVReady					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVCabinConditioning					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		EVRESSConiditioning					-	-	O	**M**	-	-	O	**M**	-	-	-	-	-	-	-	-
		EVErrorCode					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVRESSSOC					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
Welding Detection Res							-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	Response Code						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	DC_EVSEStatus						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSEIsolationStatus					-	-	**M**	O	-	-	**M**	O	-	-	-	-	-	-	-	-
		EVSE Status Code					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		Notification MaxDelay					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
		EVSE Notification					-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-
	EVSEPresentVoltage						-	-	**M**	**M**	-	-	**M**	**M**	-	-	-	-	-	-	-	-

8.6.2.2 AC

8.6.2.2.1 EIM

If an EVCC/SECC supports the EIM identification mode for AC charging it supports the corresponding Messages Sets.

[V2G2-376] If an EVCC supports AC charging and the EIM identification mode it shall support the messages and parameters marked with “M” and "O" in Table 98, column “AC Charging EIM”, sub column “EVCC”.

[V2G2-377] If an SECC supports AC charging and the EIM identification mode it shall support the messages and parameters marked with “M” and "O" in Table 98, column “AC Charging EIM”, sub column “SECC”.

If an EVCC/SECC supports the Message Set “AC Charging EIM” it may additionally support the Message Set “Value Added Services”.

[V2G2-378] If an EVCC supports the Message Set “AC Charging EIM“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: VAS”, sub column “EVCC”.

[V2G2-379] If an SECC supports the Message Set “AC Charging EIM“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: VAS”, sub column “SECC”.

8.6.2.2.2 PnC

If an EVCC/SECC supports the PnC identification mode for AC charging it supports the corresponding Messages Sets.

[V2G2-380] If an EVCC supports AC charging and the PnC identification mode it shall support the messages and parameters marked with “M” and "O" in Table 98, column “AC Charging PnC”, sub column “EVCC”.

[V2G2-381] If an SECC supports AC charging and the PnC identification mode it shall support the messages and parameters marked with “M” and "O" in Table 98, column “AC Charging PnC”, sub column “SECC”.

[V2G2-382] If an EVCC supports the Message Set “AC Charging PnC“ it shall support the messages and parameters marked with “M” and "O" in Table 98, column “MeteringReceipt”, sub column “EVCC”.

[V2G2-383] If an SECC supports the Message Set “AC Charging PnC“ it shall support the messages and parameters marked with “M” and "O" in Table 98, column “MeteringReceipt", sub column “SECC”.

If an EVCC/SECC supports the PnC identification mode for AC charging it may additionally support the Message Set, “Value Added Services”, “Certificate Update”, and “Certificate Install“.

[V2G2-384] If an EVCC supports the Message Set “AC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: VAS”, sub column “EVCC”.

[V2G2-385] If an SECC supports the Message Set “AC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: VAS”, sub column “SECC”.

[V2G2-386] If an EVCC supports the Message Set “AC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: Certificate Installation”, sub column “EVCC”.

[V2G2-387] If an SECC supports the Message Set “AC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: Certificate Installation”, sub column “SECC”.

[V2G2-388] If an EVCC supports the Message Set “AC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Certificate Update”, sub column “EVCC”.

[V2G2-389] If an SECC supports the Message Set “AC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Certificate Update”, sub column “SECC”.

8.6.2.3 DC

8.6.2.3.1 Charging EIM

If an EVCC/SECC supports the EIM identification mode for DC charging it supports the corresponding Messages Sets.

[V2G2-390] If an EVCC supports DC charging and the EIM identification mode it shall support the messages and parameters marked with “M” and "O" in Table 98, column “DC Charging EIM”, sub column “EVCC”.

[V2G2-391] If an SECC supports DC charging and the EIM identification mode it shall support the messages and parameters marked with “M” and "O" in Table 98, column “DC Charging EIM”, sub column “SECC”.

If an EVCC/SECC supports the Message Set “DC Charging EIM” it may additionally support the Message Set “Value Added Services”.

[V2G2-392] If an EVCC supports the Message Set “DC Charging EIM“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: VAS”, sub column “EVCC”.

[V2G2-393] If an SECC supports the Message Set “DC Charging EIM“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: VAS”, sub column “SECC”.

8.6.2.3.2 PnC

If an EVCC/SECC supports the PnC identification mode for DC charging it supports the corresponding Messages Sets.

[V2G2-394] If an EVCC supports DC charging and the PnC identification mode it shall support the messages and parameters marked with “M” and "O" in Table 98, column “DC Charging PnC”, sub column “EVCC”.

[V2G2-395] If an SECC supports DC charging and the PnC identification mode it shall support the messages and parameters marked with “M” and "O" in Table 98, column “DC Charging PnC”, sub column “SECC”.

If an EVCC/SECC supports the PnC identification mode for DC charging it may additionally support the Message Set “Value Added Services”, “Certificate Update”, and “Certificate Install“.

[V2G2-396] If an EVCC supports the Message Set “DC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: VAS”, sub column “EVCC”.

[V2G2-397] If an SECC supports the Message Set “DC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: VAS”, sub column “SECC”.

[V2G2-398] If an EVCC supports the Message Set “DC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: Certificate Installation”, sub column “EVCC”.

[V2G2-399] If an SECC supports the Message Set “DC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Option: Certificate Installation”, sub column “SECC”.

[V2G2-400] If an EVCC supports the Message Set “DC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Certificate Update”, sub column “EVCC”.

[V2G2-401] If an SECC supports the Message Set “DC Charging PnC“ it may additionally support the messages and parameters marked with “M” and "O" in Table 98, column “Certificate Update”, sub column “SECC”.

8.6.3 Selection of Message Sets

8.6.3.1 Message Sets for AC/DC Charging EIM/PnC

The selection of the Message Set is based on the selected payment option in the Service Payment Selection Request message.

Figure 92 shows an overview for selecting the Messages Sets “AC Charging EIM”, “DC Charging EIM”, AC Charging PnC”, and “DC Charging PnC” based on the definitions for ServicePaymentSelectionReq (refer to subclause 8.4.1.5.2) and ChargeParameterDiscoveryReq (refer to subclause 8.4.1.8.2).

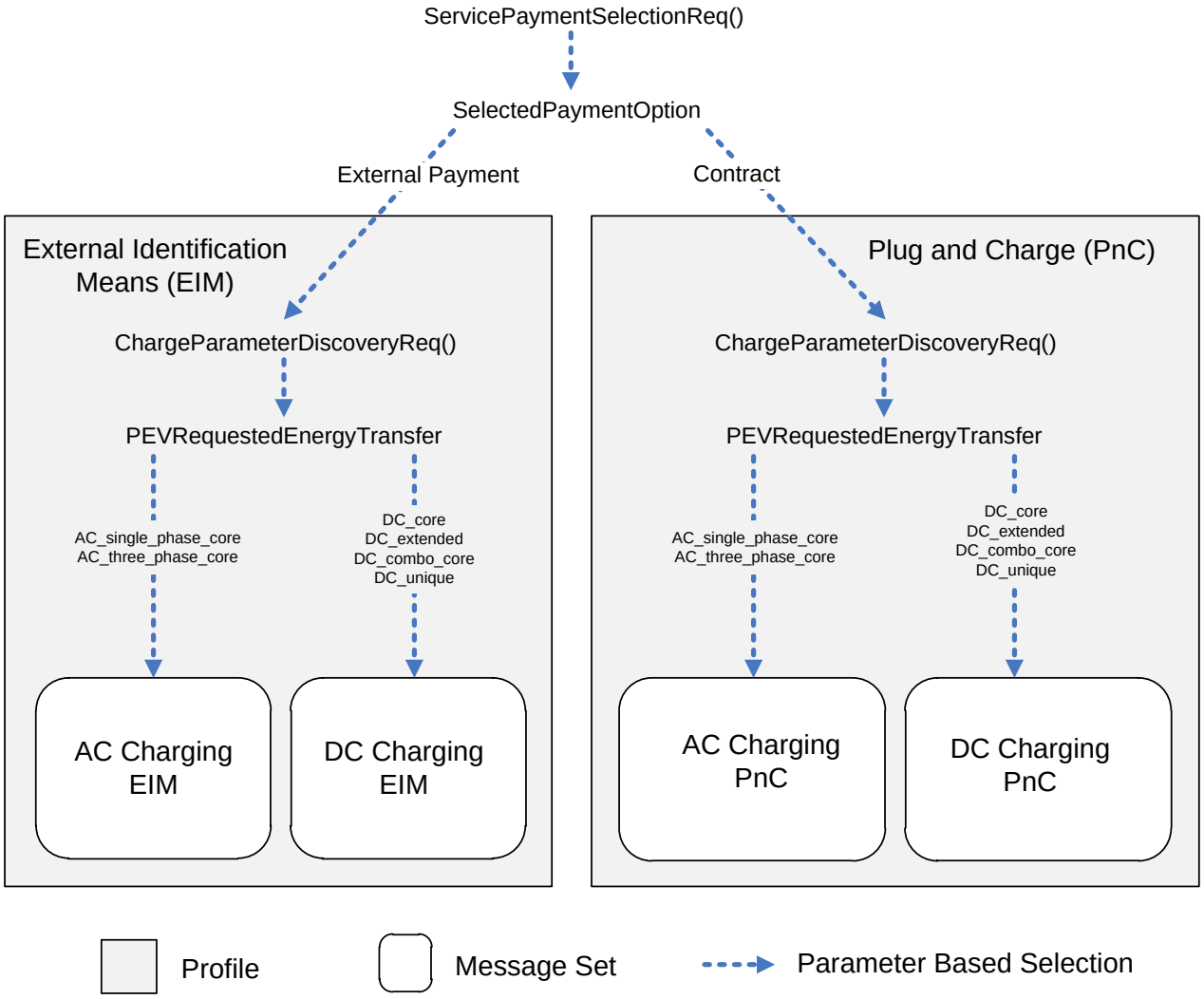


Figure 92 — Selection of Message Sets

[V2G2-402] If an EVCC sends the payment option “External Payment” in the parameter “SelectedPaymentOption” of the message “ServicePaymentSelectionReq” and sends “AC_single_phase_core” or “AC_three_phase_core” in the parameter “EVRequestedEnergyTransfer” of the Message “ChargeParameterDiscoveryReq” it shall use the Message Set “AC Charging EIM”.

[V2G2-403] If an EVCC sends the payment option “External Payment” in the parameter “SelectedPaymentOption” of the message “ServicePaymentSelectionReq” and sends “DC_core” or “DC_extended” or “DC_combo_core” or “DC_unique” in the parameter “EVRequestedEnergyTransfer” of the Message “ChargeParameterDiscoveryReq” it shall use the Message Set “DC Charging EIM”.

[V2G2-404] If an EVCC sends the payment option “Contract” in the parameter “SelectedPaymentOption” of the message “ServicePaymentSelectionReq” and sends “AC_single_phase_core” or “AC_three_phase_core” in the parameter “EVRequestedEnergyTransfer” of the Message “ChargeParameterDiscoveryReq” it shall use the Message Set “AC Charging PnC”.

[V2G2-405] If an EVCC sends the payment option “External Payment” in the parameter “SelectedPaymentOption” of the message “ServicePaymentSelectionReq” and sends “DC_core” or “DC_extended” or “DC_combo_core” or “DC_unique” in the parameter “EVRequestedEnergyTransfer” of the Message “ChargeParameterDiscoveryReq” it shall use the Message Set “DC Charging PnC”.

8.6.3.2 Message Set Metering Receipt

As shown in Figure 1, the Message Set “Meter Status Receipt” is mandatory for the EVCC/SECC if it supports the Message Set “AC Charging Receipt”. While the implementation of the Message Set “Meter Status Receipt” is mandatory to ensure compatibility, the application of the Message Set is optional.

[V2G2-406] If an EVCC selected the Message Set “AC Charging PnC” as described in subclause 8.6.3.1 and an SECC requires a metering receipt from an EVCC, the SECC shall set the Parameter “Receipt Required” in the message “ChargingStatusRes” to value “TRUE”.

[V2G2-407] If an EVCC selected the Message Set “AC Charging PnC” as described in subclause 8.6.3.1 and an SECC does not require any metering receipt from an EVCC, the SECC shall set the Parameter “Receipt Required” in the message “ChargingStatusRes” to value “FALSE”.

[V2G2-408] If an EVCC selected the Message Set “AC Charging PnC” as described in subclause 8.6.3.1 it shall use the Message Set “Metering Receipt” if the SECC sets the Parameter “ReceiptRequired” to value “Yes” in the message “ChargingStatusRes”.

[V2G2-409] If an EVCC selected the Message Set “AC Charging PnC” as described in subclause 8.6.3.1 and an SECC set the Parameter “Receipt Required” in the message “ChargingStatusRes” to value “Yes” it shall use the Message Set “Metering Receipt”

8.6.3.3 Certificate Install

[V2G2-410] If an SECC offers the certificate installation service in the parameter “ServiceList” in the message “ServiceDiscoveryRes” it shall use the Message Set “Certificate Install”.

[V2G2-411] If an EVCC intends to use certification installation services offered by an SECC in the parameter “ServiceList” in the message “ServiceDiscoveryRes” it shall use the Message Set “Certificate Install”.

8.6.3.4 Certificate Update

[V2G2-412] If an SECC offers the certificate installation service in the parameter “ServiceList” in the message “ServiceDiscoveryRes” it shall use the Message Set “Certificate Update”.

[V2G2-413] If an EVCC intends to use certification installation services offered by an SECC in the parameter “ServiceList” in the message “ServiceDiscoveryRes” it shall use the Message Set “Certificate Update”.

8.6.3.5 Message Set Value Added Services

[V2G2-414] If an SECC offers the use of Value Added Services in the parameter “ServiceList” in the message “ServiceDiscoveryRes” it shall use the Message Set “Value Added Services (VAS)”.

[V2G2-415] If an EVCC intends to use Value Added Services offered by an SECC in the parameter “ServiceList” in the message “ServiceDiscoveryRes” it shall use the Message Set “Value Added Services (VAS)”.

8.6.3.6 Selection of services

Any Service Tag as defined in 8.5.2.1 includes a Service ID (mandatory). This mandatory field allows the EVCC to identify a service.

This subclause defines the reserved ServiceID ranges and the ranges that can be used implementation specific. Table 99 defines the ServiceIDs defined and reserved by this standard. In addition it defines the parameter sets for Certificate services and InternetAccess services including the respective parameterSetIDs (refer to Table 100 and Table 101). Also this subclause includes requirements defining the usage of

ServiceDiscoveryReq, ServiceDiscoveryRes, ServiceDetailsReq, ServiceDetailsRes, ServicePaymentSelectionReq and ServicePaymentSelectionRes.

NOTE Refer to Annex D.1 for an example how the parameter set for an InternetAccess service is selected using the definition in this subclause.

[V2G2-416] The EVCC and the SECC shall use the Service IDs in the range from 1 to 3 as defined in this standard.

[V2G2-417] The EVCC and the SECC shall conform to the Service ID, ServiceName, ServiceCategory as defined in Table 99.

Table 99 — Definition of Service ID, Service Category, Service Name, and Service Scope

Service ID (unsignedshort)	ServiceName	ServiceCategory	Description
0			Reserved by ISO/IEC
1	AC_DC_Charging	EVCharging	All charging services as defined by EVSESupportedEnergyTransferType in Table 63.
2	Certificate	ContractCertificate	Service allowing to update or install contract certificates.
3	InternetAccess	Internet	Service for standard protocols like HTTP, HTTPs, FTP, etc..
4 – 60000			Reserved by ISO/IEC
60001 – 65535			Reserved for implementation specific use

[V2G2-418] The ServiceDiscoveryRes shall contain information about the offered services which requires the appropriate ResponseCode, PaymentOptions, ChargeService and optional a ServiceList.

[V2G2-419] The requirements **[V2G2-420]** and **[V2G2-421]** shall apply if a ServiceList is offered.

[V2G2-420] The ServiceList shall contain a list of offered services and for each offered service an information if the service can be used for free or if the customer needs to pay for the service.

[V2G2-421] An offered service shall be identified by it's ServiceID, the offered value shall comply with Table 99. An offered service may contain one or multiple additional information like ServiceName, ServiceCategory and ServiceScope.

[V2G2-422] The EVCC shall request ServiceDetails prior using one or multiple services offered in the ServiceList of the ServiceDiscoveryRes.

[V2G2-424] The EVCC shall provide the ServiceID for which the service details are requested. The ServiceID shall be used according to Table 99.

[V2G2-425] The SECC shall respond with a negative response code 'FAILED_ServiceIDInvalid' if the EVCC provided a not previously retrieved ServiceID in the ServiceDetailsReq.

[V2G2-426] The SECC shall respond with the ServiceParameterList containing the detailed information about the requested ServiceID.

[V2G2-427] The ServiceParameterList shall contain a ParameterSetID and details to the offered parameters.

[V2G2-428] The ServiceParameterList shall comply with Table 100 if service details for ServiceID 2 'Certificate' is requested

Table 100 — ServiceParameterList for certificate service

ParameterSetID (unsignedshort)	ParameterName = Service	Description
0		Reserved by ISO/IEC
1	stringValue = Installation	Service to install a contract certificate in the EVCC, according to 8.4.1.11.
2	stringValue = Update	Service to update a contract certificate in the EVCC, according to 8.4.1.10.
4 – 60000		Reserved by ISO/IEC
60001 – 65535		Implementation specific use

[V2G2-429] The ServiceParameterList shall comply with Table 101 if service details for ServiceID 3 'InternetAccess' is requested.

Table 101 — ServiceParameterList for internet access service

ParameterSetID (unsignedshort)	ParameterName = Protocol	ParameterName = Port	Description
0			Reserved by ISO/IEC
1	stringValue = ftp	intValue = 20	Service to use internet access using FTP protocol via port 20
2	stringValue = ftp	intValue = 21	Service to use internet access using FTP protocol via port 21
3	stringValue = http	intValue = 80	Service to use internet access using HTTP protocol via port 80
4	stringValue = https	intValue = 443	Service to use internet access using HTTPS protocol via port 443,
5 – 65535	service name according to IANA Service&PortRegistry	port number according to IANA Service&PortRegistry	Additional protocol port combinations which are supported by the SECC for internet access

[V2G2-430] If the SECC supports additional protocol / port combinations beyond the definitions in Table 101, it shall use the service names and the assinged port numbers according to IANA Service&PortRegistry (i.e. the service name defined in IANA Service&PortRegistry is transmitted as the "Protocol" and the port number defined in IANA Service&PortRegistry is transmitted as "Port").

NOTE It is assumed if a IANA Service&PortRegistry defined service name and port number combination is applicable for both transport protocols, TCP and UDP, the SECC supports connections on TCP or UDP or on both for the respective combination.

[V2G2-431] The ServicePaymentSelectionReq shall contain a list of selected services.

[V2G2-432] Each selected service shall be defined according to a ServiceID and a ParameterSetID which are previously retrieved from the SECC using ServiceDiscovery and ServiceDetail message set.

[V2G2-433] The SECC shall respond with a negative response code 'FAILED_ServiceSelectionInvalid' if the EVCC provided a not previously retrieved ServiceID, ParameterSetID pair in the ServicePaymentSelectionReq.

8.7 V2G Communication Timing

8.7.1 Overview

This subclause describes the timing and error handling for the V2G Communication Session. The error handling is based on timers enabling the EVCC and the SECC to monitor the V2G message exchange. For the detection of missing or delayed messages the EVCC and the SECC use predefined timeout values as error criteria. Whenever a timer is equal of larger than the related timeout the related error handling is processed.

A timer counts the duration from the last time it was reset. The value of a timer is the duration from the last reset to the present time. The monitoring of a V2G Communication Message is based on two Timer categories:

— Message Timer: Monitors the exchange of a request message and the corresponding response message (Request-Response-Pair).

— Sequence Timer: Monitors the exchange of multiple Request-Response-Pairs.

To enable error handling for a V2G Communication Session setup the EVCC monitors the time between plug-in and the reception of the Session Setup Response and the Power Delivery Response, respectively. This allows the EVCC to decide about a successful or failed charging session after the defined timeouts.

The monitoring of a V2G Communication Session is based on two Timer categories:

— Communication Setup Timer: Monitors the time from plug-in until the Session Setup message. It allows deciding if the communication setup was successful.

— Ready to Charge Timer: Monitors the time from plug-in until the first Power Delivery message. It allows deciding if the request for power from the EVCC was successful.

The timers are compared to predefined time values as decision criteria. The EVCC and the SECC decide between two categories:

— Timeout: If the specified time is exceeded the related error handling is initiated.

— Performance Time: If the specified time is exceeded the performance requirement is not fulfilled.

NOTE While exceeding a timeout always causes an error handling, the performance time does not necessarily cause error handling. Depending on the system behaviour (e.g. transmission time) no error may occur if the corresponding communication partner does not detect a timeout but the probability for causing a timeout is high.

8.7.2 Message sequence and communication session

8.7.2.1 Definitions

Message Timers, Sequence Timers, Timeouts, and Performance Times are defined for EVCC and SECC separately and are summarized in Table 102. Timeouts and Performance Times are parameterized for messages separately to describe different processing times. Table 103 defines the values for each V2G message type.

Table 102 — EVCC and SECC Timers, Timeouts, Performance Times

Name	Type	Applicable for	
		EVCC	SECC
V2G_EVCC_Msg_Timer	Message Timer in the EVCC	x	
V2G_SECC_Msg_Timer	Message Timer in the SECC		x
V2G_EVCC_Sequence_Timer	Sequence Timer in the EVCC	x	

V2G_SECC_Sequence_Timer	Sequence Timer in the SECC		x
V2G_EVCC_Msg_Timeout (MessageType)	Timeout for the Message Timer The value is defined by the parameter MessageType as defined in Table 103.	x	
V2G_SECC_Msg_Performance_Time (MessageType)	Performance Time for the Message Timer The value is defined by the parameter MessageType as defined in Table 103.		x
V2G_EVCC_Sequence_Performance_Time	Performance Time for the Sequence Timer as defined in Table 103.	x	
V2G_SECC_Sequence_Timeout	Timeout for the Sequence Timer as defined in Table 103.		x

Table 103 — EVCC EVCC and SECC Message sequence and session timing parameter values

Name	MessageType	Value [s]
V2G_EVCC_Msg_Timeout(MessageType)	SupportedAppProtocol	2
	SessionSetup	2
	ServiceDiscovery	2
	ServicePaymentSelection	2
	PaymentDetails	5
	ChargeParameterDiscovery	2
	ChargingStatus	2
	MeteringReceipt	2
	PowerDelivery	2
	CableCheck	25
	PreCharge	1
	CurrentDemand	0,25
	WeldingDetection	1
	SessionStop	2
	CertificateInstall	5
	CertificateUpdate	5

Name	MessageType	Value [s]
V2G_SECC_Msg_Performance_Time(MessageType)	SupportedAppProtocol	1,5
	SessionSetup	1,5
	ServiceDiscovery	1,5
	ServicePaymentSelection	1,5
	PaymentDetails	4,5
	ChargeParameterDiscovery	1,5
	ChargingStatus	1,5
	MeteringReceipt	1,5
	PowerDelivery	1,5
	CableCheck	20
	PreCharge	0,1
	CurrentDemand	0,025
	WeldingDetection	0,1
	SessionStop	1,5
	CertificateInstall	4,5
	CertificateUpdate	4,5
V2G_EVCC_Sequence_Performance_Time	(all messages)	40
V2G_SECC_Sequence_Timeout	(all messages)	60

Figure 93 illustrates how the Message Timers, Sequence Timers, Timeouts, and Performance Times are applied in the EVCC and the SECC.

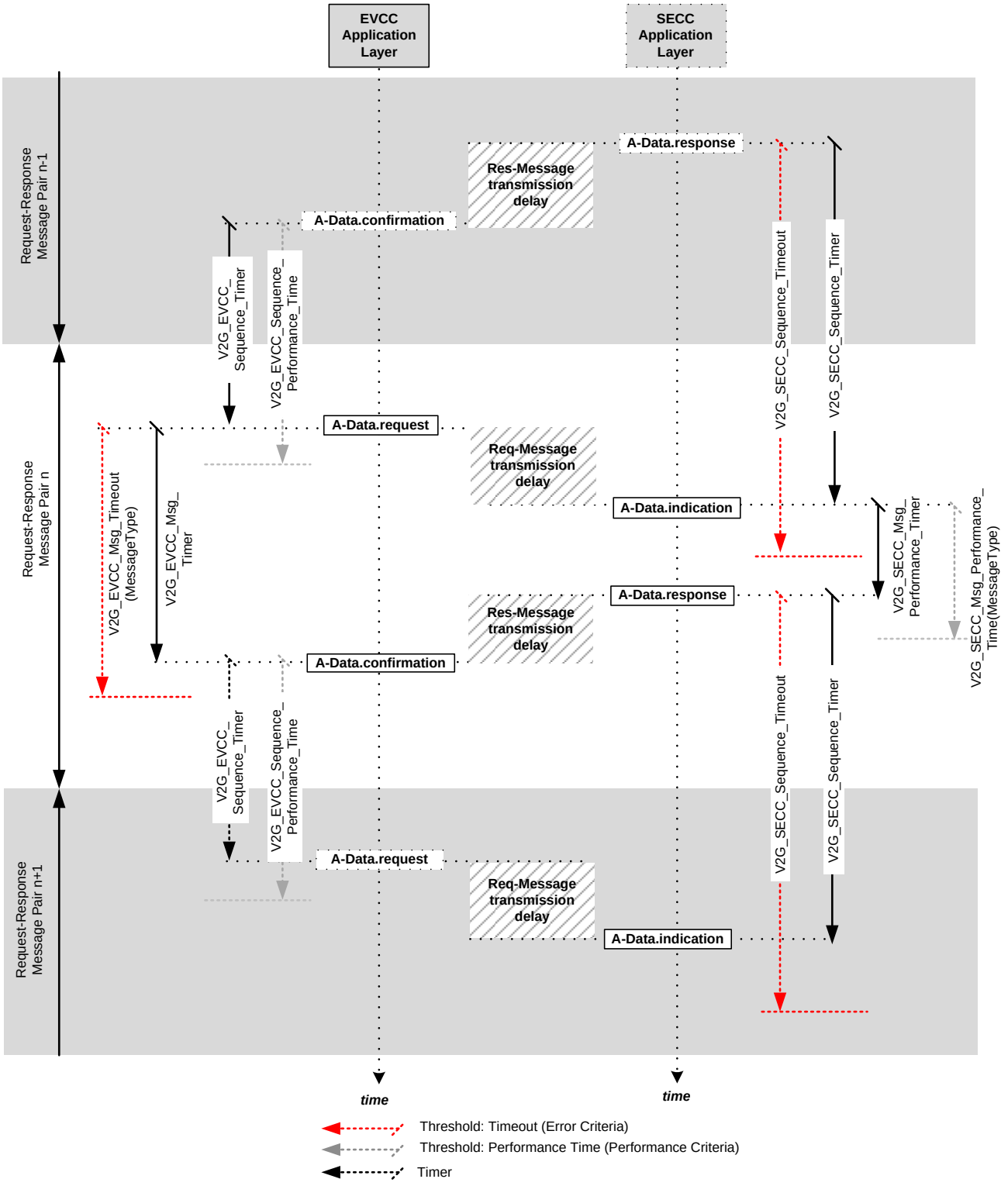


Figure 93 — Message sequence and session timing

[V2G2-434] The EVCC shall implement the EVCC specific Timeouts and Performance Times defined in Table 102 and Table 103.

[V2G2-435] The SECC shall implement the SECC specific Timeouts and Performance Times defend in Table 102 and Table 103.

8.7.2.2 EVCC Timing for Request-Response Message Pairs

[V2G2-436] The EVCC shall set the timeout V2G_EVCC_Msg_Timeout to the value MessageType as defined in Table 103, reset the V2G_EVCC_Msg_Timer and start monitoring the V2G_EVCC_Msg_Timer when it sends a request message.

NOTE 1 In this document sending a request message is described by A-DATA.request.

[V2G2-437] The EVCC shall wait for the response message corresponding to the request message sent before.

[V2G2-438] The EVCC shall stop waiting for the response message and stop monitoring the V2G_EVCC_Msg_Timer when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout(MessageType) and no response message was received. It shall then apply the error handling as defined in subclause 8.8.

NOTE 2 In this document receiving a response message is described by A-DATA.confirmation.

[V2G2-439] The EVCC shall stop waiting for the response message and stop monitoring the V2G_EVCC_Msg_Timer when V2G_EVCC_Msg_Timer is smaller than V2G_EVCC_Msg_Timeout(MessageType) and it received a response message. It shall then process the response message as defined in subclause 8.8.

NOTE 3 In this document receiving a response message is described by A-DATA.confirmation.

[V2G2-440] The EVCC shall ignore any message that is not a valid response message.

8.7.2.3 SECC Timing for Request-Response Message Pairs

[V2G2-441] The SECC shall set the timeout V2G_SECC_Sequence_Timeout to the value MessageType as defined in Table 103, reset the V2G_SECC_Sequence_Timer and start monitoring the V2G_SECC_Sequence_Timer when it sends a response message.

NOTE 1 In this document sending a response message is described by A-DATA.response.

[V2G2-442] The SECC shall wait for a request message.

[V2G2-443] The SECC shall stop waiting for a request message and stop monitoring the V2G_SECC_Sequence_Timer when V2G_SECC_Sequence_Timer is equal or larger than V2G_SECC_Sequence_Timeout and no request message was received. It shall then apply the error handling as defined in subclause 8.8.

NOTE 2 In this document receiving a request message is described by A-DATA.indication.

[V2G2-444] The SECC shall stop waiting for a request message and stop monitoring the V2G_SECC_Sequence_Timer when V2G_SECC_Sequence_Timer is smaller than V2G_SECC_Sequence_Timeout and it received a request message. It shall then process the response message as defined in subclause 8.8.

NOTE 3 In this document receiving a response message is described by A-DATA.indication.

[V2G2-445] The EVCC shall ignore any message that is not a valid request message.

8.7.3 Session setup and ready to charge

8.7.3.1 Definitions

Timing parameters applicable to the communication session setup and ready to charge time defined in this standard are shown in Table 104. Table 105 define the values for the related Performance Times and the Timeouts.

Table 104 — EVCC and SECC V2G Communication session setup timing parameters

Parameter name	Definition	Implementation	
		EVCC	SECC
V2G_EVCC_CommunicationSetup_Timer	Communication Setup Timer in the EVCC	x	
V2G_SECC_CommunicationSetup_Timer	Communication Setup Timer in the SVCC		x
V2G_EVCC_ReadyToCharge_Timer	Ready to Charge Timer in the EVCC	x	
V2G_SECC_ReadyToCharge_Timer	Ready to Charge Timer in the SECC		x
V2G_EVCC_CommunicationSetup_Timeout	Timeout for the Communication Setup Timer as defined in Table 105.	x	
V2G_SECC_CommunicationSetup_Performance_ Time	Performance Time for the Communication Setup Timer as defined in Table 105.		x
V2G_EVCC_ReadyToCharge_Timeout	Timeout for the Ready to Charge Timer as defined in Table 105.	x	
V2G_SECC_ReadyToCharge_ Performance_Time	Performance Time for the Ready to Charge Timer as defined in Table 105.		x

[V2G2-605] The EVCC and SECC shall implement the timing parameter values defined in Table 105.

Table 105 — EVCC and SECC Message sequence and session timing parameter values

Parameter name	Value [s]	Implementation	
		EVCC	SECC
V2G_SECC_ReadyToCharge_Performance_Time	28		x
V2G_EVCC_ReadyToCharge_Timeout	30	x	
V2G_SECC_CommunicationSetup_Performance_Time	18		x
V2G_EVCC_CommunicationSetup_Timeout	20	x	

Figure 94 illustrates how the timing parameters defined in Table 104 are applied.

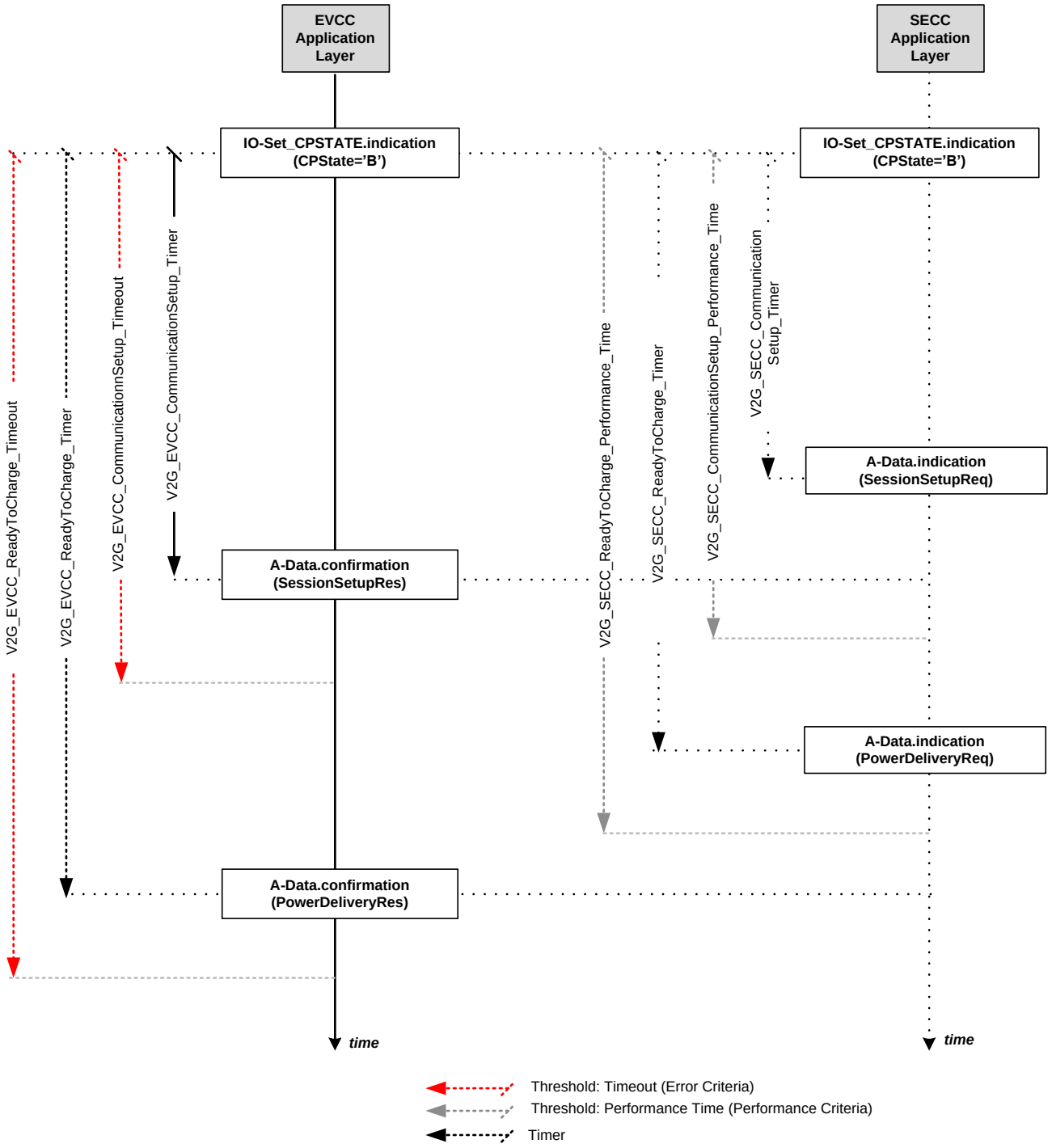


Figure 94 — Session setup setup and ready to charge performance timing

8.7.3.2 EVCC Timing for communication session setup

[V2G2-446] The EVCC shall set the timeout V2G_EVCC_CommunicationSetup_Timeout to the value as defined in Table 105, reset the V2G_EVCC_CommunicationSetup_Timer and start monitoring the V2G_EVCC_CommunicationSetup_Timer when state B is indicated.

NOTE 1 In this document state B is indicated by IO-CPSTATE.indication (CPState=B).

[V2G2-447] The EVCC shall wait for the Session Setup Response message.

[V2G2-448] The EVCC shall stop waiting for the Session Setup Response message and stop monitoring the V2G_EVCC_CommunicationSetup_Timer when V2G_EVCC_CommuncationSetup_Timer

is equal or larger than V2G_EVCC_CommunicationSetup_Timeout and no Session Setup Response message was received. It shall then apply the error handling as defined in subclause 8.8.

NOTE 2 In this document receiving the response message “Session Setup Response” is described by A-DATA.confirmation(SessionSetupRes).

[V2G2-449] The EVCC shall stop waiting for the Session Setup Response message and stop monitoring the V2G_EVCC_CommunicationSetup_Timer when V2G_EVCC_CommuncationSetup_Timer is small than V2G_EVCC_CommunicationSetup_Timeout and a Session Setup Response message was received. It shall then process the response message as defined in subclause 8.8.

NOTE 3 In this document receiving the response message “Session Setup Response” is described by A-DATA.confirmation(SessionSetupRes).

8.7.3.3 EVCC Timing for ready to charge

[V2G2-450] The EVCC shall set the timeout V2G_EVCC_ReadyToCharge_Timeout to the value as defined in Table 105, reset the V2G_EVCC_ReadyToCharge_Timer and start monitoring the V2G_EVCC_ReadyToCharge_Timer when state B is indicated.

NOTE 1 In this document state B is indicated by IO-CPSTATE.indication (CPState=B).

[V2G2-604] The EVCC shall wait for the Power Delivery Response message.

[V2G2-451] The EVCC shall stop waiting for the Power Delivery Response message and stop monitoring the V2G_EVCC_ReadyToCharge_Timer when V2G_EVCC_ReadyToCharge_Timer is equal or larger than V2G_EVCC_ReadyToCharge_Timeout and no Power Delivery Response message was received. It shall then apply the error handling as defined in subclause 8.8.

NOTE 2 In this document receiving a response message is described by A-DATA.confirmation(PowerDeliveryRes).

[V2G2-452] The EVCC shall stop waiting for the Power Delivery Response message and stop monitoring the V2G_EVCC_ReadyToCharge_Timer when V2G_EVCC_ReadyToCharge_Timer is equal or larger than V2G_EVCC_ReadyToCharge_Timeout and no Power Delivery Response message was received. It shall then process the response message as defined in subclause 8.8.

NOTE 3 In this document receiving a response message is described by A-DATA.confirmation(PowerDeliveryRes).

[V2G2-681] If the EVCC receives a V2G response message with parameter EVSEProcessing equal to 'Ongoing' the EVCC shall stop the timer 'V2G_EVCC_ReadyToCharge_Timer' and keep the value when the timer was stopped.

[V2G2-682] If the EVCC receives a V2G message response with parameter EVSEProcessing equal to 'Finished' and the timer 'V2G_EVCC_ReadyToCharge_Timer' is stopped the EVCC shall continue the timer 'V2G_EVCC_ReadyToCharge_Timer' starting with the value when the timer was stopped.

8.8 Message Sequencing and Error Handling

8.8.1 Overview

Depending on the identification mode as defined in subclause 8.6 and the error handling the EVCC and SECC comply to a defined Request-Response Message Sequence. These Request-Response Message Sequences allow both sides to synchronize the process in any situation and to control the correct behaviour of the communication partner.

8.8.2 Basic Definitions for Error Handling

The basic error handling for a Request-Response-Message Pair and a Request-Response Message Sequence is based on the Response Code included in the Response Message of the SECC. Depending on the value in the Response Code the EVCC decides if it can proceed with the standard Request-Response Message Sequence or if it has to handle an error.

In this standard, the Response Code as defined in Annex C.6 is interpreted by the EVCC as follows:

— OK:
Any Value starting with “OK” or “OK_” indicates an positive response. Detailed information may be provided by OK_<additional info>. This information may be used to differentiate the reaction on the positive response.

— FAIL:
Any Value starting with “FAIL” or “FAIL_” indicates an negative response. Detailed information may be provided by FAIL_<additional info>. This information may be used to differentiate the reaction on the negative response.

8.8.3 Response Code Handling

8.8.3.1 Common Requirements

Besides the requirements for Request-Response Message Pairs and Request-Response Message Sequences as defined in subclause 8.8.4 the EVCC and the SECC shall conform to the following requirements.

[V2G2-455] The EVCC shall change to state B as defined in IEC 61851-1 (IO-SET_CPSTATE.request (CPState=B) and IO-SET_CPSTATE.confirmation (CPState=B)) after sending a Power Delivery Request with parameter ‘ReadyToChargeState set to 'FALSE’ and receiving the message Power Delivery Response as defined in 8.4.1.9.3.

[V2G2-456] The SECC shall measure the state B as defined in IEC 61851-1 (IO-SET_CPSTATE.indicationn (CPState=B)) after receiving a Power Delivery Request with parameter ‘ReadyToChargeState set to 'FALSE’ and sending the message Power Delivery Response as defined in 8.4.1.9.3.

In general each response message can contain two types of response codes 'OK' or 'FAILED'.

[V2G2-457] A response message shall contain the ResponseCode 'OK' in the 'ResponseCode' attribute if the processing of the request message was successful. If later on a specific positive 'ResponseCode' is defined for a dedicated situation, this ResponseCode shall be used.

[V2G2-458] A response message shall contain the ResponseCode 'FAILED' in the 'ResponseCode' attribute if the processing of the request message was not successful and no specific 'ResponseCodeType' is defined for the concrete error case.

[V2G2-459] The response message shall contain the ResponseCode 'FAILED_SequenceError' if the SECC has received an unexpected request message.

[V2G2-460] The response message shall contain the ResponseCode 'FAILED_UnknownSession' if the SessionID in the request message does not fit to the SECC provided SessionID during SessionSetupRes.

[V2G2-461] The response message shall contain the ResponseCode 'FAILED_SignatureError' if the validation of the Security element in the message header failed.

[V2G2-462] The message 'SessionSetupRes' shall contain the specific ResponseCode 'OK_NewSessionEstablished' if processing of the SessionSetupReq message was successful

and a different SessionID is contained in the response message than the SessionID in the request message.

[V2G2-463] The message 'SessionSetupRes' shall contain the specific ResponseCode 'OK_OldSessionJoined' if processing of the SessionSetupReq message was successful and the same SessionID as used in the request message is contained in the response message.

[V2G2-464] The message 'ServiceDetailRes' shall contain the ResponseCode 'FAILED_ServiceIDInvalid' if the ServiceID contained in the ServiceDetailReq message was not part of the offered ServiceList during ServiceDiscovery.

[V2G2-465] The message 'ServicePaymentSelectionRes' shall contain the ResponseCode 'FAILED_PaymentSelectionInvalid' if the SelectedPaymentOption contained in the ServicePaymentSelectionReq message was not part of the offered PaymentOptions of ServiceDiscoveryRes.

[V2G2-466] The message 'ServicePaymentSelectionRes' shall contain the ResponseCode 'FAILED_PaymentSelectionInvalid' if the SelectedPaymentOption contained in the ServicePaymentSelectionReq message was not part of the offered PaymentOptions of ServiceDiscoveryRes.

[V2G2-467] The message 'ServicePaymentSelectionRes' shall contain the ResponseCode 'FAILED_ServiceSelectionInvalid ' if the SelectedServiceList contained in the ServicePaymentSelectionReq message contains a ServiceID which was not contained in the offered ServiceList of ServiceDiscoveryRes.

[V2G2-468] The message 'CertificateInstallationRes' shall contain the ResponseCode 'FAILED_CertificateExpired' if the OEMProvisioningCert contained in the CertificateInstallationReq message is not valid.

[V2G2-469] The message 'CertificateInstallationRes' shall contain the ResponseCode 'FAILED_NoCertificateAvailable' if the new certificate can not be retrieved from secondary actor within V2G_SECC_Msg_Performance_Time according to defined in Table 103.

[V2G2-470] The message 'CertificateUpdateRes' shall contain the ResponseCode 'FAILED_CertChainError' if the ContractSignatureCertChain contained in the CertificateInstallationReq message is not valid.

[V2G2-471] The message 'CertificateUpdateRes' shall contain the ResponseCode 'FAILED_NoCertificateAvailable' if the new certificate can not be retrieved from secondary actor within V2G_SECC_Msg_Performance_Time according to Table 103.

[V2G2-472] The message 'CertificateUpdateRes' shall contain the ResponseCode FAILED_ContractCanceled if the provided ContractID provided by EVCC during CertificateUpdateReq is not accepted by secondary actor.

[V2G2-473] The message 'CertificateUpdateRes' shall contain the ResponseCode 'FAILED_CertificateExpired' if the contract certificate contained in the CertificateUpdateReq message is not valid.

[V2G2-474] The message 'PaymentDetailsRes' shall contain the ResponseCode 'FAILED_CertificateExpired' if the contract certificate contained in the PaymentDetailsReq message in attribute ContractSignatureCertChain is not valid.

[V2G2-475] The message 'ContractAuthenticationRes' shall contain the ResponseCode 'FAILED_ChallengeInvalid' if the challenge response contained in the ContractAuthenticationReq message in attribute GenChallenge is not valid versus the provided GenChallenge in PaymentDetailsRes.

[V2G2-476] The message 'ChargeParameterDiscoveryRes' shall contain the ResponseCode 'FAILED_WrongEnergyTransferType' if the content of attribute 'EVRequestedEnergyTransferType' in the ChargeParameterDiscoveryReq message is not valid, or does not fit to the content of attribute EVChargeParameter.

[V2G2-477] The message 'ChargeParameterDiscoveryRes' shall contain the ResponseCode 'FAILED_WrongChargeParameter' if the content of attribute ' EVChargeParameter' in the ChargeParameterDiscoveryReq message is not valid, e.g. wrong parameter set is provided, one or multiple parameters can not be interpreted, ….

[V2G2-478] The message 'PowerDeliveryRes' shall contain the ResponseCode 'FAILED_ChargingProfileInvalid' if the content of attribute 'ChargingProfile' in the PowerDeliveryReq message violates a power limitation provided in 'ChargeParameterDiscoveryRes'.

[V2G2-479] The message 'PowerDeliveryRes' shall contain the ResponseCode 'FAILED_TariffSelectionInvalid' if the content of attribute 'ChargingProfile' in the PowerDeliveryReq message contains a SAtupleID which was not contained in the 'SASchedules' attribute provided in 'ChargeParameterDiscoveryRes'.

[V2G2-480] The message 'PowerDeliveryRes' shall contain the ResponseCode 'FAILED_PowerDeliveryNotApplied' if the EVSE is not able to deliver energy.

[V2G2-481] The message 'MeteringReceiptRes' shall contain the ResponseCode 'FAILED_MeteringSignatureNotValid' if the SECC is not able to validate the signature, or the contained meter reading does not fit to the provided meter reading during 'ChargingStatusRes' and the SECC requires that the signature is valid.

NOTE ResponseCodes that are not defined in this chapter can be used implementation specific.

Table 106 shows an overview on the response coded and the messages as defined before.

Table 106 — Overview on application of ResponseCodes

Response Code (Enum)	V2G message type																											
	supportedAppProtocolReq	supportedAppProtocolRes	SessionSetupReq	SessionSetupRes	ServiceDiscoveryReq	ServiceDiscoveryRes	ServiceDetailReq	ServiceDetailRes	ServiceandPaymentSelectiReq	ServiceandPaymentSelectionRes	PaymentDetailsReq	PaymentDetailsRes	ContractAuthenticationReq	ContractAuthenticationRes	ChargeParameterDiscoveryReq	ChargeParameterDiscoveryRes	PowerDeliveryReq	PowerDeliveryRes	ChargingStatusReq	ChargingStatusRes	MeteringReceiptReq	MeteringReceiptRes	CertificateupdateReq	CertificateupdateRes	CertificateinstallationReq	CertificateinstallationRes	SessionStopReq	SessionStopRes
OK		x		x		x		x		x		x		x		x		x		x		x		x		x		x
OK_CertificateExpiresSoon																												
OK_NewSessionEstablished				x																								
OK_OldSessionJoined				x																								
FAILED		x		x		x		x		x		x		x		x		x		x		x		x		x		x
FAILED_SequenceError		x		x		x		x		x		x		x		x		x		x		x		x		x		x
FAILED_SignatureError		x		x		x		x		x		x		x		x		x		x		x		x		x		x
FAILED_UnknownSession						x		x		x		x		x		x		x		x		x		x		x		x
FAILED_ServiceIDInvalid								x																				
FAILED_Payment SelectionInvalid										x																		
FAILED_ServiceSelection Invalid										x																		
FAILED_CertificateExpired												x												x		x		
FAILED_NoCertificate Available																										x		
FAILED_CertChainError																								x				
FAILED_ContractCanceled																								x				
FAILED_ChallengeInvalid														x														
FAILED_WrongEnergy TransferType																x												
FAILED_WrongCharge Parameter																x												
FAILED_ChargingProfile Invalid																		x										
FAILED_TariffSelection Invalid																		x										
FAILED_PowerDelivery NotApplied																		x										
FAILED_MeteringSignature NotValid																						x						

8.8.3.2 AC Specific Requirements

Besides the requirements for Request-Response Message Pairs and Request-Response Message Sequences as defined in subclause 8.8.4 the EVCC and the SECC shall conform to the following requirements.

[V2G2-453] The EVCC shall be in state C or D as defined in IEC 61851-1 (IO-SET_CPSTATE.request (CPState=C or D) and IO-SET_CPSTATE.confirmation (CPState=C or D)) before sending the message Power Delivery Request with parameter 'ReadyToChargeState set to 'TRUE' as defined in subclause 8.4.1.9.2.

[V2G2-454] The SECC shall measure state C or D as defined in IEC 61851-1 (IO-SET_CPSTATE.indication (CPState=C or D)) before receiving a Power Delivery Request with parameter 'ReadyToChargeState set to 'TRUE' for sending a Power Delivery Response Message with parameter ResponseCode set to 'OK' otherwise it shall send ResponseCode set to 'FAILED'.

8.8.3.3 DC Specific Requirements

Besides the requirements for Request-Response Message Pairs and Request-Response Message Sequences as defined in subclause 8.8.4 the EVCC and the SECC shall conform to the following requirements.

[V2G2-669] The EVCC shall be in state C or D as defined in IEC 61851-1 (IO-SET_CPSTATE.request (CPState=C or D) and IO-SET_CPSTATE.confirmation (CPState=C or D)) before sending the message Cable Check Request as defined in subclause 8.4.3.2.2.

[V2G2-670] The SECC shall measure state C or D as defined in IEC 61851-1 (IO-SET_CPSTATE.indication (CPState=C or D)) before receiving a Cable Check Request for sending a Cable Check Response Message with parameter ResponseCode set to 'OK' otherwise it shall send ResponseCode set to 'FAILED'.

[V2G2-671] The response message shall contain the ResponseCode 'FAILED' as soon as the isolation monitor fails.

8.8.4 Request-Response Message Sequence Requirements

8.8.4.1 General Requirements

[V2G2-672] During a specific V2G Communication Session the SECC shall apply unique SAScheduleTupleIDs in the parameters SAScheduleTuple.

NOTE Unique identifiers are required to ensure that the EVCC and the SECC can refer to specific SASchedule during an entire V2G communication session, This also ensures that an EVCC can select a valid charging profile during renegotiation. In general, during renegotiation the EVCC can select the currently applied charging profile again or select a new charging profile based on the latest SASchedule provided by the SECC.

[V2G2-673] If the EVCC has no valid charging profile it shall calculate a charging profile that fits the parameter limits for a SAScheduleTuple in SASchedules received in the latest message ChargeParameterDiscoveryRes.

[V2G2-674] If the EVCC sends an valid parameter ChargingProfile in the message PowerDeliveryReq the EVSE shall ensure that the charging profile can be fulfilled for all entries ChargingProfileEntryStart and ChargingProfileEntryMaxPower.

[V2G2-675] In case of renegotiation the EVCC shall decide to either send the current applied parameter ChargingProfile (same SAScheduleTupleID as before the renegotiation) or to send a new parameter ChargingProfile that fits the limits of a SAScheduleTuple in SASchedules received in the latest message ChargeParameterDiscoveryRes.

[V2G2-676] If the EVCC sends a parameter ChargingProfile in the message PowerDeliveryReq that either fulfils the limits of a SAScheduleTuple provided in the parameter SASchedules sent by the SECC in the latest ChargeParameterDiscoveryRes or the currently applied charging profile (same SAScheduleTupleID as before the renegotiation, the EVSE shall ensure that the charging profile can be fulfilled for all entries in the parameter ChargingProfile.

[V2G2-677] If the SECC does not intend to send an Notification request to the EVCC it shall set parameter EVSENotification in EVSEStatus to value 'None'.

[V2G2-678] If the parameter EVSENotification in the EVSEStatus is equal to None the EVCC shall ignore the value of the parameter NotificationMaxDelay in the EVSEStatus.

[V2G2-679] If the parameter EVSENotification in EVSEStatus is equal to StopCharging, the EV should stop charging within the number of seconds provided in NotificationMaxDelay.

NOTE After indication of a StopCharging the SECC may stop the charging from EVSE side e.g. by turning-off the pilot signal or opening the mains contactors after the duration of NotificationMaxDelay.

[V2G2-680] If the parameter EVSENotification in EVSEStatus is equal to ReNegotiation, the EV shall initiate a renegotiation within the number of seconds provided in NotificationMaxDelay (refer to [V2G2-521], [V2G2-522] and [V2G2-686]).

NOTE If a renegotiation initiated by the EVSE is accepted or not by the EV is out of scope of this specification, but depends on the specific context of the charging process e.g. the closed charging contract. A renegotiation can be in severe conflict with the user expectation and is discouraged if it is not explicitly indicated beforehand to the user during the first Negotiation. Thus the renegotiation only provides technical means to support use cases which have to be cleared beforehand with the user.

8.8.4.2 EVCC

The EVCC behavior defining all valid Request-Response Message Sequences for AC is shown in Figure 95 and for DC is shown in Figure 96.

8.8.4.2.1 Common Requirements

[V2G2-482] The EVCC shall stop the V2G Communication Session whenever it receives a response message that does not correspond to the last request message sent.

NOTE This means for example that the EVCC shall only accept an SessionSetupRes if the message sent before was a SessionSetupReq message.

[V2G2-483] The EVCC shall send a supportedAppProtocolReq.

[V2G2-484] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout of 'supportedAppProtocolRes' according to Table 103.

[V2G2-485] After receiving the supportedAppProtocolRes, the EVCC shall send a SessionSetupReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-486] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'SessionSetupRes' according to Table 103.

[V2G2-487] After receiving the SessionSetupRes, the EVCC shall send a ServiceDiscoveryReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-488] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of ' ServiceDiscoveryRes' according to Table 103.

[V2G2-489] After receiving the ServiceDiscoveryRes, the EVCC shall send a ServiceDetailReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the SECC offers a ServiceList in ServiceDiscoveryRes and the EVCC intends to use a service from the ServiceList.

[V2G2-509] After receiving the ServicePaymentSelectionRes, the EVCC shall send a ContractAuthenticationReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-490] After receiving the ServiceDiscoveryRes, the EVCC shall send a ServicePaymentSelectionReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if no service details are necessary for the remaining process.

[V2G2-491] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'ServiceDetailRes' according to Table 103.

[V2G2-492] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'ServicePaymentSelectionRes' according to Table 103.

[V2G2-493] After receiving the ServiceDetailRes, the EVCC shall send a ServicePaymentSelectionReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if there are no further ServiceDetailReq intended.

[V2G2-494] After receiving the ServiceDetailRes, the EVCC shall send an additional ServiceDetailReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if further ServiceDetailReq are necessary to retrieve the detailed information from the SECC.

[V2G2-495] After receiving the ServicePaymentSelectionRes, the EVCC shall send a PaymentDetailsReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the Message Set “AC Charing PnC” is selected and the EVCC does not intend to use the Message Sets “Certificate Install” or “Certificate Update”.

[V2G2-496] After receiving the ServicePaymentSelectionRes, the EVCC shall send a CertifiacteInstallationReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time., if the ServiceDetailRes indicates that CertificateInstall is available and the EVCC wants to use the Message Set “Certificate Install".

[V2G2-497] After receiving the ServicePaymentSelectionRes, the EVCC shall send a CertifiacteUpdateReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the ServiceDetailRes indicates that CertificateInstall is available and the EVCC wants to use the Message Set “Certificate Update".

[V2G2-498] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'CertifiacteInstallationRes' according to Table 103.

[V2G2-499] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout of 'CertifiacteUpdateRes' according to Table 103.

[V2G2-500] After receiving the CertificateInstallationRes, the EVCC shall send a PaymentDetailsReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-501] After receiving the CertificateUpdateRes, the EVCC shall send a PaymentDetailsReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-502] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'PaymentDetailsRes' according to Table 103.

[V2G2-503] After receiving the PaymentDetailsRes, the EVCC shall send a ContractAuthenticationReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-504] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'ContractAuthenticationRes' according to Table 103.

[V2G2-505] After receiving the ContractAuthenticationRes with parameter 'EVSEProcessing' set to 'Finished', the EVCC shall send a ChargeParameterDiscoveryReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-506] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'ChargeParameterDiscoveryRes' according to Table 103.

[V2G2-507] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'SessionStopRes' according to Table 103.

[V2G2-508] After receiving the SessionStopRes, the EVCC shall terminate the V2G Communication.

The EVCC may renegotiate the charging schedule as follows:

[V2G2-683] If an EVCC initiated an renegotiation (refer to [V2G2-521], [V2G2-522] and [V2G2-686]) and receives the PowerDeliveryRes the EVCC shall send a ChargeParameterDiscoveryReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

If the SECC suspends a message sequence by EVSEProcessing set to 'Ongoing' the following applies for the EVCC:

[V2G2-684] After receiving the ContractAuthenticationRes, the EVCC shall resend the identical ContractAuthenticationReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time as long as the parameter EVSEProcessing is equal to 'Ongoing'.

[V2G2-685] After receiving the ChargeParameterDiscoveryRes, the EVCC shall resend the identical ChargeParameterDiscoveryReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time as long as the parameter EVSEProcessing is equal to 'Ongoing'.

8.8.4.2.2 AC specific requirements

[V2G2-510] After receiving the ChargeParameterDiscoveryRes with parameter 'EVSEProcessing' set to 'Finished', the EVCC shall send a PowerDeliveryReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-511] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'ChargingStatusRes' according to Table 103.

[V2G2-512] After receiving the ChargingStatusRes, the EVCC shall send a MeteringReceiptReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if Message Set "Meter Status Receipt" is selected, indicated if the Parameter "Receipt Required" is set to value "TRUE" in the message "ChargingStatusRes".

[V2G2-513] After receiving the ChargingStatusRes, the EVCC shall send a PowerDeliveryReq with parameter 'ReadyToChargeState is set to FALSE', while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the charge process is stopped in the EV and Message Set "Meter Status Receipt" is not selected which means that the Parameter "Receipt Required" was set to value "FALSE" in the message "ChargingStatusRes".

[V2G2-514] After receiving the PowerDeliveryRes, the EVCC shall send a ChargingStatusReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-515] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'PowerDeliveryRes' according to Table 103.

[V2G2-516] After receiving the ChargingStatusRes, the EVCC shall send a ChargingStatusReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the charge process is continued and Message Set "Meter Status Receipt" is not selected which means that the Parameter "Receipt Required" was set to value "FALSE" in the message "ChargingStatusRes".

[V2G2-517] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'MeteringReceiptRes' according to Table 103.

[V2G2-518] After receiving the MeteringReceiptRes, the EVCC shall send a ChargingStatusReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time., if the charge process is continued.

[V2G2-519] After receiving the MeteringReceiptRes, the EVCC shall send a PowerDeliveryReq with parameter ReadyToChargeState set to FALSE, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the charge process is stopped in the EV.

[V2G2-520] After receiving the PowerDeliveryRes, the EVCC shall send a SessionStopReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if a PowerDeliveryReq with parameter ReadyToChargeState is set to FALSE was sent.

The EVCC may renegotiate the charging schedule as follows:

[V2G2-521] An EV can decide to perform a renegotiation for adapting its charge schedule as follows. After ChargingStatusRes has been received and the Parameter "Receipt Required" set to value "FALSE", it shall initiate a renegotiation by sending a PowerDeliveryReq with parameter 'ReadyToChargeState is set to FALSE', while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-522] An EV can decided to perform a renegotiation for adapting its charge schedule as follows. After MeteringReceiptRes has been received, the EVCC shall initiate a renegotiation by sending a PowerDeliveryReq with parameter 'ReadyToChargeState is set to FALSE', while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-689] If the parameter EVSENotification in EVSEStatus is equal to ReNegotiation in ChargingStatusRes, the EV shall initiate a renegotiation as described in [V2G2-521] and [V2G2-522] within the number of seconds provided in NotificationMaxDelay.

NOTE The EV can decide to renegotiate by applying the process as described in [V2G2-521] and [V2G2-522]. The EVSE triggered renegotiation is basically the same process as but triggered by the EVSE as described in [V2G2-689].

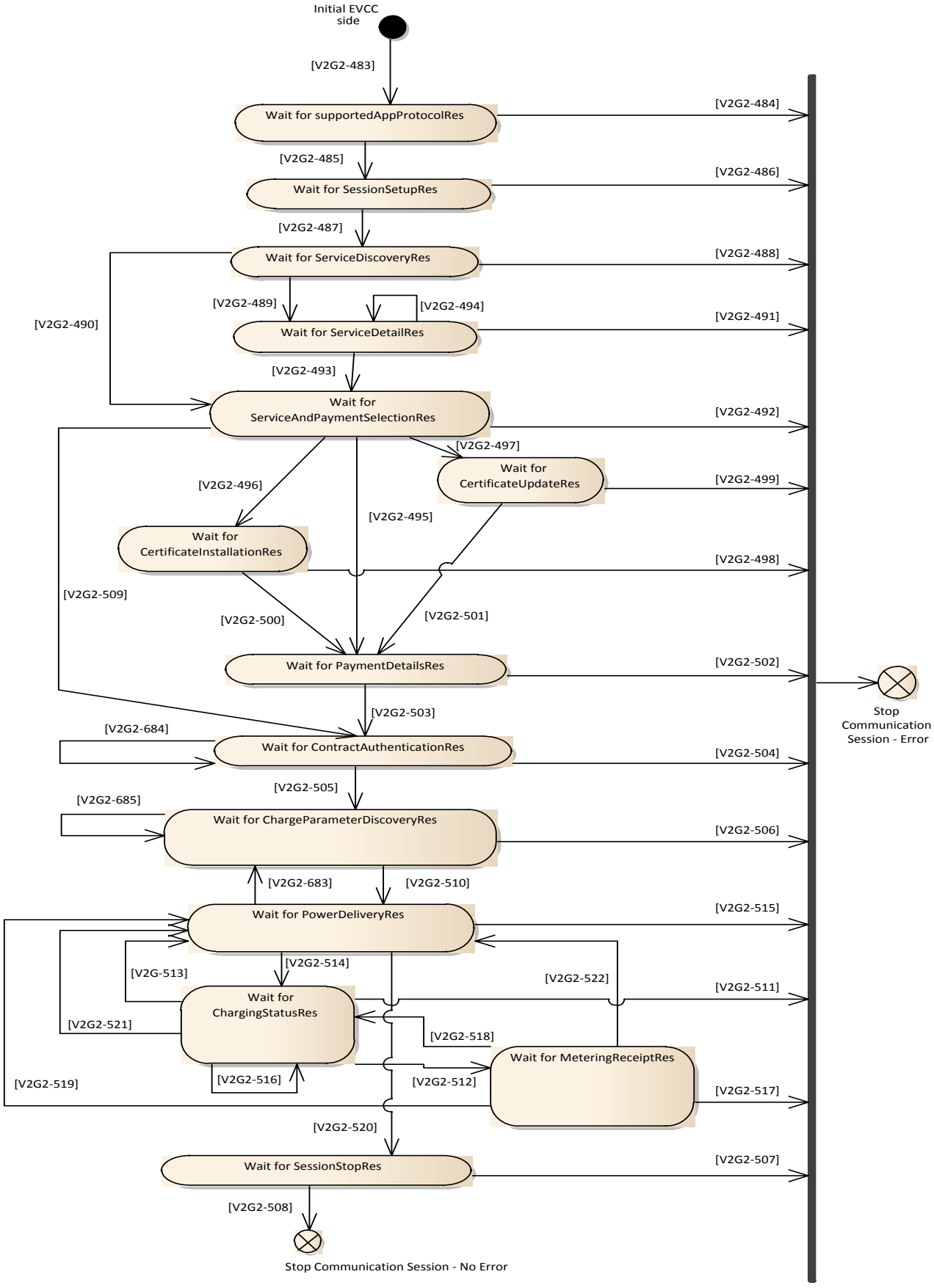


Figure 95 — EVCC Communication states for AC V2G messaging

8.8.4.2.3 DC specific requirements

[V2G2-657] The EVCC shall be in state C or D as defined in IEC 61851-1 (IO-SET_CPSTATE.request (CPState=C or D) and IO-SET_CPSTATE.confirmation (CPState=C or D)) before sending the message Cable Check Request as defined in subclause 8.4.3.2.2.

[V2G2-599] After receiving the ChargeParameterDiscoveryRes, the EVCC shall send a CableCheckReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-524] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'CableCeckRes' according to Table 103.

[V2G2-617] After receiving the CableCheckRes, the EVCC shall resend the identical CableCheckReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time as long as the parameter EVSEProcessing is equal to 'Ongoing'.

[V2G2-525] After receiving the CableCeckRes with parameter ‘EVSEProcessing' set to 'Finished’, the EVCC shall send a PreChargeReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time and the parameter.

[V2G2-526] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of ' PreChargeReq ' according to Table 103.

[V2G2-527] After receiving the CurrentDemandRes, the EVCC shall send a PowerDeliveryReq with parameter ‘ReadyToChargeState is set to FALSE’, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the charge process is stopped.

[V2G2-618] After receiving the PreChargeRes, the EVCC shall send a PreChargeReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time and the value of the parameter EVSEPresentVoltage does not fulfil the voltage threshold requirement of the EV.

NOTE In addition the EV may utilize internal voltage measurement methods for evaluating the input voltage value received in the PreChargeRes message (EVSEPresentVoltage).

[V2G2-528] After receiving the PreChargeRes, the EVCC shall send a PowerDeliveryReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time and the value of the parameter EVSEPresentVoltage fulfils the voltage threshold requirement of the EV.

[V2G2-529] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'PowerDeliveryRes' according to Table 103.

[V2G2-530] After receiving the PowerDeliveryRes, the EVCC shall send a CurrentDemandReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-619] After receiving the PowerDeliveryRes, the EVCC shall send a SessionStopReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if a PowerDeliveryReq with parameter ReadyToChargeState is set to FALSE was sent.

[V2G2-531] After receiving the CurrentDemandRes, the EVCC shall send a CurrentDemandReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the EV wants to continue the charging process.

[V2G2-532] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'CurrentDemandRes' according to Table 103.

[V2G2-533] After receiving the PowerDeliveryRes, the EVCC shall send a WeldingDetectionReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if a PowerDeliveryReq with parameter ReadyToChargeState is set to FALSE was sent.

[V2G2-534] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAIL' of 'WeldingDetectionRes' according to Table 103.

[V2G2-620] After receiving the WeldingDetectionRes, the EVCC shall send a WeldingDetectionReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time and Welding Detection function has not finished on EV side.

[V2G2-535] After receiving the WeldingDetectionRes, the EVCC shall send a SessionStopReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-686] After CurrentDemandRes has been received and the EVCC intends to renegotiate, it shall initiate a renegotiation by sending a PowerDeliveryReq with parameter 'ReadyToChargeState is set to FALSE', while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

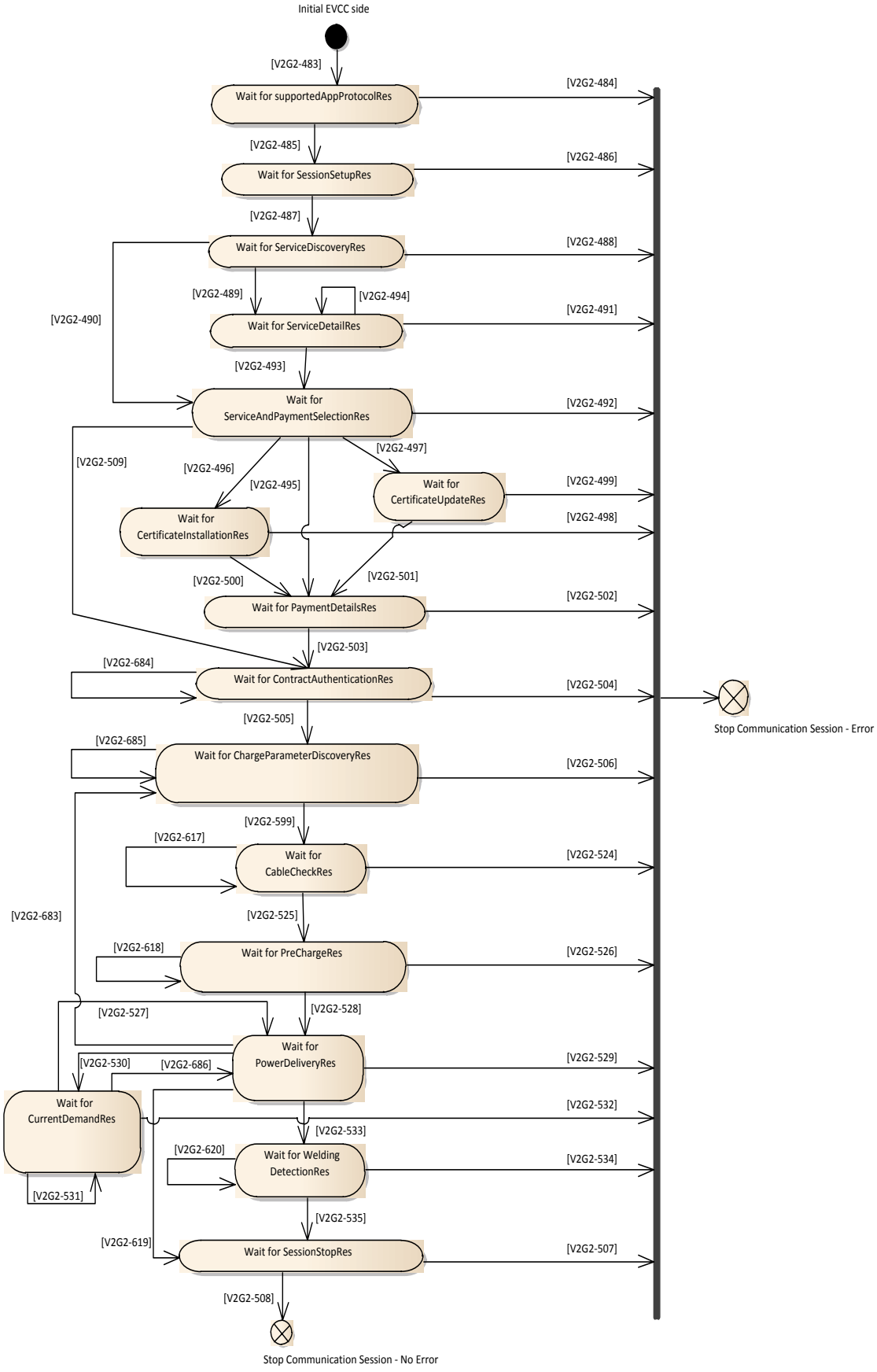


Figure 96 — EVCC Communication states for DC V2G messaging

8.8.4.3 SECC

The SECC behavior defining all valid Request-Response Message Sequences for AC is shown in Figure 97 and for DC in Figure 98.

8.8.4.3.1 Common Requirements

[V2G2-536] The SECC shall enter a wait state for supportedAppProtocolReq, set the timeout V2G_SECC_Sequence_Timeout to the value MessageType as defined in Table 103, reset the V2G_SECC_Sequence_Timer and start monitoring the V2G_SECC_Sequence_Timer.

NOTE Before the first message the SECC did not send any response message. Therefore the SECC has to start its Sequence Timer when starting to wait for the first message.

[V2G2-537] The SECC shall stop the V2G Communication Session when V2G_SECC_Sequence_Timer is equal or larger than V2G_SECC_Sequence_Timeout according to Table 103.

[V2G2-538] The SECC shall respond with the corresponding response message containing a “ResponseCode = FAILED_SequenceError' within V2G_SECC_Msg_Performance_Time according to Table 103, if request message was received that the SECC does not expect in the wait state.

[V2G2-539] The V2G Communication Session shall be stopped with an error, after sending the corresponding response message for the request message.

[V2G2-540] After receiving a supportedAppProtocolReq, the SECC shall process the received information.

[V2G2-541] The SECC shall respond with a supportedAppProtocolRes within V2G_SECC_Msg_Performance_Time according to Table 103. The allowed next request shall be ServiceSetupReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-542] After receiving a SessionSetupReq, the SECC shall process the received information.

[V2G2-543] The SECC shall respond with a SessionSetupRes containing “ResponseCode = OK' within V2G_SECC_Msg_Performance_Time according to Table 103. The allowed next request shall be ServiceDiscoveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-544] After receiving a ServiceDiscoveryReq, the SECC shall process the received information.

[V2G2-545] The SECC shall respond with a ServiceDiscoveryRes containing “ResponseCode = OK' within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is passed successfully. The allowed next request shall be ServiceDetailReq and ServiceAndPaymentSelectionReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-546] The SECC shall respond with ServiceDiscoveryRes containing “ResponseCode=FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-547] After receiving a ServiceDetailReq, the SECC shall process the received information.

[V2G2-548] The SECC shall respond with ServiceDetailRes containing “ResponseCode = OK' within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next request shall be ServiceDetailReq and ServiceAndPaymentSelectionReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-549] The SECC shall respond with ServiceDetailRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-550] After receiving a ServicePaymentSelectionReq, the SECC shall process the received information.

[V2G2-551] The SECC shall respond with ServicePaymentSelectionRes containing “ResponseCode = OK' within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next request shall be PaymentDetailsReq, CertificateInstallationReq and CertificateUpdateReq if Message Set “AC Charging PnC” is selected and ContractAuthenticationReq if Message Set “AC Charging EIM” is selected. V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-552] The SECC shall respond with ServicePaymentSelectionRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-553] After receiving a CertificateInstallationReq, the SECC shall process the received information.

[V2G2-554] The SECC shall respond with CertificateInstallationRes containing “ResponseCode = OK' within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next request shall be PaymentDetailsReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-555] The SECC shall respond with CertificateInstallationRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-556] After receiving a CertificateUpdateReq, the SECC shall process the received information.

[V2G2-557] The SECC shall respond CertificateUpdateRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next request shall be PaymentDetailsReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-558] The SECC shall respond with CertificateUpdateRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful. The allowed next request shall be PaymentDetailsReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-559] After receiving a PaymentDetailsReq, the SECC shall process the received information.

[V2G2-560] The SECC shall respond with PaymentDetailsRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next request shall be ContractAuthenticationReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-561] The SECC shall respond with PaymentDetailsRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103.

[V2G2-562] After receiving a ContractAuthenticationReq, the SECC shall process the received information.

[V2G2-563] The SECC shall respond with ContractAuthenticationRes containing “ResponseCode = OK” and "EVSEProcessing=Finished" within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed and the authentication is finished. The allowed next request shall be ChargeParameterDiscoveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-564] The SECC shall respond with ContractAuthenticationRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-565] After receiving a ChargeParameterDiscoveryReq, the SECC shall process the received information.

[V2G2-566] The SECC shall respond with ChargeParameterDiscoveryRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-567] After receiving a PowerDeliveryReq, the SECC shall process the received information.

[V2G2-568] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed and the request contained “ReadyToChargeState is set to FALSE”. The allowed next request shall be SessionStopReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-569] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-570] After receiving a SessionStopReq, the SECC shall process the received information and start the V2G_SECC_Msg_Performance_Timer.

[V2G2-571] The SECC shall respond with SessionStopRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The V2G Communication Session is then stopped without error.

[V2G2-572] The SECC shall respond with SessionStopRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-687] The SECC shall respond with ContractAuthenticationRes containing “ResponseCode = OK” and "EVSEProcessing=Ongoing" within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed and the authentication is ongoing. The allowed next request shall be ContractAuthenticationReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-688] The SECC shall respond with ChargeParameterDiscoveryRes containing “ResponseCode = OK” and "EVSEProcessing=Ongoing" and no parameter SASchedule within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed and the calculation of the parameter SASchedule is ongoing. The allowed next request shall be ChargeParameterDiscoveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

8.8.4.3.2 AC specific requirements

[V2G2-573] The SECC shall respond with ChargeParameterDiscoveryRes containing “ResponseCode = OK” and "EVSEProcessing=Finished" and a valid parameter SASchedule within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next request shall be set to PowerDeliveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-574] After receiving a ChargingStatusReq, the SECC shall process the received information.

[V2G2-575] The SECC shall respond with ChargingStatusRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The SECC sets “ReceiptRequired = FALSE" indicating that the Message Set MessageReceipt shall not be used by the EVCC. The allowed next request shall be ChargingStatusReq, PowerDeliveryReq and ChargeParameterDiscoveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-576] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed and the request contained “ReadyToChargeState is set to True”. The allowed next request shall be ChargingStatusReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-577] The SECC shall respond with ChargingStatusRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The SECC sets “ReceiptRequired = TRUE" indicating that the Message Set MessageReceipt shall be used by the EVCC. The allowed next request shall be MeteringReceiptReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-578] The SECC shall respond with ChargingStatusRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-579] After receiving a MeteringReceiptReq, the SECC shall process the received information.

[V2G2-580] The SECC shall respond with MeteringReceiptRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next request shall be ChargingStatusReq, PowerDeliveryReq and ChargeParameterDiscoveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-581] The SECC shall respond with MeteringReceiptRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

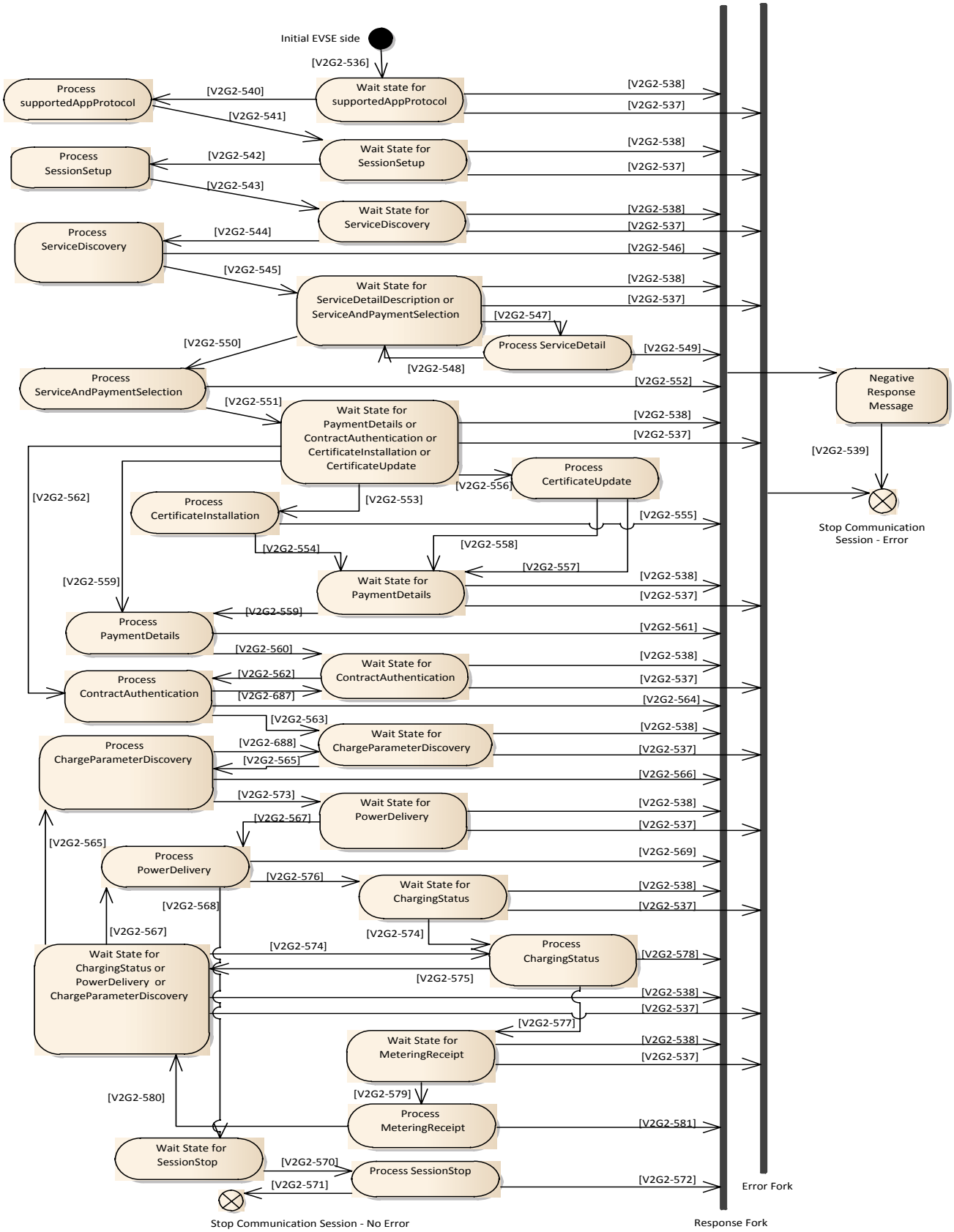


Figure 97 — SECC Communication states for AC V2G messaging

8.8.4.3.3 DC specific requirements

[V2G2-582] The SECC shall respond with ChargeParameterDiscoveryRes containing “ResponseCode = OK” and "EVSEProcessing=Finished" and a valid parameter SASchedule within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next request shall be CableCheckReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-583] After receiving a CableCheckReq, the SECC shall process the received information.

[V2G2-658] The SECC shall measure state C or D as defined in IEC 61851-1 (IO-SET_CPSTATE.indication (CPState=C or D)) before receiving a Cable Check Request for sending a Cable Check Response Message with parameter ResponseCode set to 'OK' otherwise it shall send ResponseCode set to 'FAILED'.

[V2G2-584] The SECC shall respond with CableCheckRes containing “ResponseCode = OK” and "EVSEProcessing=Finished" within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed and cable check is finnished. The allowed next request shall be PreChargeReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-621] The SECC shall respond with CableCheckRes containing “ResponseCode = OK” and "EVSEProcessing=Ongoing" within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed and the Cable Check is ongoing. The allowed next request shall be CableCheckReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-585] The SECC shall respond with CableCheckRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-586] After receiving a PrechargeReq, the SECC shall process the received information and start the V2G_SECC_Msg_Performance_Timer.

[V2G2-587] The SECC shall respond with PreChargeRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next requests shall be PrechargeReq and PowerDeliveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-588] The SECC shall respond with PreChargeRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-589] After receiving a PowerDeliveryReq, the SECC shall process the received information and start the V2G_SECC_Msg_Performance_Timer.

[V2G2-590] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed and the request contained “ReadyToChargeState = TRUE. The allowed next request shall be CurrentDemmandReq and PowerDeliveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-601] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed and the request contained “ReadyToChargeState = FALSE”. The allowed next request shall be ChargeParameterDiscoveryReq and WeldingDetectionReq and SessionStopReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-591] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-592] After receiving a CurrentDemandReq, the SECC shall process the received information and start the V2G_SECC_Msg_Performance_Timer.

[V2G2-593] The SECC shall respond with CurrentDemandRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next request shall be CurrentDemandReq and PowerDeliveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-595] The SECC shall respond with CurrentDemandRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is not successful.

[V2G2-596] After receiving a WeldingDetectionReq, the SECC shall process the received information and start the V2G_SECC_Msg_Performance_Timer.

[V2G2-597] The SECC shall respond with WeldingDetectionRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 103, if the processing of the information is successfully passed. The allowed next request shall be WeldingDetectionReq and SessionStopReq and the V2G_SECC_Sequence_Timeout is set according to Table 103.

[V2G2-598] The SECC shall respond with WeldingDetectionyRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 103 if the processing of the information is not successful.

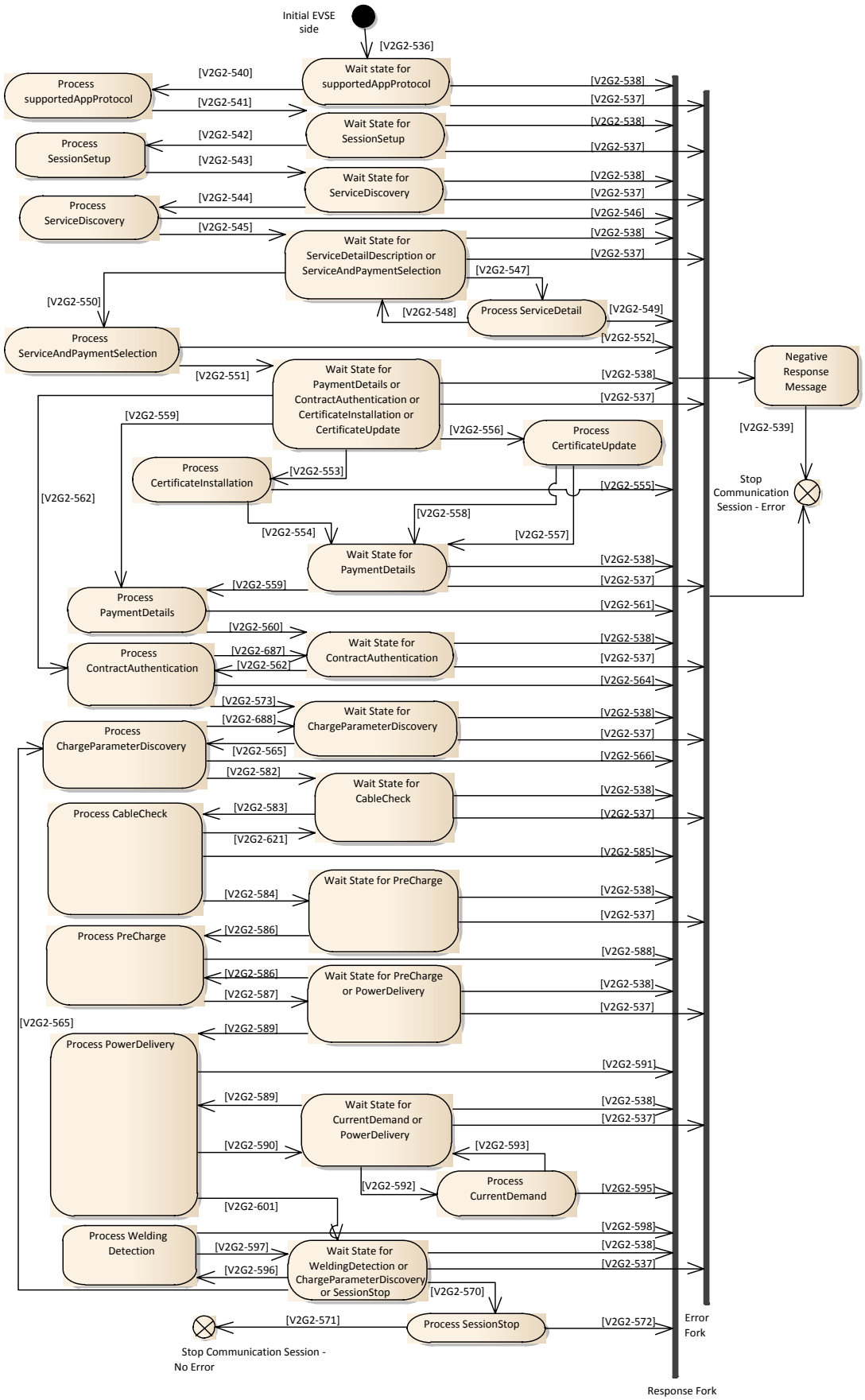


Figure 98 — SECC Communication states for DC V2G messaging

8.9 Request-Response Message Sequence Examples

8.9.1 AC

8.9.1.1 EIM

Figure 99 depicts an example for a Request-Response Message Sequence for the EIM identification mode without any errors including optional messages.

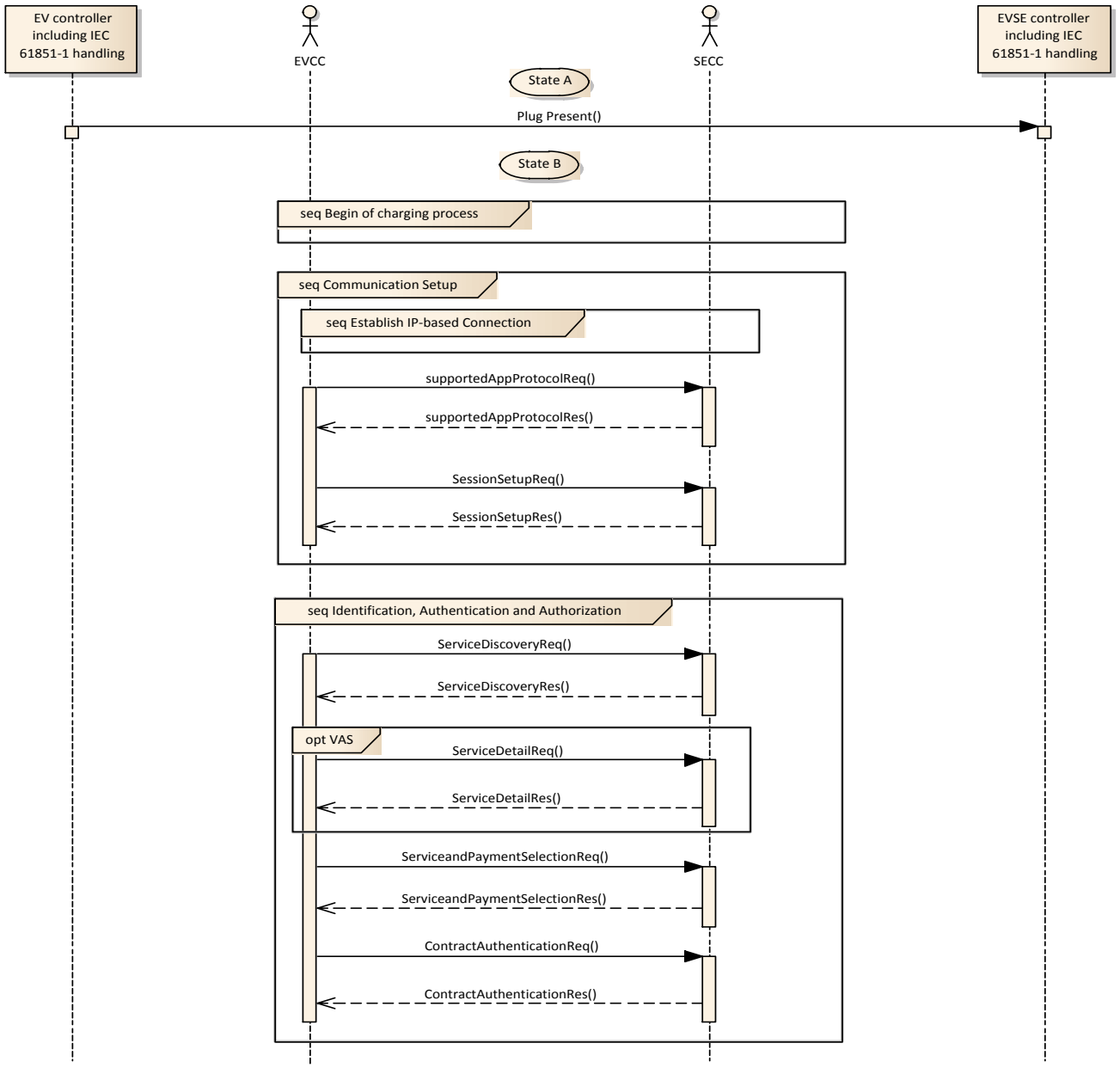


Figure 99 — Overview AC Request-Response Message Sequence EIM identification mode (1 of 2)

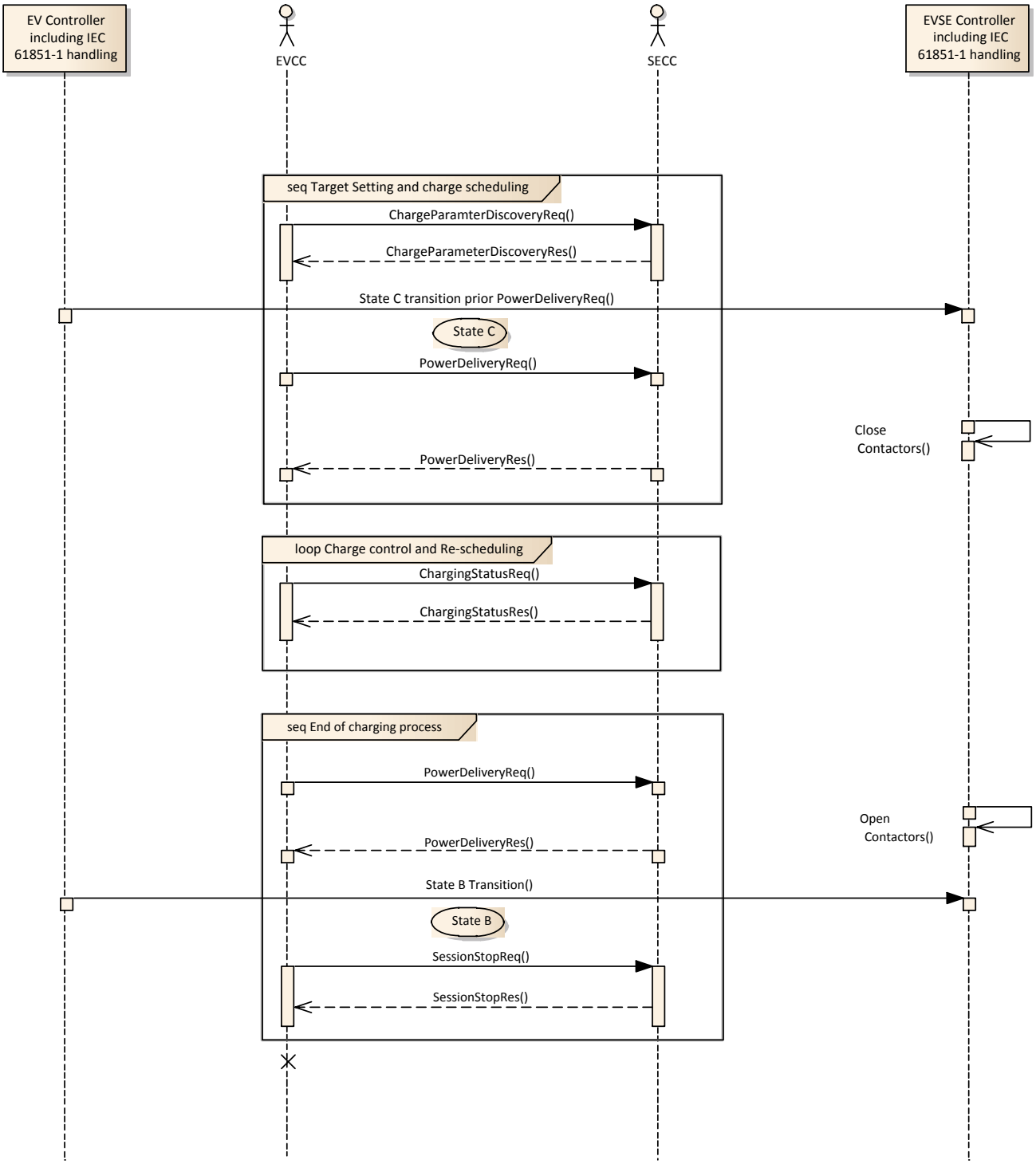


Figure 99— Overview AC Request-Response Message Sequence EIM identification mode (2 of 2)

8.9.1.2 PnC

Figure 100 depicts an example for a Request-Response Message Sequence for the PnC identification mode without any errors including optional messages.

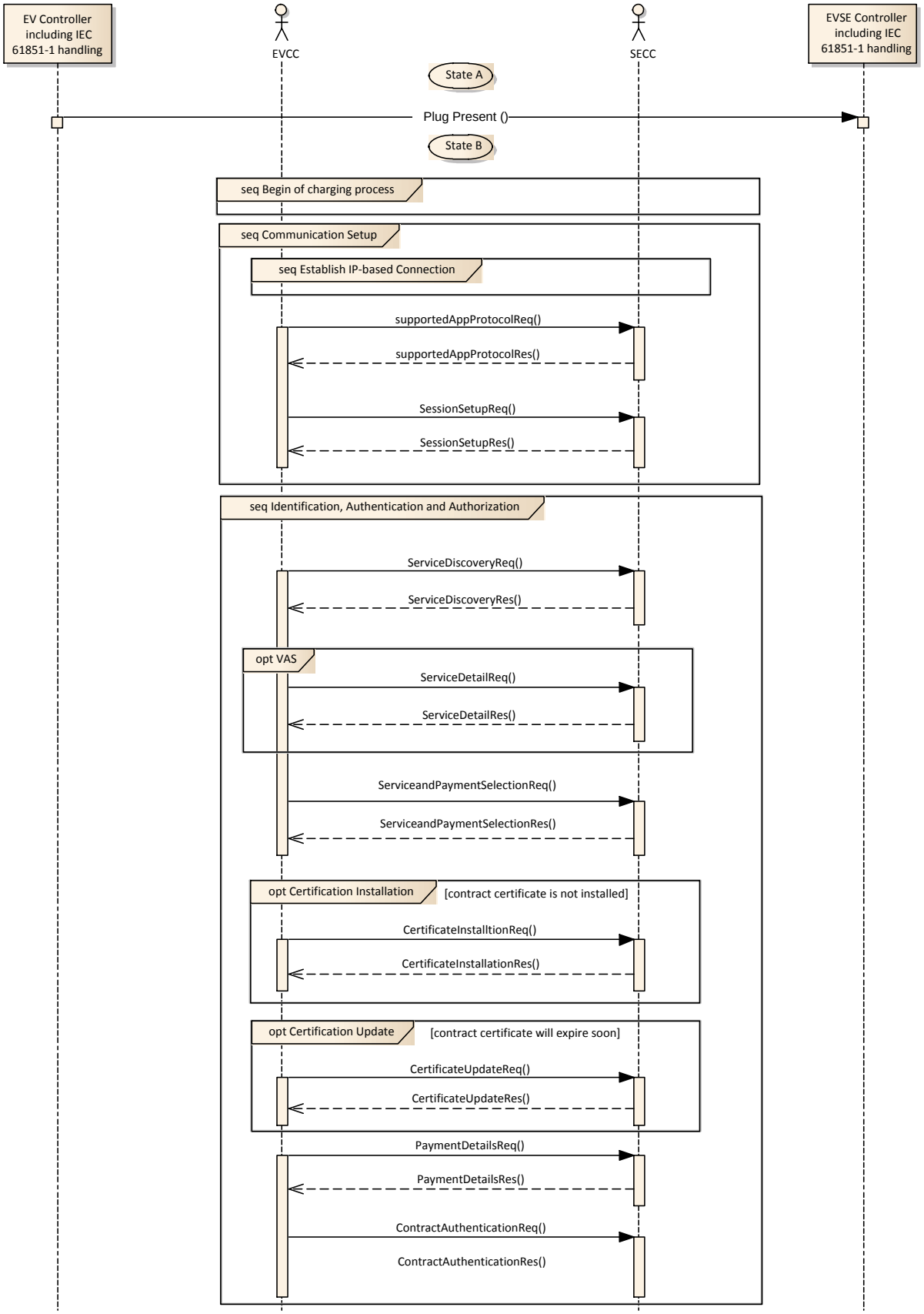


Figure 100 — Overview AC Request-Response Message Sequence PnC identification mode (1 of 2)

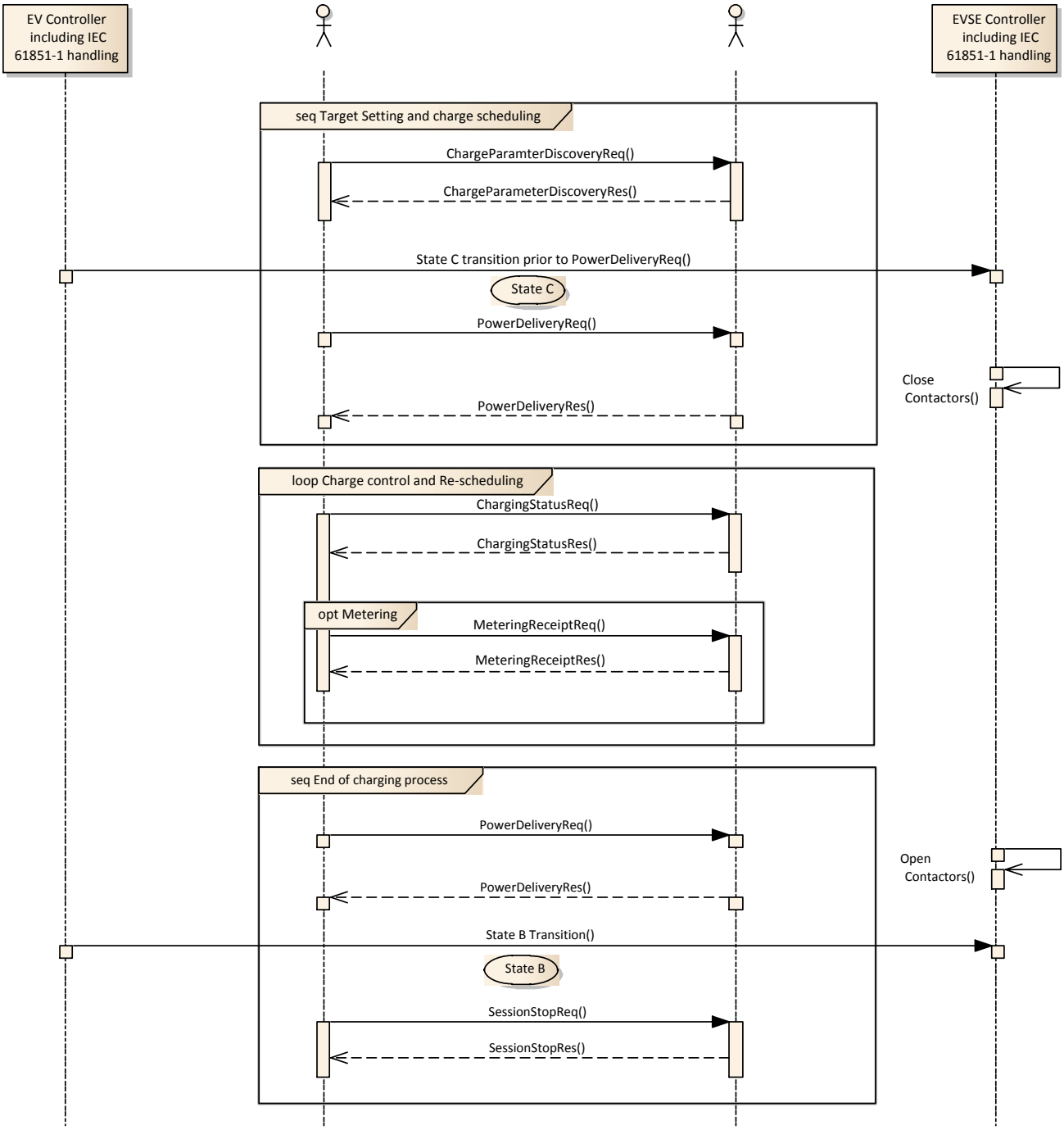


Figure 100— Overview AC Request-Response Message Sequence PnC identification mode (2 of 2)

8.9.2 DC

8.9.2.1 EIM

Figure 101 depicts an example for a Request-Response Message Sequence in EIM identification mode without any errors including optional messages.

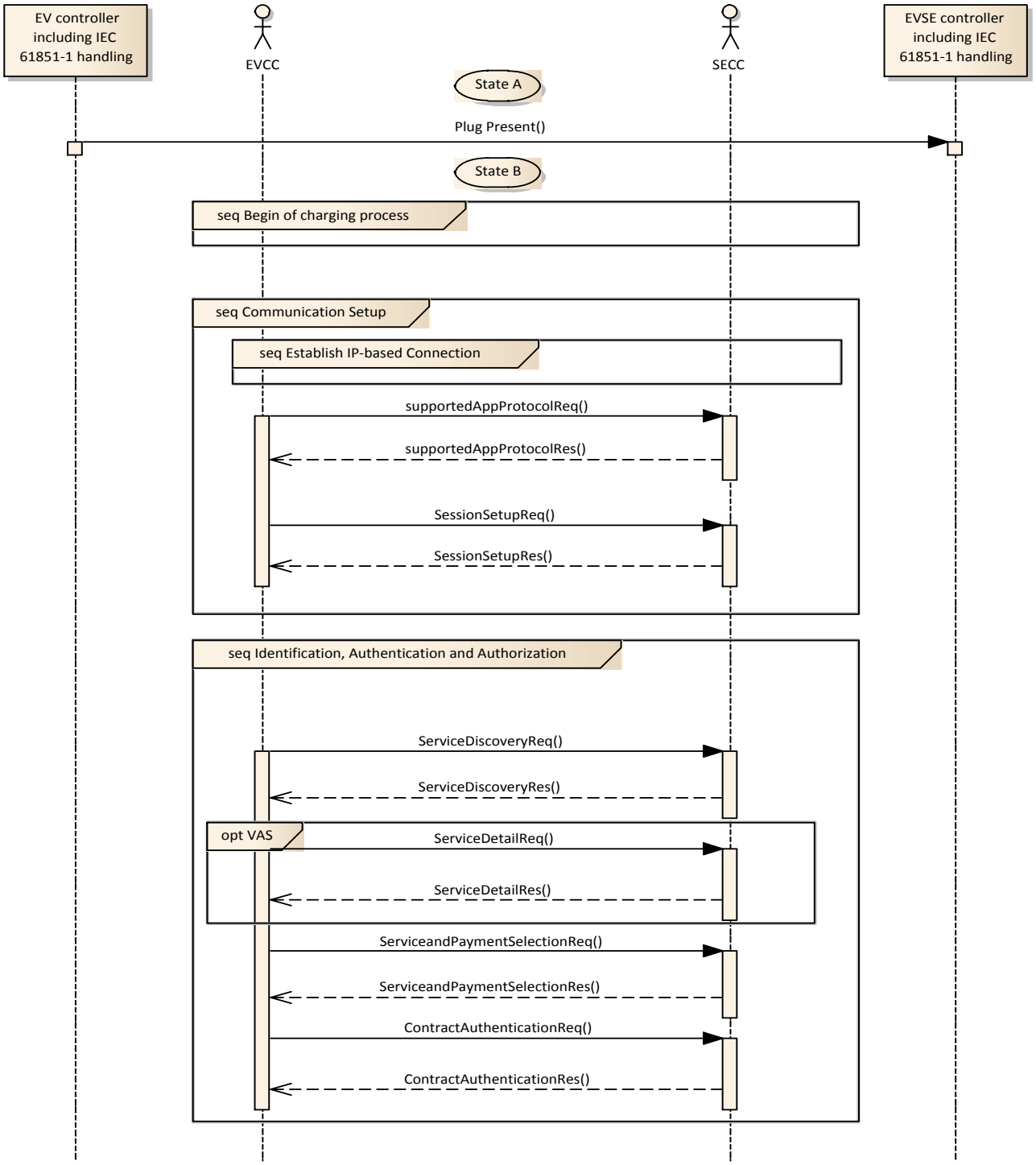


Figure 101 — Overview DC Request-Response Message Sequence EIM identification mode (1 of 2)

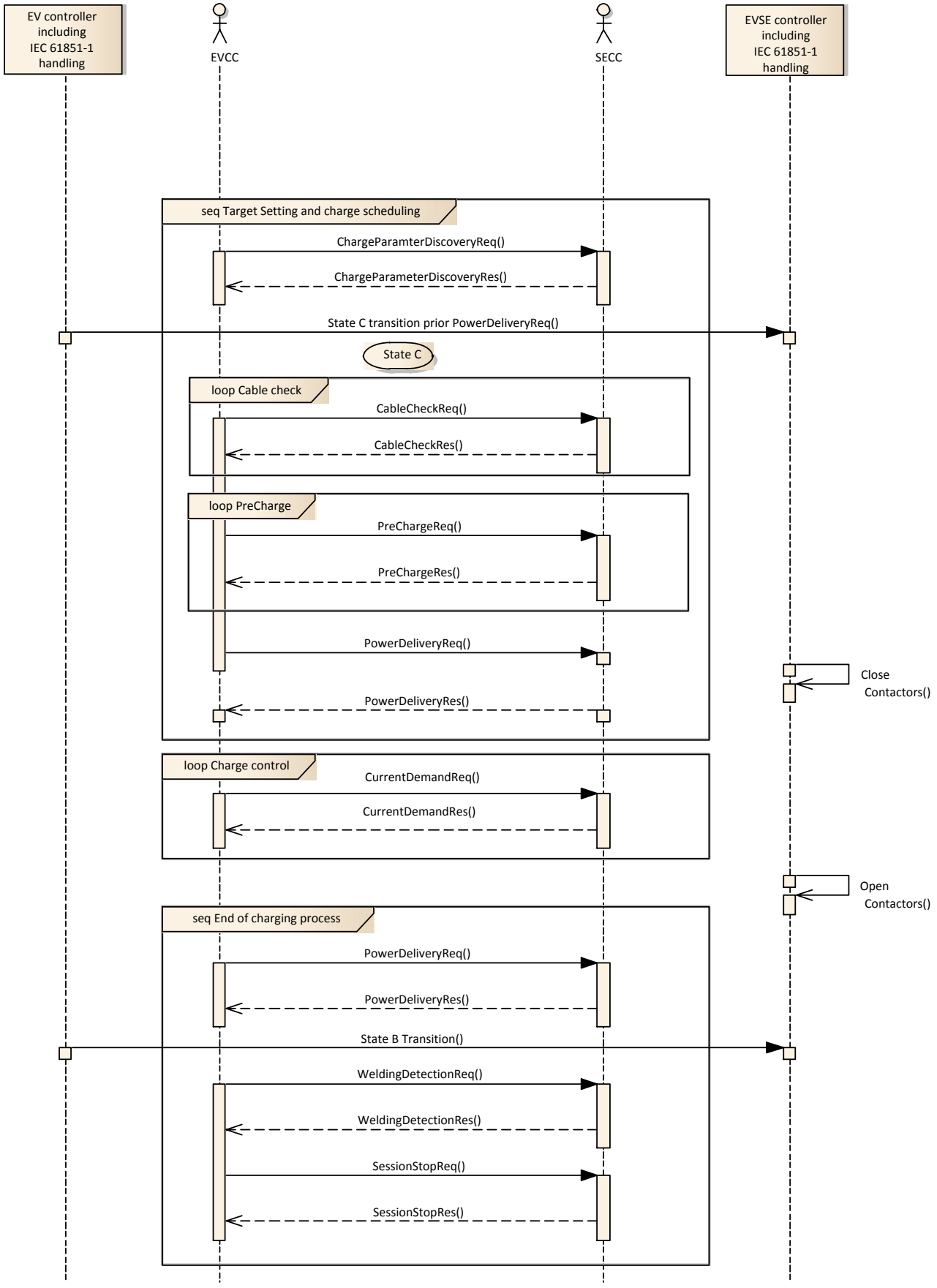


Figure 101— Overview DC Request-Response Message Sequence EIM identification mode (2 of 2)

8.9.2.2 PnC

Figure 102 depicts an example for a Request-Response Message Sequence in PnC identification mode without any errors including optional messages.

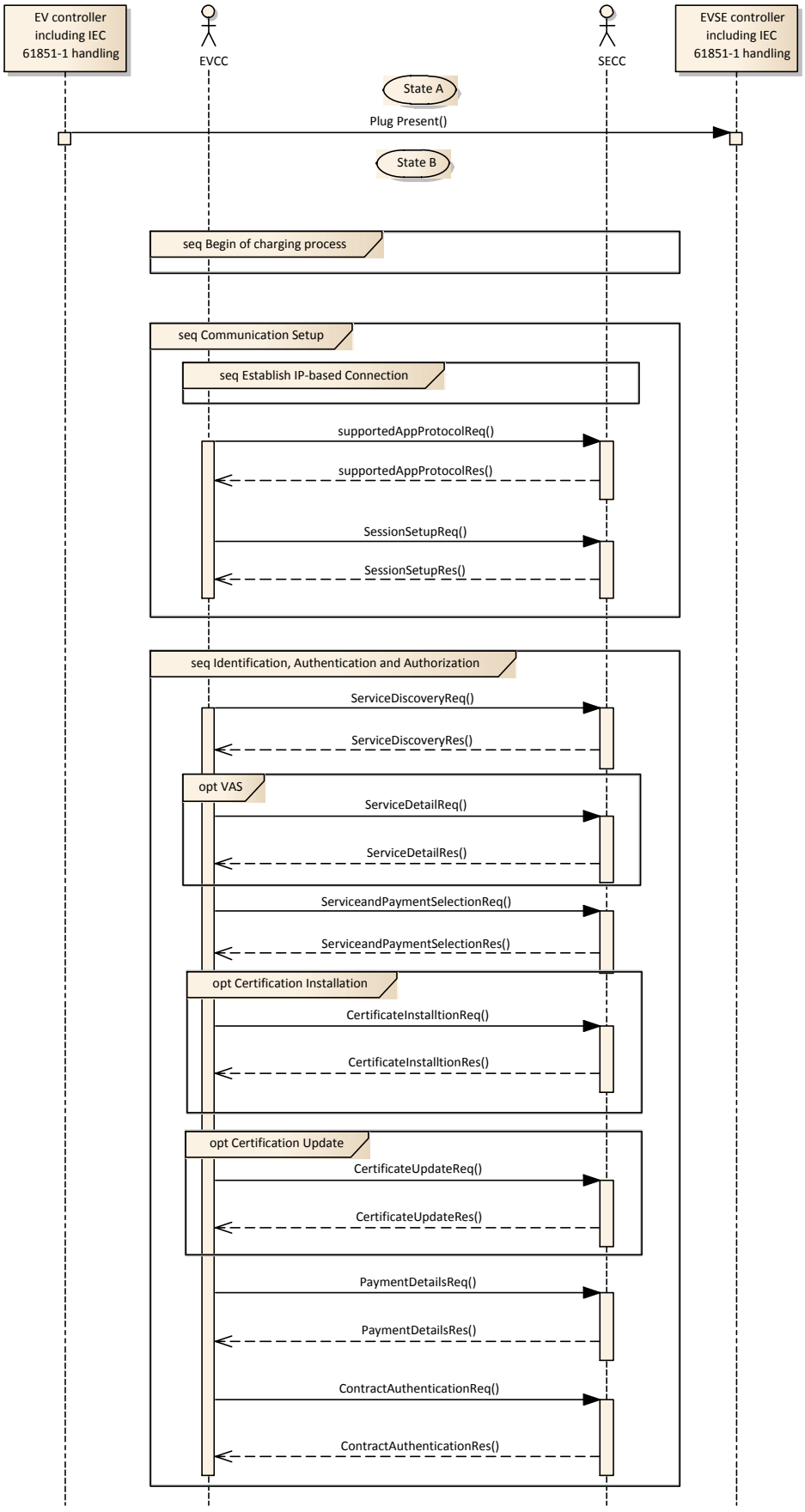


Figure 102 — Overview DC Request-Response Message Sequence PnC identification mode (1 of 2)

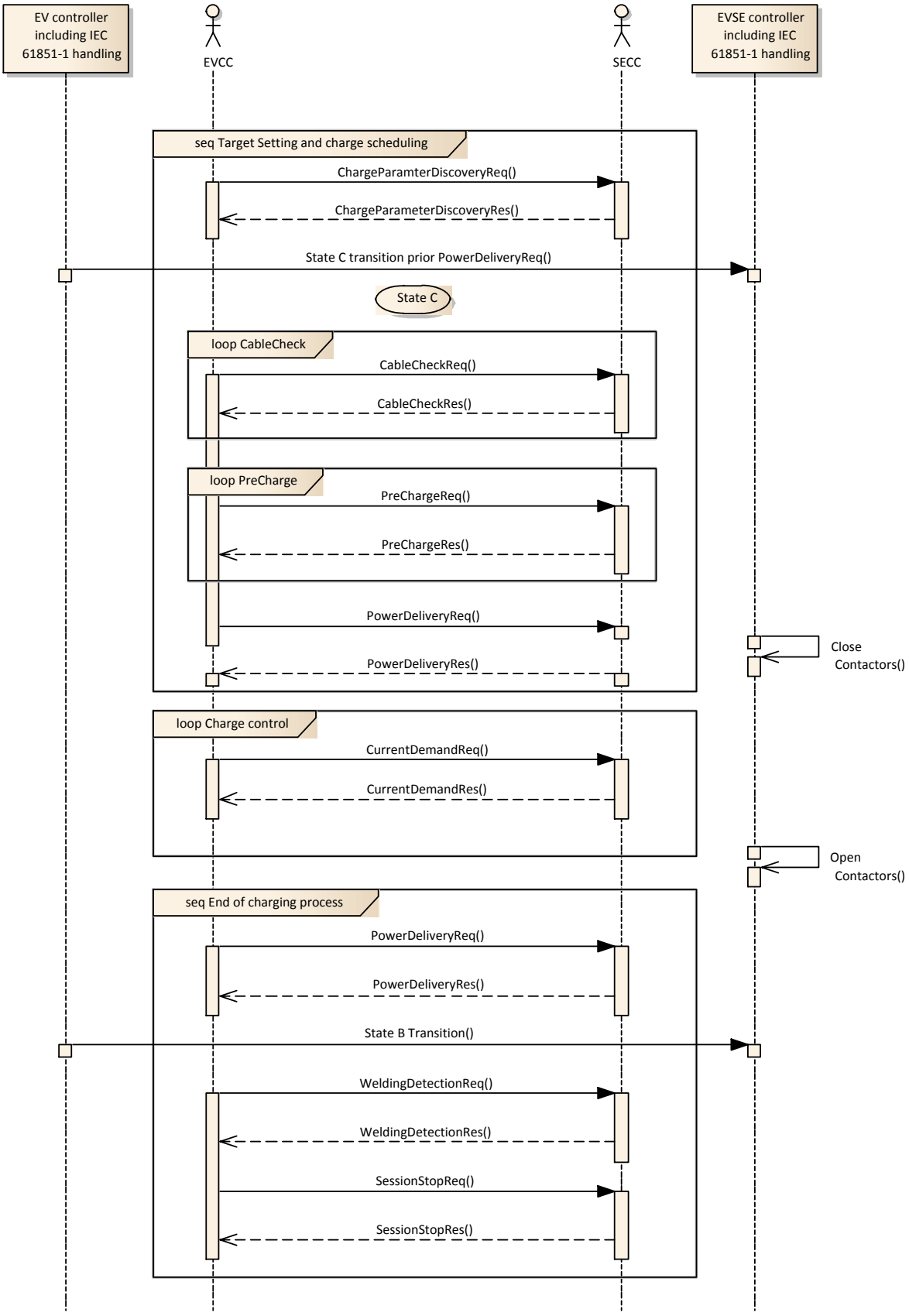


Figure 102— Overview DC Request-Response Message Sequence PnC identification mode (2 of 2)

Annex A
(informative)
Mapping of Part 1 use case elements

A.1 Relation of Identification modes and Use Case Elements

The identification modes External Identification Means (EIM) and the identification mode Plug and Charge (PnC) cover various Use Cases elements as defined in Part 1. Table A. 1 gives a detailed overview on the Message Sets as defined in subclause 8.6, and the covered Use Case Elements A1 – H1 as defined in Part 1.

Table A. 1— Message Set(s) and covered use case elements

V2G Message		Message Sets							
Name	Parameter Level	AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Update	Option: MeteringReceipt	Option: VAS
supportedAppProtocolReq		B1	B1	B1	B1	-	-	-	-
	ProtocolNamespace	B1	B1	B1	B1	-	-	-	-
	VersionNumberMajor	B1	B1	B1	B1	-	-	-	-
	VersionNumberMinor	B1	B1	B1	B1	-	-	-	-
	SchemaID	B1	B1	B1	B1	-	-	-	-
	Priority	B1	B1	B1	B1	-	-	-	-
supportedAppProtocolRes		B1	B1	B1	B1	-	-	-	-
	ResponseCode	B1	B1	B1	B1	-	-	-	-
	SchemaID	B1	B1	B1	B1	-	-	-	-
SessionSetupReq		B1	B1	B1	B1	-	-	-	-
	EVCCID	B1	B1	B1	B1	-	-	-	-
SessionSetupRes		B1	B1	B1	B1	-	-	-	-
	ResponseCode	B1	B1	B1	B1	-	-	-	-
	EVSEID	B1	B1	B1	B1	-	-	-	-
	DateTimeNow	-	-	B1	B1	-	-	-	-
ServiceDiscoveryReq		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-
	ServiceScope	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-
	ServiceCategory	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-
ServiceDiscoveryRes		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-
	ResponseCode	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-
	PaymentOption	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-

V2G Message		Message Sets							
Name	Parameter Level	AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Update	Option: MeteringReceipt	Option: VAS
	ChargeService	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-
	ServiceList	-	-	-	-	C1	C2	-	G1,G2
ServiceDetailReq		-	-	-	-	C1	C2	-	G1,G2
	ServiceID	-	-	-	-	C1	C2	-	G1,G2
ServiceDetailRes		-	-	-	-	C1	C2	-	G1,G2
	ResponseCode	-	-	-	-	C1	C2	-	G1,G2
	ServiceID	-	-	-	-	C1	C2	-	G1,G2
	ServiceParameterList	-	-	-	-	C1	C2	-	G1,G2
ServicePayment SelectionReq		D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	-	G1, G2
	SelectedPaymentOption	D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	-	G1, G2
	SelectedServiceList	D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	-	G1, G2
ServicePayment SelectionRes		D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	-	G1, G2
	ResponseCode	D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	-	G1, G2
PaymentDetailsReq		-	-	D1, D2	D1, D2	-	-	-	-
	ContractID	-	-	D1, D2	D1, D2	-	-	-	-
	ContractSignatureCertChain	-	-	D1, D2	D1, D2	-	-	-	-
PaymentDetailsRes		-	-	D1, D2	D1, D2	-	-	-	-
	ResponseCode	-	-	D1, D2	D1, D2	-	-	-	-
	GenChallenge	-	-	D1, D2	D1, D2	-	-	-	-
	DateTimeNow	-	-	D1,D2	D1,D2	-	-	-	-
ContractAuthenticationReq		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-
	GenChallenge	-	-	D1, D2	D1, D2	-	-	-	-
ContractAuthenticationRes		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-
	EVSEProcessing	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-
	ResponseCode	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-	-

V2G Message		Message Sets							
Name	Parameter Level	AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Update	Option: MeteringReceipt	Option: VAS
ChargeParameter DiscoveryReq		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	EVRequestedEnergyTransferType	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	AC_EVChargeParameter	E1, E2	-	E3, E5, F3	-	-	-	-	-
	DC_EVChargeParameter	-	E4	-	E4, F3	-	-	-	-
ChargeParameter DiscoveryRes		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	ResponseCode	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	EVSEProcessing	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	SAScheduleList	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	AC_EVSEChargeParameter	E1, E2	-	E3, E5, F3	-	-	-	-	-
	DC_EVSEChargeParameter	-	E4	-	E4, F3	-	-	-	-
PowerDeliveryReq		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	ReadyToChargeState	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	ChargingProfile	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	DC_EVPowerDeliveryParameter	-	E4	-	E4, F3	-	-	-	-
PowerDeliveryRes		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	ResponseCode	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-	-
	AC_EVSEStatus	E1, E2	-	E3, E5, F3	-	-	-	-	-
	DC_EVSEStatus	-	E4	-	E4, F3	-	-	-	-
CertificateupdateReq		-	-	-	-	C1	-	-	-
	ContractSignatureCertChain	-	-	-	-	C1	-	-	-
	ContractID	-	-	-	-	C1	-	-	-
	ListOfRootCertificateIDs	-	-	-	-	C1	-	-	-
	DHParams								
CertificateupdateRes		-	-	-	-	C1	-	-	-
	ResponseCode	-	-	-	-	C1	-	-	-
	ContractSignatureCertChain	-	-	-	-	C1	-	-	-
	ContractSignatureEncryptedPrivateKey	-	-	-	-	C1	-	-	-
	DHParams	-	-	-	-	C1	-	-	-

V2G Message		Message Sets							
Name	Parameter Level	AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Update	Option: MeteringReceipt	Option: VAS
	ContractID	-	-	-	-	C1	-	-	-
	RetryCounter	-	-	-	-	C1	-	-	-
CertificateinstallationReq		-	-	-	-	-	C2	-	-
	OEMProvisioningCert	-	-	-	-	-	C2	-	-
	ListOfRootCertificateIDs	-	-	-	-	-	C2	-	-
	DHParams								
CertificateinstallationRes		-	-	-	-	-	C2	-	-
	ResponseCode	-	-	-	-	-	C2	-	-
	ContractSignatureCertChain	-	-	-	-	-	C2	-	-
	ContractSignatureEncryptedPrivateKey	-	-	-	-	-	C2	-	-
	DHParams	-	-	-	-	-	C2	-	-
	ContractID	-	-	-	-	-	-	-	-
SessionStopReq		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-	-
SessionStopRes		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-	-
	ResponseCode	H1, F3	H1, F3	H1, F3	H1, F3	-	-	-	-
ChargingStatusReq		E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-	-
ChargingStatusRes		E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-	-
	ResponseCode	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-	-
	EVSEID	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-	-
	SAScheduleTupleID	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-	-
	EVSEMaxCurrent	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-	-
	MeterInfo	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-	-
	ReceiptRequired	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-	-
	AC_EVSEStatus	E1,	-	E3, E5,	-	-	-	-	-

V2G Message		Message Sets							
Name	Parameter Level	AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Update	Option: MeteringReceipt	Option: VAS
		E2, F0		F0, F1					
MeteringReceiptReq		-	-	-	-	-	-	F1	-
	SessionID	-	-	-	-	-	-	F1	-
	SAScheduleTupleID	-	-	-	-	-	-	F1	-
	MeterInfo	-	-	-	-	-	-	F1	-
MeteringReceiptRes		-	-	-	-	-	-	F1	-
	ResponseCode	-	-	-	-	-	-	F1	-
	AC_EVSEStatus	-	-	-	-	-	-	F1	-
CableCheckReq		-	E4	-	E4	-	-	-	-
	DC_EVStatus	-	E4	-	E4	-	-	-	-
CableCheckRes		-	E4	-	E4	-	-	-	-
	ResponseCode	-	E4	-	E4	-	-	-	-
	EVSEProcessing	-	E4	-	E4	-	-	-	-
	DC_EVSEStatus	-	E4	-	E4	-	-	-	-
PreChargeReq		-	E4	-	E4	-	-	-	-
	DC_EVStatus	-	E4	-	E4	-	-	-	-
	EVTargetVoltage	-	E4	-	E4	-	-	-	-
	EVDemandCurrent	-	E4	-	E4	-	-	-	-
PreChargeRes		-	E4	-	E4	-	-	-	-
	ResponseCode	-	E4	-	E4	-	-	-	-
	DC_EVSEStatus	-	E4	-	E4	-	-	-	-
	EVSEPresentVoltage	-	E4	-	E4	-	-	-	-
CurrentDemandReq		-	E4	-	E4	-	-	-	-
	DC_EVStatus	-	E4	-	E4	-	-	-	-
	EVTargetCurrent	-	E4	-	E4	-	-	-	-
	EVMaximumVoltageLimit	-	E4	-	E4	-	-	-	-
	EVMaximumCurrentLimit	-	E4	-	E4	-	-	-	-
	EVMaximumPowerLimit	-	E4	-	E4	-	-	-	-
	BulkChargingComplete	-	E4	-	E4	-	-	-	-
	ChargingComplete	-	E4	-	E4	-	-	-	-
	RemainingTimeToFullSoC	-	E4	-	E4	-	-	-	-
	RemainingTimeToBulkSoC	-	E4	-	E4	-	-	-	-
	EVTargetVoltage	-	E4	-	E4	-	-	-	-
CurrentDemandRes		-	E4	-	E4	-	-	-	-
	ResponseCode	-	E4	-	E4	-	-	-	-
	DC_EVSEStatus	-	E4	-	E4	-	-	-	-

V2G Message		Message Sets							
Name	Parameter Level	AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Update	Option: MeteringReceipt	Option: VAS
	EVSEPresentVoltage	-	E4	-	E4	-	-	-	-
	EVSEPresentCurrent	-	E4	-	E4	-	-	-	-
	EVSECurrentLimitAchieved	-	E4	-	E4	-	-	-	-
	EVSEVoltageLimitAchieved	-	E4	-	E4	-	-	-	-
	EVSEPowerLimitAchieved	-	E4	-	E4	-	-	-	-
	EVSEMaximumVoltageLimit	-	E4	-	E4	-	-	-	-
	EVSEMaximumCurrentLimit	-	E4	-	E4	-	-	-	-
	EVSEMaximumPowerLimit	-	E4	-	E4	-	-	-	-
WeldingDetectionReq		-	E4	-	E4	-	-	-	-
	DC_EVStatus	-	E4	-	E4	-	-	-	-
WeldingDetectionRes		-	E4	-	E4	-	-	-	-
	ResponseCode	-	E4	-	E4	-	-	-	-
	DC_EVSEStatus	-	E4	-	E4	-	-	-	-
	EVSEPresentVoltage	-	E4	-	E4	-	-	-	-

Annex B
(informative)
Mapping of ISO/IEC 15118 message element names to SAE J2847-2 terms

B.1 SAE J2847-2 Status Codes

The Table B. 1 and Table B. 2 define the name mapping of the enumerated signal types used in the context of SAE J2847-2 and their corresponding message element names in ISO/IEC 15118. In general, the term vehicle and charger are replaced by the abbreviations EV and EVSE, respectively.

Table B. 1 – Mapping of naming for SAE J2847-2 Vehicle Status Code

Enumeration	Definition in SAE J2847-2 (Vehicle Status Code)	Definition in ISO/IEC 15118
0x0	Vehicle Not Ready	DC_EVStatus: EVReady (Value = false)
0x1	Vehicle Charging or Energy Transfer	DC_EVStatus: EVReady (Value = true)
0x2	RESS Temperature Inhibit	DC_EVStatus: DC_EVErrorCode (Value = FAILED_RESSTemperatureInhibit)
0x3	Vehicle Shift Position	DC_EVStatus: DC_EVErrorCode (Value = FAILED_EVShiftPosition)
0x4	Charger Connector Lock Fault	DC_EVStatus: DC_EVErrorCode (Value = FAILED_ChargerConnectorLockFault)
0x5	Vehicle Cabin Conditioning	DC_EVStatus: DC_EVErrorCode (Value = EVCabinConditioning)
0x6	Vehicle RESS Conditioning	DC_EVStatus: DC_EVErrorCode (Value =EVRESSConditioning)
0x7	Vehicle RESS Malfunction	DC_EVStatus: DC_EVErrorCode (Value = FAILED_EVRESSMalfunction)
0x8	Charging current differential	DC_EVStatus: DC_EVErrorCode (Value = FAILED_ChargingCurrentdifferential)
0x9	Charging voltage out of range	DC_EVStatus: DC_EVErrorCode (Value = FAILED_ChargingVoltageOutOfRange)
0xA – 0xC	Reserved	DC_EVStatus: DC_EVErrorCode (Value = Reserved_A .. Reserved_C)
0xD	Charging System Incompatibility	DC_EVStatus: DC_EVErrorCode (Value = FAILED_ChargingSystemIncompatibility)
0xE	Other Vehicle Faults	Refer to Response Codes
0xF	No Data	DC_EVStatus: DC_EVErrorCode (Value = NoData)

NOTE The data being transferred correspond to the Vehicle Status Codes as described in the SAE J2847/2.

Table B. 2 – Mapping of naming for SAE J2847-2 Charger Status Code

Enumeration	Definition in SAE J2847-2 (Charger Status Code)	Definition in ISO/IEC 15118
0x0	Charger Standby / Not Ready	DC_EVSEStatusCode (value = FAILED_NotReady)
0x1	Charger Ready / Charging	DC_EVSEStatusCode (value = OK_Charging)
0x2	Charger Prepaid Limits Exceeded	responseCode (value = FAILED_ContractCanceled)
0x3	Charger Shutdown	DC_EVSEStatusCode (value = EVSE_Shutdown)
0x4	Utility Interrupt Event	DC_EVSEStatusCode (value = EVSE_UtilityInterruptEvent)
0x5	Isolation Monitoring Internal	n.a.
0x6	Isolation Monitoring Active	DC_EVSEStatusCode (value = EVSE_IsolationMonitoringActive)
0x7	Charger Emergency Shutdown	DC_EVSEStatusCode (value = EVSE_EVSE_EmergencyShutdown)
0x8 – 0xC	Reserved	DC_EVSEStatusCode (value = Reserved_8 .. Reserved_C)
0xD	Charging System Incompatibility	responseCode (value = FAILED_WrongChargeParameter)
0xE	Charger Malfunction	responseCode (value = EVSE_Malfunction)
0xF	No Data	n.a

NOTE The data being transferred correspond to the Charger Status Codes as described in the SAE J2847/2.

B.2 SAE J2847-2 Energy Transfer Types

Table B. 3 and Table B. 4 define the name mapping for the SAE J2847-2 Vehicle Requested Energy Transfer Type and Charger Supported Energy Transfer Type.

Table B. 3 – Mapping of naming for SAE J2847-2 Vehicle Requested Energy Transfer Type

Enumeration	Definition in SAE J2847-2 (Vehicle Reqested Energy Transfer Type)	Definition in ISO/IEC 15118 (EVRequestedEnergyTransferType)
0x00	(reserved for) AC Type 1/Type 2 – single phase on core pins	AC_single_phase_core
0x01	(reserved for) AC Type 2 – three phase on Type 2 core pins	AC_three_phase_core
0x02	DC Type 1/Type 2 on core pins	DC_core
0x03	DC combo 1/combo 2 on extended pins	DC_extended
0x04	DC combo 1/combo 2 on core pins	DC_combo_core
0x05	(reserved for) Dedicated DC on unique connector	DC_unique
0x06	(reserved for) Reverse Energy Flow DC	DC_reverse
0x07	(reserved for) Reverse Energy Flow AC	AC_reverse
0x08 – 0x0E	Reserved for future use	Reserved_8.. Reserved_E

Enumeration	Definition in SAE J2847-2 (Vehicle Reqested Energy Transfer Type)	Definition in ISO/IEC 15118 (EVRequestedEnergyTransferType)
0x0F	Undetermined	Undetermined

Table B. 4 – Mapping of naming for SAE J2847-2 Charger Supported Energy Transfer Type

Enumeration	Definition in SAE J2847-2 (Charger Supported Energy Transfer Type)	Definition in ISO/IEC 15118 (EVSESupportedEnergyTransferType)
0x00	(reserved for) AC Type 1/Type 2 – single phase on core pins	AC_single_phase_core
0x01	(reserved for) AC Type 2 – three phase on Type 2 core pins	AC_three_phase_core
0x02	DC Type 1/Type 2 on core pins	DC_core
0x03	DC combo 1/combo 2 on extended pins	DC_extended
0x04	DC combo 1/combo 2 on core pins	DC_combo_core
0x05	DC combo 1/combo 2 dual	DC_dual
0x06	(reserved for) AC on core pins / DC on extended pins	AC_core_DC_extended
0x07	(reserved for) AC and DC capable on core pins	AC_single_DC_core
0x08	(reserved for) AC single phase on core combo 1/combo 2 pins AND AC three phase on combo 1/combo 2 core + extended pins	AC_single_phase_three_phase_core_DC_extended
0x09-0x0E	Reserved for future use	Reserved_9.. Reserved_E
0x0F	Undetermined	Undetermined

B.3 SAE J2847-2 Signals

Table B. 5 defines the name mapping for the SAE J2847-2 signals to ISO/IEC 15118 message elements.

Table B. 5 – Mapping of naming for SAE J2847-2 signals to ISO/IEC 15118 message elements

SAE J2847-2 signal definition	ISO/IEC 15118 message element definition
Bulk Charging Complete	DC_EVPowerDeliveryParameter: BulkChargingComplete CurrentDemandRequest: BulkChargingComplete
Bulk SOC	DC_EVChargeParameter: BulkSOC
Charge Current Request	CurrentDemandRequest: EVTargetCurrent
Charger Current Limit Achieved	CurrentDemandRes: EVSECurrentLimitAchieved
Charger Current Regulation Tolerance	DC_EVSEChargeParameter: EVSECurrentRegulationTolerance
Charger Energy to be delivered	DC_EVSEChargeParameter: EVSEEnergyToBeDelivered
Charger Maximum Current Limit	DC_EVSEChargeParameter: EVSEMaximumCurrentLimit CurrentDemandRes: EVSEMaximumCurrentLimit
Charger Maximum Power Limit	DC_EVSEChargeParameter: EVSEMaximumPowerLimit CurrentDemandRes: EVSEMaximumCurrentLimit
Charger Maximum Voltage Limit	DC_EVSEChargeParameter: EVSEMaximumVoltageLimit CurrentDemandRes: EVSEMaximumCurrentLimit
Charger Minimum Current Limit	DC_EVSEChargeParameter: EVSEMinimumCurrentLimit
Charger Minimum Voltage Limit	DC_EVSEChargeParameter: EVSEMinimumVoltageLimit
Charger Peak Current Ripple	DC_EVSEChargeParameter: EVSEPeakCurrentRipple
Charger Power Limit Achieved	CurrentDemandRes: EVSEPowerLimitAchieved
Charger Status	DC_EVSEStatus
Charger Voltage Limit Achieved	CurrentDemandRes: EVSEVoltageLimitAchieved
Current Output	CurrentDemandRes: EVSEPresentCurrent
Connector Locked	n.a
Full SOC	DC_EVChargeParameter: FullSOC
Vehicle Energy Capacity	DC_EVChargeParameter: EVEnergyCapacity
Vehicle Energy Request	DC_EVChargeParameter: EVEnergyRequest

SAE J2847-2 signal definition	ISO/IEC 15118 message element definition
Vehicle Maximum Current Limit	DC_EVChargeParameter: EVMaximumCurrentLimit
Vehicle Maximum Power Limit	DC_EVChargeParameter: EVMaximumPowerLimit
Vehicle Maximum Voltage Limit	DC_EVChargeParameter: EVMaximumVoltageLimit
Vehicle RESS SOC	DC_EVStatusType: EVRESSSOC
Vehicle Status Code	DC_EVErrorCode
Voltage Output	CurrentDemandRes: EVSEPresentVoltage

Annex C
(normative)

Schema definition

C.1 Overview

The V2G application layer message specification consists of four XML Schema documents with the following scope:

— “V2G_CI_AppProtocol”: Defines the protocol handshake messages

— “V2G_CI_MsgDef”: Defines the message structure Definition

— “V2G_CI_MsgHeader”: Defines the message Header

— “V2G_CI_MsgBody”: Defines the message Body

— “V2G_CI_MsgDataTypes”: Defines the data types

Figure C.1 shows the dependency graph for all five XML Schema documents.

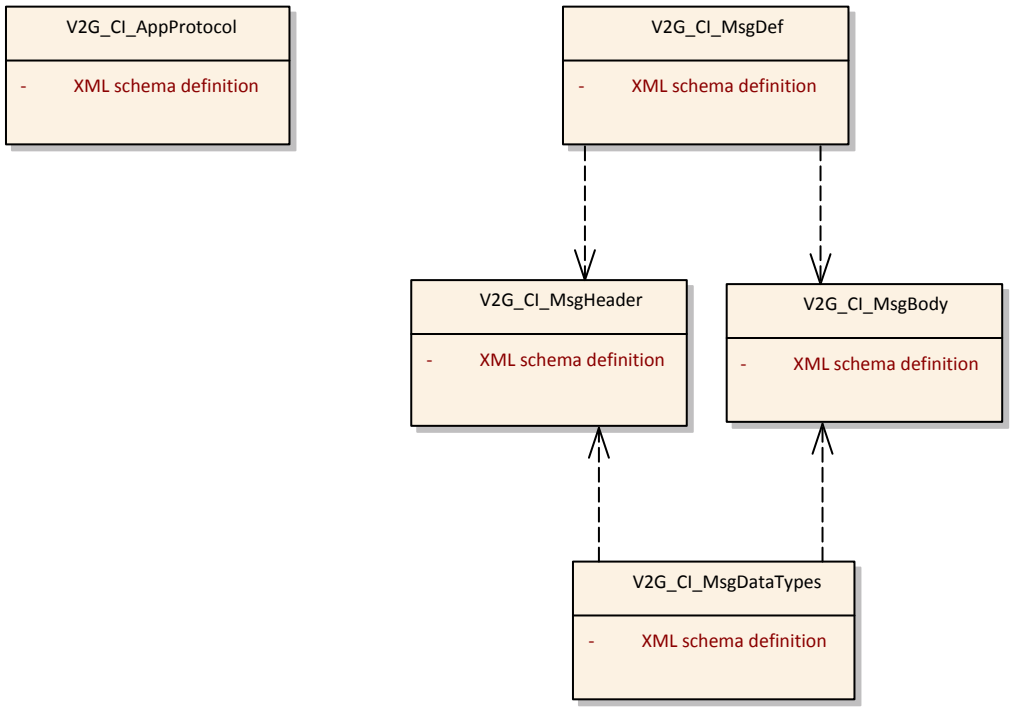


Figure C.1 — Dependency Chart of the V2G CI XML Schema Definitions

C.2 V2G_CI_AppProtocol.xsd

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns="urn:iso:15118:2:2010:AppProtocol"
        targetNamespace="urn:iso:15118:2:2010:AppProtocol">
```

```
	<xs:element name="supportedAppProtocolReq">
		<xs:complexType>
			<xs:sequence>
				<xs:element name="AppProtocol" type="AppProtocolType" maxOccurs="20"/>
			</xs:sequence>
		</xs:complexType>
	</xs:element>
	<xs:element name="supportedAppProtocolRes">
		<xs:complexType>
			<xs:sequence>
				<xs:element name="ResponseCode" type="responseCodeType"/>
				<xs:element name="SchemaID" type="idType" minOccurs="0"/>
			</xs:sequence>
		</xs:complexType>
	</xs:element>
	<xs:complexType name="AppProtocolType">
		<xs:sequence>
			<xs:element name="ProtocolNamespace" type="protocolNamespaceType"/>
			<xs:element name="VersionNumberMajor" type="xs:unsignedInt"/>
			<xs:element name="VersionNumberMinor" type="xs:unsignedInt"/>
			<xs:element name="SchemaID" type="idType"/>
			<xs:element name="Priority" type="priorityType"/>
		</xs:sequence>
	</xs:complexType>
	<xs:simpleType name="idType">
		<xs:restriction base="xs:unsignedByte"/>
	</xs:simpleType>
	<xs:simpleType name="protocolNameType">
		<xs:restriction base="xs:string">
			<xs:maxLength value="30"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="protocolNamespaceType">
		<xs:restriction base="xs:anyURI">
			<xs:maxLength value="100"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="priorityType">
		<xs:restriction base="xs:unsignedByte">
			<xs:minInclusive value="1"/>
			<xs:maxInclusive value="20"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="responseCodeType">
		<xs:restriction base="xs:string">
			<xs:enumeration value="OK_SuccessfulNegotiation"/>
			<xs:enumeration value="OK_SuccessfulNegotiationWithMinorDeviation"/>
			<xs:enumeration value="Failed_NoNegotiation"/>
		</xs:restriction>
	</xs:simpleType>
</xs:schema>
```

C.3 V2G_CI_MsgDef.xsd

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
	targetNamespace="urn:iso:15118:2:2010:MsgDef"
	xmlns="urn:iso:15118:2:2010:MsgDef"
	xmlns:v2gci_h="urn:iso:15118:2:2010:MsgHeader"
	elementFormDefault="qualified"
	attributeFormDefault="unqualified">

	<xs:import namespace="urn:iso:15118:2:2010:MsgHeader" schemaLocation="V2G_CI_MsgHeader.xsd"/>

	<!-- Message Structure -->
	<xs:element name="V2G_Message">
		<xs:complexType>
			<xs:sequence>
				<xs:element name="Header" type="v2gci_h:MessageHeaderType"/>
				<xs:element name="Body" type="BodyType"/>
			</xs:sequence>
		</xs:complexType>
	</xs:element>
	<!-- Body -->
```

```
    <xs:complexType name="BodyType">
        <xs:sequence>
            <xs:element ref="BodyElement" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="BodyElement" type="BodyBaseType"/>
    <xs:complexType name="BodyBaseType" abstract="true"/>
</xs:schema>
```

C.4 V2G_CI_MsgHeader.xsd

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="urn:iso:15118:2:2010:MsgHeader"
    xmlns="urn:iso:15118:2:2010:MsgHeader"
    xmlns:v2gci_d="urn:iso:15118:2:2010:MsgDef"
    xmlns:v2gci_t="urn:iso:15118:2:2010:MsgDataTypes"
    xmlns:xmlsig=http://www.w3.org/2000/09/xmldsig#
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

    <xs:import namespace="urn:iso:15118:2:2010:MsgDef" schemaLocation="V2G_CI_MsgDef.xsd"/>
    <xs:import namespace="urn:iso:15118:2:2010:MsgDataTypes" schemaLocation="V2G_CI_MsgDataTypes.xsd"/>
    <xs:import namespace="http://www.w3.org/2000/09/xmldsig#" schemaLocation="xmldsig-core-schema.xsd"/>

    <!-- Message Header -->
    <xs:complexType name="MessageHeaderType">
        <xs:sequence>
            <xs:element name="SessionID" type="v2gci_t:sessionIDType"/>
            <xs:element name="Notification" type="v2gci_t:NotificationType" minOccurs="0"/>
            <xs:element ref="xmlsig:Signature" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

C.5 V2G_CI_MsgBody.xsd

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="urn:iso:15118:2:2010:MsgBody"
    xmlns="urn:iso:15118:2:2010:MsgBody"
    xmlns:v2gci_d="urn:iso:15118:2:2010:MsgDef"
    xmlns:v2gci_t="urn:iso:15118:2:2010:MsgDataTypes"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

    <xs:import namespace="urn:iso:15118:2:2010:MsgDef" schemaLocation="V2G_CI_MsgDef.xsd"/>
    <xs:import namespace="urn:iso:15118:2:2010:MsgDataTypes" schemaLocation="V2G_CI_MsgDataTypes.xsd"/>
    <!-- ..................... -->
    <!-- Common Messages (AC/DC) -->
    <!-- ..................... -->
    <!--             -->
    <!-- Session Setup -->
    <!--             -->
    <xs:element name="SessionSetupReq" type="SessionSetupReqType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="SessionSetupReqType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="EVCCID" type="v2gci_t:evccIDType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="SessionSetupRes" type="SessionSetupResType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="SessionSetupResType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
```

```
                        <xs:element name="EVSEID" type="v2gci_t:evseIDType"/>
                        <xs:element name="DateTimeNow" type="xs:long" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                -->
        <!-- Service Discovery -->
        <!--                -->
        <xs:element name="ServiceDiscoveryReq" type="ServiceDiscoveryReqType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ServiceDiscoveryReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ServiceScope" type="v2gci_t:serviceScopeType" minOccurs="0"/>
                        <xs:element name="ServiceCategory" type="v2gci_t:serviceCategoryType" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="ServiceDiscoveryRes" type="ServiceDiscoveryResType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ServiceDiscoveryResType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="PaymentOptions" type="v2gci_t:PaymentOptionsType"/>
                        <xs:element name="ChargeService" type="v2gci_t:ServiceChargeType"/>
                        <xs:element name="ServiceList" type="v2gci_t:ServiceTagListType" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                -->
        <!-- Service Detail    -->
        <!--                -->
        <xs:element name="ServiceDetailReq" type="ServiceDetailReqType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ServiceDetailReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ServiceID" type="v2gci_t:serviceIDType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="ServiceDetailRes" type="ServiceDetailResType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ServiceDetailResType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="ServiceID" type="v2gci_t:serviceIDType"/>
                        <xs:element name="ServiceParameterList" type="v2gci_t:ServiceParameterListType" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                        -->
        <!-- Service Payment & Selection -->
        <!--                        -->
        <xs:element name="ServicePaymentSelectionReq" type="ServicePaymentSelectionReqType"
substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ServicePaymentSelectionReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="SelectedPaymentOption" type="v2gci_t:paymentOptionType"/>
                        <xs:element name="SelectedServiceList" type="v2gci_t:SelectedServiceListType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="ServicePaymentSelectionRes" type="ServicePaymentSelectionResType"
substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ServicePaymentSelectionResType">
```

```
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--              -->
        <!-- Payment Details -->
        <!--              -->
        <xs:element name="PaymentDetailsReq" type="PaymentDetailsReqType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="PaymentDetailsReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ContractID" type="v2gci_t:contractIDType"/>
                        <xs:element name="ContractSignatureCertChain" type="v2gci_t:CertificateChainType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="PaymentDetailsRes" type="PaymentDetailsResType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="PaymentDetailsResType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="GenChallenge" type="v2gci_t:genChallengeType"/>
                        <xs:element name="DateTimeNow" type="xs:long"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                    -->
        <!-- Contract Authentification -->
        <!--                    -->
        <xs:element name="ContractAuthenticationReq" type="ContractAuthenticationReqType"
substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ContractAuthenticationReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="GenChallenge" type="v2gci_t:genChallengeType" minOccurs="0"/>
                    </xs:sequence>
                    <xs:attribute name="Id" type="xs:IDREF" use="required"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="ContractAuthenticationRes" type="ContractAuthenticationResType"
substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ContractAuthenticationResType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="EVSEProcessing" type="v2gci_t:EVSEProcessingType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                    -->
        <!-- Charge Parameter Discovery -->
        <!--                    -->
        <xs:element name="ChargeParameterDiscoveryReq" type="ChargeParameterDiscoveryReqType"
substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ChargeParameterDiscoveryReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="EVRequestedEnergyTransferType"
type="v2gci_t:EVRequestedEnergyTransferType"/>
                        <xs:element ref="v2gci_t:EVChargeParameter"/>
                    </xs:sequence>
                </xs:extension>
```

```
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="ChargeParameterDiscoveryRes" type="ChargeParameterDiscoveryResType"
substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ChargeParameterDiscoveryResType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="EVSEProcessing" type="v2gci_t:EVSEProcessingType"/>
                        <xs:element ref="v2gci_t:SASchedules"/>
                        <xs:element ref="v2gci_t:EVSEChargeParameter"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                    -->
        <!-- Power Delivery     -->
        <!--                    -->
        <xs:element name="PowerDeliveryReq" type="PowerDeliveryReqType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="PowerDeliveryReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ReadyToChargeState" type="xs:boolean"/>
                        <xs:element name="ChargingProfile" type="v2gci_t:ChargingProfileType" minOccurs="0"/>
                        <xs:element ref="v2gci_t:EVPowerDeliveryParameter" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="PowerDeliveryRes" type="PowerDeliveryResType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="PowerDeliveryResType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element ref="v2gci_t:EVSEStatus"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                    -->
        <!-- Charging Status    -->
        <!--                    -->
        <xs:element name="ChargingStatusReq" type="ChargingStatusReqType" substitutionGroup="v2gci_d:BodyElement"/>
        <!--   <xs:element name="MeteringStatusReq" type="MeteringStatusReqType" substitutionGroup="v2gci_d:BodyElement"/> -->
        <xs:complexType name="ChargingStatusReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="ChargingStatusRes" type="ChargingStatusResType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="ChargingStatusResType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="EVSEID" type="v2gci_t:evseIDType"/>
                        <xs:element name="SAScheduleTupleID" type="v2gci_t:SAIDType"/>
                        <xs:element name="EVSEMaxCurrent" type="v2gci_t:PhysicalValueType" minOccurs="0"/>
                        <xs:element name="MeterInfo" type="v2gci_t:MeterInfoType" minOccurs="0"/>
                        <xs:element name="ReceiptRequired" type="xs:boolean"/>
                        <xs:element name="AC_EVSEStatus" type="v2gci_t:AC_EVSEStatusType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                    -->
        <!-- Metering Receipt   -->
        <!--                    -->
        <xs:element name="MeteringReceiptReq" type="MeteringReceiptReqType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="MeteringReceiptReqType">
            <xs:complexContent>
```

```
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="SessionID" type="v2gci_t:sessionIDType"/>
                    <xs:element name="SAScheduleTupleID" type="v2gci_t:SAIDType" minOccurs="0"/>
                    <xs:element name="MeterInfo" type="v2gci_t:MeterInfoType"/>
                </xs:sequence>
                <xs:attribute name="Id" type="xs:IDREF"/>
            </xs:extension>
            <!-- only needed if receipt is required -->
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="MeteringReceiptRes" type="MeteringReceiptResType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="MeteringReceiptResType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                    <xs:element name="AC_EVSEStatus" type="v2gci_t:AC_EVSEStatusType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <!--                -->
    <!-- SessionStop    -->
    <!--                -->
    <xs:element name="SessionStopReq" type="SessionStopType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="SessionStopType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="SessionStopRes" type="SessionStopResType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="SessionStopResType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <!--                -->
    <!-- Certificate Update -->
    <!--                -->
    <xs:element name="CertificateUpdateReq" type="CertificateUpdateReqType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="CertificateUpdateReqType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="ContractSignatureCertChain" type="v2gci_t:CertificateChainType"/>
                    <xs:element name="ContractID" type="v2gci_t:contractIDType"/>
                    <xs:element name="ListOfRootCertificateIDs" type="v2gci_t:ListOfRootCertificateIDsType"/>
                    <xs:element name="DHParams" type="v2gci_t:dHParamsType"/>
                    <!-- new -->
                </xs:sequence>
                <xs:attribute name="Id" type="xs:IDREF"/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="CertificateUpdateRes" type="CertificateUpdateResType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="CertificateUpdateResType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                    <xs:element name="ContractSignatureCertChain" type="v2gci_t:CertificateChainType"/>
                    <xs:element name="ContractSignatureEncryptedPrivateKey" type="v2gci_t:privateKeyType"/>
                    <xs:element name="DHParams" type="v2gci_t:dHParamsType"/>
                    <xs:element name="ContractID" type="v2gci_t:contractIDType"/>
                    <xs:element name="RetryCounter" type="xs:short"/>
                </xs:sequence>
                <xs:attribute name="Id" type="xs:IDREF" use="required"/>
            </xs:extension>
```

```
            </xs:complexContent>
        </xs:complexType>
        <!--                          -->
        <!-- Certificate Installation     -->
        <!--                          -->
        <xs:element name="CertificateInstallationReq" type="CertificateInstallationReqType"
substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="CertificateInstallationReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="OEMProvisioningCert" type="v2gci_t:certificateType"/>
                        <xs:element name="ListOfRootCertificateIDs" type="v2gci_t:ListOfRootCertificateIDsType"/>
                        <xs:element name="DHParams" type="v2gci_t:dHParamsType"/>
                    </xs:sequence>
                    <xs:attribute name="Id" type="xs:IDREF"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="CertificateInstallationRes" type="CertificateInstallationResType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="CertificateInstallationResType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="ContractSignatureCertChain" type="v2gci_t:CertificateChainType"/>
                        <xs:element name="ContractSignatureEncryptedPrivateKey" type="v2gci_t:privateKeyType"/>
                        <xs:element name="DHParams" type="v2gci_t:dHParamsType"/>
                        <xs:element name="ContractID" type="v2gci_t:contractIDType"/>
                    </xs:sequence>
                    <xs:attribute name="Id" type="xs:IDREF" use="required"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!-- ..................... -->
        <!-- DC-Messages           -->
        <!-- ..................... -->
        <!--             -->
        <!-- Cable Check   -->
        <!--             -->
        <xs:element name="CableCheckReq" type="CableCheckReqType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="CableCheckReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="DC_EVStatus" type="v2gci_t:DC_EVStatusType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="CableCheckRes" type="CableCheckResType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="CableCheckResType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="DC_EVSEStatus" type="v2gci_t:DC_EVSEStatusType"/>
                        <xs:element name="EVSEProcessing" type="v2gci_t:EVSEProcessingType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--             -->
        <!-- Pre-Charge    -->
        <!--             -->
        <xs:element name="PreChargeReq" type="PreChargeReqType" substitutionGroup="v2gci_d:BodyElement"/>
        <xs:complexType name="PreChargeReqType">
            <xs:complexContent>
                <xs:extension base="v2gci_d:BodyBaseType">
                    <xs:sequence>
                        <xs:element name="DC_EVStatus" type="v2gci_t:DC_EVStatusType"/>
                        <xs:element name="EVTargetVoltage" type="v2gci_t:PhysicalValueType"/>
                        <xs:element name="EVTargetCurrent" type="v2gci_t:PhysicalValueType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
```

```
    </xs:complexType>
    <xs:element name="PreChargeRes" type="PreChargeResType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="PreChargeResType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                    <xs:element name="DC_EVSEStatus" type="v2gci_t:DC_EVSEStatusType"/>
                    <xs:element name="EVSEPresentVoltage" type="v2gci_t:PhysicalValueType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <!--                -->
    <!-- Current Demand    -->
    <!--                -->
    <xs:element name="CurrentDemandReq" type="CurrentDemandReqType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="CurrentDemandReqType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="DC_EVStatus" type="v2gci_t:DC_EVStatusType"/>
                    <xs:element name="EVTargetCurrent" type="v2gci_t:PhysicalValueType"/>
                    <xs:element name="EVMaximumVoltageLimit" type="v2gci_t:PhysicalValueType" minOccurs="0"/>
                    <xs:element name="EVMaximumCurrentLimit" type="v2gci_t:PhysicalValueType" minOccurs="0"/>
                    <xs:element name="EVMaximumPowerLimit" type="v2gci_t:PhysicalValueType" minOccurs="0"/>
                    <xs:element name="BulkChargingComplete" type="xs:boolean" minOccurs="0"/>
                    <xs:element name="ChargingComplete" type="xs:boolean"/>
                    <xs:element name="RemainingTimeToFullSoC" type="v2gci_t:PhysicalValueType" minOccurs="0"/>
                    <xs:element name="RemainingTimeToBulkSoC" type="v2gci_t:PhysicalValueType" minOccurs="0"/>
                    <xs:element name="EVTargetVoltage" type="v2gci_t:PhysicalValueType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="CurrentDemandRes" type="CurrentDemandResType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="CurrentDemandResType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                    <xs:element name="DC_EVSEStatus" type="v2gci_t:DC_EVSEStatusType"/>
                    <xs:element name="EVSEPresentVoltage" type="v2gci_t:PhysicalValueType"/>
                    <xs:element name="EVSEPresentCurrent" type="v2gci_t:PhysicalValueType"/>
                    <xs:element name="EVSECurrentLimitAchieved" type="xs:boolean"/>
                    <xs:element name="EVSEVoltageLimitAchieved" type="xs:boolean"/>
                    <xs:element name="EVSEPowerLimitAchieved" type="xs:boolean"/>
                    <xs:element name="EVSEMaximumVoltageLimit" type="v2gci_t:PhysicalValueType" minOccurs="0"/>
                    <xs:element name="EVSEMaximumCurrentLimit" type="v2gci_t:PhysicalValueType" minOccurs="0"/>
                    <xs:element name="EVSEMaximumPowerLimit" type="v2gci_t:PhysicalValueType" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <!--                -->
    <!-- Welding Detection -->
    <!--                -->
    <xs:element name="WeldingDetectionReq" type="WeldingDetectionReqType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="WeldingDetectionReqType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="DC_EVStatus" type="v2gci_t:DC_EVStatusType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="WeldingDetectionRes" type="WeldingDetectionResType" substitutionGroup="v2gci_d:BodyElement"/>
    <xs:complexType name="WeldingDetectionResType">
        <xs:complexContent>
            <xs:extension base="v2gci_d:BodyBaseType">
                <xs:sequence>
                    <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                    <xs:element name="DC_EVSEStatus" type="v2gci_t:DC_EVSEStatusType"/>
                    <xs:element name="EVSEPresentVoltage" type="v2gci_t:PhysicalValueType"/>
```

```
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
</xs:schema>
```

C.6 V2G_CI_MsgDataTypes.xsd

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="urn:iso:15118:2:2010:MsgDataTypes"
        xmlns="urn:iso:15118:2:2010:MsgDataTypes"
        xmlns:v2gci_b="urn:iso:15118:2:2010:MsgBody"
        elementFormDefault="qualified"
        attributeFormDefault="unqualified">

        <xs:import namespace="urn:iso:15118:2:2010:MsgBody" schemaLocation="V2G_CI_MsgBody.xsd"/>

        <!-- ================== -->
        <!-- Complex types      -->
        <!-- ================== -->
        <!--                    -->
        <!-- service-related types -->
        <!--                    -->
        <xs:complexType name="ServiceType">
            <xs:sequence>
                <xs:element name="ServiceTag" type="ServiceTagType"/>
                <xs:element name="FreeService" type="xs:boolean"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="ServiceTagListType">
            <xs:sequence>
                <xs:element name="Service" type="ServiceType" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="ServiceTagType">
            <xs:sequence>
                <xs:element name="ServiceID" type="serviceIDType"/>
                <xs:element name="ServiceName" type="serviceNameType" minOccurs="0"/>
                <xs:element name="ServiceCategory" type="serviceCategoryType"/>
                <xs:element name="ServiceScope" type="serviceScopeType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="SelectedServiceListType">
            <xs:sequence>
                <xs:element name="SelectedService" type="SelectedServiceType" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="SelectedServiceType">
            <xs:sequence>
                <xs:element name="ServiceID" type="serviceIDType"/>
                <xs:element name="ParameterSetID" type="xs:short" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="ServiceParameterListType">
            <xs:sequence>
                <xs:element name="ParameterSet" type="ParameterSetType" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="ParameterSetType">
            <xs:sequence>
                <xs:element name="ParameterSetID" type="xs:short"/>
                <xs:element name="Parameter" type="ParameterType" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="ParameterType">
            <xs:choice>
                <xs:element name="boolValue" type="xs:boolean"/>
                <xs:element name="byteValue" type="xs:byte"/>
                <xs:element name="shortValue" type="xs:short"/>
                <xs:element name="intValue" type="xs:int"/>
                <xs:element name="physicalValue" type="PhysicalValueType"/>
                <xs:element name="stringValue" type="xs:string"/>
```

```
        </xs:choice>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="ValueType" type="valueType" use="required"/>
    </xs:complexType>
    <xs:simpleType name="valueType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="bool"/>
            <xs:enumeration value="byte"/>
            <xs:enumeration value="short"/>
            <xs:enumeration value="int"/>
            <xs:enumeration value="physicalValue"/>
            <xs:enumeration value="string"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:element name="ServiceCharge" type="ServiceChargeType"/>
    <xs:complexType name="ServiceChargeType">
        <xs:complexContent>
            <xs:extension base="ServiceType">
                <xs:sequence>
                    <xs:element name="EnergyTransferType" type="EVSESupportedEnergyTransferType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <!--                                    -->
    <!-- security related types -->
    <!--                                    -->
    <xs:complexType name="CertificateChainType">
        <xs:sequence>
            <xs:element name="Certificate" type="certificateType"/>
            <xs:element name="SubCertificates" type="SubCertificatesType" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="SubCertificatesType">
        <xs:sequence>
            <xs:element name="Certificate" type="certificateType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ListOfRootCertificateIDsType">
        <xs:sequence>
            <xs:element name="RootCertificateID" type="rootCertificateIDType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <!--                                        -->
    <!-- metering related types -->
    <!--                                        -->
    <xs:complexType name="MeterInfoType">
        <xs:sequence>
            <xs:element name="MeterID" type="meterIDType"/>
            <xs:element name="MeterReading" type="PhysicalValueType" minOccurs="0"/>
            <xs:element name="SigMeterReading" type="sigMeterReadingType" minOccurs="0"/>
            <xs:element name="MeterStatus" type="meterStatusType" minOccurs="0"/>
            <xs:element name="TMeter" type="xs:long" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:simpleType name="meterStatusType">
        <xs:restriction base="xs:short"/>
    </xs:simpleType>
    <!--                                    -->
    <!-- Physical value type   -->
    <!--                                    -->
    <xs:complexType name="PhysicalValueType">
        <xs:sequence>
            <xs:element name="Multiplier" type="unitMultiplierType"/>
            <xs:element name="Unit" type="unitSymbolType" minOccurs="0"/>
            <xs:element name="Value" type="xs:short"/>
        </xs:sequence>
    </xs:complexType>
    <!--                          -->
    <!-- header related types   -->
    <!--                          -->
    <xs:complexType name="NotificationType">
        <xs:sequence>
            <xs:element name="FaultCode" type="faultCodeType"/>
            <xs:element name="FaultMsg" type="faultMsgType" minOccurs="0"/>
```

```
            </xs:sequence>
        </xs:complexType>
        <!--                                -->
        <!-- Tariff related types           -->
        <!--                                -->
        <xs:complexType name="SASchedulesType" abstract="true"/>
        <xs:element name="SASchedules" type="SASchedulesType"/>
        <xs:element name="SAScheduleList" type="SAScheduleListType" substitutionGroup="SASchedules"/>
        <xs:complexType name="SAScheduleListType">
            <xs:complexContent>
                <xs:extension base="SASchedulesType">
                    <xs:sequence>
                        <xs:element name="SAScheduleTuple" type="SAScheduleTupleType" maxOccurs="unbounded"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="SAScheduleTupleType">
            <xs:sequence>
                <xs:element name="SAScheduleTupleID" type="SAIDType"/>
                <xs:element name="PMaxSchedule" type="PMaxScheduleType"/>
                <xs:element name="SalesTariff" type="SalesTariffType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="SalesTariffType">
            <xs:sequence>
                <xs:element name="SalesTariffID" type="SAIDType"/>
                <xs:element name="SalesTariffDescription" type="tariffDescriptionType" minOccurs="0"/>
                <xs:element name="NumEPriceLevels" type="xs:unsignedByte"/>
                <xs:element ref="SalesTariffEntry" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="Id" type="xs:IDREF" use="required"/>
        </xs:complexType>
        <xs:complexType name="PMaxScheduleType">
            <xs:sequence>
                <xs:element name="PMaxScheduleID" type="SAIDType"/>
                <xs:element ref="PMaxScheduleEntry" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
        <xs:element name="Entry" type="EntryType"/>
        <xs:complexType name="EntryType" abstract="true">
            <xs:sequence>
                <xs:element ref="TimeInterval"/>
            </xs:sequence>
        </xs:complexType>
        <xs:element name="SalesTariffEntry" type="SalesTariffEntryType" substitutionGroup="Entry"/>
        <xs:complexType name="SalesTariffEntryType">
            <xs:complexContent>
                <xs:extension base="EntryType">
                    <xs:sequence>
                        <xs:element name="EPriceLevel" type="xs:unsignedByte"/>
                        <xs:element name="ConsumptionCost" type="ConsumptionCostType" minOccurs="0"
maxOccurs="unbounded"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="PMaxScheduleEntry" type="PMaxScheduleEntryType" substitutionGroup="Entry"/>
        <xs:complexType name="PMaxScheduleEntryType">
            <xs:complexContent>
                <xs:extension base="EntryType">
                    <xs:sequence>
                        <xs:element name="PMax" type="PMaxType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="IntervalType" abstract="true"/>
        <xs:element name="TimeInterval" type="IntervalType"/>
        <xs:element name="RelativeTimeInterval" type="RelativeTimeIntervalType" substitutionGroup="TimeInterval"/>
        <xs:complexType name="RelativeTimeIntervalType">
            <xs:complexContent>
                <xs:extension base="IntervalType">
                    <xs:sequence>
                        <xs:element name="start" type="xs:unsignedInt"/>
                        <xs:element name="duration" type="xs:unsignedInt" minOccurs="0"/>
```

```
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ConsumptionCostType">
        <xs:sequence>
            <xs:element name="startValue" type="xs:unsignedInt"/>
            <xs:element name="Cost" type="CostType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="CostType">
        <xs:sequence>
            <xs:element name="costKind" type="costKindType"/>
            <xs:element name="amount" type="xs:unsignedInt"/>
            <xs:element name="amountMultiplier" type="unitMultiplierType" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <!--                          -->
    <!-- EV/EVSE related types   -->
    <!--                          -->
    <xs:complexType name="EVSEStatusType" abstract="true"/>
    <xs:element name="EVSEStatus" type="EVSEStatusType"/>
    <xs:element name="AC_EVSEStatus" type="AC_EVSEStatusType" substitutionGroup="EVSEStatus"/>
    <xs:complexType name="AC_EVSEStatusType">
        <xs:complexContent>
            <xs:extension base="EVSEStatusType">
                <xs:sequence>
                    <xs:element name="PowerSwitchClosed" type="xs:boolean"/>
                    <xs:element name="RCD" type="xs:boolean"/>
                    <xs:element name="NotificationMaxDelay" type="xs:unsignedInt"/>
                    <xs:element name="EVSENotification" type="EVSENotificationType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="EVStatusType" abstract="true"/>
    <xs:element name="EVStatus" type="EVStatusType"/>
    <xs:element name="DC_EVSEStatus" type="DC_EVSEStatusType" substitutionGroup="EVSEStatus"/>
    <xs:complexType name="DC_EVSEStatusType">
        <xs:complexContent>
            <xs:extension base="EVSEStatusType">
                <xs:sequence>
                    <xs:element name="EVSEIsolationStatus" type="isolationLevelType" minOccurs="0"/>
                    <xs:element name="EVSEStatusCode" type="DC_EVSEStatusCodeType"/>
                    <xs:element name="NotificationMaxDelay" type="xs:unsignedInt"/>
                    <xs:element name="EVSENotification" type="EVSENotificationType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="DC_EVStatus" type="DC_EVStatusType" substitutionGroup="EVStatus"/>
    <xs:complexType name="DC_EVStatusType">
        <xs:complexContent>
            <xs:extension base="EVStatusType">
                <xs:sequence>
                    <xs:element name="EVReady" type="xs:boolean"/>
                    <xs:element name="EVCabinConditioning" type="xs:boolean" minOccurs="0"/>
                    <xs:element name="EVRESSConditioning" type="xs:boolean" minOccurs="0"/>
                    <xs:element name="EVErrorCode" type="DC_EVErrorCodeType"/>
                    <xs:element name="EVRESSSOC" type="percentValueType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <!--                                    -->
    <!-- EVSE/EV Charge Parameter related types   -->
    <!--                                    -->
    <xs:complexType name="EVChargeParameterType" abstract="true"/>
    <xs:element name="EVChargeParameter" type="EVChargeParameterType"/>
    <xs:element name="AC_EVChargeParameter" type="AC_EVChargeParameterType" substitutionGroup="EVChargeParameter"/>
    <xs:complexType name="AC_EVChargeParameterType">
        <xs:complexContent>
            <xs:extension base="EVChargeParameterType">
                <xs:sequence>
                    <xs:element name="DepartureTime" type="xs:unsignedInt"/>
```

```
                        <xs:element name="EAmount" type="PhysicalValueType"/>
                        <xs:element name="EVMaxVoltage" type="PhysicalValueType"/>
                        <xs:element name="EVMaxCurrent" type="PhysicalValueType"/>
                        <xs:element name="EVMinCurrent" type="PhysicalValueType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="DC_EVChargeParameter" type="DC_EVChargeParameterType" substitutionGroup="EVChargeParameter"/>
        <xs:complexType name="DC_EVChargeParameterType">
            <xs:complexContent>
                <xs:extension base="EVChargeParameterType">
                    <xs:sequence>
                        <xs:element name="DC_EVStatus" type="DC_EVStatusType"/>
                        <xs:element name="EVMaximumCurrentLimit" type="PhysicalValueType"/>
                        <xs:element name="EVMaximumPowerLimit" type="PhysicalValueType" minOccurs="0"/>
                        <xs:element name="EVMaximumVoltageLimit" type="PhysicalValueType"/>
                        <xs:element name="EVEnergyCapacity" type="PhysicalValueType" minOccurs="0"/>
                        <xs:element name="EVEnergyRequest" type="PhysicalValueType" minOccurs="0"/>
                        <xs:element name="FullSOC" type="percentValueType" minOccurs="0"/>
                        <xs:element name="BulkSOC" type="percentValueType" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="EVSEChargeParameterType" abstract="true"/>
        <xs:element name="EVSEChargeParameter" type="EVSEChargeParameterType"/>
        <xs:element name="AC_EVSEChargeParameter" type="AC_EVSEChargeParameterType"
substitutionGroup="EVSEChargeParameter"/>
        <xs:complexType name="AC_EVSEChargeParameterType">
            <xs:complexContent>
                <xs:extension base="EVSEChargeParameterType">
                    <xs:sequence>
                        <xs:element name="AC_EVSEStatus" type="AC_EVSEStatusType"/>
                        <xs:element name="EVSEMaxVoltage" type="PhysicalValueType"/>
                        <xs:element name="EVSEMaxCurrent" type="PhysicalValueType"/>
                        <xs:element name="EVSEMinCurrent" type="PhysicalValueType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="DC_EVSEChargeParameter" type="DC_EVSEChargeParameterType"
substitutionGroup="EVSEChargeParameter"/>
        <xs:complexType name="DC_EVSEChargeParameterType">
            <xs:complexContent>
                <xs:extension base="EVSEChargeParameterType">
                    <xs:sequence>
                        <xs:element name="DC_EVSEStatus" type="DC_EVSEStatusType"/>
                        <xs:element name="EVSEMaximumCurrentLimit" type="PhysicalValueType"/>
                        <xs:element name="EVSEMaximumPowerLimit" type="PhysicalValueType" minOccurs="0"/>
                        <xs:element name="EVSEMaximumVoltageLimit" type="PhysicalValueType"/>
                        <xs:element name="EVSEMinimumCurrentLimit" type="PhysicalValueType"/>
                        <xs:element name="EVSEMinimumVoltageLimit" type="PhysicalValueType"/>
                        <xs:element name="EVSECurrentRegulationTolerance" type="PhysicalValueType" minOccurs="0"/>
                        <xs:element name="EVSEPeakCurrentRipple" type="PhysicalValueType"/>
                        <xs:element name="EVSEEnergyToBeDelivered" type="PhysicalValueType" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                                    -->
        <!-- EV Power Delivery related types    -->
        <!--                                    -->
        <xs:complexType name="EVPowerDeliveryParameterType" abstract="true"/>
        <xs:element name="EVPowerDeliveryParameter" type="EVPowerDeliveryParameterType"/>
        <xs:element name="DC_EVPowerDeliveryParameter" type="DC_EVPowerDeliveryParameterType"
substitutionGroup="EVPowerDeliveryParameter"/>
        <xs:complexType name="DC_EVPowerDeliveryParameterType">
            <xs:complexContent>
                <xs:extension base="EVPowerDeliveryParameterType">
                    <xs:sequence>
                        <xs:element name="DC_EVStatus" type="DC_EVStatusType"/>
                        <xs:element name="BulkChargingComplete" type="xs:boolean" minOccurs="0"/>
                        <xs:element name="ChargingComplete" type="xs:boolean"/>
                    </xs:sequence>
                </xs:extension>
```

```
        </xs:complexContent>
    </xs:complexType>
    <!--                                    -->
    <!-- ChargingProfileType                -->
    <!--                                    -->
    <xs:complexType name="ChargingProfileType">
        <xs:sequence>
            <xs:element name="SAScheduleTupleID" type="SAIDType"/>
            <xs:element name="ProfileEntry" type="ProfileEntryType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ProfileEntryType">
        <xs:sequence>
            <xs:element name="ChargingProfileEntryStart" type="xs:unsignedInt"/>
            <xs:element name="ChargingProfileEntryMaxPower" type="PMaxType"/>
        </xs:sequence>
    </xs:complexType>
    <!-- ====================   -->
    <!-- Simple types           -->
    <!-- ====================   -->
    <!--                        -->
    <!-- General Types          -->
    <!--                        -->
    <xs:simpleType name="PMaxType">
        <xs:restriction base="xs:short"/>
    </xs:simpleType>
    <xs:simpleType name="percentValueType">
        <xs:restriction base="xs:byte">
            <xs:minInclusive value="0"/>
            <xs:maxInclusive value="100"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="faultMsgType">
        <xs:restriction base="xs:string">
            <xs:maxLength value="64"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="EVSEProcessingType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Finished"/>
            <xs:enumeration value="Ongoing"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="EVSENotificationType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="None"/>
            <xs:enumeration value="StopCharging"/>
            <xs:enumeration value="ReNegotiation"/>
        </xs:restriction>
    </xs:simpleType>
    <!--                                    -->
    <!-- service related types              -->
    <!--                                    -->
    <xs:simpleType name="serviceNameType">
        <xs:restriction base="xs:string">
            <xs:maxLength value="32"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="serviceCategoryType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="EVCharging"/>
            <xs:enumeration value="Internet"/>
            <xs:enumeration value="ContractCertificate"/>
            <xs:enumeration value="OtherCustom"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="serviceScopeType">
        <xs:restriction base="xs:string">
            <xs:maxLength value="32"/>
        </xs:restriction>
    </xs:simpleType>
    <!--                                    -->
    <!-- EnergyTransferType                 -->
    <!--                                    -->
    <xs:simpleType name="EVSESupportedEnergyTransferType">
```

```
            <xs:restriction base="xs:string">
                <xs:enumeration value="AC_single_phase_core"/>
                <xs:enumeration value="AC_three_phase_core"/>
                <xs:enumeration value="DC_core"/>
                <xs:enumeration value="DC_extended"/>
                <xs:enumeration value="DC_combo_core"/>
                <xs:enumeration value="DC_dual"/>
                <xs:enumeration value="AC_core1p_DC_extended"/>
                <xs:enumeration value="AC_single_DC_core"/>
                <xs:enumeration value="AC_single_phase_three_phase_core_DC_extended"/>
                <xs:enumeration value="AC_core3p_DC_extended"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="EVRequestedEnergyTransferType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="AC_single_phase_core"/>
                <xs:enumeration value="AC_three_phase_core"/>
                <xs:enumeration value="DC_core"/>
                <xs:enumeration value="DC_extended"/>
                <xs:enumeration value="DC_combo_core"/>
                <xs:enumeration value="DC_unique"/>
            </xs:restriction>
        </xs:simpleType>
        <!--                                -->
        <!-- security types                 -->
        <!--                                -->
        <xs:simpleType name="genChallengeType">
            <xs:restriction base="xs:string"/>
        </xs:simpleType>
        <xs:simpleType name="certificateType">
            <xs:restriction base="xs:base64Binary">
                <xs:maxLength value="1200"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="rootCertificateIDType">
            <xs:restriction base="xs:string">
                <xs:maxLength value="40"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="dHParamsType">
            <xs:restriction base="xs:base64Binary">
                <xs:maxLength value="256"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="privateKeyType">
            <xs:restriction base="xs:base64Binary">
                <xs:maxLength value="128"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="sigMeterReadingType">
            <xs:restriction base="xs:base64Binary">
                <xs:maxLength value="32"/>
            </xs:restriction>
        </xs:simpleType>
        <!--                                -->
        <!-- Identification Numbers         -->
        <!--                                -->
        <xs:simpleType name="sessionIDType">
            <xs:restriction base="xs:hexBinary">
                <xs:length value="8"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="evccIDType">
            <xs:restriction base="xs:hexBinary">
                <xs:maxLength value="8"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="evseIDType">
            <xs:restriction base="xs:hexBinary">
                <xs:maxLength value="32"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="serviceIDType">
            <xs:restriction base="xs:unsignedShort"/>
        </xs:simpleType>
        <xs:simpleType name="contractIDType">
```

```
        <xs:restriction base="xs:string">
            <xs:maxLength value="24"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="meterIDType">
        <xs:restriction base="xs:string">
            <xs:maxLength value="32"/>
        </xs:restriction>
    </xs:simpleType>
    <!--                    -->
    <!-- Tariffs and payment    -->
    <!--                    -->
    <xs:simpleType name="SAIDType">
        <xs:restriction base="xs:short"/>
    </xs:simpleType>
    <xs:simpleType name="tariffDescriptionType">
        <xs:restriction base="xs:string">
            <xs:maxLength value="32"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="costKindType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="relativePricePercentage"/>
            <xs:enumeration value="RenewableGenerationPercentage"/>
            <xs:enumeration value="CarbonDioxideEmission"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="PaymentOptionsType">
        <xs:sequence>
            <xs:element name="PaymentOption" type="paymentOptionType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:simpleType name="paymentOptionType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Contract"/>
            <xs:enumeration value="ExternalPayment"/>
        </xs:restriction>
    </xs:simpleType>
    <!--                    -->
    <!-- Fault and Response Codes    -->
    <!--                    -->
    <xs:simpleType name="faultCodeType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="ParsingError"/>
            <xs:enumeration value="NoTLSRootCertificatAvailable"/>
            <xs:enumeration value="UnknownError"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="responseCodeType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="OK"/>
            <xs:enumeration value="OK_NewSessionEstablished"/>
            <xs:enumeration value="OK_OldSessionJoined"/>
            <xs:enumeration value="OK_CertificateExpiresSoon"/>
            <xs:enumeration value="FAILED"/>
            <xs:enumeration value="FAILED_SequenceError"/>
            <xs:enumeration value="FAILED_ServiceIDInvalid"/>
            <xs:enumeration value="FAILED_UnknownSession"/>
            <xs:enumeration value="FAILED_ServiceSelectionInvalid"/>
            <xs:enumeration value="FAILED_PaymentSelectionInvalid"/>
            <xs:enumeration value="FAILED_CertificateExpired"/>
            <xs:enumeration value="FAILED_SignatureError"/>
            <xs:enumeration value="FAILED_NoCertificateAvailable"/>
            <xs:enumeration value="FAILED_CertChainError"/>
            <xs:enumeration value="FAILED_ChallengeInvalid"/>
            <xs:enumeration value="FAILED_ContractCanceled"/>
            <xs:enumeration value="FAILED_WrongChargeParameter"/>
            <xs:enumeration value="FAILED_PowerDeliveryNotApplied"/>
            <xs:enumeration value="FAILED_TariffSelectionInvalid"/>
            <xs:enumeration value="FAILED_ChargingProfileInvalid"/>
            <xs:enumeration value="FAILED_EVSEPresentVoltageToLow"/>
            <xs:enumeration value="FAILED_MeteringSignatureNotValid"/>
            <xs:enumeration value="FAILED_WrongEnergyTransferType"/>
        </xs:restriction>
    </xs:simpleType>
```

```
<!--                              -->
<!-- Multiplier and Unit Types -->
<!--                              -->
<xs:simpleType name="unitMultiplierType">
    <xs:restriction base="xs:byte">
        <xs:minInclusive value="-3"/>
        <xs:maxInclusive value="3"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="unitSymbolType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="h">
            <xs:annotation>
                <xs:documentation>Time in hours</xs:documentation>
            </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="m">
            <xs:annotation>
                <xs:documentation>Time in minutes</xs:documentation>
            </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="s">
            <xs:annotation>
                <xs:documentation>Time in seconds</xs:documentation>
            </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="A">
            <xs:annotation>
                <xs:documentation>Current in Ampere</xs:documentation>
            </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="Ah">
            <xs:annotation>
                <xs:documentation>Ampere hour</xs:documentation>
            </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="V">
            <xs:annotation>
                <xs:documentation>Voltage in Volt</xs:documentation>
            </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="VA">
            <xs:annotation>
                <xs:documentation>Apparent power in Volt Ampere</xs:documentation>
            </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="W">
            <xs:annotation>
                <xs:documentation>Active power in Watt</xs:documentation>
            </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="W/s">
            <xs:annotation>
                <xs:documentation>Watt per second</xs:documentation>
            </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="Wh">
            <xs:annotation>
                <xs:documentation>Real energy in Watt hours</xs:documentation>
            </xs:annotation>
        </xs:enumeration>
    </xs:restriction>
</xs:simpleType>
<!--                -->
<!-- only DC related  -->
<!--                -->
<xs:simpleType name="DC_EVSEStatusCodeType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="EVSE_NotReady"/>
        <xs:enumeration value="EVSE_Ready"/>
        <xs:enumeration value="EVSE_Shutdown"/>
        <xs:enumeration value="EVSE_UtilityInterruptEvent"/>
        <xs:enumeration value="EVSE_IsolationMonitoringActive"/>
        <xs:enumeration value="EVSE_EmergencyShutdown"/>
        <xs:enumeration value="EVSE_Malfunction"/>
        <xs:enumeration value="Reserved_8"/>
```

```
                <xs:enumeration value="Reserved_9"/>
                <xs:enumeration value="Reserved_A"/>
                <xs:enumeration value="Reserved_B"/>
                <xs:enumeration value="Reserved_C"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="isolationLevelType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="Invalid"/>
                <xs:enumeration value="Safe"/>
                <xs:enumeration value="Warning"/>
                <xs:enumeration value="Fault"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="DC_EVErrorCodeType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="NO_ERROR"/>
                <xs:enumeration value="FAILED_RESSTemperatureInhibit"/>
                <xs:enumeration value="FAILED_EVShiftPosition"/>
                <xs:enumeration value="FAILED_ChargerConnectorLockFault"/>
                <xs:enumeration value="FAILED_EVRESSMalfunction"/>
                <xs:enumeration value="FAILED_ChargingCurrentdifferential"/>
                <xs:enumeration value="FAILED_ChargingVoltageOutOfRange"/>
                <xs:enumeration value="Reserved_A"/>
                <xs:enumeration value="Reserved_B"/>
                <xs:enumeration value="Reserved_C"/>
                <xs:enumeration value="FAILED_ChargingSystemIncompatibility"/>
                <xs:enumeration value="NoData"/>
            </xs:restriction>
        </xs:simpleType>
</xs:schema>
```

C.7 xmldsig-core-schema.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE schema PUBLIC "-//W3C//DTD XMLSchema 200102//EN" "http://www.w3.org/2001/XMLSchema.dtd" [
  <!ATTLIST schema
    xmlns:ds CDATA #FIXED "http://www.w3.org/2000/09/xmldsig#">
  <!ENTITY dsig 'http://www.w3.org/2000/09/xmldsig#'>
  <!ENTITY % p ">
  <!ENTITY % s ">
 ]>
<!-- Schema for XML Signatures
    http://www.w3.org/2000/09/xmldsig#
    $Revision: 1.1 $ on $Date: 2002/02/08 20:32:26 $ by $Author: reagle $

    Copyright 2001 The Internet Society and W3C (Massachusetts Institute
    of Technology, Institut National de Recherche en Informatique et en
    Automatique, Keio University). All Rights Reserved.
    http://www.w3.org/Consortium/Legal/

    This document is governed by the W3C Software License [1] as described
    in the FAQ [2].

    [1] http://www.w3.org/Consortium/Legal/copyright-software-19980720
    [2] http://www.w3.org/Consortium/Legal/IPR-FAQ-20000620.html#DTD
-->
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
targetNamespace="http://www.w3.org/2000/09/xmldsig#" elementFormDefault="qualified" version="0.1">
        <!-- Basic Types Defined for Signatures -->
        <simpleType name="CryptoBinary">
            <restriction base="base64Binary"/>
        </simpleType>
        <!-- Start Signature -->
        <element name="Signature" type="ds:SignatureType"/>
        <complexType name="SignatureType">
            <sequence>
```

```
                <element ref="ds:SignedInfo"/>
                <element ref="ds:SignatureValue"/>
                <element ref="ds:KeyInfo" minOccurs="0"/>
                <element ref="ds:Object" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
        <attribute name="Id" type="ID" use="optional"/>
</complexType>
<element name="SignatureValue" type="ds:SignatureValueType"/>
<complexType name="SignatureValueType">
        <simpleContent>
                <extension base="base64Binary">
                        <attribute name="Id" type="ID" use="optional"/>
                </extension>
        </simpleContent>
</complexType>
<!-- Start SignedInfo -->
<element name="SignedInfo" type="ds:SignedInfoType"/>
<complexType name="SignedInfoType">
        <sequence>
                <element ref="ds:CanonicalizationMethod"/>
                <element ref="ds:SignatureMethod"/>
                <element ref="ds:Reference" maxOccurs="unbounded"/>
        </sequence>
        <attribute name="Id" type="ID" use="optional"/>
</complexType>
<element name="CanonicalizationMethod" type="ds:CanonicalizationMethodType"/>
<complexType name="CanonicalizationMethodType" mixed="true">
        <sequence>
                <any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                <!-- (0,unbounded) elements from (1,1) namespace -->
        </sequence>
        <attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>
<element name="SignatureMethod" type="ds:SignatureMethodType"/>
<complexType name="SignatureMethodType" mixed="true">
        <sequence>
                <element name="HMACOutputLength" type="ds:HMACOutputLengthType" minOccurs="0"/>
                <any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
                <!-- (0,unbounded) elements from (1,1) external namespace -->
        </sequence>
        <attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>
<!-- Start Reference -->
<element name="Reference" type="ds:ReferenceType"/>
<complexType name="ReferenceType">
        <sequence>
                <element ref="ds:Transforms" minOccurs="0"/>
                <element ref="ds:DigestMethod"/>
                <element ref="ds:DigestValue"/>
        </sequence>
        <attribute name="Id" type="ID" use="optional"/>
        <attribute name="URI" type="anyURI" use="optional"/>
        <attribute name="Type" type="anyURI" use="optional"/>
</complexType>
<element name="Transforms" type="ds:TransformsType"/>
<complexType name="TransformsType">
        <sequence>
                <element ref="ds:Transform" maxOccurs="unbounded"/>
        </sequence>
</complexType>
<element name="Transform" type="ds:TransformType"/>
<complexType name="TransformType" mixed="true">
        <choice minOccurs="0" maxOccurs="unbounded">
                <any namespace="##other" processContents="lax"/>
                <!-- (1,1) elements from (0,unbounded) namespaces -->
                <element name="XPath" type="string"/>
        </choice>
        <attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>
<!-- End Reference -->
<element name="DigestMethod" type="ds:DigestMethodType"/>
<complexType name="DigestMethodType" mixed="true">
        <sequence>
                <any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
        <attribute name="Algorithm" type="anyURI" use="required"/>
```

```
        </complexType>
        <element name="DigestValue" type="ds:DigestValueType"/>
        <simpleType name="DigestValueType">
            <restriction base="base64Binary"/>
        </simpleType>
        <!-- End SignedInfo -->
        <!-- Start KeyInfo -->
        <element name="KeyInfo" type="ds:KeyInfoType"/>
        <complexType name="KeyInfoType" mixed="true">
            <choice maxOccurs="unbounded">
                <element ref="ds:KeyName"/>
                <element ref="ds:KeyValue"/>
                <element ref="ds:RetrievalMethod"/>
                <element ref="ds:X509Data"/>
                <element ref="ds:PGPData"/>
                <element ref="ds:SPKIData"/>
                <element ref="ds:MgmtData"/>
                <any namespace="##other" processContents="lax"/>
                <!-- (1,1) elements from (0,unbounded) namespaces -->
            </choice>
            <attribute name="Id" type="ID" use="optional"/>
        </complexType>
        <element name="KeyName" type="string"/>
        <element name="MgmtData" type="string"/>
        <element name="KeyValue" type="ds:KeyValueType"/>
        <complexType name="KeyValueType" mixed="true">
            <choice>
                <element ref="ds:DSAKeyValue"/>
                <element ref="ds:RSAKeyValue"/>
                <any namespace="##other" processContents="lax"/>
            </choice>
        </complexType>
        <element name="RetrievalMethod" type="ds:RetrievalMethodType"/>
        <complexType name="RetrievalMethodType">
            <sequence>
                <element ref="ds:Transforms" minOccurs="0"/>
            </sequence>
            <attribute name="URI" type="anyURI"/>
            <attribute name="Type" type="anyURI" use="optional"/>
        </complexType>
        <!-- Start X509Data -->
        <element name="X509Data" type="ds:X509DataType"/>
        <complexType name="X509DataType">
            <sequence maxOccurs="unbounded">
                <choice>
                    <element name="X509IssuerSerial" type="ds:X509IssuerSerialType"/>
                    <element name="X509SKI" type="base64Binary"/>
                    <element name="X509SubjectName" type="string"/>
                    <element name="X509Certificate" type="base64Binary"/>
                    <element name="X509CRL" type="base64Binary"/>
                    <any namespace="##other" processContents="lax"/>
                </choice>
            </sequence>
        </complexType>
        <complexType name="X509IssuerSerialType">
            <sequence>
                <element name="X509IssuerName" type="string"/>
                <element name="X509SerialNumber" type="integer"/>
            </sequence>
        </complexType>
        <!-- End X509Data -->
        <!-- Begin PGPData -->
        <element name="PGPData" type="ds:PGPDataType"/>
        <complexType name="PGPDataType">
            <choice>
                <sequence>
                    <element name="PGPKeyID" type="base64Binary"/>
                    <element name="PGPKeyPacket" type="base64Binary" minOccurs="0"/>
                    <any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
                </sequence>
                <sequence>
                    <element name="PGPKeyPacket" type="base64Binary"/>
                    <any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
                </sequence>
            </choice>
```

```
        </complexType>
        <!-- End PGPData -->
        <!-- Begin SPKIData -->
        <element name="SPKIData" type="ds:SPKIDataType"/>
        <complexType name="SPKIDataType">
            <sequence maxOccurs="unbounded">
                <element name="SPKISexp" type="base64Binary"/>
                <any namespace="##other" processContents="lax" minOccurs="0"/>
            </sequence>
        </complexType>
        <!-- End SPKIData -->
        <!-- End KeyInfo -->
        <!-- Start Object (Manifest, SignatureProperty) -->
        <element name="Object" type="ds:ObjectType"/>
        <complexType name="ObjectType" mixed="true">
            <sequence minOccurs="0" maxOccurs="unbounded">
                <any namespace="##any" processContents="lax"/>
            </sequence>
            <attribute name="Id" type="ID" use="optional"/>
            <attribute name="MimeType" type="string" use="optional"/>
            <attribute name="Encoding" type="anyURI" use="optional"/>
            <!-- add a grep facet -->
        </complexType>
        <element name="Manifest" type="ds:ManifestType"/>
        <complexType name="ManifestType">
            <sequence>
                <element ref="ds:Reference" maxOccurs="unbounded"/>
            </sequence>
            <attribute name="Id" type="ID" use="optional"/>
        </complexType>
        <element name="SignatureProperties" type="ds:SignaturePropertiesType"/>
        <complexType name="SignaturePropertiesType">
            <sequence>
                <element ref="ds:SignatureProperty" maxOccurs="unbounded"/>
            </sequence>
            <attribute name="Id" type="ID" use="optional"/>
        </complexType>
        <element name="SignatureProperty" type="ds:SignaturePropertyType"/>
        <complexType name="SignaturePropertyType" mixed="true">
            <choice maxOccurs="unbounded">
                <any namespace="##other" processContents="lax"/>
                <!-- (1,1) elements from (1,unbounded) namespaces -->
            </choice>
            <attribute name="Target" type="anyURI" use="required"/>
            <attribute name="Id" type="ID" use="optional"/>
        </complexType>
        <!-- End Object (Manifest, SignatureProperty) -->
        <!-- Start Algorithm Parameters -->
        <simpleType name="HMACOutputLengthType">
            <restriction base="integer"/>
        </simpleType>
        <!-- Start KeyValue Element-types -->
        <element name="DSAKeyValue" type="ds:DSAKeyValueType"/>
        <complexType name="DSAKeyValueType">
            <sequence>
                <sequence minOccurs="0">
                    <element name="P" type="ds:CryptoBinary"/>
                    <element name="Q" type="ds:CryptoBinary"/>
                </sequence>
                <element name="G" type="ds:CryptoBinary" minOccurs="0"/>
                <element name="Y" type="ds:CryptoBinary"/>
                <element name="J" type="ds:CryptoBinary" minOccurs="0"/>
                <sequence minOccurs="0">
                    <element name="Seed" type="ds:CryptoBinary"/>
                    <element name="PgenCounter" type="ds:CryptoBinary"/>
                </sequence>
            </sequence>
        </complexType>
        <element name="RSAKeyValue" type="ds:RSAKeyValueType"/>
        <complexType name="RSAKeyValueType">
            <sequence>
                <element name="Modulus" type="ds:CryptoBinary"/>
                <element name="Exponent" type="ds:CryptoBinary"/>
            </sequence>
        </complexType>
        <!-- End KeyValue Element-types -->
```

```
        <!-- End Signature -->
</schema>
```

Annex D
(informative)

Message examples

D.1 Value Added Service selection

The following XSD give an example how the selection of VAS can be implemented. The following assumptions is made: EVCC wants to use InternetAccess with the protocols HTTP and FTP, if offered.

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_d="urn:iso:15118:2:2010:MsgDef"
 xmlns:v2gci_t="urn:iso:15118:2:2010:MsgDataTypes"
 xmlns:v2gci_h="urn:iso:15118:2:2010:MsgHeader"
 xmlns:v2gci_b="urn:iso:15118:2:2010:MsgBody"
 xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="urn:iso:15118:2:2010:MsgDef file:/D:/Data/MCHN4240/Siemens/eMobility/xsd/svn/trunk/15118-
2_XMLSchemas/V2G_CI_MsgDef.xsd">
   <v2gci_d:Header>
        <v2gci_t:SessionID>3031323334353637</v2gci_t:SessionID>
   </v2gci_d:Header>
   <v2gci_d:Body>
      <v2gci_b:ServiceDiscoveryRes>
         <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
         <v2gci_b:PaymentOptions>
            <v2gci_t:PaymentOption>Contract</v2gci_t:PaymentOption>
            <v2gci_t:PaymentOption>ExternalPayment</v2gci_t:PaymentOption>
         </v2gci_b:PaymentOptions>
         <v2gci_b:ChargeService>
            <v2gci_t:ServiceTag>
               <v2gci_t:ServiceID>1</v2gci_t:ServiceID>
               <v2gci_t:ServiceName>AC_DC_Charging</v2gci_t:ServiceName>
            </v2gci_t:ServiceTag>
            <v2gci_t:FreeService>true</v2gci_t:FreeService>
            <v2gci_t:EnergyTransferType>AC_core3p_DC_extended</v2gci_t:EnergyTransferType>
         </v2gci_b:ChargeService>
         <v2gci_b:ServiceList>
              <v2gci_t:Service>
               <v2gci_t:ServiceTag>
                  <v2gci_t:ServiceID>2</v2gci_t:ServiceID>
                  <v2gci_t:ServiceName> InternetAccess</v2gci_t:ServiceName>
                  <v2gci_t:ServiceCategory>Internet</v2gci_t:ServiceCategory>
               </v2gci_t:ServiceTag>
               <v2gci_t:FreeService>false</v2gci_t:FreeService>
              </v2gci_t:Service>
            <v2gci_t:Service>
               <v2gci_t:ServiceTag>
                  <v2gci_t:ServiceID>3</v2gci_t:ServiceID>
                  <v2gci_t:ServiceName> Certificate </v2gci_t:ServiceName>
                  <v2gci_t:ServiceCategory>ContractCertificate</v2gci_t:ServiceCategory>
               </v2gci_t:ServiceTag>
               <v2gci_t:FreeService>false</v2gci_t:FreeService>
            </v2gci_t:Service>
         </v2gci_b:ServiceList>
      </v2gci_b:ServiceDiscoveryRes>
   </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 6 – ServiceDiscoveryRes message

In this message the SECC offers AC and DC Charging, Internet Access, Certificate handling. The offered payment options are Contract and ExternalPayment

Now the EVCC requests the details for Certificate and InternetAccess using two times the ServiceDetailsReq. The ServiceDetailReq/Res pair for Internet Service looks like:

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2010:MsgHeader"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:v2gci_b="urn:iso:15118:2:2010:MsgBody"
 xmlns:v2gci_d="urn:iso:15118:2:2010:MsgDef"
 xmlns:v2gci_t="urn:iso:15118:2:2010:MsgDataTypes"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="urn:iso:15118:2:2010:MsgDef file:/D:/Data/MCHN4240/Siemens/eMobility/xsd/svn/trunk/15118-
2_XMLSchemas/V2G_CI_MsgDef.xsd">
   <v2gci_d:Header>
         <v2gci_t:SessionID>3031323334353637</v2gci_t:SessionID>
   </v2gci_d:Header>
   <v2gci_d:Body>
      <v2gci_b:ServiceDetailReq>
         <v2gci_b:ServiceID>2</v2gci_b:ServiceID>
      </v2gci_b:ServiceDetailReq>
   </v2gci_d:Body>
</v2gci_d:V2G_Message>



<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2010:MsgHeader"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:v2gci_b="urn:iso:15118:2:2010:MsgBody"
 xmlns:v2gci_d="urn:iso:15118:2:2010:MsgDef"
 xmlns:v2gci_t="urn:iso:15118:2:2010:MsgDataTypes"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="urn:iso:15118:2:2010:MsgDef file:/D:/Data/MCHN4240/Siemens/eMobility/xsd/svn/trunk/15118-
2_XMLSchemas/V2G_CI_MsgDef.xsd">
   <v2gci_d:Header>
         <v2gci_t:SessionID>3031323334353637</v2gci_t:SessionID>
   </v2gci_d:Header>
   <v2gci_d:Body>
      <v2gci_b:ServiceDetailRes>
         <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
         <v2gci_b:ServiceID>2</v2gci_b:ServiceID>
         <v2gci_b:ServiceParameterList>
            <v2gci_t:ParameterSet>
               <v2gci_t:ParameterSetID>1</v2gci_t:ParameterSetID>
               <v2gci_t:Parameter Name="Protocol" ValueType="string">
                  <v2gci_t:stringValue>HTTP</v2gci_t:stringValue>
               </v2gci_t:Parameter>
               <v2gci_t:Parameter Name="Port" ValueType="int">
                  <v2gci_t:intValue>80</v2gci_t:intValue>
               </v2gci_t:Parameter>
            </v2gci_t:ParameterSet>
            <v2gci_t:ParameterSet>
               <v2gci_t:ParameterSetID>2</v2gci_t:ParameterSetID>
               <v2gci_t:Parameter Name="Protocol" ValueType="string">
                  <v2gci_t:stringValue>HTTPS</v2gci_t:stringValue>
               </v2gci_t:Parameter>
               <v2gci_t:Parameter Name="Port" ValueType="int">
                  <v2gci_t:intValue>81</v2gci_t:intValue>
               </v2gci_t:Parameter>
            </v2gci_t:ParameterSet>
         </v2gci_b:ServiceParameterList>
      </v2gci_b:ServiceDetailRes>
   </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 7 – ServiceDetailsReq and ServiceDetailsRes message

The response message shows that the SECC offers internet access via HTTP using port 80 and via HTTPS using port 81, therefore the EVCC is not allowed to use FTP but to use HTTP.

Now, the EVCC request the desired services using the ServicePaymentSelectionReq

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2010:MsgHeader"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:v2gci_b="urn:iso:15118:2:2010:MsgBody"
 xmlns:v2gci_d="urn:iso:15118:2:2010:MsgDef"
 xmlns:v2gci_t="urn:iso:15118:2:2010:MsgDataTypes"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="urn:iso:15118:2:2010:MsgDef file:/D:/Data/MCHN4240/Siemens/eMobility/xsd/svn/trunk/15118-
2_XMLSchemas/V2G_CI_MsgDef.xsd">
   <v2gci_d:Header>
         <v2gci_t:SessionID>3031323334353637</v2gci_t:SessionID>
   </v2gci_d:Header>
   <v2gci_d:Body>
      <v2gci_b:ServicePaymentSelectionReq>
         <v2gci_b:SelectedPaymentOption>Contract</v2gci_b:SelectedPaymentOption>
         <v2gci_b:SelectedServiceList>
            <v2gci_t:SelectedService>
               <v2gci_t:ServiceID>1</v2gci_t:ServiceID> <!-- charge service -->
            </v2gci_t:SelectedService>
            <v2gci_t:SelectedService>
               <v2gci_t:ServiceID>2</v2gci_t:ServiceID>  <!-- internet service -->
               <v2gci_t:ParameterSetID>1</v2gci_t:ParameterSetID>  <!—selection of http via port 80 -->
            </v2gci_t:SelectedService>
         </v2gci_b:SelectedServiceList>
      </v2gci_b:ServicePaymentSelectionReq>
   </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 8 – ServicePaymentSelectionReq message

D.2 EXI encoded message examples

In the following subclause you will find 3 examples (SessionSetupRes, ChargeParameterDiscoveryReq, and CurrentDemandRequest) of plain XML V2G message instances and the equivalent EXI format representation based on the EXI settings declared in section

D.2.1 SessionSetupReq message

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2010:MsgHeader"
 xmlns:v2gci_d="urn:iso:15118:2:2010:MsgDef"
 xmlns:v2gci_t="urn:iso:15118:2:2010:MsgDataTypes"
 xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
 xmlns:v2gci_b="urn:iso:15118:2:2010:MsgBody"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <v2gci_d:Header>
      <v2gci_h:SessionID>3031323334353637</v2gci_h:SessionID>
   </v2gci_d:Header>
   <v2gci_d:Body>
      <v2gci_b:SessionSetupRes>
         <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
         <v2gci_b:EVSEID>abc123</v2gci_b:EVSEID>
      </v2gci_b:SessionSetupRes>
   </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 9 – Plain XML representation of a SessionSetupReq message

```
80 9A 02 0C 0C 4C 8C CD 0D 4D 8D D1 E0 00 0E AF 04 8C 80
```

V2G message example 10 – EXI data stream representation of the SessionSetupReq message

D.2.2 ChargeParameterDiscoveryReq message (AC-based)

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2010:MsgHeader"
 xmlns:v2gci_d="urn:iso:15118:2:2010:MsgDef"
 xmlns:v2gci_t="urn:iso:15118:2:2010:MsgDataTypes"
 xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
 xmlns:v2gci_b="urn:iso:15118:2:2010:MsgBody"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <v2gci_d:Header>
      <v2gci_h:SessionID>3031323334353637</v2gci_h:SessionID>
   </v2gci_d:Header>
   <v2gci_d:Body>
      <v2gci_b:ChargeParameterDiscoveryReq>
         <v2gci_b:EVRequestedEnergyTransferType>AC_three_phase_core</v2gci_b:EVRequestedEnergyTransferType>
         <v2gci_t:AC_EVChargeParameter>
            <v2gci_t:DepartureTime>100</v2gci_t:DepartureTime>
            <v2gci_t:EAmount>
               <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
               <v2gci_t:Unit>Wh</v2gci_t:Unit>
               <v2gci_t:Value>1000</v2gci_t:Value>
            </v2gci_t:EAmount>
            <v2gci_t:EVMaxVoltage>
               <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
               <v2gci_t:Unit>V</v2gci_t:Unit>
               <v2gci_t:Value>200</v2gci_t:Value>
            </v2gci_t:EVMaxVoltage>
            <v2gci_t:EVMaxCurrent>
               <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
               <v2gci_t:Unit>A</v2gci_t:Unit>
               <v2gci_t:Value>100</v2gci_t:Value>
            </v2gci_t:EVMaxCurrent>
            <v2gci_t:EVMinCurrent>
               <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
               <v2gci_t:Unit>A</v2gci_t:Unit>
               <v2gci_t:Value>20</v2gci_t:Value>
            </v2gci_t:EVMinCurrent>
         </v2gci_t:AC_EVChargeParameter>
      </v2gci_b:ChargeParameterDiscoveryReq>
   </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 11 – Plain XML representation of a ChargeParameterDiscoveryReq message (AC-based)

```
80 9A 02 0C 0C 4C 8C CD 0D 4D 8D D0 70 81 90 18 48 74 03 81 82 86 40 08 18 18 32 01 81 80 A0 00
```

V2G message example 12 – EXI data stream representation of the SessionSetupReq message

D.2.3 CurrentDemandReq message

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2010:MsgHeader"
 xmlns:v2gci_d="urn:iso:15118:2:2010:MsgDef"
 xmlns:v2gci_t="urn:iso:15118:2:2010:MsgDataTypes"
 xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
 xmlns:v2gci_b="urn:iso:15118:2:2010:MsgBody"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <v2gci_d:Header>
      <v2gci_h:SessionID>3031323334353637</v2gci_h:SessionID>
   </v2gci_d:Header>
   <v2gci_d:Body>
      <v2gci_b:CurrentDemandReq>
         <v2gci_b:DC_EVStatus>
            <v2gci_t:EVReady>true</v2gci_t:EVReady>
            <v2gci_t:EVErrorCode>NO_ERROR</v2gci_t:EVErrorCode>
            <v2gci_t:EVRESSSOC>55</v2gci_t:EVRESSSOC>
         </v2gci_b:DC_EVStatus>
         <v2gci_b:EVTargetCurrent>
            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
            <v2gci_t:Unit>A</v2gci_t:Unit>
            <v2gci_t:Value>100</v2gci_t:Value>
         </v2gci_b:EVTargetCurrent>
         <v2gci_b:ChargingComplete>false</v2gci_b:ChargingComplete>
         <v2gci_b:EVTargetVoltage>
            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
            <v2gci_t:Unit>V</v2gci_t:Unit>
            <v2gci_t:Value>200</v2gci_t:Value>
         </v2gci_b:EVTargetVoltage>
      </v2gci_b:CurrentDemandReq>
   </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 13 – Plain XML representation of a CurrentDemandReq message

```
80 9A 02 0C 0C 4C 8C CD 0D 4D 8D D0 D1 40 0D C0 C0 C1 90 82 18 28 64 00 80
```

V2G message example 14 – EXI data stream representation of the SessionSetupReq message

D.3 Schedules and Tariff Information

This clause provides an overview for best practices on how different tariff models may be implemented as part of the ChargeParameterDiscovery Req/Res pattern. The following examples are given:

- Dynamic GridSchedule w/o SalesTariff over ISO/IEC 15118 V2G CI (see D.3.1)
- “Time Of Use”-based SalesTariff including constant value for GridSchedule (see D.3.2)
- “Time Of Use”-based SalesTariff with dynamic GridSchedule (see D.3.3)
- “Consumption”-based SalesTariff with constant value for GridSchedule (see D.3.4)
- Multiple SalesTariffs with different Demand Limits in GridSchedule (see D.3.5)

NOTE 1: All following examples show the *body* element of the *ChargeParameterDiscovery Response* for AC-based charging.

NOTE 2: The list of examples given in this Annex is not exhaustive. Variations & combinations of such schedule and tariff information may be applicable based on the underlying e.g. business model.

D.3.1 Dynamic GridSchedule w/o SalesTariff over ISO/IEC 15118 V2G CI

```
<v2gci_d:Body>
    <v2gci_b:ChargeParameterDiscoveryRes>
      <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
      <v2gci_t:SAScheduleList>
        <v2gci_t:SAScheduleTuple>
          <v2gci_t:SAScheduleTupleID>24512</v2gci_t:SAScheduleTupleID>
          <v2gci_t:PMaxSchedule>
            <v2gci_t:PMaxScheduleID>23451</v2gci_t:PMaxScheduleID>
            <v2gci_t:PMaxScheduleEntry>
              <v2gci_t:RelativeTimeInterval>
                <v2gci_t:start>0</v2gci_t:start>
              </v2gci_t:RelativeTimeInterval>
              <v2gci_t:PMax>9600</v2gci_t:PMax>
            </v2gci_t:PMaxScheduleEntry>
            <v2gci_t:PMaxScheduleEntry>
              <v2gci_t:RelativeTimeInterval>
                <v2gci_t:start>2341</v2gci_t:start>
              </v2gci_t:RelativeTimeInterval>
              <v2gci_t:PMax>22000</v2gci_t:PMax>
            </v2gci_t:PMaxScheduleEntry>
            <v2gci_t:PMaxScheduleEntry>
              <v2gci_t:RelativeTimeInterval>
                <v2gci_t:start>9541</v2gci_t:start>
              </v2gci_t:RelativeTimeInterval>
              <v2gci_t:PMax>9600</v2gci_t:PMax>
            </v2gci_t:PMaxScheduleEntry>
            <v2gci_t:PMaxScheduleEntry>
              <v2gci_t:RelativeTimeInterval>
                <v2gci_t:start>23941</v2gci_t:start>
              </v2gci_t:RelativeTimeInterval>
              <v2gci_t:PMax>22000</v2gci_t:PMax>
            </v2gci_t:PMaxScheduleEntry>
            <v2gci_t:PMaxScheduleEntry>
              <v2gci_t:RelativeTimeInterval>
                <v2gci_t:start>31141</v2gci_t:start>
                <v2gci_t:duration>7200</v2gci_t:duration>
              </v2gci_t:RelativeTimeInterval>
              <v2gci_t:PMax>12000</v2gci_t:PMax>
            </v2gci_t:PMaxScheduleEntry>
          </v2gci_t:PMaxSchedule>
        </v2gci_t:SAScheduleTuple>
      </v2gci_t:SAScheduleList>
      <v2gci_t:AC_EVSEChargeParameter>
        <v2gci_t:AC_EVSEStatus>
          <v2gci_t:PowerSwitchClosed>false</v2gci_t:PowerSwitchClosed>
          <v2gci_t:RCD>false</v2gci_t:RCD>
          <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
          <v2gci_t:IndicationTargetTime>0</v2gci_t:IndicationTargetTime>
        </v2gci_t:AC_EVSEStatus>
        <v2gci_t:EVSEMaxVoltage>
          <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
          <v2gci_t:Value>230</v2gci_t:Value>
        </v2gci_t:EVSEMaxVoltage>
        <v2gci_t:EVSEMaxCurrent>
          <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
          <v2gci_t:Value>50</v2gci_t:Value>
        </v2gci_t:EVSEMaxCurrent>
        <v2gci_t:EVSEMinCurrent>
          <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
```

```
            <v2gci_t:Value>0</v2gci_t:Value>
          </v2gci_t:EVSEMinCurrent>
        </v2gci_t:AC_EVSEChargeParameter>
      </v2gci_b:ChargeParameterDiscoveryRes>
    </v2gci_d:Body>
```

V2G message example 15 – ChargeParameterDiscovery Response example for GridSchedule w/o SalesTariff

D.3.2 "Time Of Use"-based SalesTariff with constant value for GridSchedule

```
<v2gci_d:Body>
  <v2gci_b:ChargeParameterDiscoveryRes>
    <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
    <v2gci_t:SAScheduleList>
      <v2gci_t:SAScheduleTuple>
        <v2gci_t:SAScheduleTupleID>24512</v2gci_t:SAScheduleTupleID>
        <v2gci_t:PMaxSchedule>
          <v2gci_t:PMaxScheduleID>23451</v2gci_t:PMaxScheduleID>
          <v2gci_t:PMaxScheduleEntry>
            <v2gci_t:RelativeTimeInterval>
              <v2gci_t:start>0</v2gci_t:start>
              <v2gci_t:duration>30000</v2gci_t:duration>
            </v2gci_t:RelativeTimeInterval>
            <v2gci_t:PMax>9600</v2gci_t:PMax>
          </v2gci_t:PMaxScheduleEntry>
        </v2gci_t:PMaxSchedule>
        <v2gci_t:SalesTariff Id="ID001"> <!-- used by XML Signature Framework for signature assignment in msg header -->
          <v2gci_t:SalesTariffID>10001</v2gci_t:SalesTariffID>
          <v2gci_t:SalesTariffDescription>SalesTariffDescription1</v2gci_t:SalesTariffDescription>
          <v2gci_t:NumEPriceLevels>3</v2gci_t:NumEPriceLevels>
          <v2gci_t:SalesTariffEntry>
            <v2gci_t:RelativeTimeInterval>
              <v2gci_t:start>0</v2gci_t:start>
            </v2gci_t:RelativeTimeInterval>
            <v2gci_t:EPriceLevel>1</v2gci_t:EPriceLevel>
          </v2gci_t:SalesTariffEntry>
          <v2gci_t:SalesTariffEntry>
            <v2gci_t:RelativeTimeInterval>
              <v2gci_t:start>2147</v2gci_t:start>
            </v2gci_t:RelativeTimeInterval>
            <v2gci_t:EPriceLevel>2</v2gci_t:EPriceLevel>
          </v2gci_t:SalesTariffEntry>
          <v2gci_t:SalesTariffEntry>
            <v2gci_t:RelativeTimeInterval>
              <v2gci_t:start>9874</v2gci_t:start>
            </v2gci_t:RelativeTimeInterval>
            <v2gci_t:EPriceLevel>3</v2gci_t:EPriceLevel>
          </v2gci_t:SalesTariffEntry>
          <v2gci_t:SalesTariffEntry>
            <v2gci_t:RelativeTimeInterval>
              <v2gci_t:start>14937</v2gci_t:start>
            </v2gci_t:RelativeTimeInterval>
            <v2gci_t:EPriceLevel>2</v2gci_t:EPriceLevel>
          </v2gci_t:SalesTariffEntry>
          <v2gci_t:SalesTariffEntry>
            <v2gci_t:RelativeTimeInterval>
              <v2gci_t:start>24431</v2gci_t:start>
              <v2gci_t:duration>30000</v2gci_t:duration>
            </v2gci_t:RelativeTimeInterval>
            <v2gci_t:EPriceLevel>1</v2gci_t:EPriceLevel>
          </v2gci_t:SalesTariffEntry>
```

```
            </v2gci_t:SalesTariff>
          </v2gci_t:SAScheduleTuple>
        </v2gci_t:SAScheduleList>
        <v2gci_t:AC_EVSEChargeParameter>
          <v2gci_t:AC_EVSEStatus>
            <v2gci_t:PowerSwitchClosed>false</v2gci_t:PowerSwitchClosed>
            <v2gci_t:RCD>false</v2gci_t:RCD>
            <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
            <v2gci_t:IndicationTargetTime>0</v2gci_t:IndicationTargetTime>
          </v2gci_t:AC_EVSEStatus>
          <v2gci_t:EVSEMaxVoltage>
            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
            <v2gci_t:Value>230</v2gci_t:Value>
          </v2gci_t:EVSEMaxVoltage>
          <v2gci_t:EVSEMaxCurrent>
            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
            <v2gci_t:Value>50</v2gci_t:Value>
          </v2gci_t:EVSEMaxCurrent>
          <v2gci_t:EVSEMinCurrent>
            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
            <v2gci_t:Value>0</v2gci_t:Value>
          </v2gci_t:EVSEMinCurrent>
        </v2gci_t:AC_EVSEChargeParameter>
      </v2gci_b:ChargeParameterDiscoveryRes>
    </v2gci_d:Body>
```

V2G message example 16 – ChargeParameterDiscovery Response example for "Time of Use"-based SalesTariff with constant value for GridSchedule

D.3.3 "Time Of Use"-based SalesTariff with dynamic GridSchedule

```
    <v2gci_d:Body>
      <v2gci_b:ChargeParameterDiscoveryRes>
        <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
        <v2gci_t:SAScheduleList>
          <v2gci_t:SAScheduleTuple>
            <v2gci_t:SAScheduleTupleID>24512</v2gci_t:SAScheduleTupleID>
            <v2gci_t:PMaxSchedule>
              <v2gci_t:PMaxScheduleID>23451</v2gci_t:PMaxScheduleID>
              <v2gci_t:PMaxScheduleEntry>
                <v2gci_t:RelativeTimeInterval>
                  <v2gci_t:start>0</v2gci_t:start>
                </v2gci_t:RelativeTimeInterval>
                <v2gci_t:PMax>9600</v2gci_t:PMax>
              </v2gci_t:PMaxScheduleEntry>
              <v2gci_t:PMaxScheduleEntry>
                <v2gci_t:RelativeTimeInterval>
                  <v2gci_t:start>2341</v2gci_t:start>
                </v2gci_t:RelativeTimeInterval>
                <v2gci_t:PMax>22000</v2gci_t:PMax>
              </v2gci_t:PMaxScheduleEntry>
              <v2gci_t:PMaxScheduleEntry>
                <v2gci_t:RelativeTimeInterval>
                  <v2gci_t:start>9541</v2gci_t:start>
                </v2gci_t:RelativeTimeInterval>
                <v2gci_t:PMax>9600</v2gci_t:PMax>
              </v2gci_t:PMaxScheduleEntry>
              <v2gci_t:PMaxScheduleEntry>
                <v2gci_t:RelativeTimeInterval>
                  <v2gci_t:start>23941</v2gci_t:start>
                </v2gci_t:RelativeTimeInterval>
                <v2gci_t:PMax>22000</v2gci_t:PMax>
```

```
                </v2gci_t:PMaxScheduleEntry>
                <v2gci_t:PMaxScheduleEntry>
                  <v2gci_t:RelativeTimeInterval>
                    <v2gci_t:start>31141</v2gci_t:start>
                    <v2gci_t:duration>36000</v2gci_t:duration>
                  </v2gci_t:RelativeTimeInterval>
                  <v2gci_t:PMax>12000</v2gci_t:PMax>
                </v2gci_t:PMaxScheduleEntry>
              </v2gci_t:PMaxSchedule>
              <v2gci_t:SalesTariff Id="ID001"> <!-- used by XML Signature Framework for signature assignment in msg header -->
                <v2gci_t:SalesTariffID>10001</v2gci_t:SalesTariffID>
                <v2gci_t:SalesTariffDescription>SalesTariffDescription1</v2gci_t:SalesTariffDescription>
                <v2gci_t:NumEPriceLevels>3</v2gci_t:NumEPriceLevels>
                <v2gci_t:SalesTariffEntry>
                  <v2gci_t:RelativeTimeInterval>
                    <v2gci_t:start>0</v2gci_t:start>
                  </v2gci_t:RelativeTimeInterval>
                  <v2gci_t:EPriceLevel>1</v2gci_t:EPriceLevel>
                </v2gci_t:SalesTariffEntry>
                <v2gci_t:SalesTariffEntry>
                  <v2gci_t:RelativeTimeInterval>
                    <v2gci_t:start>2147</v2gci_t:start>
                  </v2gci_t:RelativeTimeInterval>
                  <v2gci_t:EPriceLevel>2</v2gci_t:EPriceLevel>
                </v2gci_t:SalesTariffEntry>
                <v2gci_t:SalesTariffEntry>
                  <v2gci_t:RelativeTimeInterval>
                    <v2gci_t:start>9874</v2gci_t:start>
                  </v2gci_t:RelativeTimeInterval>
                  <v2gci_t:EPriceLevel>3</v2gci_t:EPriceLevel>
                </v2gci_t:SalesTariffEntry>
                <v2gci_t:SalesTariffEntry>
                  <v2gci_t:RelativeTimeInterval>
                    <v2gci_t:start>14937</v2gci_t:start>
                  </v2gci_t:RelativeTimeInterval>
                  <v2gci_t:EPriceLevel>2</v2gci_t:EPriceLevel>
                </v2gci_t:SalesTariffEntry>
                <v2gci_t:SalesTariffEntry>
                  <v2gci_t:RelativeTimeInterval>
                    <v2gci_t:start>24431</v2gci_t:start>
                    <v2gci_t:duration>36000</v2gci_t:duration>
                  </v2gci_t:RelativeTimeInterval>
                  <v2gci_t:EPriceLevel>1</v2gci_t:EPriceLevel>
                </v2gci_t:SalesTariffEntry>
              </v2gci_t:SalesTariff>
            </v2gci_t:SAScheduleTuple>
          </v2gci_t:SAScheduleList>
          <v2gci_t:AC_EVSEChargeParameter>
            <v2gci_t:AC_EVSEStatus>
              <v2gci_t:PowerSwitchClosed>false</v2gci_t:PowerSwitchClosed>
              <v2gci_t:RCD>false</v2gci_t:RCD>
              <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
              <v2gci_t:IndicationTargetTime>0</v2gci_t:IndicationTargetTime>
            </v2gci_t:AC_EVSEStatus>
            <v2gci_t:EVSEMaxVoltage>
              <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
              <v2gci_t:Value>230</v2gci_t:Value>
            </v2gci_t:EVSEMaxVoltage>
            <v2gci_t:EVSEMaxCurrent>
              <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
              <v2gci_t:Value>50</v2gci_t:Value>
            </v2gci_t:EVSEMaxCurrent>
```

```
            <v2gci_t:EVSEMinCurrent>
               <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
               <v2gci_t:Value>0</v2gci_t:Value>
            </v2gci_t:EVSEMinCurrent>
         </v2gci_t:AC_EVSEChargeParameter>
      </v2gci_b:ChargeParameterDiscoveryRes>
   </v2gci_d:Body>
```

V2G message example 17 – ChargeParameterDiscovery Response example for "Time of Use"-based SalesTariff with dynamic GridSchedule

D.3.4 "Consumption"-based SalesTariff with constant value for GridSchedule

```
   <v2gci_d:Body>
      <v2gci_b:ChargeParameterDiscoveryRes>
         <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
         <v2gci_t:SAScheduleList>
            <v2gci_t:SAScheduleTuple>
               <v2gci_t:SAScheduleTupleID>24512</v2gci_t:SAScheduleTupleID>
               <v2gci_t:PMaxSchedule>
                  <v2gci_t:PMaxScheduleID>23451</v2gci_t:PMaxScheduleID>
                  <v2gci_t:PMaxScheduleEntry>
                     <v2gci_t:RelativeTimeInterval>
                        <v2gci_t:start>0</v2gci_t:start>
                        <v2gci_t:duration>36000</v2gci_t:duration>
                     </v2gci_t:RelativeTimeInterval>
                     <v2gci_t:PMax>9600</v2gci_t:PMax>
                  </v2gci_t:PMaxScheduleEntry>
               </v2gci_t:PMaxSchedule>
               <v2gci_t:SalesTariff Id="ID001"> <!-- used by XML Signature Framework for signature assignment in msg header -->
                  <v2gci_t:SalesTariffID>10001</v2gci_t:SalesTariffID>
                  <v2gci_t:SalesTariffDescription>Standard</v2gci_t:SalesTariffDescription>
                  <v2gci_t:NumEPriceLevels>1</v2gci_t:NumEPriceLevels>
                  <v2gci_t:SalesTariffEntry>
                     <v2gci_t:RelativeTimeInterval>
                        <v2gci_t:start>0</v2gci_t:start>
                        <v2gci_t:duration>36000</v2gci_t:duration>
                     </v2gci_t:RelativeTimeInterval>
                     <v2gci_t:EPriceLevel>1</v2gci_t:EPriceLevel>
                     <v2gci_t:ConsumptionCost>
                        <v2gci_t:startValue>0</v2gci_t:startValue>
                        <v2gci_t:Cost>
                           <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
                           <v2gci_t:amount>90</v2gci_t:amount>
                           <v2gci_t:amountMultiplier>0</v2gci_t:amountMultiplier>
                        </v2gci_t:Cost>
                     </v2gci_t:ConsumptionCost>
                     <v2gci_t:ConsumptionCost>
                        <v2gci_t:startValue>10000</v2gci_t:startValue>
                        <v2gci_t:Cost>
                           <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
                           <v2gci_t:amount>95</v2gci_t:amount>
                           <v2gci_t:amountMultiplier>0</v2gci_t:amountMultiplier>
                        </v2gci_t:Cost>
                     </v2gci_t:ConsumptionCost>
                     <v2gci_t:ConsumptionCost>
                        <v2gci_t:startValue>20000</v2gci_t:startValue>
                        <v2gci_t:Cost>
                           <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
                           <v2gci_t:amount>100</v2gci_t:amount>
                           <v2gci_t:amountMultiplier>0</v2gci_t:amountMultiplier>
                        </v2gci_t:Cost>
```

```
                    </v2gci_t:ConsumptionCost>
                  </v2gci_t:SalesTariffEntry>
                </v2gci_t:SalesTariff>
              </v2gci_t:SAScheduleTuple>
            </v2gci_t:SAScheduleList>
            <v2gci_t:AC_EVSEChargeParameter>
              <v2gci_t:AC_EVSEStatus>
                <v2gci_t:PowerSwitchClosed>false</v2gci_t:PowerSwitchClosed>
                <v2gci_t:RCD>false</v2gci_t:RCD>
                <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
                <v2gci_t:IndicationTargetTime>0</v2gci_t:IndicationTargetTime>
              </v2gci_t:AC_EVSEStatus>
              <v2gci_t:EVSEMaxVoltage>
                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                <v2gci_t:Value>230</v2gci_t:Value>
              </v2gci_t:EVSEMaxVoltage>
              <v2gci_t:EVSEMaxCurrent>
                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                <v2gci_t:Value>50</v2gci_t:Value>
              </v2gci_t:EVSEMaxCurrent>
              <v2gci_t:EVSEMinCurrent>
                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                <v2gci_t:Value>0</v2gci_t:Value>
              </v2gci_t:EVSEMinCurrent>
            </v2gci_t:AC_EVSEChargeParameter>
          </v2gci_b:ChargeParameterDiscoveryRes>
        </v2gci_d:Body>
```

V2G message example 18 – ChargeParameterDiscovery Response example for “Consumption”-based SalesTariff with constant value for GridSchedule

D.3.5 Multiple SalesTariffs with different Demand Limits in GridSchedule

```
<v2gci_d:Body>
  <v2gci_b:ChargeParameterDiscoveryRes>
    <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
    <v2gci_t:SAScheduleList>
      <v2gci_t:SAScheduleTuple>
        <v2gci_t:SAScheduleTupleID>24512</v2gci_t:SAScheduleTupleID>
        <v2gci_t:PMaxSchedule>
          <v2gci_t:PMaxScheduleID>23451</v2gci_t:PMaxScheduleID>
          <v2gci_t:PMaxScheduleEntry>
            <v2gci_t:RelativeTimeInterval>
              <v2gci_t:start>0</v2gci_t:start>
              <v2gci_t:duration>3600</v2gci_t:duration>
            </v2gci_t:RelativeTimeInterval>
            <v2gci_t:PMax>22000</v2gci_t:PMax>
          </v2gci_t:PMaxScheduleEntry>
        </v2gci_t:PMaxSchedule>
        <v2gci_t:SalesTariff Id="ID001"> <!-- used by XML Signature Framework for signature assignment in msg header -->
          <v2gci_t:SalesTariffID>10001</v2gci_t:SalesTariffID>
          <v2gci_t:SalesTariffDescription>SalesTariffDescription1</v2gci_t:SalesTariffDescription>
          <v2gci_t:NumEPriceLevels>2</v2gci_t:NumEPriceLevels>
          <v2gci_t:SalesTariffEntry>
            <v2gci_t:RelativeTimeInterval>
              <v2gci_t:start>0</v2gci_t:start>
              <v2gci_t:duration>3600</v2gci_t:duration>
            </v2gci_t:RelativeTimeInterval>
            <v2gci_t:EPriceLevel>1</v2gci_t:EPriceLevel>
          </v2gci_t:SalesTariffEntry>
        </v2gci_t:SalesTariff>
      </v2gci_t:SAScheduleTuple>
```

```
            <v2gci_t:SAScheduleTuple>
               <v2gci_t:SAScheduleTupleID>24513</v2gci_t:SAScheduleTupleID>
               <v2gci_t:PMaxSchedule>
                  <v2gci_t:PMaxScheduleID>23452</v2gci_t:PMaxScheduleID>
                  <v2gci_t:PMaxScheduleEntry>
                     <v2gci_t:RelativeTimeInterval>
                        <v2gci_t:start>0</v2gci_t:start>
                        <v2gci_t:duration>3600</v2gci_t:duration>
                     </v2gci_t:RelativeTimeInterval>
                     <v2gci_t:PMax>11000</v2gci_t:PMax>
                  </v2gci_t:PMaxScheduleEntry>
               </v2gci_t:PMaxSchedule>
               <v2gci_t:SalesTariff Id="ID002"> <!-- used by XML Signature Framework for signature assignment in msg header -->
                  <v2gci_t:SalesTariffID>10002</v2gci_t:SalesTariffID>
                  <v2gci_t:SalesTariffDescription>SalesTariffDescription1</v2gci_t:SalesTariffDescription>
                  <v2gci_t:NumEPriceLevels>2</v2gci_t:NumEPriceLevels>
                  <v2gci_t:SalesTariffEntry>
                     <v2gci_t:RelativeTimeInterval>
                        <v2gci_t:start>0</v2gci_t:start>
                        <v2gci_t:duration>3600</v2gci_t:duration>
                     </v2gci_t:RelativeTimeInterval>
                     <v2gci_t:EPriceLevel>2</v2gci_t:EPriceLevel>
                  </v2gci_t:SalesTariffEntry>
               </v2gci_t:SalesTariff>
            </v2gci_t:SAScheduleTuple>
         </v2gci_t:SAScheduleList>
         <v2gci_t:AC_EVSEChargeParameter>
            <v2gci_t:AC_EVSEStatus>
               <v2gci_t:PowerSwitchClosed>false</v2gci_t:PowerSwitchClosed>
               <v2gci_t:RCD>false</v2gci_t:RCD>
               <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
               <v2gci_t:IndicationTargetTime>0</v2gci_t:IndicationTargetTime>
            </v2gci_t:AC_EVSEStatus>
            <v2gci_t:EVSEMaxVoltage>
               <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
               <v2gci_t:Value>230</v2gci_t:Value>
            </v2gci_t:EVSEMaxVoltage>
            <v2gci_t:EVSEMaxCurrent>
               <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
               <v2gci_t:Value>50</v2gci_t:Value>
            </v2gci_t:EVSEMaxCurrent>
            <v2gci_t:EVSEMinCurrent>
               <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
               <v2gci_t:Value>0</v2gci_t:Value>
            </v2gci_t:EVSEMinCurrent>
         </v2gci_t:AC_EVSEChargeParameter>
      </v2gci_b:ChargeParameterDiscoveryRes>
   </v2gci_d:Body>
```

V2G message example 19 – Charge Parameter Discovery Response example for Multiple SalesTariffs with different Demand Limits in GridSchedule

Annex E
(informative)

Application of certificates

E.1 General

This Annex provides an overview about the application of certificates and the requirements, which need to be fullfiled covered by the certificate concept implemented in this standard.It als explain the certificate parameters like validity, size, chain length and others.

Informative note:

In the charge protocol the following types of certificates are used:

— V2G Root Certificates: These are globally valid (top level) root certificates of the PKI (Public Key Infrastructure). They are used to check the authenticity of the following two types of certificates.

— Contract Certificate: This kind of certificate is used in the “Plug and Charge” use case (i.e. contract-based charging) to check whether a contract exists for the vehicle: If such a contract exists a contract certificate is stored in the vehicle and transmitted to the EVSE at the beginning of the charging session. If the contract certificate is authentic and valid, charging can be started. Contract certificates are derived from the V2G root certificates mentioned above.

— SECC Certificate: This kind of certificate is used by the vehicle to check whether the communication really happens with the EVSE and is not a man-in-the-middle attack. These SECC certificates are also derived from the V2G root certificates mentioned above.

— OEM Root Certificate: Each OEM may create a (top level) root certificate and distribute it to the secondary actors. Such an OEM root certificate can be used to check the authenticity of an OEM provisioning certificate (described below). The root certificate of an OEM is not part of the global PKI; i.e., it is not derived from the the V2G root certificates mentioned above.

— OEM Provisioning Certificate: This kind of certificate is individual (i.e. different) for each vehicle (installed e.g. at vehicle production) and used to identify a vehicle at the beginning of the provisioning process (see also Annex J for Certificate Provisioning). After successful vehicle identification, it is checked whether a contract exists for this vehicle. If this is the case, a contract certificate is installed automatically into the vehicle (EVCC) via the charge protocol.

In the following we explain the general requirements and restrictions caused by the OEMs (refer to clause E.1) and the secondary actors (refer to clause E.2) with respect to these kinds of certificates. Using this starting point, we argue why the decisions made in this document are necessary or at least meaningful (refer to clause E.3). Finally, we give a visual overview of the resulting certificate structure and usage (refer to clause E.4).

E.2 Requirements of the OEM

An OEM typically has the following general requirements that result from the fact that control units (here the SECC) must not become very expensive and manual treatment of a control unit (e.g. in a workshop) causes much effort and has to be avoided.

R1 Installing a certificate into a vehicle is only simple at vehicle production. Later, installation actions result in much effort in a workshop. To enable certificate installation only at vehicle production, a certificate must be “static”. That means, the certificate must have a very long validity. Since vehicles may be used

for 20 years or longer, the certificate must have an even longer validity. Such static certificates for instance may be the root certificates (of the PKI) that are stored in the vehicle.

R2 Unfortunately, static certificates cannot be used for all purposes: The validity of a contract certificate (used for Plug and Charge) typically is only as long as the validity of the contract. Furthermore, the contract may not exist at vehicle production time and, therefore, the contract certificate then still does not exist. It must be possible to install “non-static” certificates into a vehicle in an acceptable manner: Ideally, the certificate installation happens automatically via the charge protocol. If this is not possible for any reason, the certificate may be sent from the secondary actor to the customer as a file and has to be installed into the vehicle by using an online connection or the diagnosis interface (in a workshop). Thereby, format transformations have to be avoided to reduce costs and guarantee compatibility with all secondary actors. Therefore a standardized file format is required for certificate files (especially for contract certificates because they cannot be of static manner).

R3 Control units with much (persistent) storage are expensive, especially if the storage has to allow a large number of write cycles. In order to reduce the amount of storage (e.g. flash), memory required for certificates has to be kept small. This results in the following sub-requirements: (R3a) The size of a single certificate has to be small. (R3b) Whenever the EVCC has to store a whole certificate chain, the chain length has to be short. (R3c) Whenever it is necessary to store multiple certificates of the same type (e.g. multiple root certificates) in the EVCC, the number of these certificates has to be small.

E.3 Requirements of the Secondary Actors

A secondary actor resp. the organization that manages its PKI (Public Key Infrastructure) typically has the general requirements listed below. They result from the fact that the organizational overhead to manage a PKI has to be kept small. This overhead increases if multiple companies or organizations have to coordinate their actions or when task cannot be distributed to multiple organizations because of the certificate structure defined in this standard.

R4 In order to be able to distribute a common root certificate (that is for instance installed in each vehicle), the secondary actors have to build a group that uses a common (i.e. single) root certificate. All certificates created by these secondary actors are derived (indirectly) from this root certificate (i.e. the certificates are signed with the private key that belongs to this root certificate). It is difficult to achieve that all secondary actors (world-wide) work together in only one group. Probably, multiple groups are required for secondary actors of different continents or of different kind (e.g. operators of EVSEs, utilities, mobility providers). If many different groups are built it becomes easier to organize these groups (each with its own root certificate).

R5 A central organization of each group creates its root certificate and the corresponding (very secret) private key. It is not realistic that this organization creates (and signs) all certificates that are required by all secondary actors (e.g. all contract certificates of all customers). Instead, each secondary actor must have the possibility to create its certificates itself. For this purpose, it requires an “intermediate certificate” that is signed by the root certificate and used to sign the (leaf) certificates created by this secondary actor. That means, a chain of certificates is required. Furthermore, it is not realistic that the central (root) organization directly signs the “intermediate certificates” of all secondary actors. Instead, “intermediate organizations” are required (e.g. one for each country). This results in a certificate chain with 2 “intermediate certificate”. If more “intermediate certificates” (i.e. levels for intermediate organizations) are used it becomes easier to manage the cooperation between the different organizations.

R6 Communication between a SECC and a SA IT system results in communication costs. Furthermore, such communication delays the start of the charging procedure and reduces the availability of the EVSE (since communication may fail). Therefore, such communication should be avoided when possible (from the perspective of a secondary actor). Best, it shall be possible that an EVSE stays offline during a whole charging procedure.

NOTE The requirement R6 respects communication costs and delays before start charging. However, there may be other priorities of the participants as well. Furthermore, there may exist scenarios where no relevant communication costs are created in fact (e.g. if the EVSE possesses a LAN or WLAN connection to the internet resp. its backend system). For example in the use case [C2] of *Part 1* fo of this standard, the EVCC communicates directly with the Secondary Actor

using a direct TLS channel. If it is intended that this communication link exists permanently during charging, requirement R6 is not applicable. As another example, the SECC may offer an internet connection or a TCP/IP connection to an arbitrary server as (value added) service to the vehicle. This connection may exist the whole time during charging (or even longer) and results in communication traffic as well. In both examples, the implementation of the SECC has to handle the communication traffic appropriately.

E.4 Decisions

Based on the requirements presented in clause E.2 and E.3 we now present and explain the decisions made in this standard. Since some of the requirements result in conflictive goals, some of the decisions are a compromise between the requirements of the OEMs and the requirements of the secondary actors.

D1 Size of a single certificate: Because of requirement R3a, certificate size must be limited. [V2G2-010] defines that the size of a certificate in DER encoded form shall be not bigger than 800 Bytes. This can be reached by the issuing secondary actor if no irrelevant information is included in the certificate (e.g. no address of the issuer).

D2 Length of certificate chains: Because of requirement R3b complete certificates incl. their chains have to be small. Requirement R5, however, demands long chains since they are easier to manage. As a compromise, [V2G2-009] defines that the path length is limited to 3 (i.e. chains with 3 certificates below the root certificate). That means that there may exist 2 “intermediate certificates” between the root certificate and the leaf certificate. This compromise is realistic, because now each group (cf. R1) is able to create an arbitrary number of “intermediate organizations” (cf. Figure E.1) and sign the “intermediate certificates” with the (top level) root private key. Each “intermediate organization” can create “company certificates” for their secondary actors which may be used by the secondary actor itself to create leaf certificates (which represent for instance contract certificates that are distributed to the customers).

D3 Number of root certificates: All existing root certificates have to be stored in each vehicle. Because of requirement R3c, a small number of root certificates is preferred. Requirement R4, however, demands a large number of root certificates. As a compromise requirement [V2G2-008] defines that at least one root certificate is required, but the note related to it recommends a minimum of 5 certificates. This number (5) corresponds to the number of continents. In principle, it should be possible that the secondary actors of each continent build one common group, because this has been also achieved for EURO5/EU5: Here, one single trust center with one single root certificate was established for Europe.

D4 Validity of root certificates: Because of requirement R1 the (static) root certificates (installed already at vehicle production time) must have an almost infinite validity. Respecting the lifetime of a vehicle, [V2G2-012] requests for any point in time there shall be a root certificate available which is valid for at least 35 years. In order to avoid the necessity to create root certificates very often (manageability of certificates, cf. R4 and R5), [V2G2-011] requests that the validity period of a newly created root certificate shall be 40 years.

D5 Validity of OEM provisioning certificates: OEM provisioning certificates are static certificates as well. For the same reasons as discussed in D4, it is requested that new OEM provisioning certificates shall have a validity period of 30 years.

D6 Installation of contract certificates: As described in R1, the best solution for the OEM would be to install all certificates at vehicle production. Because of R6, it is not possible to use the OEM provisioning certificate directly for (contract-based) charging since this would prevent offline EVSEs. The reason is that without communication to the backend system, it is not possible to verify whether a contract exists for a vehicle that only transmits a static OEM provisioning certificate (installed already at vehicle production independent of the existence of a charge contract).

a. Therefore, contract certificates have to be installed into the vehicle and exchanged from time to time. To avoid costs for the customer, OEM, and secondary actor this has to happen automatically via the charge protocol. [V2G2-235] and [V2G2-237] request that the SECC has to support the messages Certificate Installation Request / Response. [V2G2-228] and [V2G2-231] requests that the SECC has to support the messages Certificate Update Request / Response. With these messages, installation of the first contract certificate resp. installation of further contract certificates (if the contract certificate installed in the vehicle will expire soon) becomes possible.

b. In all cases where it is not possible to perform certificate installation automatically, the effort for manual contract certificate installation has to be minimized. As described in requirement R2, this can be achieved by using a standardized file format when shipping a contract certificate as file to the customer. The file format is defined by [V2G2-648].

D7 Validity of contract certificates: Contract certificates should not be exchanged very often in a vehicle in order to avoid unnecessary write cycles for the storage of the EVCC (cf. requirement R3). To avoid that contract certificates are replaced extremely often (e.g. at each charging session) [V2G2-104] requests that the minimum lifetime of such a certificate shall be 4 weeks (unless the contract lifetime is shorter). For the case that the contract certificate is shipped to the customer and has to be installed manually in a workshop (cf. requirement R2 and decision D6b) its lifetime has to be much longer to avoid inacceptable effort and costs for the customer. Then, the validity of the contract certificate should be as long as the lifetime of the contract.

D8 Validity period of SECC certificates: Mechanism to revoke SECC certificates are not required but instead, it is required to be short term certificates, with a validity period less than 4 weeks.

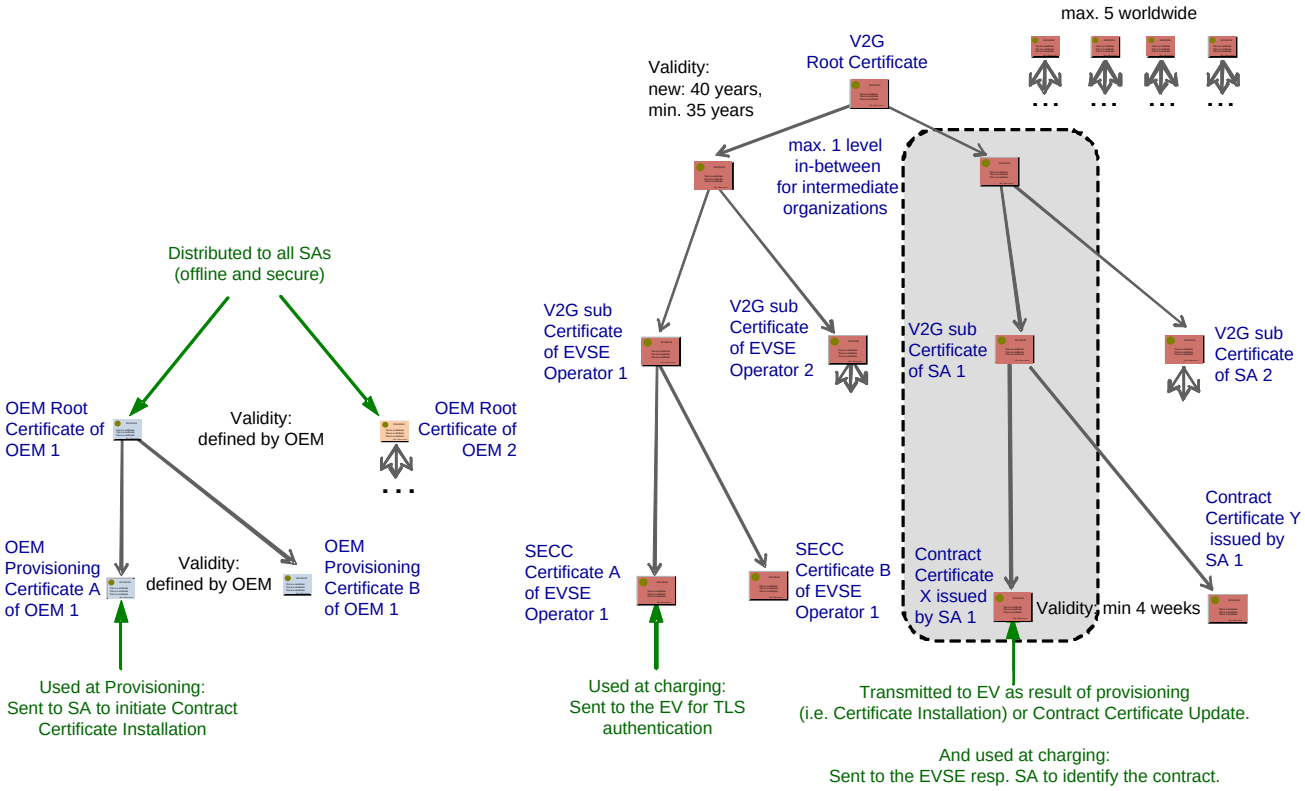
E.5 Overview of the resulting Certificate Structure

Figure E.1 provides a visual overview of the resulting certificate structure and relevant validity periods.

As you can see, the OEM provisioning certificates (depicted in blue) are independent from the PKI of the secondary actors below the (global) root certificates (depicted in red). The root certificate of an OEM provisioning certificate is created by the OEM itself. Therefore, there is no need to have a (longer) certificate chain. (For an explanation of the usage of the OEM root certificate and the OEM provisioning certificate refer to Annex J).

The certificate chain below the V2G root certificates has a length of max. 3 (refer to decision D2 in subclause E.4); i.e. including the root certificate 4 certificates are involved. (The leaf certificate of this chain is used for signature only. As specified, for encryption a Diffie-Hellman approach is used.)

The certificate of an SECC has the same structure as a contract certificate; i.e. a certificate chain with a length of 3. This chain is transmitted to the EVCC to enable an authenticity check of the SECC before a TLS connection is established (cf. above: in order to avoid man-in-the-middle attacks).



= a Certificate

= Contract Certificate incl. Certificate Chain

Figure E.1 — Overview Certificate Structure and required Validity Periods

Annex G provides an example of simplified certificate management in trusted environments.

Annex F
(informative)

Security appliances and their associated certificates

Table F.1 — Security use cases and their associated certificates

Security Appliances	Comment	Credentials EV	Credentials EVSE	Credentials Secondary Actor	Algorithm
Simplified Certificate Management in Trusted Environment Please refer also to Annex G	DC/AC Charging with transport security	Private Operator Root Certificate	SECC certificate (certified by PrivateOperator Root Certificate) with corresponding private key	none	SHA256 AES128 ECC NIST FIPS PUB 180-3 For TLS Handshake only
DC/AC Charging with EIM transport security, payment at EVSE	TLS Authentication of EVSE by EV, using EVCC Cert Confidential Data Exchange between EV and EVSE (Service Discovery, Power Delivery, Metering) Integrity of Data between EV and EVSE (Service Discovery, Power Delivery, Metering) (AES used for Stream / Session Encryption, ECDH used for KeyExchange)	V2G Root Certificates	SECC certificate with corresponding private key, and itscertificate chain to all V2G Root Certificate OCSP response regarding SECC certificate		SHA256 AES128 ECC NIST FIPS PUB 180-3 For TLS

Security Appliances	Comment	Credentials EV	Credentials EVSE	Credentials Secondary Actor	Algorithm
DC/AC Charging with Plug & Charge DC/AC Charging with transport security, an contracts based payment	**All of TLS security plus:** Authentication of EV against EVSE of backend based on Contract Signer Key / Cert / Signing Integrity of metering data based on digital signature with Contract Signer Key	**All of TLS security plus:** Contract Certificate with corresponding signing key.	**All of TLS security plus:**	V2G Root Certificates For Meter receipts	
AC Charging with Plug & Charge with Meter Status Receipt DC/AC Charging with transport security, an contracts based payment,with Meter Status Receipt	**All of PnC plus:** Integrity of tariff data based on digital signature with Secondary Actor in key chain of abvailable V2G Root)	**All of PnC plus:**	**All of PnC plus:**	Public Key Certificate for Secondary Actor and corresponding signing key	
AC Charging with Plug & Charge with Meter Status Receipt, Contract Key updates and Provisioning DC/AC Charging with transport security, an contracts based payment & Contract Certificate Update and Provisioning	**All of PnC& Meter Status Receipt plus:** Integrity of Contract Keys / Certs based on digital signature of Secondary Actor Confidentiality of private key of new Contract Certificates are based on encryption by Secondary Actor (AES used for Data Encryption, ECDH used for KeyExchange)	**All of PnC& Meter Status Receipt plus:** OEM privisioning Certificate and corresponding signing key.	**All of PnC& Meter Status Receipt plus:**	OEM Root Certificate Public Key Certificate for Secondary Actor and corresponding signing key	

Annex G
(informative)

Simplified Certificate Management in Trusted Environment

G.1 Overview (Motivation)

In addition to the public charge stations (that were in mind in the major part of this document), the EVSE can also be a private wall-box. A private wall-box is located in a private resp. small-company environment (e.g. private parking garage, garage or parking lot of a company with its own EV fleet). In order to keep the production and operation costs of a private wall-box low, the following requirements are mandatory:

— Since a private wall-box does not have the possibility to communicate, to a backend server for example, it must be possible that it always stays offline. This is feasible because the transmitted energy in a private environment is already paid with the regular house resp. company (corresponding meter). Therefore, it is not necessary to transmit any (signed) bills with the charged energy to a secondary actor (in fact, no secondary actor is involved in this scenario at all).

— For the same reasons, it cannot be accepted that service actions are performed periodically by humans with that kind of wall-box; e.g., to install data into the wall-box.

Since it is not possible to install data into this wall-box (automatically or manually) after its production time, a problem with SECC certificates arises: In order to avoid CRLs in the vehicle, the SECC certificates are valid for 4 weeks (refer to subclause 7.7.3.4). As explained, it is not possible, however, to install a new certificate (chain) into the SECC every 4 weeks. But without a valid SECC certificate a vehicle (EVCC) does not communicate with the SECC. This is because [V2G2-069] requests that the EVCC checks the validity of the SECC certificate.

But even in a private environment, the communication between SECC and EVCC is required; e.g. to enable the transmission of charge profiles (which may contain data with the preferred charging time, which is at night from 0:30-6:00 when lots of energy is available). Therefore, as described below a special solution for private environments is required.

G.2 Solution for private environments

For each local environment, an individual Private Operator Root Certificate (including a private key) is created (e.g. by the manufacturer of the wall-box). Different Private Operator Root Certificates have to be used, for different wall-boxes (even if the wall boxes are manufacturered by one supplier only), which are located in different local environments. A SECC certificate (signed with a Private Operator Root Certificate of this private environment) is installed into all wall-boxes that need to be used in this private environment (see figure below). These wall-boxes are delivered to the customer, together with the corresponding Private Operator Root Certificate. This Private Operator Root Certificate has to be installed into each vehicle that shall be able to charge in this private environment.

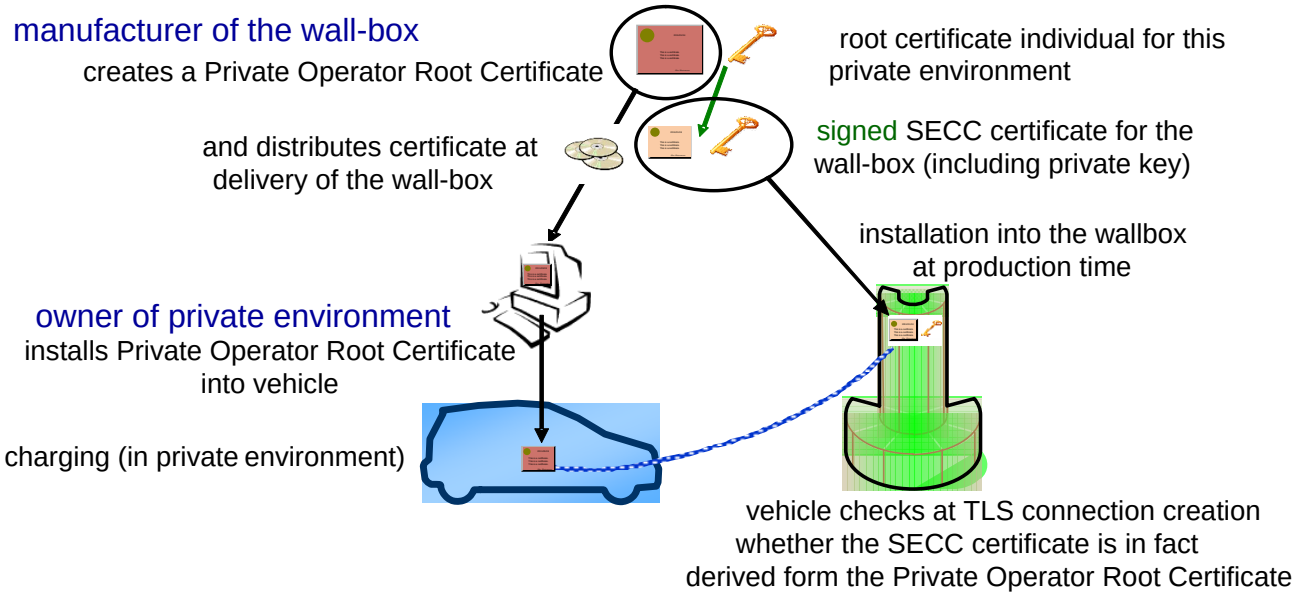


Figure G.1 — Charging in private environments

In order to realize this solution, the vehicle must be able to store at least one Private Operator Root Certificate (it requires one Private Operator Root Certificate for each private environment that it wants to use):

Each EVCC should be able to store Private Operator Root Certificates. They are needed, in addition to the 5 V2G Root certificates, recommended by NOTE 1 in subclause 7.3.3 as well as handled by the EVCC in the same way when establishing a charging session.

In order to avoid the exchange of certificates in the wall-box, their SECC certificates for TLS requires a special validity:

The validity of a SECC certificate installed in a private wall-box and derived from a Private Operator Root Certificate can be as long as defined by the Private Operator. This means, that the requirements that SECC certificates, are valid for 4 weeks, are not applied in the private wall-box certificates.

This almost arbitrary validity does not result in a problem for the security of other vehicles: The SECC certificate is only accepted by vehicles, which have the corresponding Private Operator Root Certificate installed; i.e., the vehicles of this private environment. Therefore, a compromised SECC certificate results only in a problem for the vehicles that belongs to this individual private environment. It is the responsibility of the Private Operator, of this private environment, to ensure that the SECC certificates cannot be stolen from the wall-boxes as well as to perform appropriate actions for the case that a leaf certificate was stolen despite (refer to subclause G.2.3).

G.2.1 Installation of a Private Root Certificate into a Vehicle

The owner (resp. manager) of the local environment gets the Private Operator Root Certificate that belongs to the corresponding wall-boxes (e.g. from their producer). This Private Operator Root Certificate has to be installed into all vehicles that shall be able to charge at this private environment. (Then, it is ensured that the vehicle can check the SECC certificate of the wall-box and that it will perform the TLS handshake.) The corresponding installation procedure is out of scope of this document. It has to be performed by using any communication channel to the vehicle the OEM offers for this purpose. This communication channels may be for instance an online connection to a vehicle web page, a diagnosis interface of the vehicle used at a workshop, an USB interface contained in the vehicle, etc.

G.2.2 Charging in a Private Environment

From the perspective of the vehicle, charging in a private environment is exactly the same as in a public environment. This is important since the vehicle does not know where it is currently located. In a private environment the wall-box transmits its (private) SECC certificate to the EVCC. The EVCC checks (as always)

whether the certificate is still valid and whether it is derived (perhaps indirectly via a certificate chain) from one of the root certificates stored in the vehicle. Thereby, V2G Root Certificates as well as Private Operator Root Certificates are respected.

The EVCC checks only whether the SECC certificate is currently valid. Therefore, the procedure is the same as for public charging.

G.2.3 Compromised Certificate of a Wall-box

A SECC certificate of a private wall-box may become compromised; e.g., when it was stolen. Each vehicle that belongs to this private environment (i.e. possessing the corresponding Private Operator Root Certificate) can be attacked with this stolen leaf certificate by a man in the middle attack (all vehicles that do not belong to this private environment are still save). This attack is possible at each EVSE, even outside of the private environment. Therefore, in that case, it is the responsibility of the owner of the private environment to delete or substitute the Private Operator Root Certificate in all vehicles of the private environment. This can be done by any communication channel mentioned in G.2.1. After completion, no attacks are possible any more. To enable charging in this private environment again, substitution of the Private Operator Root Certificates in the vehicles is not sufficient. In addition, the SECC certificates of the (private) wall-boxes have to be substituted as well (by new SECC certificates derived from the new Private Operator Root Certificate installed in the vehicles).

As the description of the process shows, two aspects are crucial in this scenario:

a) It must be possible that the SECC certificates in the wall-boxes are secured safely respectively that their theft is detected reliably.

b) The effort that results in case of a theft from exchanging the Private Operator Root Certificates of all vehicles and all wall-boxes of this environment must be acceptable. Therefore, the described solution is only suitable for closed and small environments; i.e. a small number of vehicles and a small number of EVSEs in a private environment or an environment of a small organizational unit.

Annex H
(informative)

Certificate profiles

Table H.1 — Certificate profiles

ISO 15118-2 Certificate Profiles			OEM Root Certificate	OEM provisioning certificate	V2G Root Certificate	sub CA Certificate	Private Operator Root Certificate	SECC Certificate	Contract Certificate	Certificates for Secondary Actor (Mobility Operator, OEM)
tbsCertificate										
	Version		2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)
	SerialNumber		Integer	Integer	Integer	Integer	Integer	Integer	Integer	Integer
	Signature		OID - TBD	OID - TBD	OID - TBD	OID - TBD	OID - TBD	OID - TBD	OID - TBD	OID - TBD
Issuer										
	Country		(x)	(x)	(x)	(x)	(x)	x	x	x
	State		-	-	-	-	-	-	-	-
	Local		-	-	-	-	-	-	-	-
	Organization		x	x	x	x	x	x	x	x
	Organization Unit		(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)
	Common Name		x	x	x	x	x	x	x	x
	E-Mail Address		-	-	-	-	-	-	-	-
	Domain Component		-	-	-	-	-	-	-	-

ISO 15118-2 Certificate Profiles			OEM Root Certificate	OEM provisioning certificate	V2G Root Certificate	sub CA Certificate	Private Operator Root Certificate	SECC Certificate	Contract Certificate	Certificates for Secondary Actor (Mobility Operator, OEM)
Validity			up to OEM	up to OEM	40 years	max. 40 years	up to Private Operator	max. 4 weeks	min(4 weeks or lenth of Contract) max 2 years	up to Secondary Actor
Subject										
	Country		(x)	-	(x)	x	(x)	x	(x)	(x)
	State		-	-	-	-	-	-	-	-
	Local		-	-	-	-	-	-	-	-
	Organization		x	x	x	x	x	x	x	x
	Organization Unit		(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)
	Common Name		x	VIN (17)	x	x	x	x	Contract-ID	x
	E-Mail Address		-	-	-	-	-	-	-	-
	Domain Component		-	-	-	-	-	-	-	-
SubjectPublicKeyInfo										
	Public Key		x	x	x	x	x	x	x	x
	Cryptographic Algorithm		id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey
	Parameters		ECParameters (namedCurve)	ECParameters (namedCurve)	ECParameters (namedCurve)	ECParameters (namedCurve)	ECParameters (namedCurve)	ECParameters (namedCurve)	ECParameters (namedCurve)	ECParameters (namedCurve)
Extensions										
	AuthorityKeyIdentifier		(x) / nc	(x) / nc	(x) / nc	(x) / nc	(x) / nc	(x) / nc	(x) / nc	(x) / nc

ISO 15118-2 Certificate Profiles		OEM Root Certificate	OEM provisioning certificate	V2G Root Certificate	sub CA Certificate	Private Operator Root Certificate	SECC Certificate	Contract Certificate	Certificates for Secondary Actor (Mobility Operator, OEM)
SubjectKeyIdentifier		(x) / nc	(x) / nc	(x) / nc	(x) / nc	(x) / nc	(x) / nc	(x) / nc	(x) / nc
KeyUsage		c	c	c	c	c	c	c	c
	digitalSignature						x		
	nonRepudiation (contentCommitment)		x					x	x
	keyEncipherment							x	
	dataEncipherment								
	keyAgreement		x				x		
	keyCertSign	x		x	x	x			
	cRLSign	(x)		(x)	(x)	(x)			
	encipherOnly								
	decipherOnly								
ExtendedKeyUsage							(x) / c	x	
CertificatePolicies		(x) / nc		(x) / nc		(x) / nc			
BasicConstraints		c	c	c	c	c	c	c	c
	CA	true	false	true	true	true	false	false	false
	PathLength	-	-	-	0	-	-	-	-
CRLDistributionPoints			(x) / nc		(x) / nc		(x) / nc	(x) / nc	(x) / nc

ISO 15118-2 Certificate Profiles			OEM Root Certificate	OEM provisioning certificate	V2G Root Certificate	sub CA Certificate	Private Operator Root Certificate	SECC Certificate	Contract Certificate	Certificates for Secondary Actor (Mobility Operator, OEM)
	Authority Information Access (OCSP)			(x) / nc id-ad-ocsp / location of the OCSP responder		(x) / nc id-ad-ocsp / location of the OCSP responder		(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder
Signature Algorithm										
	Cryptographic Algorithm		ecdsa_with_sha256	ecdsa_with_sha256	ecdsa_with_sha256	ecdsa_with_sha256	ecdsa_with_sha256	ecdsa_with_sha256	ecdsa_with_sha256	ecdsa_with_sha256
	Parameters		-	-	-	-	-	-	-	-
Signature Value			Octect-String	Octect-String	Octect-String	Octect-String	Octect-String	Octect-String	Octect-String	Octect-String

Legend:

x – a ‘x’ means that field is to be used

(x) – a ‘x’ in braces means optional

c – means this extension is critical, see [RFC 3280]. If a implementation recognices that a “critical” extension is present, but the implementation can not interpret the extension, the implementation has to reject the certificate

nc - means this extension is non-critical, see [RFC 3280]. If a implementation recognices that a “non-critical” extension is present, but the implementation can not interpret the extension, the extension can be ignored.

Therefore all optional fields should also be “non-critical” (nc)

quote from RFC 3280: “Each extension in a certificate is designated as either critical or non-critical. A certificate using system MUST reject the certificate if it encounters a critical extension it does not recognize; however, a non-critical extension MAY be ignored if it is not recognized.”

OCSP

Online Certificate Status Protocol (OCSP), defined in [RFC 2560] is a protocol used for obtaining the revocation status of an X.509 certificate. OCSP is as an alternative to certificate revocation lists (CRL), OCSP is an online service. Meaning that a backend infrastructure has to support OCSP services in order to be able to use that service. As access to OCSP services can not be guranteed during charging, the usage of OCSP can only be recommended but not be mandatory.

Within a x.509 certificate, the id-ad-ocsp OID is used within the Authority Information Access as accessMethod if revocation information for the certificate containing this extension is available using the Online Certificate Status Protocol (OCSP). When id-ad-ocsp appears as accessMethod, the accessLocation field is the location of the OCSP responder, using the conventions defined in [RFC 2560].

Annex I
(normative)
Using Contract Certificates for XML encryption

I.1 Overview

I.1.1 Scope

In the ISO 15118, ECDSA is the algorithm used for signing certificates (and corresponding private keys). This algorithm is a variant of the DSA algorithm using elliptic curves and enables digital signatures, but may not be used for public key encryption. Hence, a mechanism is needed allowing for the encrypted transport of information with minimal roundtrips. The goal is to have the option to encrypt information for the EV and sent this information encrypted (and signed) to the EV within a pair of request and response message.

Having the boundary conditions of ECDSA regarding encryption in mind, sending encrypted information can be achieved using different approaches:

Key negotiation can be achieved by different means. For the use cases in the context of ISO 15118 the Diffie Hellman (DH) key agreement, which is oulined in the following overview, provides appropriate means.

Precondition here is that both entities, the EV and the Secondary Actor(SA) have means to generate and to validate a digital signature using the available ECDSA based key pair (certificate and corresponding private key).

I.1.2 Diffie Hellman Key Agreement

The Diffie Hellman key agreement describes an option to establish a secret shared key between two (or more) entities using asymmetric cryptographic algorithms. Both entities generate messages with DH parameter sets and exchange these messages. Both parties can then calculate the DH secret, which can be used directly or from which session keys may be derived by a key derivation function. The security of the Diffie Hellman key agreement relies on the discrete logarithm problem, which would need to be solved by a potential attacker. The general proceeding is depicted in Figure G.2.

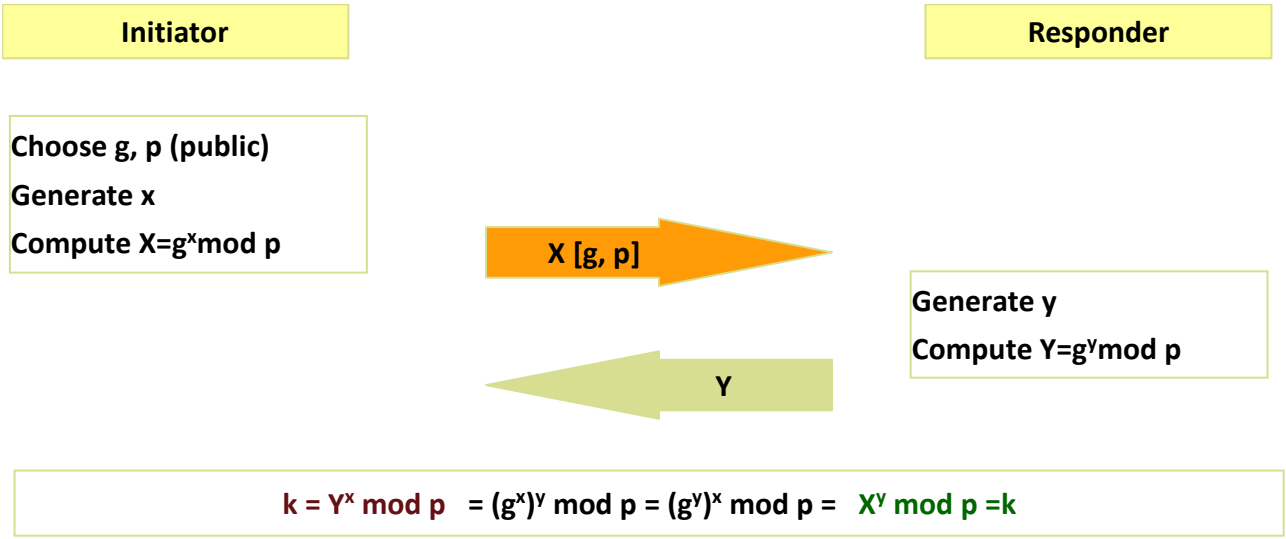


Figure G.2 — Diffie Hellman Key Agreement

Diffie Hellman has two public parameters *g* and *p*, which need to be available to both communication peers. The other parameters are generated during the proceeding of the DH key agreement. Note that the private part of the DH parameters (*x* and *y* in the figure above) must be kept secret.

The problem of the Diffie Hellman key agreement is that in its pure form, it is susceptible against Man-in-the-Middle attacks. Therefore, the parameter exchange is typically signed using the private keys of each entity allowing a source authentication and integrity check at the receiving end.

I.2 Proposal

I.2.1 General

Diffie Hellman Key agreement can also be performed using elliptic curves (ECDH). Since Contract certificates uses ECDSA for ISO 15118, Diffie Hellman key agreement using elliptic curves (ECDH) shall also be supported by the participating entities (EV, SA).

Using Diffie Hellman there are two ways to achieve the exchange of a secret key:

a) Usage of ephemeral DH parameter on EV and SA. This may be achieved by generating a fresh DH parameter set on the EV side. This DH parameter set is then signed by the EV's private key and can be sent to the SA through EVSE. The SA in turn also generates fresh DH parameter, signs the parameter and sends them back to the EV

b) Usage of static DH parameter options:

   1) Usage of static DH parameter from the EV side and ephemeral DH parameter from SA

   2) Usage of static DH parameter from the EV side and from SA

In ISO 15118, there are two options, namely either a) or b1).The first option, a) is a typical ECDH scheme, however, requires extra computation on the EV. The second option b1) can be implemented using ECDSA keys as the static DH parameter set, which are stored in Contract Certificates and OEM Provisioning Certificates. Either the Contract Certificate and/or OEM Provisioning Certificate is available at SA (either E-mobility operator or OEM.). This enables the SA to pre-calculate responses with, e.g., encrypted private key informationn, for all contracts. If an EVSE then connects to SA with the contract based information of the connected EV, the private key information can be provided without any delay. Using ephemeral keys from the SA ensures that there are different encryption keys used for each session. Although there is no formal

analysis to prove the security of this combination of ECDSA and ECDH, no reports have been provided to threat such use.

The next subsection describes the application.

I.2.2 How to choose the option

In ISO 15118, some messages involve encryption, for example the Certificate Update Messages. Request for CertificateUpdate will be generated by EVCC and signed using current Contract Certificate, and sent to SECC. The SECC forwards this request to Secondary Actor (presumably E-mobility Operator) and receives an updated Contract Certificate for the EVCC together with its corresponding encrypted private key. Similarly, Request for Certificate Install will be generated by EVCC and signed using OEM provisioning Certificate, and sent to SECC. The SECC forwards this request to the Secondary Actor and receives a new Contract Certificate for the EVCC together with corresponding encrypted private key. Which keys to use to encrypt the corresponding private keys are provided by ECDH scheme.

It is always EV to chose which option, either ephemeral ECDH or static ECDH to generate encryption keys. If EV wishes to use ephemeral DH, that is the option a), EV generates fresh DH prameters and send it as DHparams in its request. EV keeps the private data corresponding the sent DH parameter.

The secondary actor, receiving this request forwarded by SECC, freshly computes its DH parameters and corresponding private data. The encryption key to encrypt the response message will be generated using DHparams forwardeded by SECC and the private data that was generated with its own DH parameters. The secondary actor encrypts the message to encrypt (namely private key corresponding to either updated or newly installed Contract Certificate) using a key generated by ephemeral ECDH and sends back to the EVCC through SECC, together with DH parameters SA generated.

EVCC receives DH parameters from SA, derives key from this DH parameters and private data that he had stored in sending his DH parameters, and decrypts the encrypted message and receives the private key corresponding to the new Contract Certificate.

If EV decides to use static DH, that is the option b1), elemet DHparams will be blank.

If no DHparams are forwarded, the backend draws out parameters in the Contract Certificate of the corresponding EVCC and replace that as DH parameters that should had been given from EVCC. The procedure will then follow similar to the option a).

I.2.3 Using static DH keys in the Certificate

The certificates to be applied in ISO 15118 are X.509 certificates. An X.509 certificate is defined in IETF RFC 5280 as follows:

```
Certificate          ::=  SEQUENCE  {
     tbsCertificate            TBSCertificate,
     signatureAlgorithm        AlgorithmIdentifier,
     signatureValue            BIT STRING  }

TBSCertificate ::=  SEQUENCE  {
     version                [0]    Version must be v3,
     serialNumber              CertificateSerialNumber,
     signature                 AlgorithmIdentifier,
     issuer                    Name,
     validity                  Validity,
     subject                   Name,
     subjectPublicKeyInfo      SubjectPublicKeyInfo,
     issuerUniqueID         [1]    IMPLICIT UniqueIdentifier OPTIONAL,
                               -- If present, version MUST be v2 or v3
     subjectUniqueID        [2]    IMPLICIT UniqueIdentifier OPTIONAL,
```

```
                                -- If present, version MUST be v2 or v3
      extensions            [3]    EXPLICIT Extensions OPTIONAL
                                -- If present, version MUST be v3
      }

   SubjectPublicKeyInfo  ::=  SEQUENCE  {
     algorithm            AlgorithmIdentifier,
     subjectPublicKey     BIT STRING  }
```

NOTE 1 There is no structural difference between ECDH and ECDSA keys.

NOTE 2 A certificate issuer may use X.509 v3 `keyUsage` and `extendedKeyUsage` extensions to restrict the use of an ECC public key to certain computations.

IETF RFC 5280 defines the usage of the public key in an extension:

```
id-ce-keyUsage OBJECT IDENTIFIER ::=  { id-ce 15 }

KeyUsage ::= BIT STRING {
     digitalSignature        (0),
     nonRepudiation          (1), -- recent editions of X.509 have
                          -- renamed this bit to contentCommitment
     keyEncipherment         (2),
     dataEncipherment        (3),
     keyAgreement            (4),
     keyCertSign             (5),
     cRLSign                 (6),
     encipherOnly            (7),
     decipherOnly            (8) }
```

For the application in ISO 15118 the usage `digitalSignature` and `keyAgreement` is mandatory.

Since we are using secp256 curve, the parameters for the curve are already agreed based on the OID of the secp256 curve, which is:

```
secp256r1 OBJECT IDENTIFIER ::= {
   ansi-X9-62 curves(3) prime(1) 7 }
```

The D-H parameters are used for generating a Diffie-Hellman key pair and must be given to the EVCC. The generated parameters are not sensitive and need not be kept secret. They have to be provided in a Base64 encoded format. The maximum length needed (with a key size of 1024) is 288 bytes.

The D-H key (shared secret) is always less than the used parameters, which are for 15118 256 Bits. It is provided in a PKCS#12 format.wih file extension p12 or pfx. For an XML-transfer it has to be base64 encoded first.

Annex J
(informative)

Use of OEM Provisioning Certificates

The difficulty with the handling of contract certificates (as defined in this document) for the OEM is to bring such a certificate into a vehicle. This becomes necessary in many situation as e.g. vehicle hand-over to customer, energy contract conclusion, changing the mobility provider, exchanging the component containing the contract certificate at vehicle repair, expired contract certificate, etc.

Using a diagnosis tool to write a file containing a contract certificate (that the customer received from the mobility provider at contract conclusion) into the vehicle is unfortunate, since such a manual procedure in a workshop causes high costs. Other solutions like installation via a customer web page also cause effort, are error-prone, and require the existence of a communication channel into the vehicle. Therefore, an automatic procedure for the installation of such a contract certificate is required: certificate provisioning. This procedure is supported by the charge protocol by offering the messages Certificate Installation Request / Response. These messages transmit a contract certificate from a secondary actor (e.g. mobility provider) to the vehicle for installation and are secure by using encrypted communication. In order to enable certificate provisioning, activities which happen outside the charge protocol are required additionally. These activities are sketched in Figure J.1 and described in the following:

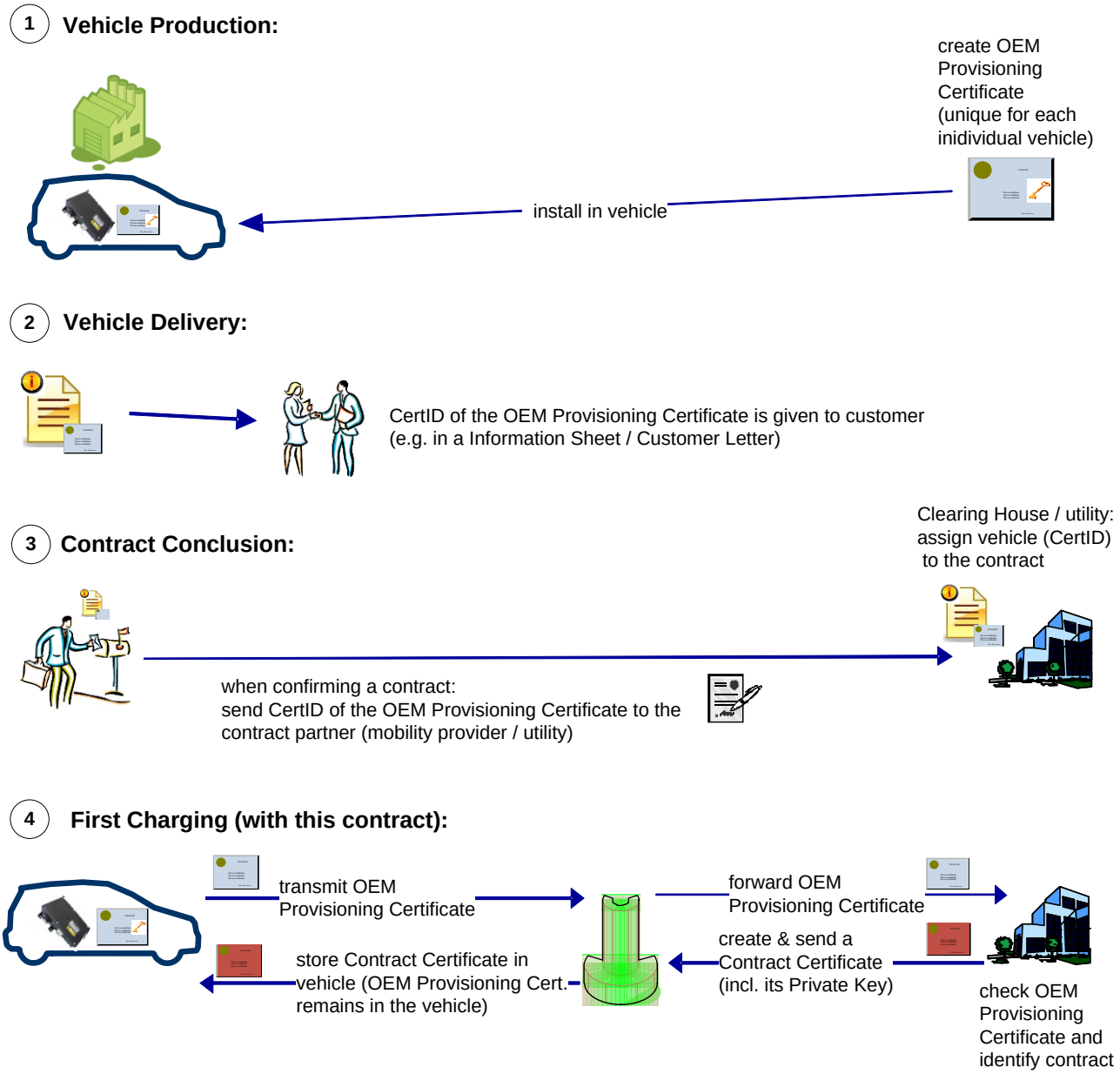


Figure J.1 — Activities required for OEM Certificate Provisioning

At vehicle production time, a unique OEM provisioning certificate and the corresponding private key are installed in each vehicle (refer to Figure J.1 – Step 1). Each OEM provisioning certificate contains a unique ID (for short: CertID) and is signed with the root certificate of the issuing OEM. (This root certificate was distributed to the secondary actors and clearing houses prior to that process step. It is not necessary that this OEM root certificate is derived from one of the 5 globally valid root certificates mentioned earlier in this document.)

At or before vehicle hand-over, the customer is informed about his/her CertID (refer to Figure J.1 – Step 2) e.g. by distributing information sheets, integrating the CertID in the vehicle documentation, offering online access to the CertID, etc.

At conclusion of an energy contract (refer to Figure J.1 – Step 3) the customer forwards the CertID to the contract partner (mobility provider, utility, etc). The contract partner assigns (in its IT system) the contract information with this CertID. Furthermore, the contract partner creates a contract certificate and assigns the contract certificate with this CertID as well. The information about the existence of a contract for this CertID is

forwarded to the Clearing House (of this country) – ideally together with the contract certificate (in order to avoid delays later on during communication in Step 4).

At first charging of the vehicle (or whenever it does not possess a valid contract certificate), the vehicle transmits its OEM provisioning certificate to the SECC using the message Certificate Installation Request (refer to Figure J.1 – Step 4). The SECC forwards this message resp. certificate to the clearing house (or all known resp. its own mobility provider(s) if there does not exist a central clearing house in this country). The clearing house checks (i) whether the OEM provisioning certificate is authentic (i.e. valid) by using the root certificate of the OEM and (ii) whether there exists a contract that belongs to this OEM provisioning certificate; i.e., that is assigned to the CertID contained in the OEM provisioning certificate. If the corresponding contract certificate was not send to the clearing house in Step 3 the clearing house requests a contract certificate from the mobility provider which concluded in the contract. Then, the contract certificate (i.e. both certificate chains, both private keys, and the ContractID) are sent to the charge station and then to the vehicle by using the message Certificate Installation Response (see below). Since this message contains secret data (private keys) it is encrypted with the public key contained in the OEM provisioning certificate of this vehicle.

This certificate provisioning procedure is used in all cases mentioned above; e.g. conclusion of a new contract, changing the mobility provider, etc. – or regarded from the perspective of the vehicle if no contract certificate is available in the vehicle, the contract certificate is expired, or it was revoked by the SECC; i.e. whenever the vehicle does not possess a valid contract certificate. Therefore, supporting this automatic certificate installation procedure avoids manual effort in all situations.

If the contract certificate is not expired yet, but it will expire soon, the messages Certificate Update Request / Response may be used to refresh the contract certificate. This avoids manual effort for exchanging an expired contract certificate (what would happen some time later) as well. The Certificate Update messages may even be used by vehicles which do not posses a provisioning certificate at all or do not realize the provisioning procedure. However, if the contract certificate is expired in fact (e.g. because the vehicle was not charged for a longer time – at least at a public / online charge station) the provisioning procedure described above is required despite to refresh the contract certificate (or manual effort becomes necessary).

When in case of a vehicle repair, the component containing the contract certificate must be exchanged, the new component may contain a different OEM provisioning certificate with a different CertID. The reason is that the private key belonging to the original OEM provisioning certificate perhaps was stored solely inside the defect component and is lost (since it can not or must not be read from it). In such a case, the new CertID of the vehicle has to be transmitted to the clearing house resp. contract partner. It has to assign the new CertID with the contract data and at next charging a (new) contract certificate is transmitted to the vehicle using the regular certificate provisioning procedure. That means, also after a vehicle repair, no manual treatment of contract certificates becomes necessary. However, the data (CertID) has to be refreshed (triggered manually by the customer resp. workshop or automatically by realizing communication of the involved IT systems).

Annex K
(informative)

Summary of Requirements

This Annex list in Table K.1 the requirements defined in this standard to allow for easy requirement management. Also it used for maintaining a change history concering the requirements in this standard.

Table K.1 — Requirement summary

Subclause including requirement(s)	List of requirement numbers
5.3 Usage of RFC references	[V2G2-001][V2G2-002][V2G2-003]
7.3.2 Certificate and key management	[V2G2-004] [V2G2-005][V2G2-006][V2G2-007]
7.3.3 Number of root certificates and root validity, certificate depth and size	[V2G2-008][V2G2-009][V2G2-010][V2G2-011][V2G2-012]
7.4 V2G communication states	[V2G2-014][V2G2-015][V2G2-016][V2G2-017][V2G2-018][V2G2-019][V2G2-020][V2G2-021][V2G2-022][V2G2-023][V2G2-024][V2G2-025][V2G2-026][V2G2-027][V2G2-028][V2G2-029][V2G2-030][V2G2-031][V2G2-032][V2G2-033][V2G2-034][V2G2-645][V2G2-646][V2G2-647]
7.5 Data Link Layer	[V2G2-035][V2G2-036]
7.6.2.1 IPv6	[V2G2-037][V2G2-038][V2G2-039][V2G2-040][V2G2-041][V2G2-042]
7.6.2.2 Dynamic Host Control Protocol (DHCPV6)	[V2G2-043][V2G2-044]
7.6.2.3 Neighbor Discovery (ND)	[V2G2-045][V2G2-046]
7.6.2.4 Internet Control Message Protocol (ICMP)	[V2G2-047][V2G2-048][V2G2-049]
7.6.3.2 Stateless auto address configuration (SLAAC)	[V2G2-050][V2G2-051][V2G2-052][V2G2-053]
7.6.3.3 Address selection	[V2G2-054]
7.7.1.2 Applicable RFCs, limitations and protocol parameter settings (TCP)	[V2G2-055]
7.7.1.3 TCP Performance and checksum requirements	[V2G2-057][V2G2-058][V2G2-059][V2G2-060][V2G2-061][V2G2-062][V2G2-063][V2G2-064]
7.7.2.2 Applicable RFC, limitations and protocol parameter settings (UDP)	[V2G2-065]
7.7.3.2 Applicable RFCs (TLS)	[V2G2-067]
7.7.3.3 Transport Layer Security Usage	[V2G2-068][V2G2-069][V2G2-070][V2G2-649][V2G2-650][V2G2-651]
7.7.3.4 Transport Layer Security Credentials and Cipher Suites	[V2G2-071][V2G2-072][V2G2-602][V2G2-603]
7.8.2 Supported ports (V2GTP)	[V2G2-073][V2G2-074][V2G2-075][V2G2-076][V2G2-077][V2G2-078][V2G2-079][V2G2-080][V2G2-081]
7.8.3.1 PDU Structure (V2GTP)	[V2G2-082][V2G2-083][V2G2-084][V2G2-085][V2G2-086][V2G2-087][V2G2-088]
7.8.3.2 PDU Header Processing (V2GTP)	[V2G2-089][V2G2-090][V2G2-091][V2G2-092][V2G2-093][V2G2-094][V2G2-095][V2G2-096]
7.9.1.1 Overview (XML/EXI)	[V2G2-097]
7.9.1.3 EXI Settings for application layer messages	[V2G2-098][V2G2-099][V2G2-100][V2G2-101][V2G2-102][V2G2-600]

Subclause including requirement(s)	List of requirement numbers
7.9.2.1 Application layer credentials and cipher suites (XML Security)	[V2G2-103][V2G2-104]
7.9.2.2 Contract Certificates as XML signature credentials	[V2G2-108]
7.9.2.3 XML Encryption Credentials	[V2G2-114][V2G2-115][V2G2-116]
7.9.2.4 XML Security specifics for 'PnC' Message Set(s)	[V2G2-117] [V2G2-119][V2G2-121][V2G2-122][V2G2-652][V2G2-653][V2G2-654][V2G2-655]
7.10.1.1 General Information (SDP)	[V2G2-123]
7.10.1.2 Supported ports (SDP)	[V2G2-124][V2G2-125][V2G2-126]
7.10.1.3 Protocol Data Unit	[V2G2-127][V2G2-128][V2G2-129][V2G2-130][V2G2-131][V2G2-132][V2G2-133][V2G2-134]
7.10.1.4 SECC Discovery Request Message	[V2G2-135][V2G2-136][V2G2-137][V2G2-138][V2G2-139][V2G2-140][V2G2-141][V2G2-142][V2G2-622][V2G2-623][V2G2-624]
7.10.1.5 SECC Discovery Response Message	[V2G2-143][V2G2-144][V2G2-145][V2G2-146][V2G2-147][V2G2-148][V2G2-149][V2G2-150][V2G2-151][V2G2-152][V2G2-153][V2G2-154][V2G2-155][V2G2-156]
7.10.1.6 Timing and Error Handling	[V2G2-157][V2G2-158][V2G2-159][V2G2-160][V2G2-161][V2G2-162]
7.10.1.7 Protocol and Security Options Handling	[V2G2-625][V2G2-626][V2G2-627][V2G2-628][V2G2-629]
7.10.1.8 Support and Application of TLS	[V2G2-630][V2G2-631][V2G2-632][V2G2-633][V2G2-634][V2G2-635][V2G2-636][V2G2-637][V2G2-638][V2G2-639][V2G2-640][V2G2-641][V2G2-642][V2G2-643][V2G2-644]
7.10.1.9 SECC Discovery service primitives	[V2G2-163][V2G2-164]
8.2.1 Handshake sequence (Protocol handshake)	[V2G2-165][V2G2-166][V2G2-167][V2G2-168][V2G2-169][V2G2-170][V2G2-171][V2G2-172][V2G2-173][V2G2-174]
8.2.2 Message definition supportedAppProtocolReq and supportedAppProtocolRes	[V2G2-175][V2G2-176]
8.2.3 Semantics decription supportedAppProtocol messages	[V2G2-178]
8.3.2 Message definition (V2G messages)	[V2G2-179][V2G2-180]
8.3.3 Message Header Definition (V2G messages)	[V2G2-181][V2G2-182]
8.3.4 Message Body Definition (V2G messages)	[V2G2-183]
8.4.1.2 Session Setup	[V2G2-184][V2G2-185][V2G2-186][V2G2-187][V2G2-188][V2G2-189][V2G2-190][V2G2-191][V2G2-192]
8.4.1.3 Service Discovery	[V2G2-193][V2G2-194][V2G2-195][V2G2-196]
8.4.1.4 Service Detail	[V2G2-197][V2G2-198][V2G2-199][V2G2-200]
8.4.1.5 Service and Payment Selection	[V2G2-201][V2G2-202][V2G2-203][V2G2-204]
8.4.1.6 Payment Details	[V2G2-205][V2G2-206][V2G2-207][V2G2-208][V2G2-209]
8.4.1.7 Contract Authentication	[V2G2-210][V2G2-211][V2G2-212][V2G2-213]
8.4.1.8 Charge Parameter Discovery	[V2G2-214][V2G2-215][V2G2-216][V2G2-217][V2G2-218][V2G2-219][V2G2-220]
8.4.1.9 Power Delivery	[V2G2-221][V2G2-222][V2G2-223][V2G2-224][V2G2-225][V2G2-226]
8.4.1.10 Certificate Update	[V2G2-227][V2G2-228][V2G2-229][V2G2-230][V2G2-231][V2G2-232][V2G2-233]
8.4.1.11 Certificate Installation	[V2G2-234][V2G2-235][V2G2-236][V2G2-237][V2G2-238][V2G2-648]

Subclause including requirement(s)	List of requirement numbers
8.4.1.12 Session Stop	[V2G2-239][V2G2-240][V2G2-241]
8.4.2.2 Charging Status	[V2G2-242][V2G2-243][V2G2-244]
8.4.2.3 Metering Receipt	[V2G2-245][V2G2-246][V2G2-247][V2G2-248]
8.4.3.2 Cable Check	[V2G2-249][V2G2-250][V2G2-251][V2G2-252]
8.4.3.3 Pre Charg	[V2G2-253][V2G2-254][V2G2-255][V2G2-256]
8.4.3.4 Current Demand	[V2G2-257][V2G2-258][V2G2-259][V2G2-260]
8.4.3.5 Welding Detection	[V2G2-261][V2G2-262][V2G2-263][V2G2-264]
8.5.2.1 ServiceTagType	[V2G2-265][V2G2-266]
8.5.2.2 ServiceTagListType	[V2G2-267][V2G2-268]
8.5.2.3 ServiceType	[V2G2-269][V2G2-270]
8.5.2.4 ServiceChargeType	[V2G2-271][V2G2-272][V2G2-273]
8.5.2.5 CertificateChainType	[V2G2-274][V2G2-275]
8.5.2.6 MeterInfoType	[V2G2-276][V2G2-277]
8.5.2.7 PhysicalValueType	[V2G2-278][V2G2-279]
8.5.2.8 NotificationType	[V2G2-280][V2G2-281]
8.5.2.9 PaymentOptionsType	[V2G2-282][V2G2-283]
8.5.2.10 ChargingProfileType	[V2G2-284][V2G2-285][V2G2-286][V2G2-287] [V2G2-606]
8.5.2.11 ProfileEntryType	[V2G2-288][V2G2-289][V2G2-290][V2G2-291][V2G2-292][V2G2-293] [V2G2-607]
8.5.2.12 SAScheduleListType	[V2G2-294][V2G2-295][V2G2-296][V2G2-297][V2G2-298][V2G2-608]
8.5.2.13 SAScheduleTupleType	[V2G2-299][V2G2-300][V2G2-301] [V2G2-303][V2G2-304][V2G2-305][V2G2-306][V2G2-307][V2G2-308][V2G2-309] [V2G2-609]
8.5.2.14 PMaxScheduleType	[V2G2-310][V2G2-311][V2G2-312] [V2G2-610]
8.5.2.15 PMaxScheduleEntryType	[V2G2-313][V2G2-314][V2G2-315][V2G2-611]
8.5.2.16 SalesTariffType	[V2G2-316][V2G2-317][V2G2-318][V2G2-319][V2G2-320][V2G2-612]
8.5.2.17 SalesTariffEntryType	[V2G2-321][V2G2-322][V2G2-323][V2G2-324][V2G2-325][V2G2-326][V2G2-613]
8.5.2.18 RelativeTimeIntervalType	[V2G2-327][V2G2-328][V2G2-329][V2G2-330][V2G2-331][V2G2-614]
8.5.2.19 ConsumptionCostType	[V2G2-332][V2G2-333][V2G2-334][V2G2-615]
8.5.2.20 CostType	[V2G2-335][V2G2-336][V2G2-337][V2G2-338][V2G2-339][V2G2-340][V2G2-341][V2G2-342] [V2G2-616]
8.5.2.21 ServiceParameterListType	[V2G2-343][V2G2-344]
8.5.2.22 ParameterSetType	[V2G2-345][V2G2-346]
8.5.2.23 ParameterType	[V2G2-347][V2G2-348]
8.5.2.24 SelectedServiceListType	[V2G2-349][V2G2-350]
8.5.2.25 SelectedServiceType	[V2G2-351][V2G2-352]
8.5.2.26 SubCertificatesType	[V2G2-353][V2G2-354][V2G2-656]
8.5.2.27 ListOfRootCertificateIDsType	[V2G2-355][V2G2-356][V2G2-357]
8.5.3.1 AC_EVSEStatusType	[V2G2-358][V2G2-359]

Subclause including requirement(s)	List of requirement numbers
8.5.3.2 AC_EVChargeParameterType	[V2G2-360][V2G2-361]
8.5.3.3 AC_EVSEChargeParameterType	[V2G2-362][V2G2-363]
8.5.4.1 DC_EVSEStatusType	[V2G2-364][V2G2-365][V2G2-366]
8.5.4.2 DC_EVStatusType	[V2G2-367][V2G2-368][V2G2-369]
8.5.4.3 DC_EVChargeParameterType	[V2G2-370][V2G2-371]
8.5.4.4 DC_EVSEChargeParameterType	[V2G2-372][V2G2-373]
8.5.4.5 DC_EVPowerDeliveryParameterType	[V2G2-374][V2G2-375]
8.6.2.1 Overview (Supported Message Set(s))	[V2G2-659][V2G2-660][V2G2-661][V2G2-662][V2G2-663][V2G2-664][V2G2-665][V2G2-666][V2G2-667][V2G2-668]
8.6.2.2 AC (Supported Message Set(s))	[V2G2-376][V2G2-377][V2G2-378][V2G2-379][V2G2-380][V2G2-381][V2G2-382][V2G2-383][V2G2-384][V2G2-385][V2G2-386][V2G2-387][V2G2-388][V2G2-389]
8.6.2.3 DC (Supported Message Set(s))	[V2G2-390][V2G2-391][V2G2-392][V2G2-393][V2G2-394][V2G2-395][V2G2-396][V2G2-397][V2G2-398][V2G2-399][V2G2-400][V2G2-401]
8.6.3.1 Message Sets for AC/DC Charging EIM/PnC	[V2G2-402][V2G2-403][V2G2-404] [V2G2-405]
8.6.3.2 Message Set Metering Receipt	[V2G2-406][V2G2-407][V2G2-408][V2G2-409]
8.6.3.3 Certificate Install	[V2G2-410][V2G2-411]
8.6.3.4 Certificate Update	[V2G2-412][V2G2-413]
8.6.3.5 Message Set Value Added Services	[V2G2-414][V2G2-415]
8.6.3.6 Selection of services	[V2G2-416][V2G2-417][V2G2-418][V2G2-419][V2G2-420][V2G2-421][V2G2-422][V2G2-424][V2G2-425][V2G2-426][V2G2-427][V2G2-428][V2G2-429][V2G2-430][V2G2-431][V2G2-432][V2G2-433]
8.7.2.1 Definitions (Message Timing)	[V2G2-434][V2G2-435]
8.7.2.2 EVCC Timing for Request-Response Message Pairs	[V2G2-436][V2G2-437][V2G2-438][V2G2-439][V2G2-440]
8.7.2.3 SECC Timing for Request-Response Message	[V2G2-441][V2G2-442][V2G2-443][V2G2-444][V2G2-445]
8.7.3.1 Definitions (Session Setup Timing)	[V2G2-605]
8.7.3.2 EVCC Timing for communication session setup	[V2G2-446][V2G2-447][V2G2-448][V2G2-449]
8.7.3.3 EVCC Timing for ready to charge	[V2G2-450][V2G2-451][V2G2-452][V2G2-604][V2G2-681][V2G2-682]
8.8.3.1 Common Requirements (Response Code Handling – Message Sequencing)	[V2G2-455][V2G2-456][V2G2-457][V2G2-458][V2G2-459][V2G2-460][V2G2-461][V2G2-462][V2G2-463][V2G2-464][V2G2-465][V2G2-466][V2G2-467][V2G2-468][V2G2-469][V2G2-470][V2G2-471][V2G2-472][V2G2-473][V2G2-474][V2G2-475][V2G2-476][V2G2-477][V2G2-478][V2G2-479][V2G2-480][V2G2-481]
8.8.3.2 AC Specific Requirements	[V2G2-453][V2G2-454]
8.8.3.3 DC Specific Requirements	[V2G2-669][V2G2-670][V2G2-671]
8.8.4.1 General Requirements (Message Sequencing)	[V2G2-672][V2G2-673][V2G2-674][V2G2-675][V2G2-676][V2G2-677][V2G2-678][V2G2-679][V2G2-680]

Subclause including requirement(s)	List of requirement numbers
8.8.4.2 EVCC (Message Sequencing)	[V2G2-453][V2G2-482][V2G2-483][V2G2-484][V2G2-485][V2G2-486][V2G2-487][V2G2-488][V2G2-489][V2G2-490][V2G2-491][V2G2-492][V2G2-493][V2G2-494][V2G2-495][V2G2-496][V2G2-497][V2G2-498][V2G2-499][V2G2-500][V2G2-501][V2G2-502][V2G2-503][V2G2-504][V2G2-505][V2G2-506][V2G2-507][V2G2-508] [V2G2-509][V2G2-510][V2G2-511][V2G2-512][V2G2-513][V2G2-514][V2G2-515][V2G2-516][V2G2-517][V2G2-518][V2G2-519][V2G2-520][V2G2-521][V2G2-522][V2G2-524][V2G2-525][V2G2-526][V2G2-527][V2G2-528][V2G2-529][V2G2-530][V2G2-531][V2G2-532][V2G2-533][V2G2-534][V2G2-535][V2G2-617][V2G2-618][V2G2-619][V2G2-620][V2G2-657][V2G2-683][V2G2-684][V2G2-685][V2G2-686][V2G2-689]
8.8.4.3 SECC (Message Sequencing)	[V2G2-454][V2G2-536][V2G2-537][V2G2-538][V2G2-539][V2G2-540][V2G2-541][V2G2-542][V2G2-543][V2G2-544][V2G2-545][V2G2-546][V2G2-547][V2G2-548][V2G2-549][V2G2-550][V2G2-551][V2G2-552][V2G2-553][V2G2-554][V2G2-555][V2G2-556][V2G2-557][V2G2-558][V2G2-559][V2G2-560][V2G2-561][V2G2-562][V2G2-563][V2G2-564][V2G2-565][V2G2-566][V2G2-567][V2G2-568][V2G2-569][V2G2-570][V2G2-571][V2G2-572][V2G2-573][V2G2-574][V2G2-575][V2G2-576][V2G2-577][V2G2-578][V2G2-579][V2G2-580][V2G2-581][V2G2-582][V2G2-583][V2G2-584][V2G2-585][V2G2-586][V2G2-587][V2G2-588][V2G2-589][V2G2-590] [V2G2-591][V2G2-592][V2G2-593][V2G2-595][V2G2-596][V2G2-597][V2G2-598][V2G2-599][V2G2-601][V2G2-621][V2G2-658][V2G2-687][V2G2-688]

Bibliography

[1] IETF RFC 5871, IANA Allocation Guidelines for the IPv6 Routing Header (Mai 2010)

[2] ISO 10731, Information technology - Open Systems Interconnection - Basic Reference Model - Conventions for the definition of OSI services

[3] IETF RFC 5220, Problem Statement for Default Address Selection in Multi-Prefix Environments: Operational Issues of RFC 3484 Default Rules (July 2008)

[4] Altova XMLSpy Manual [viewed 2011-01-12], Available from , <http://manual.altova.com/XMLSpy/spyprofessional/index.html?xseditingviews_schview_contmodview.htm>

[5] IETF RFC 5280, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile (May 2008)

[6] Object Identifier (OID) Repository [viewed 2011-01-12], Available from <http://www.oid-info.com/>

[7] W3C EXI Profile, Efficient XML Interchange (EXI) (August 2011)

[8] W3C XML, Extensible Markup Language (XML) 1.0 (Fifth Edition) (November 2008)

[9] W3C XMLSchema 0, XML Schema Part 0: Primer Second Edition (October 2004)

[10] W3C XMLSchema 1, XML Schema Part 1: Structures Second Edition (October 2004)

[11] W3C XMLSchema 2, XML Schema Part 2: Datatypes Second Edition (October 2004)

[12] W3C XMLSig, XML Signature Syntax and Processing (Second Edition) (June 2008)

[13] W3C XMLEnc, XML Encryption Syntax and Processing Version 1.1 (January 2012)

[14] IETF RFC 1421, Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures (February 1993)