INTERNATIONAL STANDARD

ISO 15118-2

First edition
2014-04-01

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

Reference number
ISO 15118-2:2014(E)

COPYRIGHT PROTECTED DOCUMENT

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

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. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information

The committee responsible for this document is ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 15118-2 was developed in conjunction with IEC TC 69, *Electric road vehicles and electric industrial trucks*.

ISO 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 and data link layer requirements*[1]

[1] To be published.

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 completely 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 economical 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 part of ISO 15118 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 15118 is designed to support the energy transfer from an EVSE to an EV. ISO 15118-1 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 International Standard.

The purpose of this part of ISO 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 ISO/IEC FDIS 15118-2:2013(E)

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

Figure 1 — Communication relationship among EVCC, SECC and secondary actor

This part of ISO 15118 defines messages, data model, XML/EXI based data representation format, usage of V2GTP, TLS, TCP and IPv6. In addition, it 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 ISO 15118-3.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 3166-1, *Codes for the representation of names of countries and their subdivisions — Part 1: Country codes*

ISO 15118-1, *Road vehicles — Vehicle to grid communication interface — Part 1: General information and use-case definition*

IEC 61851-1, *Electric vehicle conductive charging system — Part 1: General requirements (Ed 2.0 2010)*

IEC 61851-22, *Electric vehicle conductive charging system - Part 22: AC electric vehicle charging station*

IEC CDV 61851-23, *Electric vehicle conductive charging system - Part 23: D.C. electric vehicle charging station (Ed 1.0 2012)*

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

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

W3C XML Signature Syntax and Processing Version 1.1, - W3C Recommendation (April 2013)

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

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

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

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

IETF RFC 6960, X*.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP (June 2013)*

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 6582, *The NewReno Modification to TCP's Fast Recovery Algorithm (April 2012)*

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 5234, *Augmented BNF for Syntax Specifications: ABNF (January 2008)*

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

IETF RFC 5280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile (May 2008)*

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

IETF RFC 5480, *Elliptic Curve Cryptography Subject Public Key Information (March 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 6961, *The Transport Layer Security (TLS) Multiple Certificate Status Request Extension (June 2013)*

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

NIST FIPS PUB 180-4: Secure Hash Standard (SHS) (March 2012)

NIST Special Publication 800-56A: Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised) (March 2007)

NIST Special Publication 800-38A: Recommendation for Block Cipher Modes of Operation - Methods and Techniques (2001)

3 Terms and definitions

For the purposes of this document, the terms in ISO 15118-1 and the following apply.

3.1
Basic Charging
BC
charging phase during a charging session controlled by IEC 61851-1 only

3.2
charging limits
set of physical constraints (e.g. voltage, current, energy, power) that is negotiated during a V2G Communication Session for a charging session

3.3
Communication Setup Timer
Timer monitoring the time from plug-in until the Session Setup message

3.4
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

3.5
CP State
Control Pilot (Vehicle) State according to IEC 61851-1 signalled on Control Pilot Line

3.6
credentials
anything that provides the basis for confidence, belief, credit, etc.

EXAMPLE Examples include certificates, passwords, user names etc.

3.7
Data Link Setup
setup phase for establishing the data link

Note 1 to entry: Entry Condition: Any valid control pilot signal according to IEC 61851-1; Exit Condition: D-LINK_READY.indication(DLINKSTATUS=LinkEstablished).

3.8
Distinguished Encoding Rules = ASN-1 encoding rule
DER
method for encoding a data object, such as an X.509 certificate, to be digitally signed or to have its signature verified

3.9
global address
IP address with unlimited scope

3.10
High Level Communication Charging
HLC-C
charging phase during a charging session controlled by ISO 15118

3.11
link-local address
IP address with link-only scope that can be used to reach neighbouring interfaces attached to the same link

3.12
Identification Mode
mandatory and optional messages and parameters with respect to charging scenarios using External Identification Means (EIM) and charging scenarios using Plug and Charge (PnC) for identification

Note 1 to entry: An Identification Mode covers a set of similar charging scenarios for a specific identification means.

3.13
(IP) address
IP-layer identifier for an interface or a set of interfaces

3.14
Maximum Transfer Unit
MTU
maximum size (in bytes) of the largest protocol data unit that the Data Link Layer that can be pass onwards

3.15
Message Set
set of mandatory V2G messages and parameters for the EVCC or SECC covering one or multiple use case elements

3.16
Message Timer
Timer monitoring the exchange of a Request-Response-Pair

3.17
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 addresses.

3.18
node
device that implements IPv6

3.19
OEM Provisioning Certificate
certificate issued to the EVCC, so that a Contract Certificate can be securely requested and received from a secondary actor

3.20
Performance Time
non-functional timing requirement defining the time a V2G Entity shall not exceed when executing or processing certain functionality

Note 1 to entry: This is a fixed time value.

3.21
private environment
area with (physical) access limited to a small number of vehicles (EVs), which may be a private parking garage or a garage / parking lot of a company with its own EV fleet, where one or several private wall-box(es) are used instead of public charging stations as EVSE, and where in order to keep the private wall-box simple and cheap in production and operation it is allowed to stay offline permanently, which allows a private wall-box to use leaf certificates with a longer maximum validity than allowed for public charging stations and using a private root certificate which is different to the V2G root certificates and which has to be installed into each EV that is allowed to charge within this specific private environment, resulting in a limited number of EVs belonging to one private environment, the difference to a “trusted environment” being that in a (pure; i.e. not additionally “trusted”) private environment TLS and the corresponding data encryption at connection level is always used, and solely certificate handling is simplified for the private wall-box (EVSE) since it may stay offline permanently, resulting in unrestricted certificate validity periods, shorter certificate chain length, omitting OCSP, and an additional “pairing mode”

3.22
Identification Mode
group of mandatory and optional Message Sets covering a set of similar charging scenarios for a specific identification means

3.23
renegotiation
messaging for updating the agreement on the charging schedule between EV and EVSE during a V2G Communication Session by retransmitting the parameters SASchedule and ChargingProfile

3.24
Request-Response Message Pair
request message and the corresponding response message

3.25
Request-Response Message Sequence
predefined sequence of Request-Response Message Pairs

3.26
SDP Client
V2G Entity that uses the SDP server to get configuration information about the SECC to be able to access the SECC

3.27
SDP Server
V2G Entity providing configuration information for accessing the SECC

3.28
SECC Certificate
certificate issued to SECC either by V2G Root CA or by Sub-CA, which is used in TLS so that the EVCC can verify the authenticity of the SECC

3.29
Sequence Timer
Timer monitoring a Request-Response Message Sequence

3.30
Sub-CA
subordinate certificate authority who issues SECC Certificates and/or Contract Certificates on behalf of the V2G Root CA

Note 1 to entry: The ability of issuing the certificates are delegated from V2G Root CA, and V2G Root CA can revoke the Sub-CA at any time.

3.31
Sub-CA Certificate
certificate issued to Sub-CA

3.32
TCP_DATA
socket/interface for data transfer based on TCP connection

3.33
Timeout
timing requirement defining the time a V2G Entity monitors the communication system for a certain event to occur

Note 1 to entry: If the specified time is exceeded the respective V2G Entity initiates the related error handling. This is a fixed time value.

3.34
Timer
device or piece of software used in an implementation for measuring time.

Note 1 to entry: Depending on the specific use case a timer is used to trigger certain system events as well.

3.35
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), which is something where a person or instance is responsible for, for example (not limited to) a person with its home garage, a car sharing operator or a taxi operator

3.36
V2G Charging Loop
V2G messaging phase for controlling the charging process by ISO 15118

3.37
V2G Communication Session
association of two specific V2G Entities for exchanging V2G messages

3.38
V2G Entity
primary actor participating in the V2G communication using a mandatory or optional transmission protocol defined by ISO 15118-2

3.39
V2G Message
message exchanged on application layer

Note 1 to entry: Refer to Clause 8 Application Layer messages.

3.40
V2G Setup
setup phase for V2G messaging

Note 1 to entry: Entry Condition: D-LINK_READY.indication(DLINKSTATUS=LinkEstablished); Exit Condition: PowerDeliveryReq with ChargeProgress equals Start or Stop.

3.41
V2G Transfer Protocol
communication protocol to transfer V2G messages between two V2GTP entities

3.42
V2GTP Entity
V2G Entity supporting the V2G Transfer Protocol

3.43
V2G Root CA
certificate Authority (CA) who issues Contract Certificates and/or SECC Certificates, or who delegates ability to issue such Certificates to Sub-CA

3.44
V2G Root Certificate
certificate issued to V2G Root CA

4 Symbols and abbreviated terms

For the purposes of this document, the following abbreviated terms 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
EMAID	E-Mobility Account Identifier
EMOCH	E-Mobility Operator Clearing House (see also 15118-1, [12])
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
PHEV	Plug-in Hybrid Electric Vehicle
PKI	Public Key Infrastructure
PLC	Power Line Communication
PnC	Plug and Charge
SA	secondary actor

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
UDP	User Datagram Protocol
XML	Extensible Markup Language

5 Conventions

5.1 Definition of OSI based services

ISO 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 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 15118 set of standards,

— Y represents the document part of the ISO 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 “shall/ shall 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 15118 referenced RFCs, are fully applicable in this standard.

5.4 Notation used for XML schema diagrams

This standard makes 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:

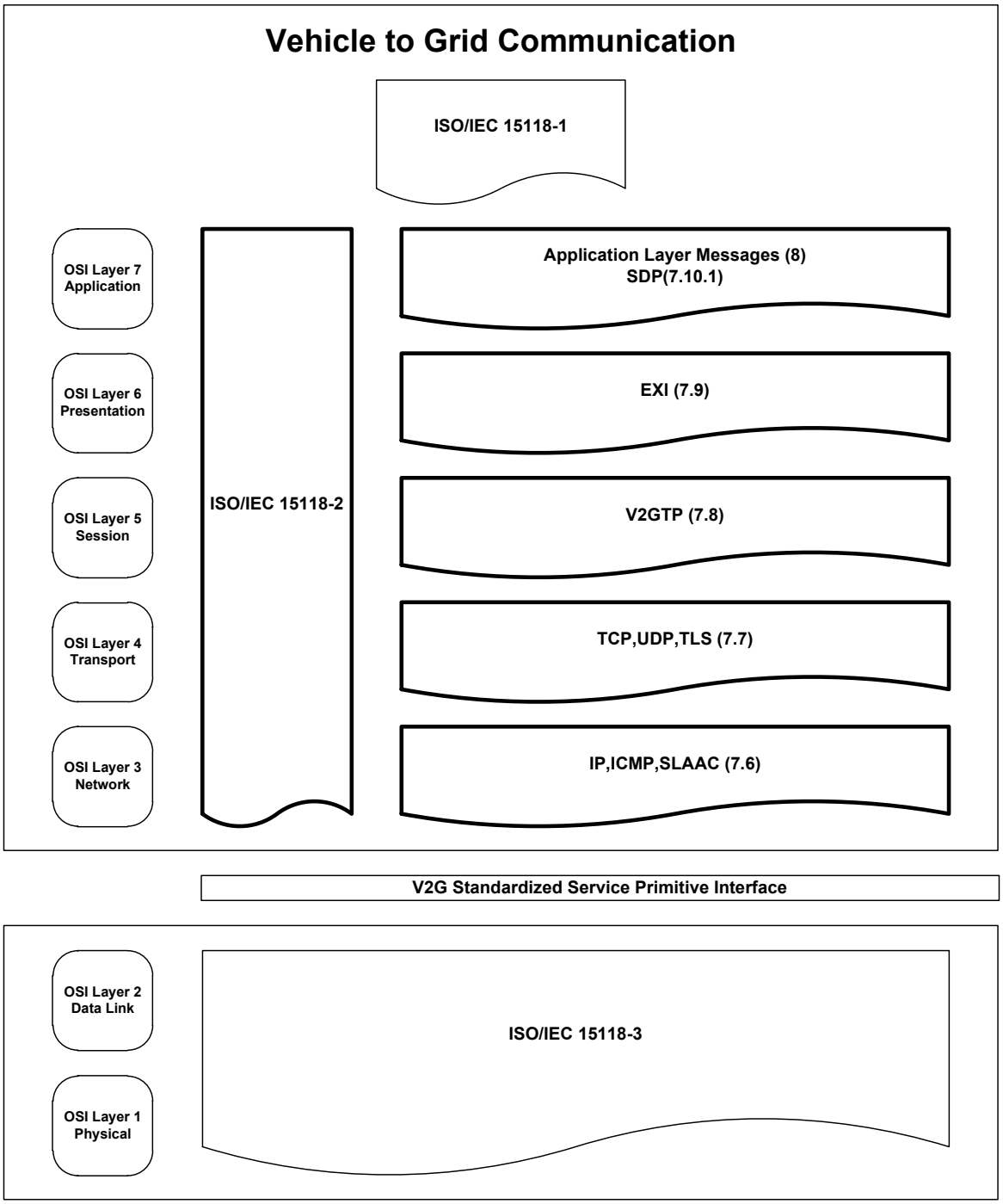
— complex type use capitalized first letters

— simple types use non capitalized first letters

6 Document overview

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

As indicated by the bold framed shapes this Part of ISO 15118 defines requirements applicable to layers 3-7 according to the OSI layered architecture. 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 15118

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

Figure 2 — Vehicle to Grid Communication document overview

7 Basic requirements for V2G communication

7.1 General information

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

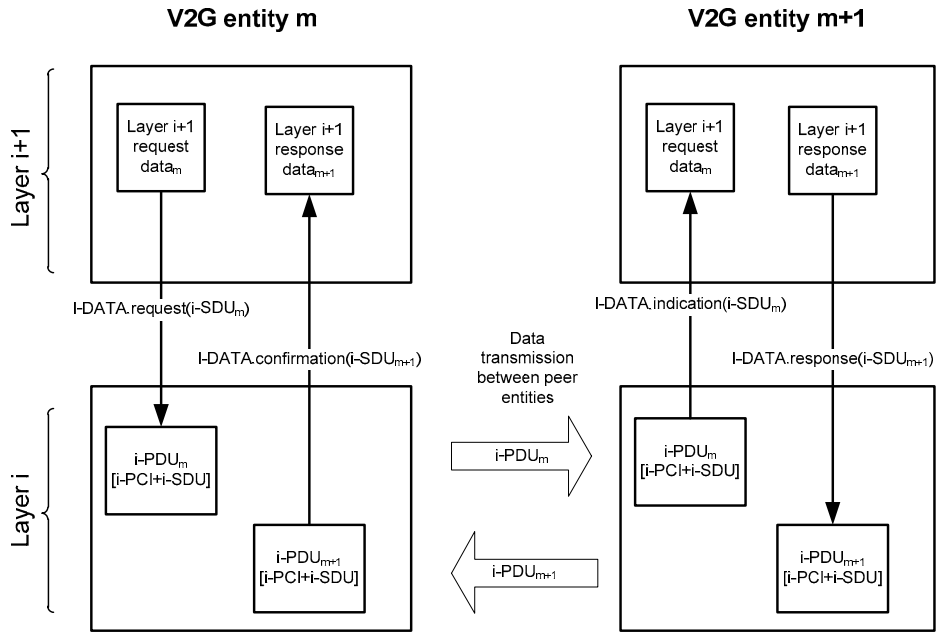
7.2 Service primitive concept of OSI layered architecture

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 use case for a semi-online connection for a Plug and Charge scenario:

In this Plug-and-Charge example, all TCP/IP based communication is protected using a unilaterally 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 support the billing process (refer to 8.4.3.13.1). 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.

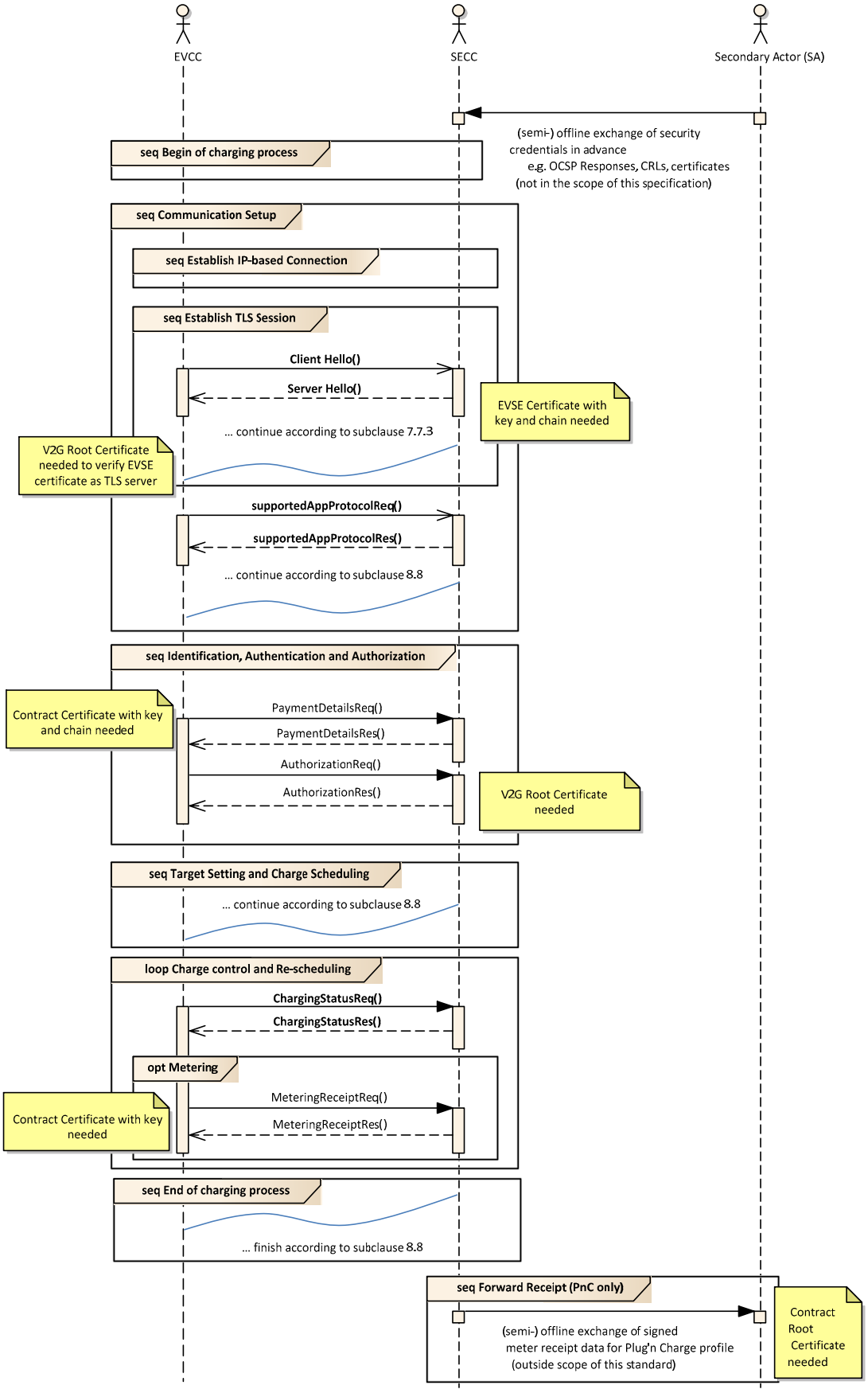


Figure 4 — Example for semi-online communication (1 of 2)

Figure 5 shows an example for an online Plug and Charge use case:

As in the semi-online case, in this Plug-and-Charge example, all TCP/IP based communication is protected using a unilaterally 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 eMAID or the vehicles credentials to be able so sign the tariff information. The EVCC calculates a charging profile (refer to subclause 8.4.3.9.2) 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.

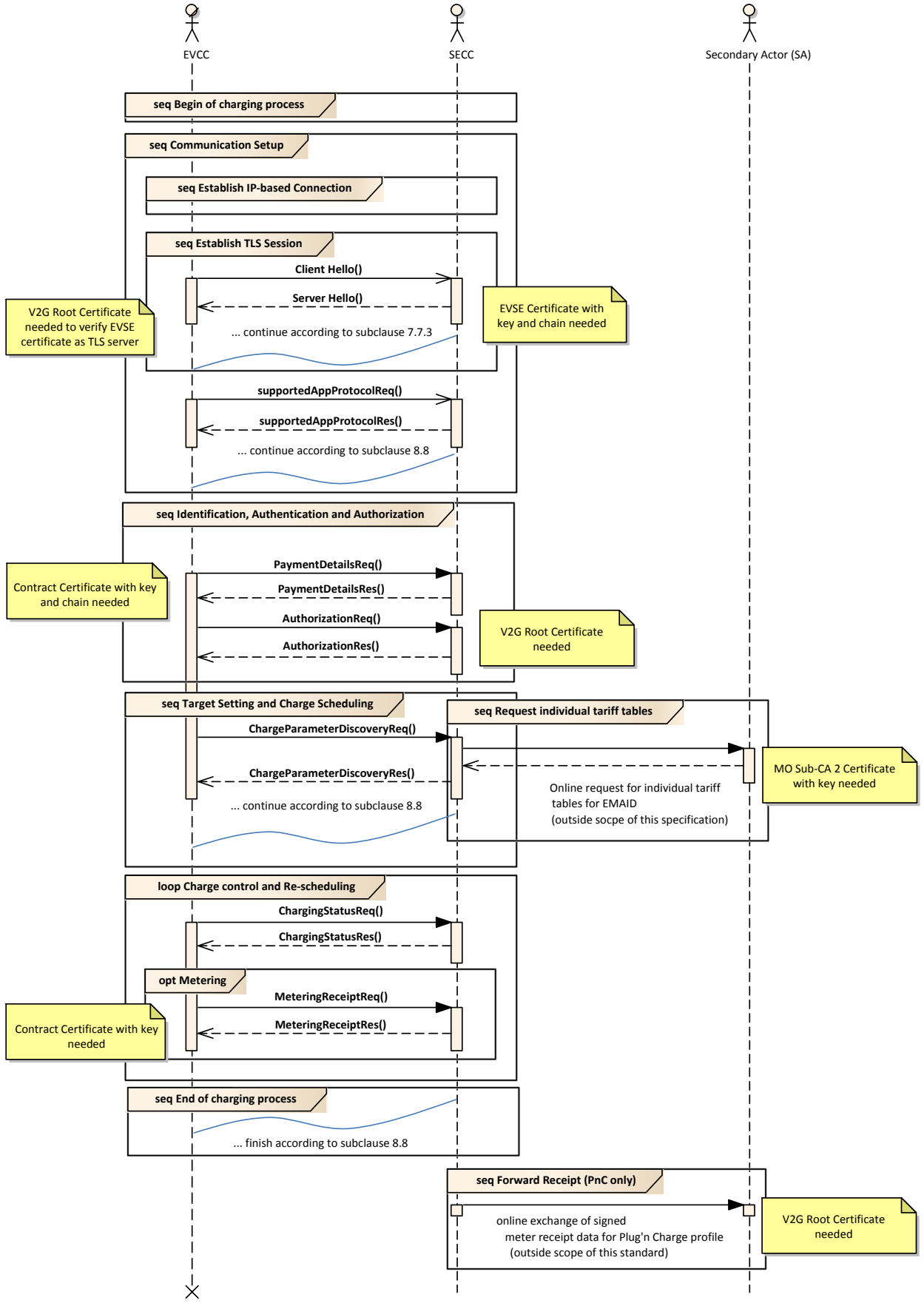


Figure 5 — Example for online communication

There are further use cases where security mechanism are needed on application layer. These are the initial enrolment 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 0. An overview on all needed certificates can be found in Annex F.

7.3.2 Certificate and key management

The call flows shown in subclause 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 at the application layer to authenticate against an SECC and/or secondary actor. There are V2G Root Certificates and possibly Sub-CA Certificates which certify SECC Certificates and Contract Certificates.

Separate from above certificates, there may be OEM Root Certificates and OEM Provisioning Certificates to be used for installing and updating Contract Certificates.

For an overview over all possible certificate profiles, see Annex F.

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

Table 1 shows what fields a X.509v3-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 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 6960 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, refer to: Object Identifier (OID) Repository.

[V2G2-005] Each V2G Entity shall support Hash-operation SHA-256 (signature process) according to NIST FIPS PUB 180-4 (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 uses shall be 256 bit.

[V2G2-885] All certificate validations shall be carried out in conformance with RFC 5280. The EVCC and the SECC may cache certificate validation results during one charging session.

[V2G2-926] The certificate extensions mentioned in Annex F shall be supported. Deviations are specified where necessary.

[V2G2-925] A leaf certificate shall be treated as invalid, if the trust anchor at the end of the chain does not match the specific root certificate required for a certain use, or if the required Domain Component value is not present.

[V2G2-910] The EVCC should implement a mechanism to check the validity periods of certificates and OCSP responses.

[V2G2-886] The EVCC may choose the accuracy of its time source at its own discretion, but it shall respect the chosen accuracy when using the time source. The accuracy should be at least one day.

NOTE 3 This requirement theoretically even allows for precisions of e.g. one year. The implementer should carefully weigh the security consequences of such a decision.

NOTE 4 If a TLS connection is built up successfully, the EVCC might use the time from a EVSETimeStamp field sent from the SECC through said connection for synchronization of its internal clock, however the implementer should carefully weigh the security consequences of such a decision.

[V2G2-887] The SECC shall ensure that at any point in time it has a certificate for itself, which is valid for at least one more hour.

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

[V2G2-008] Each EVCC shall support at least one V2G Root Certificate.

[V2G2-877] Each SECC shall support the storage of at least 10 V2G Root Certificates.

NOTE 1 Due to overlapping validity periods, up to 10 concurrently valid certificates may exist. See V2G2-012

NOTE 2 A number of five (5) V2G Root Certificates is strongly recommended. 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 3 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.

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

NOTE 4 The path length constraint defines the number of non self signed 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. For transmission, all certificates shall be DER encoded.

[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 the next 35 years.

NOTE 5 For explanations of certificate validity, number of root certificates, certificate depth and size refer to Annex E.1.

[V2G2-878] The maximum number of concurrently valid V2G Root Certificates for one Root CA shall be never more than 10.

[V2G2-867] A V2G root shall not issue a leaf certificate.

NOTE 6 Leaf certificates can only be issued by a Sub-CA.

[V2G2-869] No CA or Sub-CA shall issue and sign a certificate which is valid for longer than the validity period of itself as the signing entity.

[V2G2-911] If only one Sub-CA layer is used, i.e. a Sub-CA signed by a Root CA directly signs leaf certificates, the profile of Sub-CA 2 shall apply for that Sub-CA.

The following requirements enable simplified certificate handling within Private Environments. See Annex E.2 for a more detailed explanation.

[V2G2-882] Each EVCC shall be able to store at least one Private Operator Root Certificate which shall be replaceable by the car owner.

[V2G2-927] The EVCC shall consider stored Private Operator Root Certificates as trust anchors for SECC Certificate verification.

[V2G2-883] The validity times of SECC Certificates having a trusted Private Operator Root Certificate as their trust anchor may be chosen by the certificate signer at his own discretion.

[V2G2-868] For a Private Environment, Sub-CAs and OCSP responses are not supported.

NOTE 7 That means, a Private Operator Root Certificate directly signs a leaf certificate.

7.3.4 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 and Message Set EIM charging DC, in both cases 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 PaymentOptionList of the ServiceDiscoveryRes message as defined in subclause 8.4.3.3.3.

[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 standard.

[V2G2-639] If TLS is not applied both sides have to ensure that an 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 3 and Table 4.

Table 3 — TLS implementation / support for EIM Identification Modes

	Other Environment (not trusted)				Trusted Environment [c]			
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	optional	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	refer to Table 4	refer to Table 4	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

[c] See Section 3 for a definition

NOTE 8 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 4 — SDP handshake with AC and DC EIM Identification Mode in trusted environment

EVCC TLS support	SECC TLS support	SDP request by EVCC	SDP response SECC
EVCC has TLS support	SECC has TLS support	EVCC signals TLS	SECC shall respond TLS
		EVCC signals TCP	SECC shall respond TCP
	SECC has no TLS support	EVCC signals TLS	SECC can indicate that it does no support TLS with indicating "0x10 = No transport layer security" in its SDP response. The EV can decide, if it wants to establish TCP without TLS, or otherwise stop the communication setup.
		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 9 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 10 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.

NOTE 11 Any functional safety related risks occurring through overvoltage and overcurrent (accidental or purposeful) need to be addressed by implementing the related electrical safety standards (e.g. IEC 61851 and ISO 17409)

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

[V2G2-643] The EVSE shall support the safety measures described in IEC 61851-1 and IEC 61851-22 to protect against overvoltage and overcurrent when using the AC EIM Identification Mode

[V2G2-644] The EVSE shall support the safety measures described in IEC 61851-1 and IEC CDV 61851-23 to protect against overvoltage and overcurrent when using the DC EIM Identification Mode.

7.4 V2G communication states and data link handling

V2G messaging on application layer requires the establishment of the lower layer protocols to enable V2G message exchange between EVCC and SECC. This document focuses on protocols above the Data-Link layer. The data link layer is described in Part 3 of this standard.

This subclause defines the basic requirements for establishing communication above the data link layer. The standard does not imply any implementation. The figures in this chapter only show the basic sequencing for managing the protocol stack.

The timer , timeout and performance time 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.

The following applies for the EVCC:

[V2G2-014] After data link layer connection is established (D-LINK_READY.indication(DLINKSTATUS=Link established), the EVCC shall initiate the IP 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_CommunicationSetup_Timer is equal or larger than V2G_EVCC_CommunicationSetup_Timeout.

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

[V2G2-019] The EVCC shall start the SDP client 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_CommunicationSetup_Timer is equal or larger than V2G_EVCC_CommunicationSetup_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 establish a TCP connection to the SECC as described in subclause 7.7.1 after the SECC IP address, port and available transport protocol and security options are discovered.

[V2G2-022] Depending on **[V2G2-021]** and **[V2G2-646]**, the EVCC shall attempt to establish a TLS or TCP connection according to 7.7.3.

[V2G2-023] The EVCC shall stop to attempt to establish a TLS or TCP connection when V2G_EVCC_CommunicationSetup_Timer is equal or larger than V2G_EVCC_CommunicationSetup_Timeout.

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

[V2G2-025] The EVCC shall terminate the TLS or TCP connection after stopping the V2G Communication Session.

[V2G2-717] If the EVCC sent the message SessionStopReq with parameter ChargingSession equal to "Terminate" it shall terminate the Data-Link (D-LINK_TERMINATE.request()) after receiving the message SessionStopRes.

[V2G2-718] If the EVCC sent the message SessionStopReq with parameter ChargingSession equal to "Pause" it shall pause the Data-Link (D-LINK_PAUSE.request()) after receiving the message SessionStopRes and continue with **[V2G2-014]**.

[V2G2-719] Whenever the EVCC receives the indication for a missing Data-Link (D-LINK_READY.indication (DLINKSTATUS=No link), the EVCC shall continue with **[V2G2-014]**.

[V2G2-720] If the EVCC identifies any error it shall indicate an Data-Link error (D-LINK_ERROR.request()) and continue with **[V2G2-014]**.

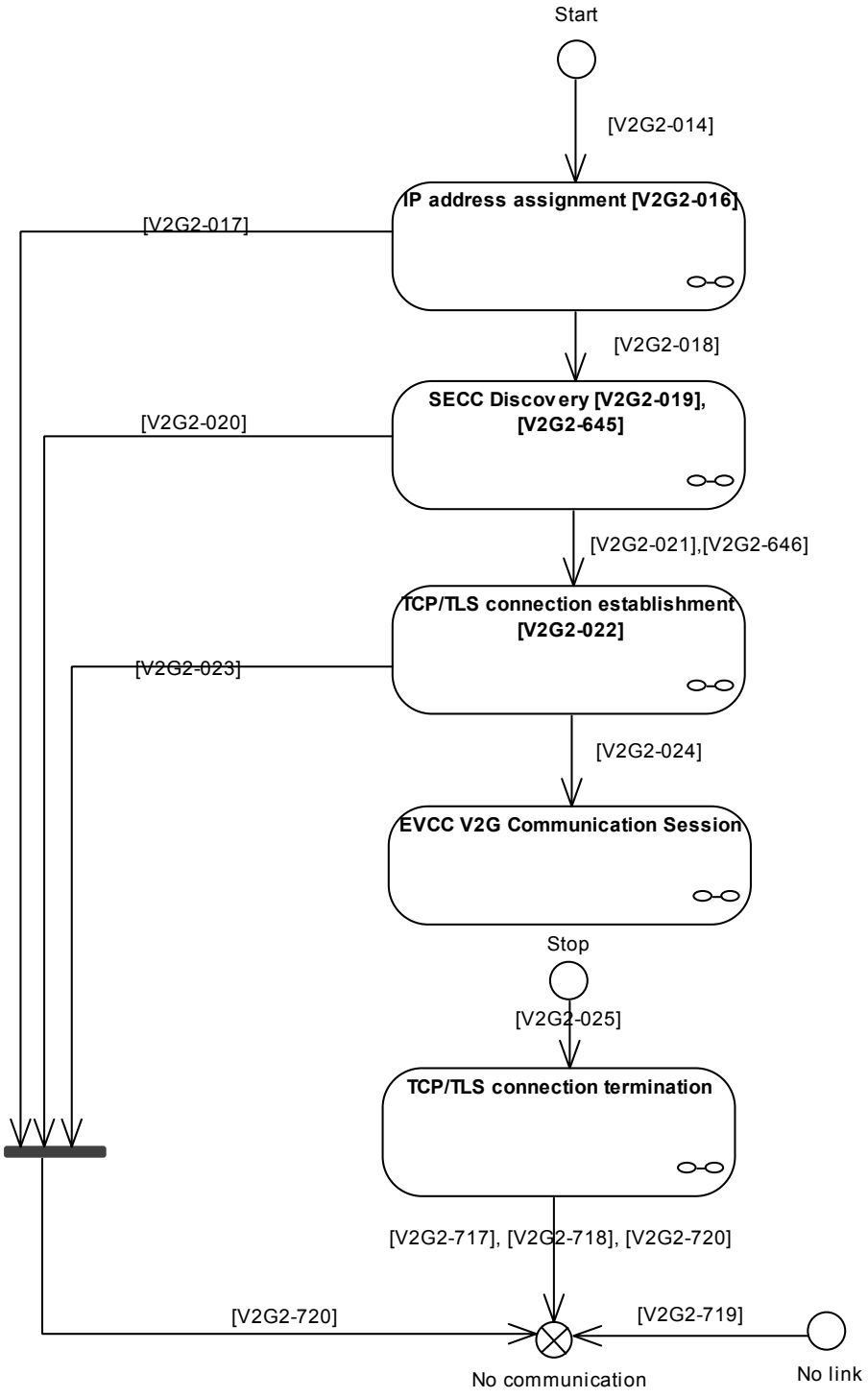


Figure 6 — Overview V2G communication states EVCC

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

The following applies for the SECC:

[V2G2-721] After indication of a successful Data-Link establishment (D-LINK_READY.indication(DLINKSTATUS=Link established), the SECC shall initiate the address assignment mechanism.

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

NOTE 1 To enable the integration of an SECC in different communication infrastructures, the definition of the address assignment for the SECC is out of scope of this standard. This enables the operator to assign an appropriate mechanism. E.g. the operator may chose any existing mechanism providing a valid IP address like e.g. a static IP or the mechanism described in subclauses 7.6.3.2 and 7.6.3.3.

NOTE 2 The SECC shall be configured with an valid IP address for enabling a connection to a EVCC. The IP assignment mechanism for the SECC has no impact on compatibility and is not defined by this standard.

[V2G2-027] After the IP address is assigned, the SECC shall start the SDP server as defined in subclause 7.10.1.

NOTE 3 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.

[V2G2-723] The SECC shall stop the SDP server when SECC_CommunicationSetup_Timer is equal or larger than V2G_SECC _CommunicationSetup_Performance_Time.

[V2G2-029] The SECC shall stop the IP address assignment mechanism when V2G_SECC_CommunicationSetup_Timer is equal or larger than V2G_SECC_CommunicationSetup_Performance_Time.

[V2G2-030] After the SDP server is started successfully, the SECC shall wait for a TLS or TCP connection initialization depending on the SDP response message as defined in subclause 7.10.1.5..

[V2G2-031] The SECC shall wait until the TLS or TCP connection is established.

[V2G2-032] The SECC shall stop waiting for establishing the TLS connection when V2G_SECC_CommunicationSetup_Timer is equal or larger than V2G_SECC_CommunicationSetup_Performance_Time.

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

[V2G2-722] If **[V2G2-033]** applies the SECC can stop the SDP server after successful establishment of a TCP or TLS connection.

[V2G2-034] The SECC shall terminate the TLS or TCP connection after stopping the V2G Communication Session.

[V2G2-724] If the SECC received the message SessionStopReq with parameter ChargingSession equal to "Terminate" it shall terminate the Data-Link (D-LINK_TERMINATE.request()) after sending the message SessionStopRes.

[V2G2-725] If the SECC received the message SessionStopReq with parameter ChargingSession equal to "Pause" it shall pause the Data-Link (D-LINK_PAUSE.request()) after sending the message SessionStopRes and continue with **[V2G2-721]**.

[V2G2-726] Whenever the SECC receives the indication for a missing Data-Link (D-LINK_READY.indication (DLINKSTATUS=No link), the SECC continue with **[V2G2-721]**.

[V2G2-727] If the SECC identifies any error it shall indicate an Data-Link error (D-LINK_ERROR.request()) and continue with **[V2G2-721]**.

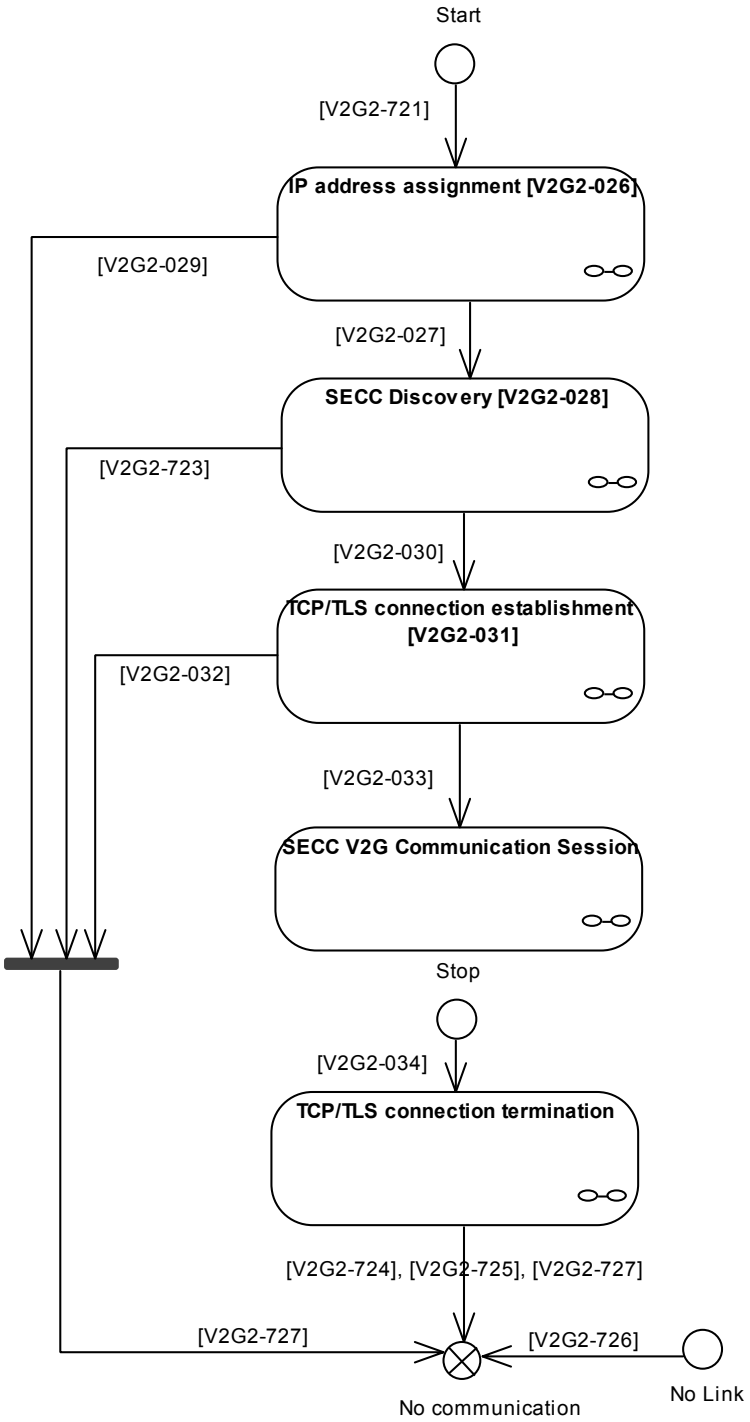


Figure 7 — Overview V2G Communication Session states SECC

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 15118-3.

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

[V2G2-043] If an EVCC chooses to implement a DHCPv6 client it shall implement it according to IETF RFC 3315.

[V2G2-044] If the infrastructure the EV is connected to (EVSE) chooses to implement a DHCPv6 server, it shall implement it according to IETF RFC 3315.

7.6.2.3 Neighbor Discovery (ND)

The EVCC uses IPv6 stateless address auto configuration for generating addresses for its interface. All interfaces have a link-local address. To ensure unique addresses and to support global addresses the neighbour discovery protocol is used.

[V2G2-045] The EVCC shall implement neigborhood discovery 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.

[V2G2-049] Each V2G Entity shall implement the RFCs referred to in column 'Reference' of Table 5 describing the implementation details for the respective ICMP message type.

Table 5 — 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 clause 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 Certificate 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.

7.7 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 retransmission 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 6582.

[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 Entity 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 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 unilaterally 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 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.

[V2G2-070] Each EVCC shall check the validity of the Sub-CA Certificates (in the SECCs certificate chain) via an OCSP response according to IETF RFC 6960, when TLS is used. The OCSP Requests

shall be transported as part of the TLS handshake using the extension defined in IETF RFC 6961.

NOTE 1 It is not intended, that the EVCC requests an OCSP response for an SECC leaf certificate during TLS handshake.

A "pairing mode" supports simple certificate management in private environments (see Annex E.2).

[V2G2-870] The EV can be set into a “pairing mode” using an OEM proprietary mechanism. This “pairing mode” shall only be activatable by explicit user interaction. This mode shall automatically be reset after 120 seconds.

NOTE 2 The pairing mode can be re-activated any time.

NOTE 3 The OEM is suggested to explain the usage and the threats of the pairing mode e.g. in the user manual.

[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.

[V2G2-876] The validity period of an OCSP response for a Sub-CA 1 and Sub-CA 2 shall be not longer that 4 weeks.

NOTE 4 This is aligned to the validity period of an SECC leaf certificate.

[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 5 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.

[V2G2-871] If an SECC is deployed in an environment that is not a private environment, it shall send its own certificate and a chain up to a root (excluding the root certificate itself) which shall be one of those roots previously indicated by the EVCC as being present in the EVCC. If the EVCC has requested OCSP responses, for each certificate that the TLS server responds with, an OCSP response shall be sent to EVCC, as defined in IETF RFC 6961. The OCSP responder shall fill the field “certs” in the OCSP response (BasicOCSPResponse) structure according to IETF RFC 6960sec 4.2.1. The SECC shall also send the certificate of the OCSP responder if it is not the corresponding Sub-CA itself.

NOTE 6 As defined in IETF RFC2560, an OCSP Responder might either be the Sub-CA itself, or it might be an entity which is directly signed by the corresponding Sub-CA using a key pair with a special extended key usage flag in the certificate.

[V2G2-923] If the SECC cannot provide a certificate with a chain up to one of the roots which the EVCC signalled as being present in EVCC, the SECC shall provide within the TLS handshake a valid certificate including a chain to a root certificate (The root certificate itself shall not be transmitted).

[V2G2-924] If the SECC has provided a certificate chain that does not trace up to a root certificate in the list of trusted certificates, the EVCC shall only accept such a certificate, if it successfully validated the certificate chain using an out of band validation mechanism (e.g. server based certificate validation protocol, SCVP, according to RFC 5055). If the validation using such a service is not done, gives a negative result or fails (e.g. due to missing connectivity), the EVCC shall treat the certificate chain as not validated.

[V2G2-872] If an SECC is deployed in a private environment, the SECC shall not send an OCSP response , but the SECC shall include its own private root certificate in the certificate chain it sends.

NOTE 7 This supports the “pairing mode”

[V2G2-873] In case the EVCC previously required an OCSP response via TLS “client hello”, and the TLS server has not sent OCSP responses in “server hello”, the EVCC shall do the following:

- If the TLS server certificate chain contains only one leaf certificate and one root certificate, which is one of the root certificates that the EV has stored as a private root, the EV ignores the fact, that the server has not sent OCSP responses.
- If the TLS server certificate chain is linked to a V2G root, the EVCC shall abort the TLS connection.

[V2G2-874] If the pairing mode is currently activated, the EVCC receives a new root certificate from the SECC during TLS handshake, and the new root certificate fulfils the requirements for private root certificates (see [Certificate Profiles]), then the EVCC shall accept the new root certificate, store it permanently, and treat it as a private root certificate.

[V2G2-875] The EVCC shall verify the SECC Certificate, the complete certificate chain and all OCSP responses (if applicable, including the validity of OCSP responder). The EVCC shall also verify that the SECC Certificate has the Domain Component set to "CPO". If at least one of these verifications fails, the EVCC shall abort the TLS setup process and apply **[V2G2-023]**.

[V2G2-810] The SECC shall support Maximum Fragment Length Negotiation according to IETF RFC 6066**.**

[V2G2-811] The SECC shall be able to support, but not be limited to a maximum fragment length of 2^9 bytes according to 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 Certificate. The validity check of Sub-CA Certificate in the Certificate chain is performed via the OCSP response received during TLS-handshake (for details refer to IETF RFC 6066). Mechanisms to revoke SECC Certificates are not required but instead, it is required to be short term certificates.

As SECC is not aware of which V2G Root Certificate EVCC has, among 10 currently valid V2G Root Certificates for one region, thus 50 currently 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 throughout the session. (There can be no session hijacking.)

[V2G2-071] Each V2G Entity shall provide the credentials and security information stated in Table 6, if TLS is used.

Table 6 — 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 regularly (at least once per week).

[V2G2-602] The SECC shall support all cipher suites defined in Table 7, if TLS is used.

[V2G2-603] The EVCC shall support at least one cipher suite as listed in Table 7, 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 7 gives. One is sufficient allowing for compact implementations. For interoperability, of course, the SECC shall support all cipher suites as listed in Table 7.

Table 7 — 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

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 Dynamic 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 number in the range of Dynamic Ports (49152-65535) as defined in IETF RFC 6335. 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 7.10.1.5) 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.

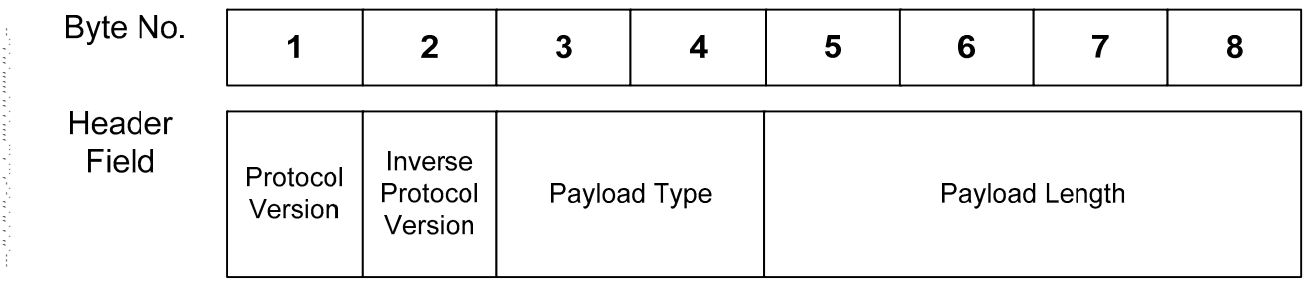


Figure 9 — V2GTP Message header structure

[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.

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

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

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

[V2G2-800] If the V2GTP header contains wrong data (e.g., not supported payload type, wrong payload length, or not supported V2GTP version) the V2GTP entity shall ignore this message.

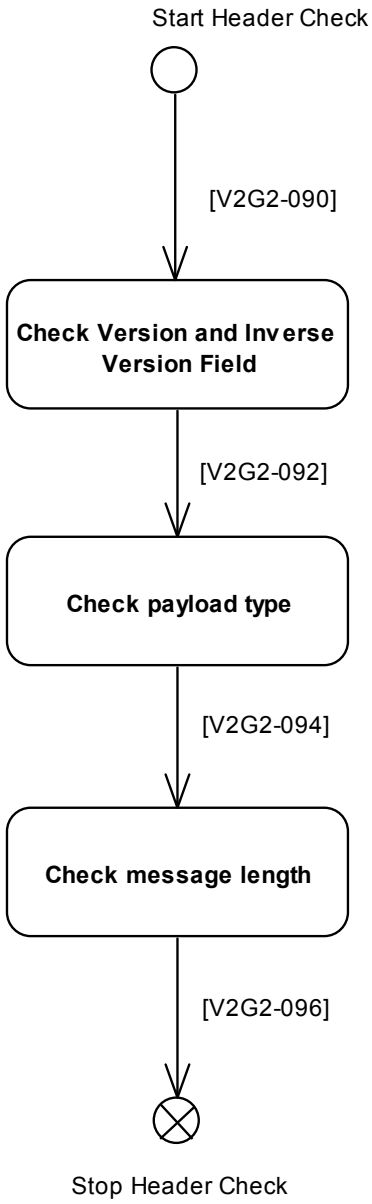


Figure 10 — V2GTP generic header handler

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 according 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 15118 in terms of efficient processing, less resources usage, message size, and message extendibility schema-aware settings should be selected (see subclause 7.9.1.3 for the requirement EXI settings for ISO 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 to 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 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 15118 application messages. Meanwhile, the SECC may be able to handle the data as data structure and/or the more resource intensive Document Object Model (DOM) or in a traditional plain-text XML variant.

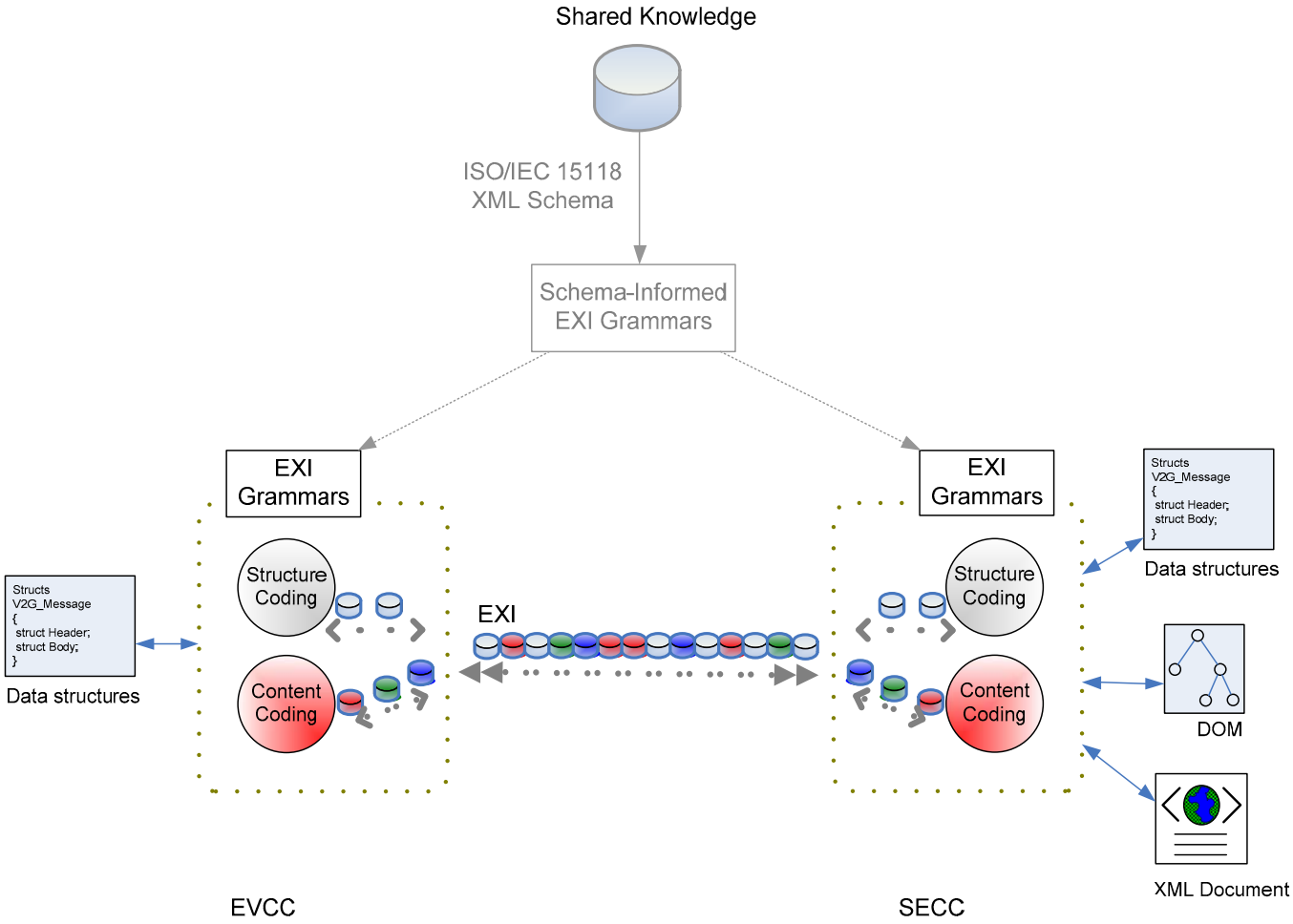


Figure 11 — Basic concept of EXI applied to V2G communication

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:15118:2:2013:MsgDef” that represents this ISO 15118 version 2.0 (major version “2”, minor version “0”) shall be used for encoding and decoding EXI streams.

[V2G2-099] The EXI coder for encoding and decoding of the ISO 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 15118 value
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 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 15118 message. Each EXI implementation (on EVCC or SECC side) shall discard messages that do not respect the ISO 15118 EXI header options.

[V2G2-101] An element/attribute which is not defined in “urn:iso:15118:2:2013: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 15118 communication shall use the EXI profile settings W3C EXI Profileaccording to Table 12.

NOTE For details describing the EXI profile refer to W3C EXI Profile.

Table 12 — EXI profile settings

EXI profile parameter	Description	ISO 15118 value
maximumNumberOfBuiltInElementGrammars	This option is the maximum number of built-in element grammars for which dynamically productions other than AT(xsi:type) productions have been added.	0
maximumNumberOfBuiltInProductions	This option is the maximum number of top-level productions that can be dynamically inserted in built-in element grammars excluding AT(xsi:type) productions.	0

[V2G2-102] A simple type/value of an element/attribute which is not defined in the “urn:iso:15118:2:2013:MsgDef” namespace shall be encoded and decoded type-aware.

7.9.2 Message Security

7.9.2.1 Overview

XML Signature is a W3C recommendation that addresses the authenticity requirements of some data fragments (e.g. metering information) of the XML-based V2G. 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. For protecting confidentiality, a hybrid encryption scheme based on the Diffie-Hellman-Protocol is applied.

7.9.2.2 Application layer credentials and cipher suites

Credentials to be applied on application layer shall be suitable for the targeted XML security. Here, XML signature is chosen to protect billing relevant information between EVCC, SECC and/or SA. Moreover, binary encryption provides a confidentiality protected way for private key provisioning to the EVCC without having intermediaries access this private key. Both approaches require asymmetric key material. The credentials for EVCC signing are provided by the Contract Certificate and credentials for EVCC receiving encrypted data are provided by an ECDH key exchange as described in Annex G.

[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 refer to Annex E.1

NOTE 2 If the certificate is not used for charging it might be used for provisioning. An OEM Provisioning Certificate enables an EVCC to request a valid Contract Certificate for plug and charge.

NOTE 3 The private keys of the Contract Certificate and the OEM Provisioning Certificate are sensitive data. Once a private key gets compromised, it cannot be trusted anymore and therefore should not be used anymore. If one of these private keys could be read by an attacker, the attacker could impersonate the original EVCC and receive electricity at the expense of the original vehicle owner. As a countermeasure, the EVCC may choose to additionally protect said keys. This may range from simple security measures such as setting readout protection fuse bits in the EVCC microcontroller to more advanced protection via secure boot facilities, debug access protection, and embedded hardware security modules (HSM). The latter offer a high level of security including protection against physical attacks. A suitable HSM provides a

secure environment for private key operations. It offers interfaces to use the private key but ensures that the key cannot be read out or manipulated, even from a program running on the EVCC. A customized HSM may even accept input of an encrypted Contract Certificate private key, so that the private key is never accessible outside of the HSM in plain, unencrypted form. Such a HSM may be integrated into the EVCC as a separate physical package or as an on-chip microcontroller extension.

NOTE 4 In order to reduce the number of required certificate update requests for the EV, the mobility operator which issues the charging contract certificates, is suggested to choose longer validity periods (up to maximum), if this is meaningful with regards to contract validity period and cancellation period.

7.9.2.3 Contract Certificates as XML signature credentials

Contract Certificate is bound to a EMAID 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 EMAID shall be encoded in the subject of the certificate.

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.

Typical information exchanged between EVCC and SECC:

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 an introduction to XML Signatures.

NOTE 1 For an Overview on XML Signatures see also Annex J.

XML Signatures as defined in W3C XML Signature Syntax and Processing Version 1.1 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 hashed and cryptographically signed. This standard uses detached XML Signatures according to W3C XML Signature Syntax and Processing Version 1.1. That means the signature and data can be in separate (externally detached) or in the same XML document (internally detached) as sibling elements. 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.

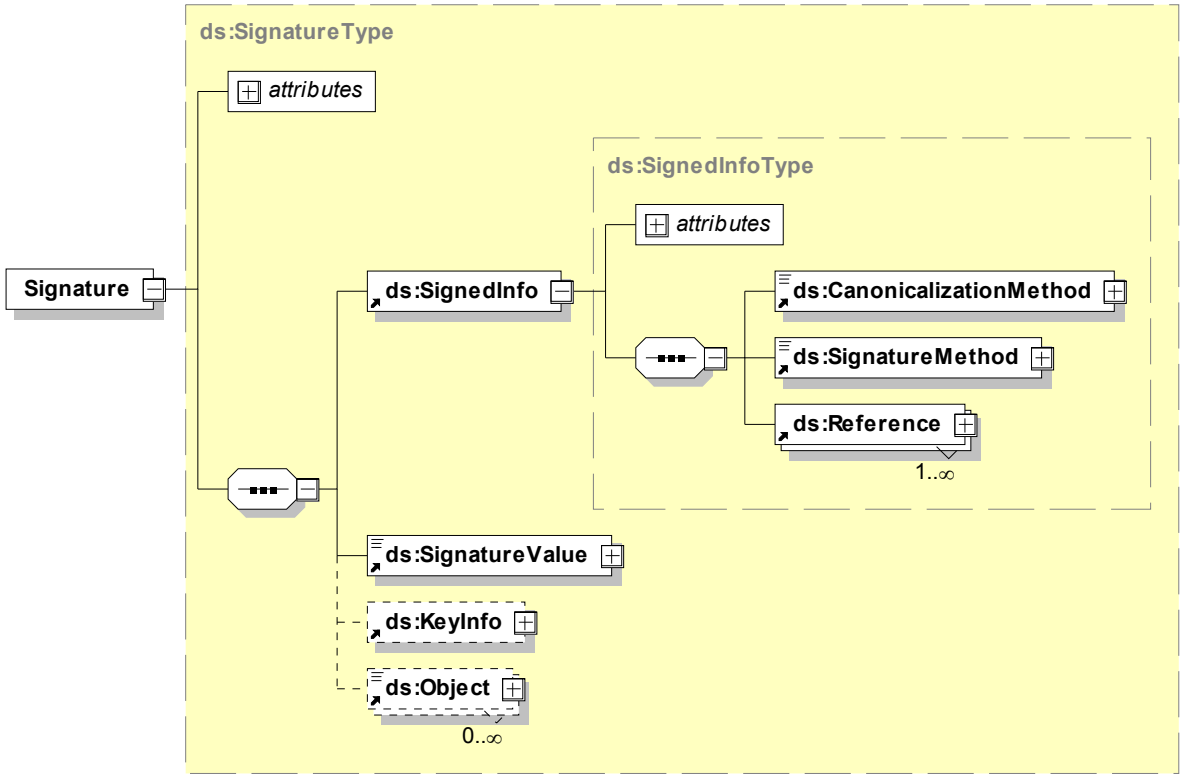


Figure 12 — Schema Diagram – XML Signature

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

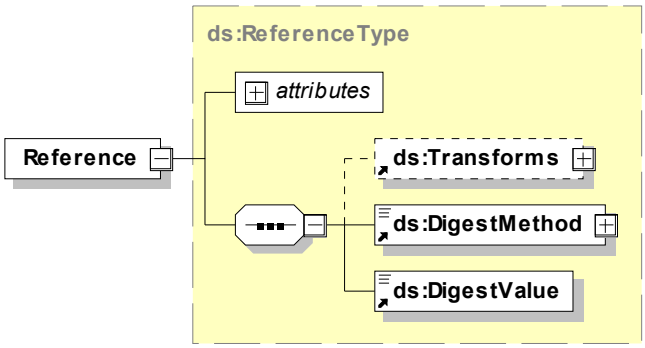


Figure 13 — Schema Diagram – Element Reference included in element SignedInfo

NOTE 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 detached XML signatures.

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

[V2G2-764] As canonicalization method EXI with schema-informed fragment grammar shall be used.

[V2G2-765] Each message, that needs the XML Signature framework shall use the value "http://www.w3.org/TR/canonical-exi/" as Algorithm attribute within the CanonicalizationMethod element.

[V2G2-766] Each message, that needs the XML Signature framework shall use the value "http://www.w3.org/TR/canonical-exi/".as Algorithm attribute within the Transform element.

[V2G2-767] The maximum number of Transforms is limited to one (1) (i.e. per referenced element where a signature is to be transmitted for, just one single Transform algorithm can be indicated).

[V2G2-768] "The parts which have to be signed in the XML-based messages shall be encoded as EXI schema-informed fragment.

[V2G2-769] Each message, that needs the XML Signature framework shall use the value "http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256" as Algorithm attribute within the SignatureMethod element.

[V2G2-770] Each message, that needs the XML Signature framework shall use the value " http://www.w3.org/2001/04/xmlenc#sha256" as Algorithm attribute within the DigestMethod element.

[V2G2-771] The following message elements of the XML Signature framework shall not be used when transmitting signatures in the header of the a V2G message:

- Id (attribute in SignedInfo)
- ##any in SignedInfo – CanonicalizationMethod
- HMACOutputLength in SignedInfo – SignatureMethod
- ##other in SignedInfo – SignatureMethod
- Type (attribute in SignedInfo-Reference)
- ##other in SignedInfo – Reference – Transforms – Transform
- XPath in SignedInfo – Reference – Transforms – Transform
- ##other in SignedInfo – Reference – DigestMethod
- Id (attribute in SignatureValue)
- Object (in Signature)
- KeyInfo

[V2G2-909] The signature shall not reference more than 4 signed elements.

NOTE 3 This allows to determine an upper bound for the size of the signature header.

For the application of XMLDsig, any element which should be signed must be addressable. In this standard, this is achieved by an URI referencing the ID attribute of such an element. Therefore, any message element that is signed carries an ID attribute. If a specific element is to be signed in all use cases, the ID attribute is marked mandatory in the XSD. Otherwise, if it is signed in only some use cases, the ID attribute is marked optional and can be omitted when not needed.

NOTE 4 Presence of an ID attribute does not necessarily indicate that a signature is used, i.e. if no signature is used, an ID can be present nevertheless.

7.9.2.4.3 Encryption mechanism

Private keys belonging to contract certificates need to be protected (i.e. encrypted) when distributed from the SA to the EVCC. For a more detailed explanation please refer to Annex G.

[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.

[V2G2-814] The private key corresponding to the contract certificate shall be transmitted only in encrypted format within ContractSignatureEncryptedPrivateKey in the messages CertificateInstallationRes and CertificateUpdateRes.

[V2G2-815] The private key corresponding to the contract certificate shall be encrypted by the sender (the SA) using the session key derived in the ECDH protocol (see **[V2G2-818]**), applying the algorithm AES-CBC-128 according to NIST Special Publication 800-38A. The initialization vector IV shall be randomly generated before encryption and shall have a length of 128 bit and never be reused. The IV shall be transmitted in the 16 most significant bytes of the ContractSignatureEncryptedPrivateKey field.

NOTE 5 It is implied that proper random generation of IVs will result in them being not reused with an extremely high probability equivalent to certainty for cryptographic needs.

[V2G2-816] The byte order of the private key shall be big endian, including leading zeros.

NOTE 6 As ECDSA with secp256r1 curve is used, the private key is 256 bit long.

NOTE 7 Padding of the plain text (private key) is not required as its length is a multiple of the block size of the used encryption algorithm.

[V2G2-817] The private key corresponding to the contract certificate shall be decrypted by the receiver (EVCC) using the session key derived in the ECDH protocol (see **[V2G2-818]**), applying the algorithm AES-CBC-128 according to NIST Special Publication 800-38A. The initialization vector IV shall be read from the 16 most significant bytes of the ContractSignatureEncryptedPrivateKey field.

[V2G2-818] For session key agreement, the ephemeral-static ECDH protocol “6.2.2.2 One-Pass Diffie-Hellman, C(1, 1, ECC CDH)” as defined in NIST Special Publication 800-56A shall be used. The KDF shall be the “concatenation KDF”, where the hash function shall be SHA256. The SA shall act as party U (as defined in said NIST document) and the EVCC shall act as party V. The protocol shall use elliptic curves as defined in Annex F. The algorithm ID shall be one character 0x01. The sender name ID_U shall be one character "U" = 0x55 , the receiver name ID_V shall be one character "V" = 0x56. A symmetric encryption key of exactly 128 bits shall be derived.

NOTE 8 The authenticity of the transmission is ensured by the surrounding signature. The authenticity check is vital for the security of the ECDH protocol.

[V2G2-819] Ephemeral public keys shall be encoded as “Subject Public Key” as defined in IETF RFC 5480, and be contained in the XML element “DHpublickey”. The uncompressed form shall be used exclusively.

NOTE 9 DHpublickey is 65 bytes long. The first byte has the fixed value 0x04 indicating the uncompressed form.

[V2G2-820] In the message CertificateInstallationRes, the receiver public key QV shall be the public key contained in the existing OEM Provisioning Certificate.

[V2G2-821] In the message CertificateUpdateRes, the receiver public key QV shall be the public key contained in the existing Contract Certificate.

[V2G2-822] A certificate whose key pair is used for ECDH shall have its key usage flags keyAgreement set. This applies to all Contract Certificates and all OEM Provisioning Certificates. This shall be enforced by all participating parties.

[V2G2-823] Upon receipt of a contract certificate, the EVCC shall verify that the private key received with the certificate is a valid private key for that certificate: Its value must be strictly smaller than the

order of the base point, and multiplication of the base point with this value must generate a key matching the public key of the contract certificate.

7.9.2.4.4 Random Number Generation

The ECDSA algorithm strongly relies on the fact that an attacker does not have information about the random nonce. Otherwise, signatures would leak information about the secret key. Depending on the type of defect of the random generator implementation, this may enable an attacker to derive the secret key after a certain number of signatures. In the worst case, simple reuse of the random nonce will enable an attacker to calculate the secret key after only two signatures. Therefore, it is vital to have a suitable random number generator (RNG), a so-called cryptographically secure RNG. In addition, for generating the Initialization Vector in AES-CBC, and for generating a challenge a cryptographically secure RNG is needed.

[V2G2-835] Whenever a V2G Entity requires random numbers within this standard, a state-of-the-art cryptographically secure random number generator shall be used.

NOTE 10 There may already be a suitable, cryptographically secure random number generator available in the implementation. As the actual implementation does not affect interoperability, this standard does not prescribe a specific way in order to be future-proof and to avoid duplication. However, established standards should be followed, for example NIST SP 800-90 A [13] and the German BSI AIS 20/AIS 31 [14].

7.9.2.4.5 Application of security mechanisms to XML message

In general, two pairs of 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;
Asymmetric encryption is applied. The entity, which creates the message, encrypts a single binary field of the XML message. The receiver decrypts that binary field.

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

Table 13 — Overview of applied XML based signatures

XML Message	protected fields	signing entity (sender)	verifying entity (receiver)
AuthorizationReq	message body / all fields (if GenChallenge present)	EVCC	SECC
CertificateUpdateReq	message body / all fields	EVCC; signed with Contract Certificate, chain transmitted in message body element	secondary actor
CertificateUpdateRes	ContractSignatureCertChain ContractSignatureEncryptedPrivateKey DHpublickey eMAID	secondary actor: Certificate Provisioning Service	EVCC; certificate chain stored after successful verification
CertificateInstallationReq	message body / all fields	EVCC; signed with OEM Provision Certificate, transmitted in message body element	secondary actor
CertificateInstallationRes	ContractSignatureCertChain ContractSignatureEncryptedPrivateKey DHpublickey ContractID	secondary actor: Certificate Provisioning Service	EVCC; certificate chain stored after successful verification
MeteringReceiptReq	message body / all fields (optionally)	EVCC	SECC
ChargeParameterDiscoveryRes	SalesTariff (optional, required for PnC)	secondary actor: Mobility Operator Sub-CA 2	EVCC

Table 14 — Overview of applied 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.

7.9.2.5 Certificate Provisioning

The Mobility Operator creates credentials for the EVCC, consisting of a Contract Certificate including a private key, which has been asymmetrically encrypted specifically for that EVCC. The Mobility Operator then forwards the credentials to the Certificate Provisioning Service. The Certificate Provisioning Service asserts the correctness and authenticity of the credentials with a signature and relays the signed message fragment to the SECC, who compiles a CertificateInstallationRes or CertificateUpdateRes message and transmits that to the EVCC. The Certificate Provisioning service is considered trustworthy. It is therefore not required that the EVCC is able to verify the certificate it has received.

The role "Certificate Provisioning Service" can be assumed, for example, by the mobility operator himself, the SECC operator, a completely stand-alone service, or (under careful consideration of the security implications) even the EVCC itself.

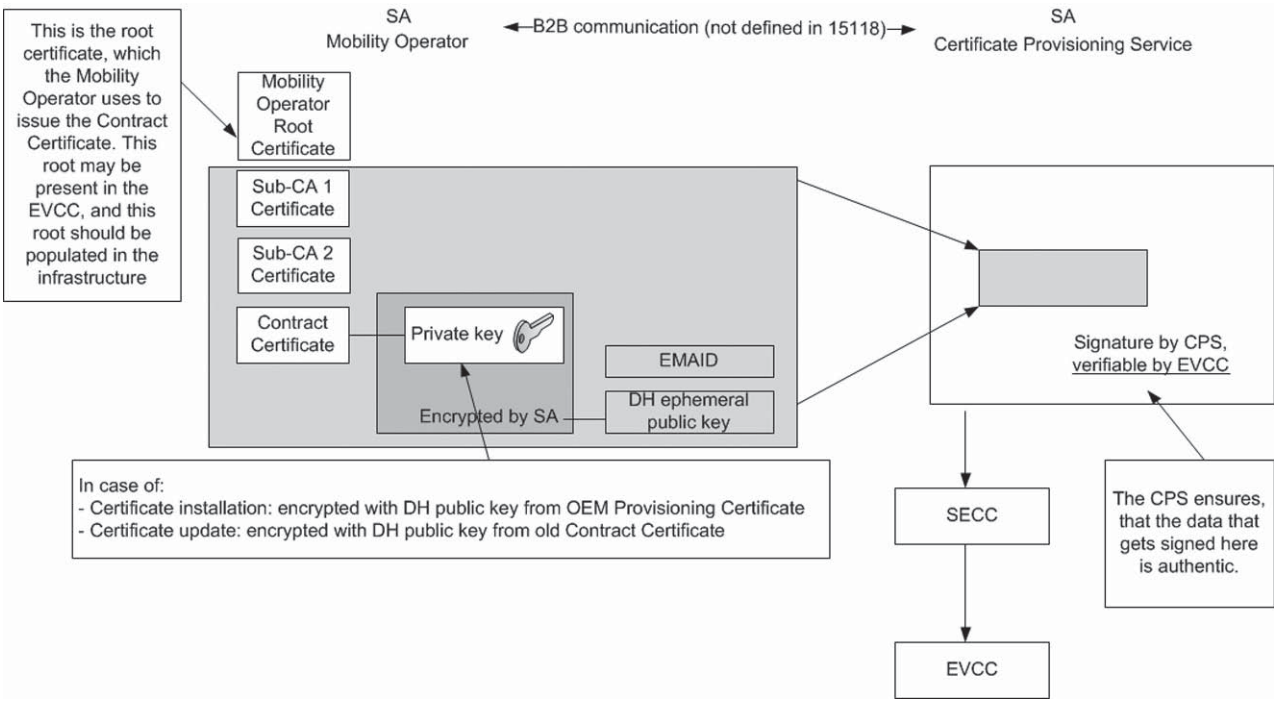


Figure 14 — Process for CertificateInstallationRes and CertificateUpdateRes

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 transport layer connection to the SECC (refer to subclause 7.3.4).

[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. If this is not the case, optimistic DAD as specified in RFC 4429 won't lead to a benefit.

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 an 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 15.

[V2G2-622] An SDP client 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 2.

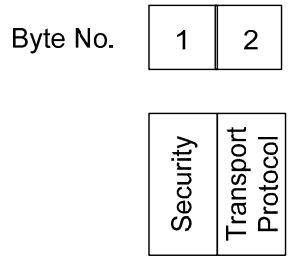


Figure 15 — SECC Discovery request message payload

[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

Description	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

7.10.1.5 SECC Discovery Response Message

The SDP server uses the SDP Response message to respond to an SDP 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 1 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 after an SECC Discovery Request message has been received.

NOTE 2 For detailed timing requirements refer to requirements **[V2G2-159]** - **[V2G2-162]**.

[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 20.

[V2G2-154] An SDP server shall send the SECC Discovery Response message with the payload as defined in Figure 16.

[V2G2-155] An SDP server shall send the payload in the order as shown in Figure 16. 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 20.

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

NOTE 4 The source IP address and the source port of a received UDP packet is usually provided by the TCP/IP stack.

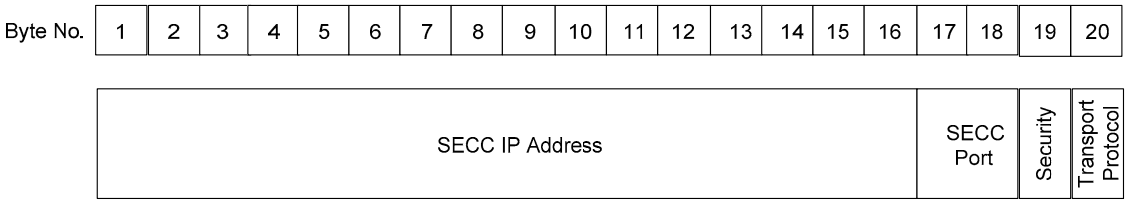


Figure 16 — SECC Discovery response message payload

[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.

7.10.1.6 Timing and Error Handling

The process of SECC discovery is based on the application time out definitions as described in subclause 7.10.1. 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 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 50 consecutive SECC Discovery Request messages it shall stop the SECC Discovery.

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

7.10.1.7 Protocol and Security Options Handling

For SDP request and response handling, the SDP client and SDP server shall implement the following requirements:

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

[V2G2-626] If the SDP client sends an SDP request with a security option code for TLS ("secured with TLS " in Table 16), and the SECC supports TLS, the SDP server shall send an SDP response with security option code TLS.

[V2G2-627] If the SDP client sends an SDP request with a security option code for TLS ("secured with TLS " in Table 16), and the SECC does not support TLS, the SDP server shall send an SDP response with security option code "No transport layer security".

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 SDP response from the SDP server 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 indicated in the SDP response from the SDP server, the SECC shall not accept this communication.

[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.request

The A-Data.request notifies an action for V2G messaging. Table 17 describes the service primitive and its parameter(s).

Table 17 — A-Data.confirmation service primitive

Primitive name	A-Data.request
Entity to support	EVCC
Parameter Name	**Description**
A_Msg	- Session Setup - Service Discovery - Service Detail - Service and Payment Selection - Payment Details - Charge Authorization - 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.request (A_Msg= “message name”) requests the lower layer to send out a V2G request message for the V2G message type that is given by A_Msg.

EXAMPLE A-Data.request (A_Msg= Session Setup) initiates the sending of the SessionSetupReq message as defined in subclause 8.4.3.2.1.

7.10.3.2 A-Data.indication

The A-Data.indication notifies an action for V2G messaging. Table 18 describes the service primitive and its parameter(s).

Table 18 — A-Data.indication service primitive

Primitive name	A-Data.indication
Entity to support	SECC
Parameter Name	**Description**
A_Msg	- Session Setup - Service Discovery - Service Detail - Service and Payment Selection - Payment Details - Charge Authorization - 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.indication (A_Msg= “message name”) indicates the reception of a V2G request message for the V2G message type that is given by A_Msg to the higher layer.

EXAMPLE A-Data.indication (A_Msg= Session Setup) initiates the reception of the SesstionSetupReq message as defined in subclause 8.4.3.2.1.

7.10.3.3 A-Data.response

The A-Data.response notifies about the status V2G messaging. Table 19 describes the service primitive and its parameter(s).

Table 19 — 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 - Charge Authorization - 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 8.4.3.2.2.

7.10.3.4 A-Data.confirmation

The A-Data.confirmation notifies 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 - Charge Authorization - 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 reception of a response message for the V2G message that is given by A_Msg to the higher layer.

EXAMPLE A-Data.confirmation (A_Msg= Session Setup) indicates the successful reception of a SessionSetupRes Message as defined in subclause 8.4.3.2.2.

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 initiates 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, 0. 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 defines 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)

[V2G2-809] When transmitting application layer messages the big endian byte order rule shall be applied: The most significant byte is sent first, the least significant byte is sent last.

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 17 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 unique 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 Priority 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 18 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 indicates 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, however 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 elements 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 performed prior to 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 defined in Figure 17.

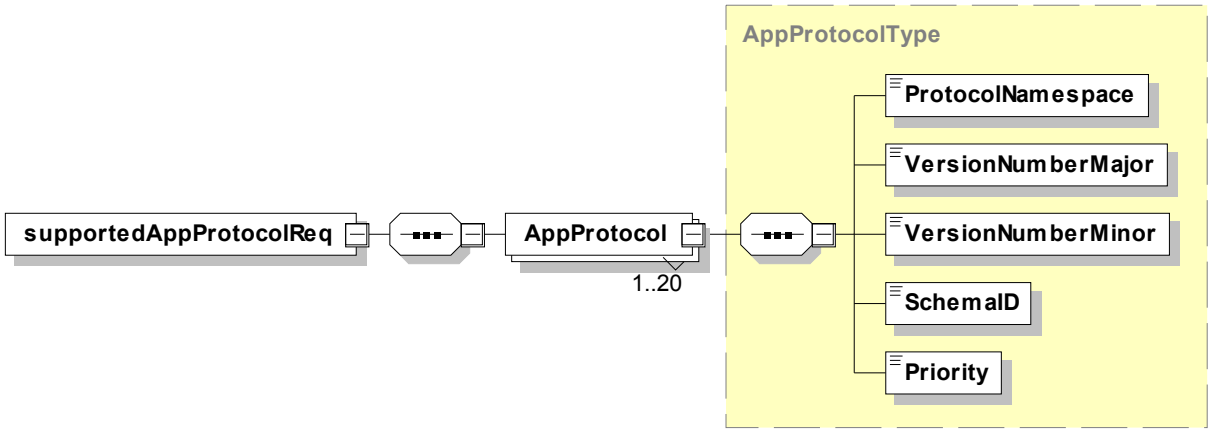


Figure 17 — Schema Diagram – supportedAppProtocolReq

[V2G2-176] The SECC and EVCC shall implement the message and message elements as defined in Figure 18.

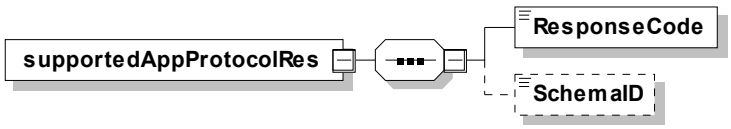


Figure 18 — Schema Diagram – supportedAppProtocolRes

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

8.2.3 Semantics description supportedAppProtocol messages

[V2G2-178] The message elements of the messages defined in Figure 17 and Figure 18 shall be used as defined in Table 21 and Table 22.

Table 21 — Semantics and type definition for supportedAppProtocolReq 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 protocols. 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.
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. See also **[V2G2-167]**

Table 22 — Semantics and type definition for supportedAppProtocolRes message elements

Element/Attribute Name	Type	Semantics
SchemaID	simpleType: unsignedByte refer to Annex C.2 for the type definition	Optional: This message element is used by the SECC to reference one of the EVCC supported protocols received in the request message.
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.2 for the type definition	This message element is used by the SECC for indicating whether 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 supportedAppProtocol messages between the EVCC and the SECC. In the request message, the EVCC sends a prioritized list of supported application layer protocols (15118:2:2013 with version 2.0, 15118:2:2010 with version 1.0) to the SECC where the first has the higher priority. In the response message the SECC confirms protocol 15118:2:2013 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"?>
<n1:supportedAppProtocolRes xsi:schemaLocation="urn:iso:15118:2:2010:AppProtocol ../V2G_CI_AppProtocol.xsd"
xmlns:n1="urn:iso:15118:2:2010:AppProtocol"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <AppProtocol>
            <ProtocolNamespace>urn:iso:15118:2:2013: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>2</Priority>
      </AppProtocol>
</n1:supportedAppProtocolReq>
```

V2G message example 1 – supportedAppProtocolReq: protocol prioritization

```
<?xml version="1.0" encoding="UTF-8"?>
<n1:supportedAppProtocolRes xsi:schemaLocation="urn:iso:15118:2:2010:AppProtocol ../V2G_CI_AppProtocol.xsd"
xmlns:n1="urn:iso:15118:2:2010:AppProtocol"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <ResponseCode>OK_SuccessfulNegotiation</ResponseCode>
      <SchemaID>10</SchemaID>
</n1: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 supportedAppProtocol messages between the EVCC and the SECC. In the request message, the EVCC sends just one supported application layer protocol (15118:2:2013 with version 2.0) to the SECC. The SECC support protocol version 2.1 only. In the response message the SECC confirms protocol 15118:2:2013 with VersionNumberMajor equal to two (2) 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 are not known to the EVCC but can be ignored.

```
<?xml version="1.0" encoding="UTF-8"?>
<n1:supportedAppProtocolRes xsi:schemaLocation="urn:iso:15118:2:2010:AppProtocol ../V2G_CI_AppProtocol.xsd"
xmlns:n1="urn:iso:15118:2:2010:AppProtocol"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <AppProtocol>
            <ProtocolNamespace>urn:iso:15118:2:2013:MsgDef</ProtocolNamespace>
            <VersionNumberMajor>2</VersionNumberMajor>
            <VersionNumberMinor>0</VersionNumberMinor>
            <SchemaID>1</SchemaID>
            <Priority>1</Priority>
      </AppProtocol>
</n1:supportedAppProtocolReq>
```

V2G message example 3 – supportedAppProtocolReq: deviation in minor version

```
<?xml version="1.0" encoding="UTF-8"?>
<n1:supportedAppProtocolRes xsi:schemaLocation="urn:iso:15118:2:2010:AppProtocol ../V2G_CI_AppProtocol.xsd"
xmlns:n1="urn:iso:15118:2:2010:AppProtocol"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <ResponseCode>OK_SuccessfulNegotiationWithMinorDeviation</ResponseCode>
      <SchemaID>1</SchemaID>
</n1: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 15118 application layer Message Set is signalized by the XML schema namespace " urn:iso:15118:2:2013: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 19 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 19.

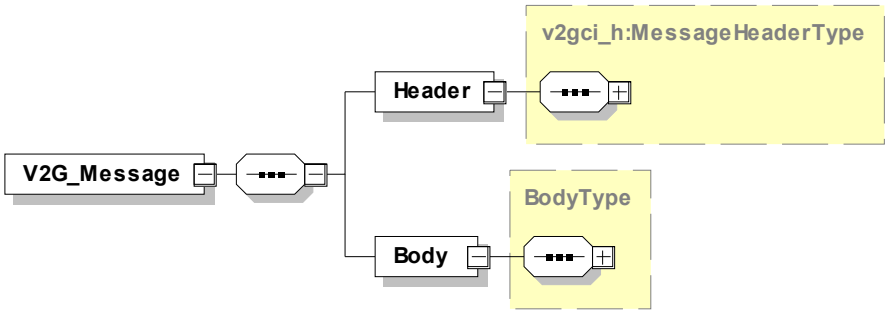


Figure 19 — Schema Diagram – V2G message

[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 0.
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 subclause 0.

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 be started. 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:15118:2:2013:MsgBody"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <v2gci_d:Header>
            <v2gci_h:SessionID>00</v2gci_h:SessionID>
      </v2gci_d:Header>
      <v2gci_d:Body>
            <v2gci_b:SessionSetupReq>
                  <v2gci_b:EVCCID>0123456789AB</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 20 shows the schema definition of the V2G message header.

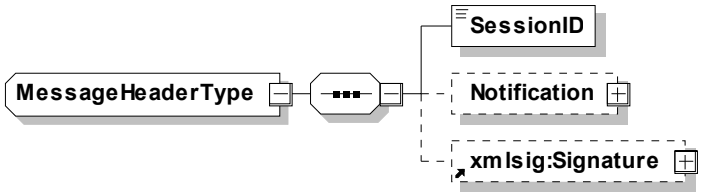


Figure 20 — Schema Diagram – Message header

[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 subclause 8.4.2 for requirements relative to this message element.
Notification	complexType: NotificationType, see subclause 8.5.2.8	Optional: This element is used by the SECC for transmitting additional fault information when an error occurred on the SECC side.
xmlsig:Signature	separate namespace: "http://www.w3.org/2000/09/xmldsig#"	Optional: This element is used if a certain V2G message requires to be signed.

[V2G2-182] Each V2G message containing signed elements shall include the xmlsig: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 21 shows the schema definition of the V2G message body. The messages described in the following clause 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 0.

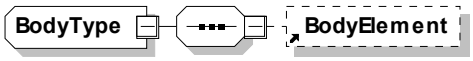


Figure 21 — Schema Diagram – Message body

[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 0	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 clause 0 is instantiated.

8.4 V2G Communication Session and BodyElement Definitions

8.4.1 General

A V2G Communication Session in this document is defined as the exchange of V2G messages between two V2G Entities in a predefined sequence (see chapter 8.8) for managing the charging process. A V2G Communication Session always starts with the SessionSetupReq/Res message pair and always ends with the SessionStopReq/Res message pair.

All messages of a V2G Communication Session carry a SessionID that allows to manage the V2G Communication Sessions between V2G Entities on application level. The SessionID is negotiated by the EVCC and the SECC in the SessionSetupReq/Res message pair. All V2G messages of a V2G Communication Session except the SessionSetupReq message use the same SessionID.

The SessionID enables pausing and resuming of a charging session using multiple V2G Communication Sessions. For this, the EVCC and the SECC apply the same SessionID in all V2G Communication Sessions during a charging session. Applying multiple V2G Communication Sessions for pausing and resuming a charging session enables the EVCC and the SECC to stop all layers and restart all layers when resuming while keeping the same application context and charging process management data. This for example allows the EV and the EVSE to switch off the ISO 15118 communication module completely during pausing to save energy.

In ISO 15118 a V2G Communication Session pausing is controlled by the parameter ChargingSession in SessionStopReq with its values “Terminate” and ”Pausing”. The parameter can be used independently from the charging profile. This means that an EV can initiate a pausing at any time after sending PowerDeliveryReq with ChargeProgress equal to 'Stop' and according to [V2G2-739].

The example in Figure 22 — V2G Communication Session handling shows two V2G communication sessions with pausing and their relation over time to the TCP/TLS connections, the data link establishment, and the charging session.

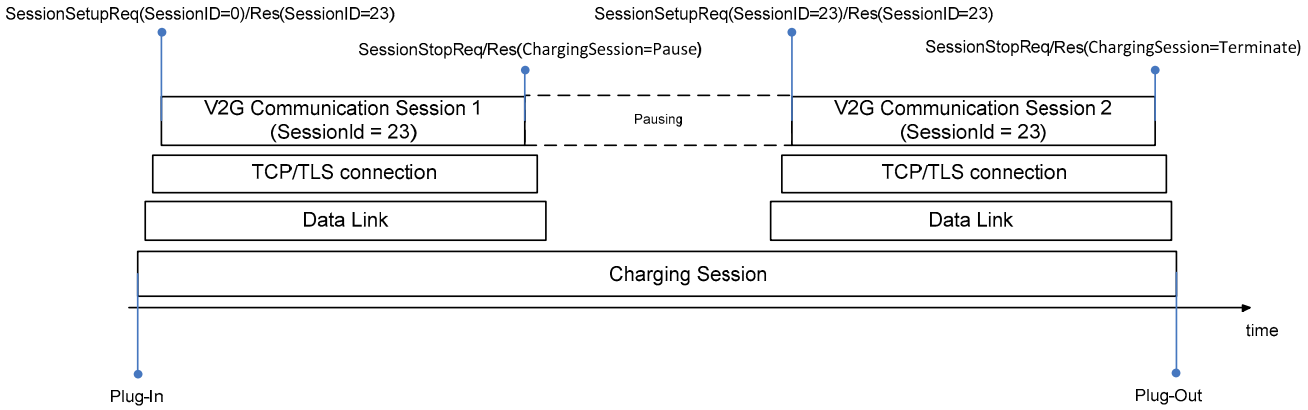


Figure 22 — V2G Communication Session handling

8.4.2 Session Handling

The following applies for the EVCC:

[V2G2-739] An EVCC shall pause a V2G Communication Session with the parameter ChargingSession set to value ‚Pause' in message SessionStopReq and shall stop the V2G Communication Session (terminate the transport layer).

NOTE 1 When pausing a V2G Communication Session, the EVCC terminates its transport layer connection which induces as well a charging pause. No value added services are usable during a charging pause either.

[V2G2-740] If an EVCC resumes a previously paused V2G Communication Session, the following parameter values provided by the EVCC in the previous V2G Communication Session shall be provided again for the resumed V2G Communication Session:

- SessionID which was communicated in the header of the SessionSetupRes message in the previous V2G Communication Session (for all request messages starting from SessionSetupReq).
- SelectedPaymentOption (PaymentServiceSelectionReq)
- RequestedEnergyTransferMode (ChargeParameterDiscoveryReq)

[V2G2-742] If an EVCC wants to resume a previously paused V2G Communication Session, it shall send the parameter DepartureTime in ChargeParameterDiscoveryReq reduced by the elapsed time.

[V2G2-743] If an EVCC wants to resume a previously paused V2G Communication Session, it shall send the parameter EAmount in ChargeParameterDiscoveryReq reduced by the energy that was already charged.

[V2G2-744] If **[V2G2-739]** applies, the EVCC shall take care that **[V2G2-740]** is fulfilled as long as no CP State A, E, or F was detected in the EVCC as defined in IEC 61851-1.

[V2G2-746] When sending the first SessionSetupReq message after connecting the EV with the EVSE (plug-in), the EVCC shall set the parameter SessionID in the message header equal to zero (0).

[V2G2-747] The header of any message sent by the EVCC during an active V2G Communication Session shall include the SessionID value returned by the SECC in the response to the SessionSetupReq initiating the currently active V2G Communication Session.

[V2G2-748] An EVCC can resume a charging session by sending a SesstionSetupReq with a message header including the SessionID value from the previously paused V2G Communication Session.

[V2G2-749] If the V2G Communication Session is resumed according to [V2G2-754] and if the EVCC wants to change the charging parameters listed in [V2G2-740], it shall use the renegotiation mechanism defined in this standard.

The following applies for the SECC:

[V2G2-741] If an EVCC resumes a previously paused V2G Communication Session, the following parameter values provided by the SECC in the previous V2G Communication Session shall be provided again for the resumed V2G Communication Session:

- SessionID which was communicated in the header of the SessionSetupRes message in the previous V2G Communication Session if the SessionID communicated in the header of SessionSetupReq matches the stored value (for all request messages starting from SessionSetupReq)
- PaymentOptionList (ServiceDiscoveryRes). Only the payment option previously selected by the EVCC shall be provided.
- ChargeService (ServiceDiscoveryRes)
- SAScheduleTuple (ChargeParameterDiscoveryRes). At least the SAScheduleTuple (including the related PMaxSchedule and SalesTariff data) whose ID was selected by the EVCC in its ChargingProfile in the previous V2G Communication Session shall be provided. This tuple ID must not change during a V2G Communication Session. The period of time this SAScheduleTuple applies for shall be reduced by the time already elapsed.

NOTE 2 This is to ensure that a ChargingProfile calculated by an EVCC during the previous V2G Communication Session(s) is still valid after a charging break for the entire charging session.

NOTE 3 The EV can choose between following the stored ChargingProfile or creating a new one based on the additional SAScheduleTuples provided in the SAScheduleList of the ChargeParameterDiscoveryRes.

[V2G2-745] If **[V2G2-739]** applies, the SECC shall take care that **[V2G2-741]** is fulfilled as long as no CP State A, E, or F was detected in the SECC as defined in IEC 61851-1.

[V2G2-750] When receiving the SessionSetupReq with the parameter SessionID equal to zero (0), the SECC shall generate a new (not stored) SessionID value different from zero (0) and return this value in the SessionSetupRes message header.

[V2G2-751] The SessionID value returned by the SECC in the SessionSetupRes message shall not change as long as the V2G Communication Session is not terminated (i.e. value of parameter ChargingSession of SessionStopReq has not been set to ‘Terminate’ at any time).

[V2G2-752] The header of any message sent by the SECC during an active V2G Communication Session shall include the SessionID value returned by the SECC in the response to the SessionSetupReq initiating the currently active V2G Communication Session.

[V2G2-753] If an EVCC chooses to resume a charging session by sending a SesstionSetupReq with a message header including the SessionID value from the previously paused V2G Communication Session, the SECC shall compare this value to the value stored from the preceding V2G Communication Session.

[V2G2-754] If the SessionID value received in the current SessionSetupReq is equal to the value stored from the preceding V2G Communication Session, the SECC shall confirm the continuation of the charging session by sending a SessionSetupRes message including the stored SessionID value and indicating the resumed V2G Communication Session with the ResponseCode set to "OK_OldSessionJoined" (refer also to **[V2G2-463]** for selecting the appropriate response code).

[V2G2-755] If the V2G Communication Session is resumed according to **[V2G2-754]** and if the SECC wants to change the charging parameters listed in **[V2G2-741]**, it shall use the renegotiation mechanism defined in this standard.

[V2G2-756] If the SECC receives a SessionSetupReq including a SessionID value which is not equal to zero (0) and not equal to the SessionID value stored from the preceding V2G Communication Session, it shall send a SessionID value in the SessionSetupRes message that is unequal to "0" and unequal to the SessionID value stored from the preceding V2G Communication Session and indicate the new V2G Communication Session with the ResponseCode set to "OK_NewSessionEstablished" (refer also to **[V2G2-462]** for applicability of this response code).

NOTE 4 Refer to subclause 8.8.3 for additional requirements relative to the usage of in Part 2 defined response codes applicable to the parameter SessionID.

8.4.3 Common Messages

8.4.3.1 Overview

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

8.4.3.2 SessionSetupReq/Res

8.4.3.2.1 SessionSetupReq

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 104 and according to Figure 23.

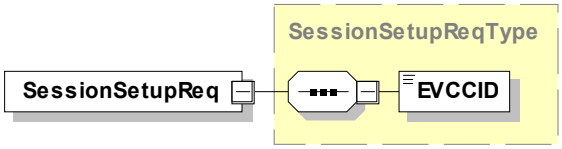


Figure 23 — Schema Diagram – SessionSetupReq

[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: 6) 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 as six hexBinary encoded bytes, i.e. the element shall have a length of six bytes.

[V2G2-879] The EVCC shall transmit an EVCCID with six byte in length, and it shall fill it with its MAC address.

8.4.3.2.2 SessionSetupRes

By using the SessionSetupRes the SECC responds to a SessionSetupReq. With the SessionSetupRes the SECC notifies the EVCC with an enclosed ResponseCode, 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 in Table 104 and according to Figure 24.

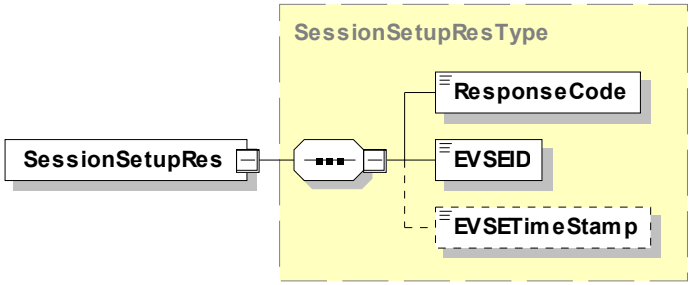


Figure 24 — Schema Diagram – SessionSetupRes

[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	ResponseCode indicating the acknowledgment status of any of the V2G messages received by the SECC.
EVSEID	simpleType: evseIDType string (min length: 7, max length:37)	Any ID that uniquely identifies the EVSE and the power outlet the vehicle is connected to. The format of this message element is defined in Annex H. If an SECC cannot provide such ID data, the value of the EVSEID is set to zero ("ZZ00000").
EVSETimeStamp	simpleType long refer to Annex C.6 for the type definition	Optional: Timestamp of the current SECC time. Format is “Unix Time Stamp”. This message element may be used by the EVCC to decide whether a specific contract certificate can be used for contract based charging during the current communication session. Based on this information the EVCC might implement a strategy when certificate updates are required. Based on this information the EV might synchronize its local time, if no other means are available in the EV.

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

8.4.3.3 ServiceDiscoveryReq/Res

8.4.3.3.1 ServiceDiscoveryReq/Res 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 value added services is already considered and offers means for extensibility. Therefore the Service Discovery differentiates between various service types and scopes.

8.4.3.3.2 ServiceDiscoveryReq

By sending the ServiceDiscoveryReq 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 in Table 104 and according to Figure 25.

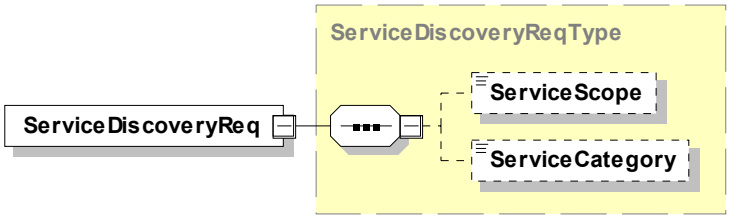


Figure 25 — Schema Diagram – ServiceDiscoveryReq

[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: 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.). 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 all scopes if no specific ServiceScope has been indicated in request message. The definition of an URI adheres to IETF RFC 1630 and is implementation specific.
ServiceCategory	simpleType: serviceCategoryType enumeration refer to Annex C.6 for the type definition	Optional: 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 ServiceCategory message element.

8.4.3.3.3 ServiceDiscoveryRes

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 ResponseCode 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 in Table 104 and according to Figure 26.

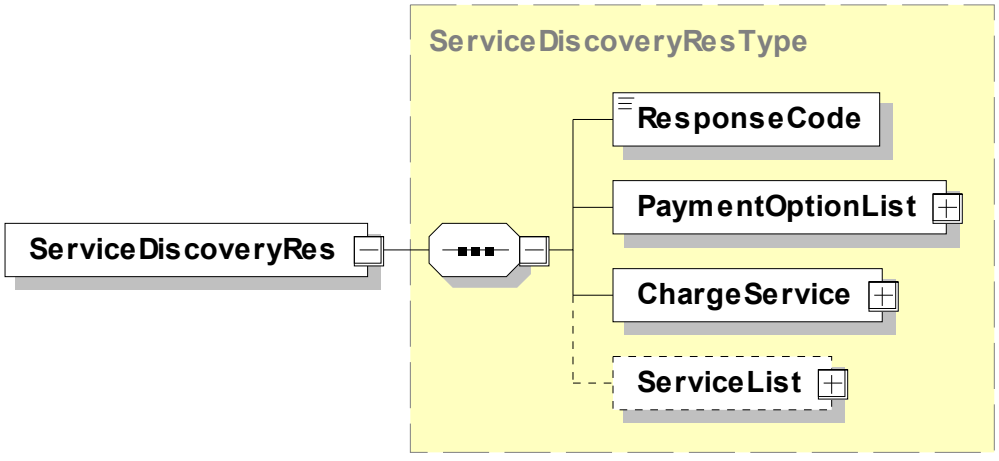


Figure 26 — Schema Diagram – ServiceDiscoveryRes

[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	ResponseCode indicating the acknowledgment status of any of the V2G messages received by the SECC.
PaymentOptionList	complexType: PaymentOptionListType 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 chosen to pay for the services. The EVCC can only select one payment method for all services used by the EVCC.
ChargeService	complexType: ChargeServiceType refer to subclause 8.5.2.3	Available charging services supported by the EVSE.
ServiceList	complexType: ServiceListType 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. The number of service elements is limited to eight

8.4.3.4 ServiceDetailReq/Res

8.4.3.4.1 ServiceDetailReq

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 in Table 104 and according to Figure 27.

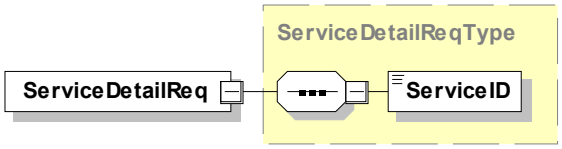


Figure 27 — Schema Diagram – ServiceDetailReq

[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 identifies a service which has been offered by the SECC in the ServiceDiscoveryRes message.

8.4.3.4.2 ServiceDetailRes

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 in Table 104 and according to Figure 28.

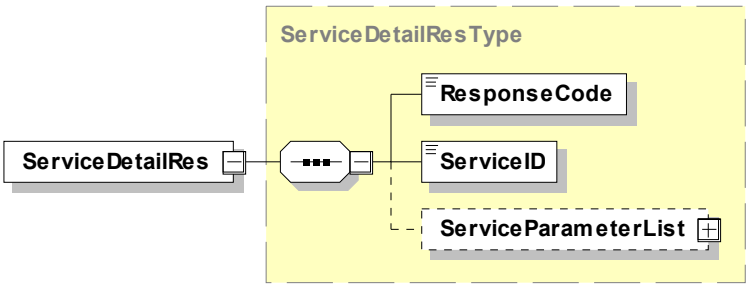


Figure 28 — Schema Diagram – ServiceDetailRes

[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	ResponseCode 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 identifies 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.

8.4.3.5 PaymentServiceSelectionReq/Res

8.4.3.5.1 Payment and Service 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.

8.4.3.5.2 PaymentServiceSelectionReq

This request message transports the information on the selected services and on how all the selected services are paid (see subclause 8.6.3.1).

[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 104 and according to Figure 29.

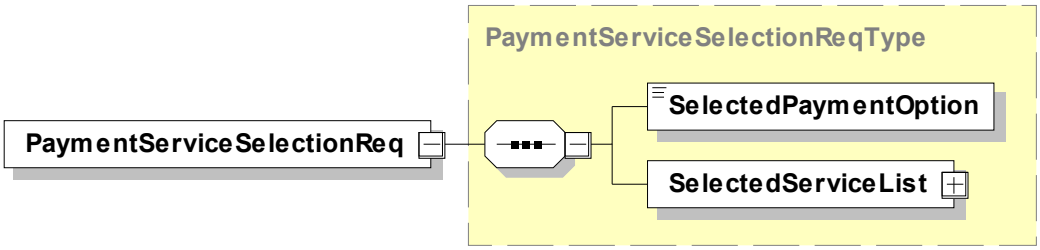


Figure 29 — Schema Diagram – PaymentServiceSelectionReq

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

Table 32 — Semantics and type definition for PaymentServiceSelectionReq

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.

8.4.3.5.3 PaymentServiceSelectionRes

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 in Table 104 and according to Figure 30.

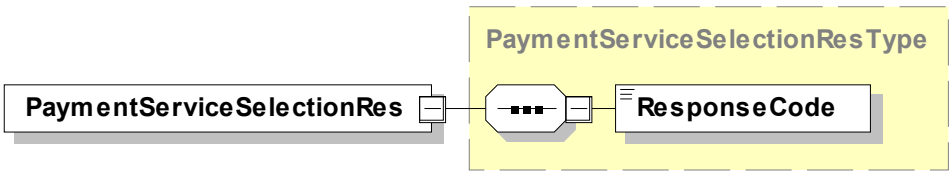


Figure 30 — Schema Diagram – PaymentServiceSelectionRes

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

Table 33 — Semantics and type definition for PaymentServiceSelectionRes

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

8.4.3.6 PaymentDetailsReq/Res

8.4.3.6.1 PaymentDetailsReq/Res 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 an eMAID would be necessary).

8.4.3.6.2 PaymentDetailsReq

With the PaymentDetailsReq the EVCC provides the payment details in case the selected payment was “Contract”. By sending the signature certificate chain and eMAID, 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 104 and according to Figure 31.

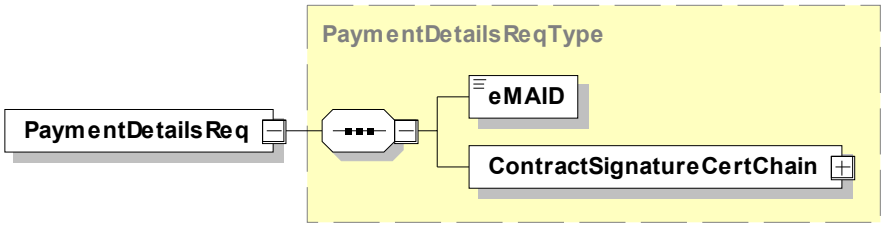


Figure 31 — Schema Diagram – PaymentDetailsReq

[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
eMAID	simpleType: eMAIDType string (min length: 14, max length: 15) refer to Annex C.6 for the type definition	This element identifies the charging contract. The format is defined in Annex H.
ContractSignatureCertChain	complexType: CertificateChainType refer to subclause 8.5.2.4	This element contains the Contract Certificate and optional SubCertificates

8.4.3.6.3 PaymentDetailsRes

With the PaymentDetailsRes 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 in Table 104 and according to Figure 32.

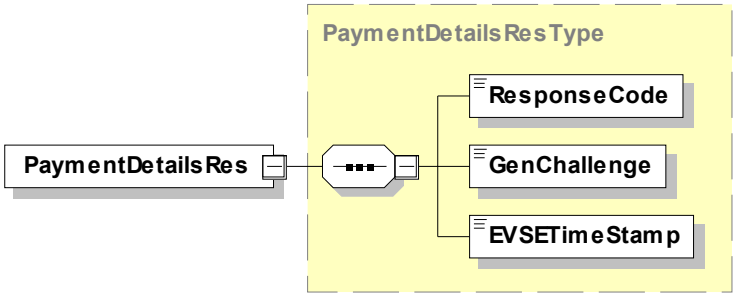


Figure 32 — Schema Diagram – PaymentDetailsRes

[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	ResponseCode indicating the acknowledgment status of any of the V2G messages received by the SECC.
GenChallenge	simpleType genChallengeType base64Binary (length 16) refer to Annex C.6 for the type definition	The challenge sent by the SECC. This element contains the generated random number.
EVSETimeStamp	simpleType long refer to Annex C.6 for the type definition	Timestamp of the current SECC time. Format is “Unix Time Stamp”. This message element may be used by the EVCC to understand, why the SECC has not accepted the transferred contract certificate and/or chain (as transferred from EVCC to SECC in message “PaymentDetailsReq”. Based on this information the EVCC might implement a strategy when certificate updates are required (for future sessions). Based on this information the EV might synchronize its local time, if no other means are available in EV.

[V2G2-825] The GenChallenge field shall be exactly 128 bits long.

[V2G2-826] The entropy of the GenChallenge field shall be at least 120 bits.

[V2G2-898] In case of PnC, the EVCC shall send its Contract Certificate including the certificate chain (up to, but not including the root certificate) in the element ContractSignatureCertChain of the message PaymentDetailsReq.

[V2G2-899] In case of PnC, the SECC shall receive the Contract Certificate including its certificate chain in the element ContractSignatureCertChain of the message PaymentDetailsReq. It shall validate the certificate including its chain, verify that it traces back to a trusted Mobility Operator Root Certificate. If any of the aforementioned verifications and validations fail, the SECC shall consider the Contract Certificate as invalid. It shall also not use said Contract Certificate for any further verifications.

8.4.3.7 AuthorizationReq/Res

8.4.3.7.1 AuthorizationReq

If a generated challenge was sent by the SECC in a previous message, the EVCC sends back this challenge and the respective signature. Otherwise, the message is empty.

[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 in Table 104 and according to Figure 33.

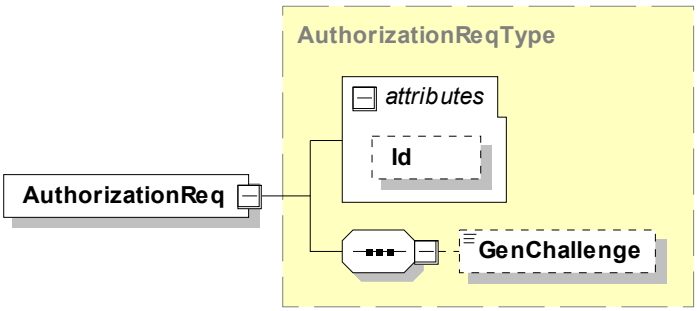


Figure 33 — Schema Diagram – AuthorizationReq

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

Table 36 — Semantics and type definition for AuthorizationReq

Element Name	Type	Semantics
Id	simpleType: ID string refer to Annex C.6 for the type definition	Optional: This attribute is used for referencing the message body in the signature header when a signature needs to be applied.
GenChallenge	simpleType genChallengeType base64Binary (length 16) refer to Annex C.6 for the type definition	Optional: The challenge sent by the SECC in PaymentDetailsRes message. This element contains the generated random number.

[V2G2-697] The GenChallenge field shall be exactly 128 bits long.

[V2G2-698] The entropy of the GenChallenge field shall be at least 120 bits.

8.4.3.7.2 AuthorizationRes

Finally, the SECC verifies the challenge signature (and the certificate with its chain if not done before) and sends the corresponding authorization response message.

[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 in Table 104 and according to Figure 34.

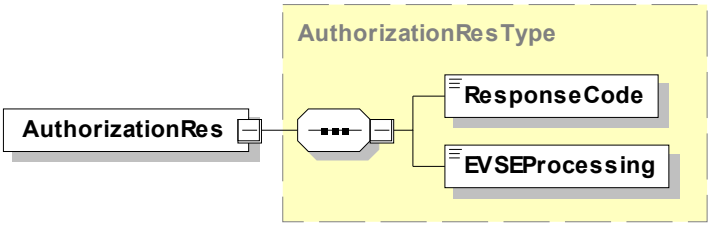


Figure 34 — Schema Diagram – AuthorizationRes

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

Table 37 — Semantics and type definition for AuthorizationRes

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 AuthorizationReq 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	ResponseCode indicating the acknowledgment status of any of the V2G messages received by the SECC.

[V2G2-900] In case of PnC, and if it sends a challenge previously received from the SECC, the EVCC shall sign the body element of the AuthorizationReq using the Contract Certificate private key it has sent in PaymentDetailsReq in this session.

[V2G2-901] In case of PnC, the SECC shall verify, that the body element of the AuthorizationReq is correctly signed by the private key corresponding to the Contract Certificate previously received in the PaymentDetailsReq message in this session, and that the GenChallenge field has the same content as the GenChallenge field the SECC has previously sent in the PaymentsDetailsRes message. If any of the aforementioned verifications and validations fail, the SECC shall consider the Signature as invalid. See also **[V2G2-475]**.

8.4.3.8 ChargeParameterDiscoveryReq/Res

8.4.3.8.1 ChargeParameterDiscoveryReq/Res Handling

After being authorized for charging at the EVSE (SECC) the EVCC and the SECC negotiate the charging parameters with the ChargeParameterDiscoveryReq/Res message pair.

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

The charging parameters negotiation that precedes 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 EVSE 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 shall be allowed to cope with difficulties encountered, locally or regionally.

Such re-negotiation shall 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.

8.4.3.8.2 ChargeParameterDiscoveryReq

By sending the ChargeParameterDiscoveryReq 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 recharging the vehicle, capabilities of the EV charging system and the point in time the vehicle operator intends to leave the EVSE.

[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 in Table 104 and according to Figure 35.

[V2G2-215] If there is no DepartureTime information available the EVCC shall not include this message element in the ChargeParameterDiscoveryReq message.

[V2G2-761] If the EVCC does not send a departure time as part of the ChargeParameterDiscoveryReq, the SECC shall assume that the EV intends to begin charging without any delay.

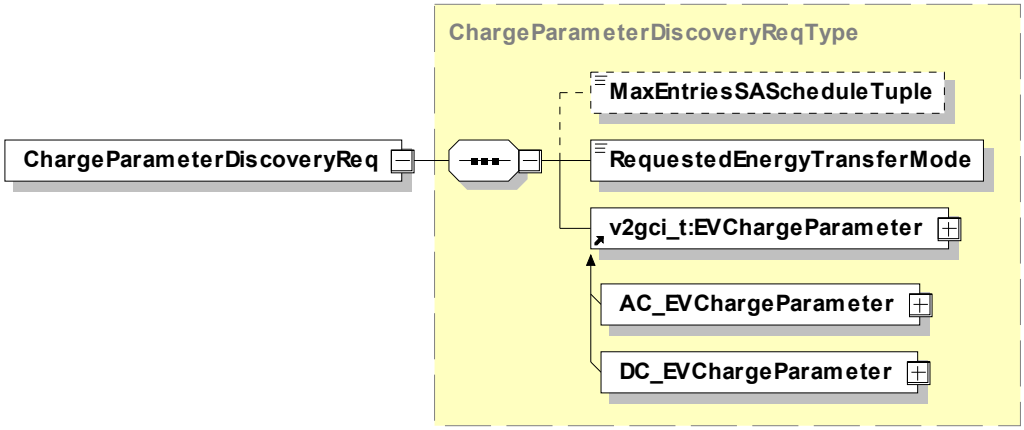


Figure 35 — Schema Diagram – ChargeParameterDiscoveryReq

[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
MaxEntriesSAScheduleTuple	simpleType: unsignedShort refer to Annex C.6 for the type definition	Optional: Indicates the maximal number of entries in the SAScheduleTuple (applies for both Pmax and Tariff). The EVSE can transmit up to the maximum number of entries defined in the parameter.
RequestedEnergyTransferMode	simpleType: EnergyTransferModeType enumeration refer to Annex C.6 for the type definition and the Table 63	Selected energy transfer mode for charging that is requested by the EVCC.
AC_EVChargeParameter	complexType: AC_EVChargeParameterType substitutes abstract type EV_ChargeParameterType refer to subclause 8.5.3.2	This element is used by the EVCC for initiating 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 by the EVCC for initating the target setting process for DC charging.

The definition of EnergyTransferModeType supports the connectors Type 1, 2, and 3 as defined in IEC 62196-2 and the connectors as defined in IEC 62196-3 Annex cc, dd, ee, and ff. Based on the supported connectors the EVCC can choose the charging services as defined in subclause 8.5.2.4.

[V2G2-784] An EVCC shall support 12 entries for PMaxScheduleEntry and SalesTariffEntry elements inside one SAScheduleTuple if MaxEntriesSAScheduleTuple is not transmitted in ChargeParameterDiscoveryReq.

[V2G2-786] An SECC shall support in minimum 12 entries for PMaxScheduleEntry and SalesTariffEntry elements inside one SAScheduleTuple.

[V2G2-785] If an SECC can not provide the number of entries for PMaxScheduleEntry and SalesTariffEntry elements inside one SAScheduleTuple that is required by MAXEntriesSAScheduleTuple, it shall send the maximum number of entries that is supported in SAScheduleTuple.

8.4.3.8.3 ChargeParameterDiscoveryRes

With the ChargeParameterDiscoveryRes 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.

NOTE 1 By "cost", the EVSE may stimulate - not force - the EV to charge in periods where it is advantageous for the grid (e.g. because in these periods, a lot of wind-power is to be expected, etc.)

[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 in Table 104 and according to Figure 36.

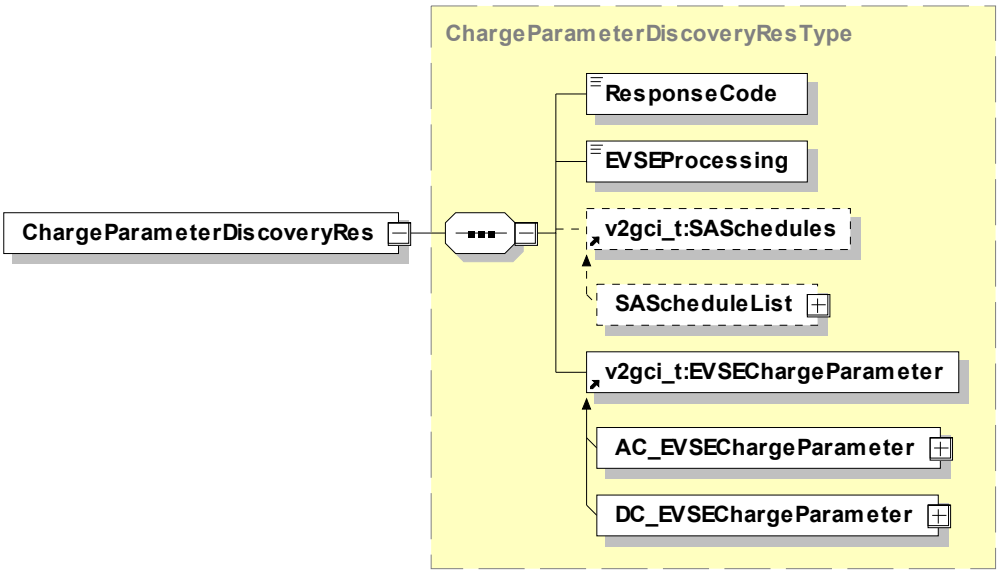


Figure 36 — Schema Diagram – ChargeParameterDiscoveryRes

[V2G2-219] All sales tariffs provided in the SASchedules element shall be originating from the same SA.

[V2G2-220] The message elements of this message shall be used as defined in Table 39.

Table 39 — 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 ChargeParameterDiscoveryReq or that 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	ResponseCode 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	Optional: Includes several tuples of schedules from secondary actors. The SECC shall only omit the parameter 'SAScheduleList' in case EVSEProcessing is set to 'Ongoing'.
AC_EVSEChargeParameter	complexType: AC_EVSEChargeParameterType substitutes abstract type EVSE_ChargeParameterType refer to subclause 8.5.3.3	This element is used by the SECC for initiating 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 by the SECC for initating the target setting process for DC charging.

NOTE 2 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 continuing the communication session while fulfilling the performance and timeout requirements.

8.4.3.9 PowerDeliveryReq/Res

8.4.3.9.1 PowerDeliveryReq/Res Handling

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

[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 ChargeParameterDiscoveryRes message (see subclause 8.4.3.8.3).

8.4.3.9.2 PowerDeliveryReq

By sending the PowerDeliveryReq the EVCC requests the SECC to provide power on and transmits the ChargingProfile the EVCC will follow during the charging process.

NOTE 1 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 in Table 104 and according to Figure 37.

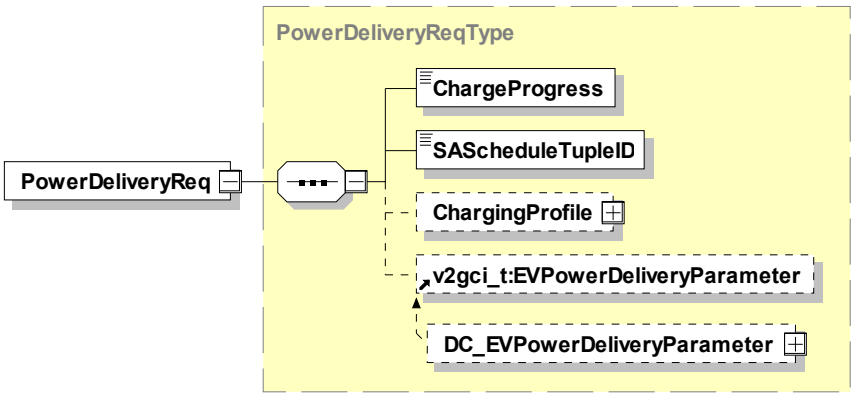


Figure 37 — Schema Diagram – PowerDeliveryReq

[V2G2-222] The message elements of this message shall be used as defined in Table 40.

Table 40 — Semantics and type definition for PowerDeliveryReq

Element Name	Type	Semantics
ChargeProgress	simpleType: chargeProgressType enumeration 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 ChargeProgress is equal to 'Start' the EVSE is requested to prepare the energy flow for an immediate start, if ChargeProgress is equal to 'Stop' the EVSE is requested to stop the energy flow, if ChargeProgress is equal to 'Renegotiate' the energy flow is neither stopped nor started, instead the renegotiation mechanisms defined in this standard apply.
SAScheduleTupleID	simpleType: SAIDType short refer to Annex C.6 for the type definition	Unique identifier within a charging session for a SAScheduleTuple element. An SAID remains a unique identifier for one schedule throughout a charging session.
ChargingProfile	complexType: ChargingProfileType refer to subclause 8.5.2.10	Optional: 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: This element is used by the EVCC for transmitting the parameters for power delivery

NOTE 2 In Part 1 of the this standard the term 'ChargingProfile' is also referred to as 'charging schedule'. Refer to Part 1 for the description of 'charging schedule'.

8.4.3.9.3 PowerDeliveryRes

After receiving the PowerDeliveryReq message of the EVCC the SECC sends the PowerDeliveryRes 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 in Table 104 and according to Figure 38.

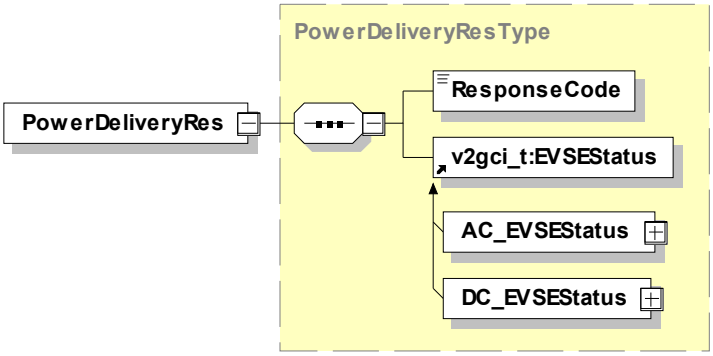


Figure 38 — Schema Diagram – PowerDeliveryRes

[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 last ChargeParameterDiscoveryRes message sent by the SECC (see subclause 8.4.3.8.3).

[V2G2-777] The maximum delay time allowed for the EV to adjust to a new nominal power level (falling edge) during charging shall be 5 seconds. Only if the delay time is exceeded and the power drawn by the EV exceeds the nominal value by more than 10%, the EVSE may drop the EV.

[V2G2-225] The SECC shall send the negative ResponseCode 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 last ChargeParameterDiscoveryRes message sent by the SECC (see subclause 8.4.3.8.3).

[V2G2-226] The message elements of this message shall be used as defined in Table 41.

Table 41 — Semantics and type definition for PowerDeliveryRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	ResponseCode indicating the acknowledgment status of any of the V2G messages received by the SECC.
AC_EVSEStatus	complexType: AC_EVSEStatusType substitutes abstract type EVSEStatusType refer to subclause 8.5.3.1	This element is used by the SECC for indicating the EVSE status and for signalling an event the SECC expects the EVCC to react to.
DC_EVSEStatus	complexType: DC_EVSEStatusType substitutes abstract type EVSEStatusType refer to subclause 8.5.4.1	This element is used by the SECC for indicating the EVSE status and for signalling an event the SECC expects the EVCC to react to.

8.4.3.10 CertificateUpdateReq/Res

8.4.3.10.1 CertificateUpdateReq/Res Handling

Updating the certificate of the EVCC is required when the certificate is still valid but is about to expire. In this use case the EVCC requests new certificate from the SECC to be installed into the EVCC.

[V2G2-227] The EVCC shall request the required certificate at the SECC, provided the SECC offers this service as previously indicated in the element ServiceList in the message ServiceDiscoveryRes.

NOTE 1 If the SECC is not able to deliver the requested certificate an appropriate error handling has to be performed, see **[V2G2-471]**. 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 ContractSignatureCertChain is not corrupted; i.e. they shall 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 CertificateInstallationReq and CertificateInstallationRes.

8.4.3.10.2 CertificateUpdateReq

By sending the CertificateUpdateReq the vehicle requests the SECC to deliver new certificate that belongs 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 in Table 104 and according to Figure 39.

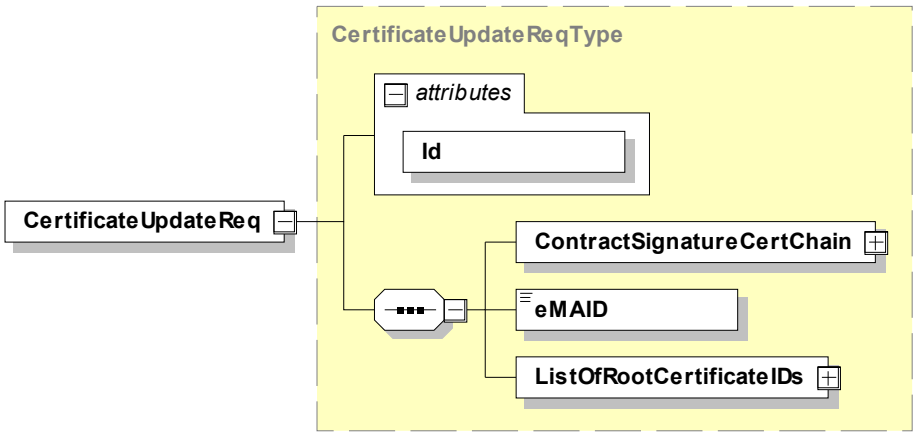


Figure 39 — Schema Diagram –CertificateUpdateReq

[V2G2-229] The message elements of this message shall be used as defined in Table 42.

[V2G2-888] The EVCC shall sign the body element of CertificateUpdateReq using the Contract Certificate for which this update is requested.

[V2G2-889] The Provisioning Service shall verify the signature of the body element of CertificateUpdateReq using the included Contract Certificate from the field ContractSignatureCertChain. The Provisioning Service shall validate said Certificate including the chain from the same field and ensure that it traces back to a trusted Mobility Operator Root CA. If any of the aforementioned fails, the Certificate Provisioning Service shall regard the signature as invalid, abort, and induce the SECC to return an error.

Table 42 — Semantics and type definition for CertificateUpdateReq

Element Name	Type	Semantics
Id	simpleType: ID string refer to Annex C.6 for the type definition	This element is used for referencing the entire message body in the message header when a signature needs to be applied.
ContractSignatureCertChain	complexType: CertificateChainType 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.
eMAID	simpleType: eMAIDType string (min length: 14, max length: 15) refer to Annex C.6 for the type definition	This element identifies the charging contract. The format is defined in Annex H.
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.

8.4.3.10.3 CertificateUpdateRes

After receiving the CertificateUpdateReq of the EVCC the SECC retrieves the requested certificate from the SA. It therefore needs to establish an online connection. Then the SECC sends the CertificateUpdateRes 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 in Table 104 and according to Figure 40.

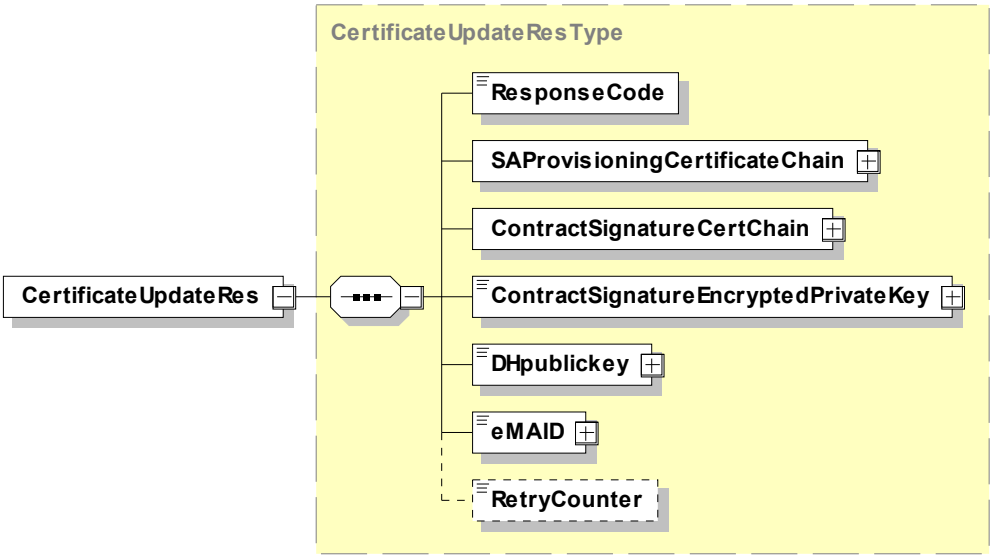


Figure 40 — Schema Diagram – CertificateUpdateRes

[V2G2-232] The message elements of this message shall be used as defined in Table 43.

[V2G2-928] If the ResponseCode in this messages indicates failure (i.e. starts with FAILED), the EVCC shall ignore all other elements except the RetryCounter.

Table 43 — Semantics and type definition for CertificateUpdateRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	ResponseCode indicating the acknowledgment status of any of the V2G messages received by the SECC.
SAProvisioningCertificateChain	complexType: CertificateChainType refer to subclause 8.5.2.5	The transmitted certificate chain is used by the EVCC to verify the signature in the message header.
ContractSignatureCertChain	complexType: CertificateChainType refer to subclause 8.5.2.5	The new certificate chain for signature purposes that has to be installed in the EVCC.
ContractSignatureEncryptedPrivateKey	complexType: ContractSignatureEncryptedPrivateKeyType refer to subclause 8.5.2.28	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 CertificateUpdateReq) and using the calculated DH secret for encryption as described in 7.9.2.4.3.
DHpublickey	complexType: DiffieHellmanPublickeyType refer to 8.5.2.29 for the type definition	Diffie Hellman ephemeral public key parameter from the SA for generating the session key at the EVCC in order to encrypt the Contract Signature Private Key at the SA and decrypt it a the EVCC.
eMAID	complexType: EMAIDType refer to 8.5.2.30 for the type definition	This element identifies the charging contract. The format is defined in Annex H.
RetryCounter	simpleType short refer to Annex C.6 for the type definition	Optional: 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..

[V2G2-696] When there is a failed response, the response message shall also provide the RetryCounter element.

NOTE 1 The SECC offers the service for certificate installation or update only in case it has connectivity to the issuing mobility operator.

NOTE 2 If the SECC is not able to deliver the requested certificate an appropriate error handling has to be performed, see **[V2G2-469]**. 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 3 It is only allowed to use the Certificate Update (as described here) if the ContractSignatureCert was not corrupted; i.e. they shall 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 CertificateInstallationReq and CertificateInstallationRes.

[V2G2-827] A contract certificate intended to replace a contract certificate already installed in a vehicle shall be usable (validity period) by an EV as soon as it has been transmitted with the CertificateUpateRes.

NOTE 4 This is to prevent that the EVCC is required to store and manage two contract certificates at once.

[V2G2-890] The Certificate Provisioning Service shall sign the following elements of the CertificateUpdateRes Message: ContractSignatureCertChain, ContractSignatureEncryptedPrivateKey, DHpublickey, eMAID. It shall include the certificate chain for verification of this signature in the field SAProvisioningCertChain. The Certificate Provisioning Service also shall verify that the data, which is covered by the signature, is received from the preceding Secondary Actor in an authentic manner.

[V2G2-891] The EVCC shall verify the signature of the CertificateUpdateRes Message using the signer certificate chain SAProvisioningCertChain, validate said chain and ensure that it traces back to a valid V2G Root Certificate. It shall ensure that the signer certificate has a DC (DomainComponent) field with the content "CPS" set. It shall ensure that exactly the following elements are included in the signature: ContractSignatureCertChain, ContractSignatureEncryptedPrivateKey, DHpublickey, ContractID. The EVCC shall validate the signer certificate and ensure that it traces back to a trusted V2G Root Certificate. If any of the aforementioned fails, the EVCC shall regard the message as invalid, and discard the message. Else, the EVCC shall store the parts of the certificate chain that it does not already have.

[V2G2-892] The ContractSignatureCertChain received in the CertificateUpdateRes Message shall be stored persistently in such a way, that it can be applied later on for the purpose of verifying tariff tables. The Mobility Operator Sub-CA2 certificate is required to verify tariff tables.

NOTE 5 It is not required that the EVCC is able to verify the certificate it has received. It is the task of the Certificate Provisioning Service to provide only valid certificates.

8.4.3.11 CertificateInstallationReq/Res

8.4.3.11.1 CertificateInstallationReq/Res Handling

Installing the contract certificate into the EVCC is required if EVCC currently does not possess a valid contract certificate; e.g. because no contract certificate is stored, or existing contract certificates are expired or revoked. 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 standard. For details on OEM Provisioning Certificates refer to Annex 0.

[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). If the EVCC does not possess a valid certificate assigned with its contract, it is not able to use plug and charge until a certificate installation process is performed successfully.

NOTE 3 For a more detailed explanation of the certificate installation process, see Annex 0.

8.4.3.11.2 CertificateInstallationReq

By sending the CertificateInstallationReq the vehicle requests the SECC to deliver the certificate 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 in Table 104 and according to Figure 41.

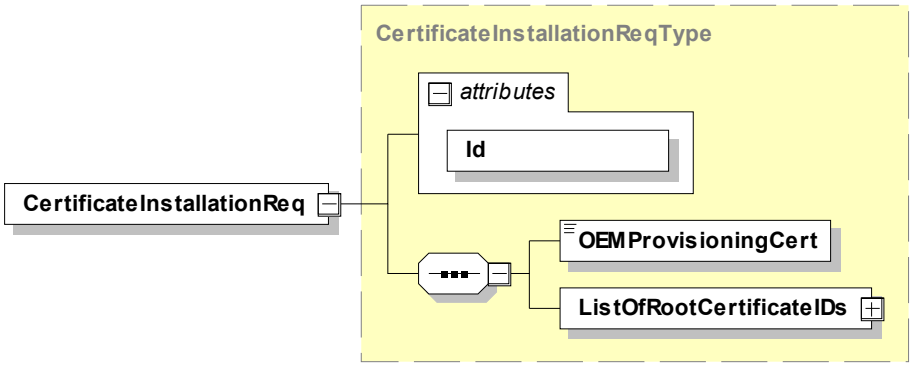


Figure 41 — Schema Diagram – CertificateInstallationReq

[V2G2-236] The message elements of this message shall be used as defined in Table 44.

Table 44 — Semantics and type definition for CertificateInstallationReq

Element Name	Type	Semantics
Id	simpleType: ID string refer to Annex C.6 for the type definition	This element is used for referencing the entire message body in the message header when a signature needs to be applied.
OEMProvisioningCert	simpleType certificateType base64Binary (max. length 800) 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 at the SA (contract partner) is 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 in the CertificateInstallationRes later on. The certificate is DER encoded. Please refer also to Annex E.1 and Annex 0.
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.

[V2G2-893] The EVCC shall sign the body element of CertificateInstallationReq using its OEM provisioning certificate.

[V2G2-894] The Certificate Provisioning Service shall verify the signature of the body element of CertificateInstallationReq using the included OEM Provisioning Certificate from the field OEMProvisioningCert. The Provisioning Service shall validate said OEM Provisioning Certificate including the chain from the same field and ensure that it traces back to a trusted OEM Root Certificate. It shall verify, that the OEM Provisioning Certificate has the DC (DomainComponent) "OEM" set. If any of the aforementioned fails, the Certificate Provisioning Service shall regard the signature as invalid, abort, and induce the SECC to return an error.

8.4.3.11.3 CertificateInstallationRes

After receiving the CertificateInstallationReq from the EVCC, the SECC sends the CertificateInstallationRes 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 in Table 104 and according to Figure 42.

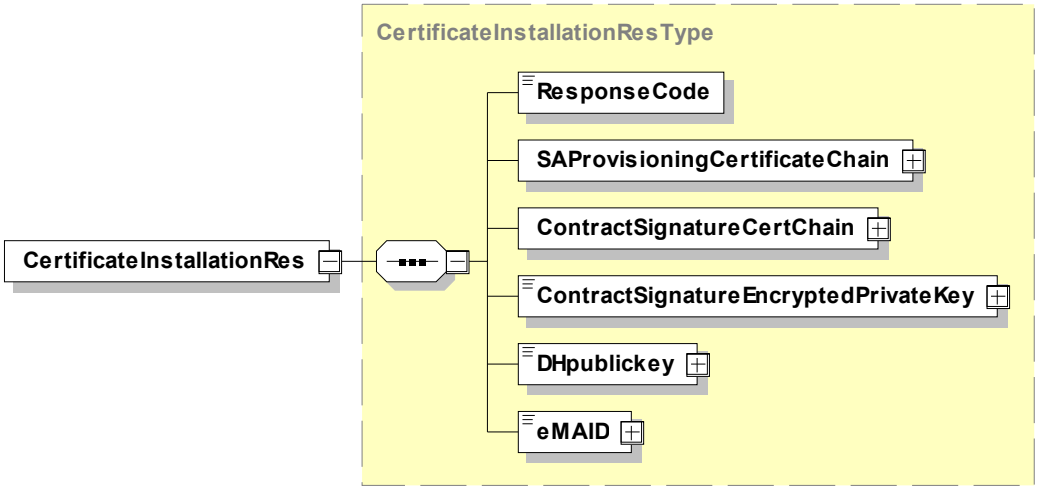


Figure 42 — Schema Diagram – CertificateInstallationRes

[V2G2-238] The message elements of this message shall be used as defined in Table 45.

Table 45 —Semantics and type definition for CertificateInstallationRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	ResponseCode indicating the acknowledgment status of any of the V2G messages received by the SECC.
SAProvisioningCertificateChain	complexType: CertificateChainType refer to subclause 8.5.2.5	The transmitted certificate chain is used by the EVCC to verify the signature in the message header.
ContractSignatureCertChain	complexType: CertificateChainType refer to subclause 8.5.2.5	The new certificate chain for signature purposes that has to be installed in the EVCC.
ContractSignatureEncryptedPrivateKey	complexType: ContractSignatureEncryptedPrivateKeyType refer to subclause 8.5.2.28	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 provided OEM Provisioning Certificate of the EVCC (based on the OEMProvisioningCert contained in message Certificate Installation Request) and using the calculated DH secret for encryption as described in 7.9.2.4.3
DHpublickey	complexType: DiffieHellmanPublickeyType refer to 8.5.2.29 for the type definition	Diffie Hellman public key from the SA for generating the session key at the EVCC in order to encrypt the Contract Signature Private Key at the SA and decrypt it a the EVCC.
eMAID	complexType: EMAIDType refer to 8.5.2.30 for the type definition	This element identifies the charging contract. The format is defined in Annex H.

[V2G2-895] The Certificate Provisioning Service shall sign the following elements of the CertificateInstallationRes Message using the Provisioning Service Certificate: ContractSignatureCertChain, ContractSignatureEncryptedPrivateKey, DHpublickey, ContractID. It shall include the certificate chain for verification of this signature in the field SAProvisioningCertChain. The Certificate Provisioning Service also shall verify that the data, which is covered by the signature, is received from the preceding Secondary Actor in an authentic manner.

[V2G2-896] The EVCC shall verify the signature of the CertificateInstallationRes Message using the signer certificate chain SAProvisioningCertificateChain, validate said chain, and ensure that it traces back to a valid V2G Root Certificate. It shall ensure that the signer certificate has a DC (DomainComponent) field with the content "CPS" set. It shall ensure that exactly the following elements are included in the signature: ContractSignatureCertChain, ContractSignatureEncryptedPrivateKey, DHpublickey, ContractID. If any of the aforementioned fails, the EVCC shall regard the message as invalid, and discard the message. Else, the EVCC shall store the parts of the certificate chain that it does not already have.

[V2G2-897] The ContractSignatureCertChain received in the CertificateInstallationRes Message shall be stored persistently in such a way, that it can be applied later on for the purpose of verifying tariff tables. The Mobility Operator Sub-CA 2 certificate is required to verify tariff tables.

NOTE It is not required that the EVCC is able to verify the certificate it has received. It is the task of the Certificate Provisioning Service to provide only valid certificates.

8.4.3.11.4 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 CertificateInstallationReq / 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 (similarly 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).

8.4.3.12 SessionStopReq/Res

8.4.3.12.1 SessionStopReq/Res Handling

This V2G message pair shall be used for terminating a V2G Communication Session initiated by preceding SessionSetupReq message.

8.4.3.12.2 SessionStopReq

By sending the SessionStopReq 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 in Table 104 and according to Figure 43.

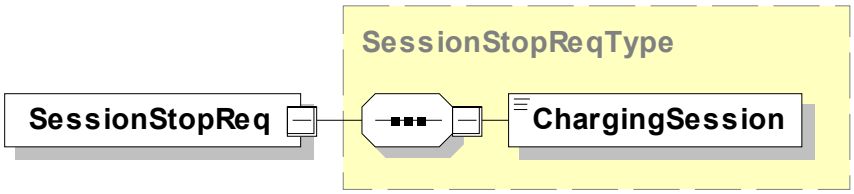


Figure 43 — Schema Diagram – SessionStopReq

[V2G2-738] The message elements of this message shall be used as defined in Table 46 — Semantics and type definition for SessionStopReq

Table 46 — Semantics and type definition for SessionStopReq

Element Name	Type	Semantics
ChargingSession	simpleType: ChargingSessionType enumeration refer to Annex C.6 for the type definition	ChargingSession indicates the intention of the EVCC to either pause or terminate a V2G Communication Session.

8.4.3.12.3 SessionStopRes

After receiving the SessionStopReq of the EVCC the SECC sends the SessionStopRes 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 in Table 104 and according to Figure 44.

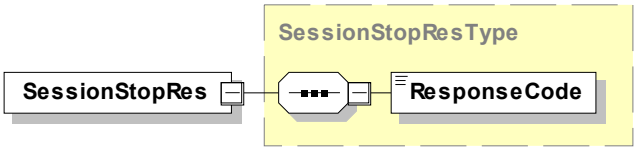


Figure 44 — Schema Diagram – SessionStopRes

[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	ResponseCode indicating the acknowledgment status of any of the V2G messages received by the SECC.

8.4.3.13 MeteringReceiptReq/Res

8.4.3.13.1 MeteringReceiptReq/Res Handling

When sending a MeteringReceiptReq message the EVCC acknowledges that the data elements MeterInfo record, SessionID and the SAScheduleTupleID included in the ChargingStatusRes message prior to this request have been received from the SECC. This confirmation is implemented by applying a signature to the message body of the MeteringReceiptReq message. The signature applied by the EVCC is intended only to confirm that the EVCC, together with the currently applied charging contract certificate, which was selected for this charging session, has received the data elements MeterInfo record, SessionID and the SAScheduleTupleID included in the Charging Status Res message prior to this request. It is not intended to confirm that the amount of energy indicated in the previous MeterInfo record is correct in the context of billing. However, the signed meter info record might be used by any mobility operator for billing purposes in general. It is out of scope of this standard to define the billing process which is subject to local regulations.

[V2G2-902] The element MeterInfo sent by SECC in the message ChargingStatusRes shall in any case be exactly the same data which the meter puts out. It shall contain the raw meter reading in a form which is in general machine readable.

NOTE This would allow the EVCC in general to analyse the meter reading, if desired.

8.4.3.13.2 MeteringReceiptReq

When sending the MeteringReceiptReq 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 of 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 in Table 104 and according Figure 45.

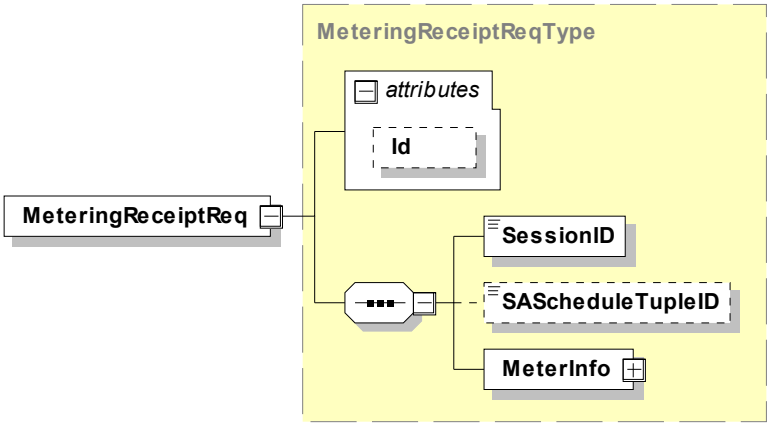


Figure 45 — Schema Diagram – MeteringReceiptReq

[V2G2-246] The message elements of this message shall be used as defined in Table 48.

Table 48 — Semantics and type definition for MeteringReceiptReq

Element Name	Type	Semantic
Id	simpleType: ID string refer to Annex C.6 for the type definition	Optional: This attribute is used for referencing the message body in the signature header when a signature needs to be applied.
SessionID	simpleType: SessionIDType hexBinary (max length: 8) refer to Annex C.6 for the type definition	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 either in the ChargingStatusRes (during AC charging) or the CurrentDemandRes (during DC charging) message from the SECC. An SAID remains a unique identifier for one schedule throughout a charging session.
MeterInfo	complexType MeterInfoType refer to subclause 8.5.2.6.	If the SECC indicated in the ChargingStatusRes (AC charging)/ CurrentDemandRes (DC charging) that a MeteringReceiptReq is required this message element is the echo of the MeterInfo record received in the ChargingStatusRes (AC charging)/ CurrentDemandRes (DC charging) from the SECC.

[V2G2-776] If included, the message element SAScheduleTupleID shall always be equal to the SAScheduleTupleID value received in the preceding ChargingStatusRes (AC charging)/ CurrentDemandRes (DC charging) message.

[V2G2-903] The EVCC shall sign the body MeteringReceiptReq message using the private key belonging to the Contract Certificate it has sent in PaymentDetailsReq in this session.

[V2G2-904] The SECC/SA may verify the signature of the body element of the MeteringReceiptReq message using the Contract Certificate it has received in PaymentDetailsReq in this session. If this verification fails, the metering receipt shall be regarded as invalid.

NOTE 1 The SECC may submit the Contract Certificate to a Secondary Actor along with the signed MeteringReceipt, so that the Secondary Actor can later re-verify the receipt. In case the Secondary Actor has stored the Contract Certificate already, an ID and the issuing CA for the Contract Certificate may suffice.

NOTE 2 The EVCC might not be able to verify the signature SigMeterReading, which was generated by the SECC and which is included in the MeterInfo Element, e.g. because some data necessary for the signature is not transmitted. Therefore the signature applied by the EVCC in this subclause only serves as an acknowledgement that the EVCC has seen this data, and does not constitute any assertion of correctness.

8.4.3.13.3 MeteringReceiptRes

After receiving the MeteringReceiptReq from the EVCC the SECC sends the MeteringReceiptRes 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 in Table 104 and according to Figure 46.

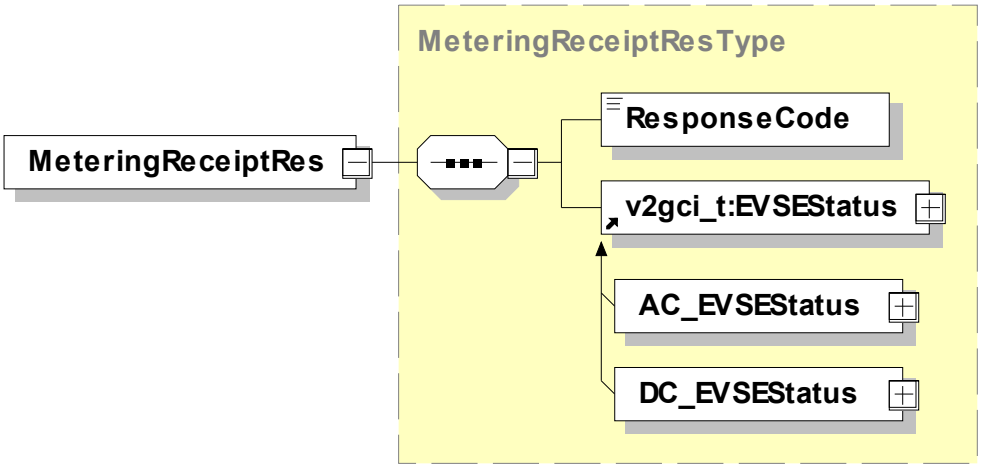


Figure 46 — Schema Diagram – MeteringReceiptRes

[V2G2-248] The message elements of this message shall be used as defined in Table 49.

Table 49 — Semantics and type definition for MeteringReceiptRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	ResponseCode indicating the acknowledgment status of any of the V2G messages received by the SECC.
AC_EVSEStatus	complexType: AC_EVSEStatusType substitutes abstract type EVSEStatusType refer to subclause 8.5.3.1	This element is used by the SECC for indicating the EVSE status and for signalling an event the SECC expects the EVCC to react to.
DC_EVSEStatus	complexType: DC_EVSEStatusType substitutes abstract type EVSEStatusType refer to subclause 8.5.4.1	This element is used by the SECC for indicating the EVSE status and for signalling an event the SECC expects the EVCC to react to.

8.4.4 AC-Messages

8.4.4.1 Overview

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

8.4.4.2 ChargingStatusReq/Res

8.4.4.2.1 ChargingStatusReq/Res Handling

The Charging Status message pair provides sanity checks on the meter readings provided by the SECC. 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 ChargingStatusRes 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).

8.4.4.2.2 ChargingStatusReq

[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 in Table 104 and according to Figure 47.

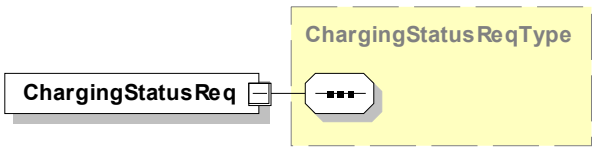


Figure 47 — Schema Diagram – ChargingStatusReq

8.4.4.2.3 ChargingStatusRes

After receiving the ChargingStatusReq from the EVCC, the SECC sends the ChargingStatusRes. 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 reading of the meter failed no meter reading are provided. The failure is indicated by the ResponseCode.

[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 in Table 104 and according to Figure 48.

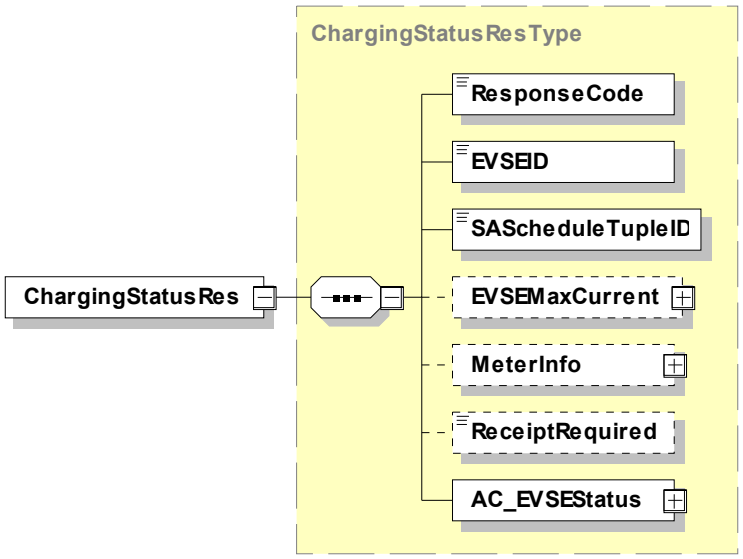


Figure 48 — Schema Diagram – ChargingStatusRes

[V2G2-244] The message elements of this message shall be used as defined in Table 50.

Table 50 — Semantics and type definition for ChargingStatusRes

Element Name	Type	Semantics
ResponseCode	simpleType: responseCodeType enumeration refer to Annex C.6 for the type definition	ResponseCode indicating the acknowledgment status of any of the V2G messages received by the SECC.
EVSEID	simpleType: evseIDType string (min length: 7, max length:37) refer to Annex C.6 for the type definition	Any ID that uniquely identifies the EVSE and the power outlet the vehicle is connected to. The format of this message element is defined in Annex H. If an SECC cannot provide such ID data, the value of the EVSEID is set to zero ("ZZ00000").
SAScheduleTupleID	simpleType: SAIDType short refer to Annex C.6 for the type definition	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. An SAID remains a unique identifier for one schedule throughout a charging session.
EVSEMaxCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: This element is used by the SECC to indicate the maximum line current per phase 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	Optional: This element is used by the SECC to indicate that the EVCC is required to send 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 False the EVCC is not required to send a MeteringReceiptReq message.
AC_EVSEStatus	complexType: AC_EVSEStatusType substitutes abstract type EVSEStatusType refer to subclause 8.5.3.1	This element is used the by the SECC for indicating the SECC status.

8.4.5 DC-Messages

8.4.5.1 Overview

Messages defined as DC-Messages belong to the DC Message Set(s) (for details refer to 8.6.2.4)

8.4.5.2 CableCheckReq/Res

8.4.5.2.1 CableCheckReq/Res Handling

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

8.4.5.2.2 CableCheckReq

The CableCheckReq 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 in Table 104 and according to Figure 49.

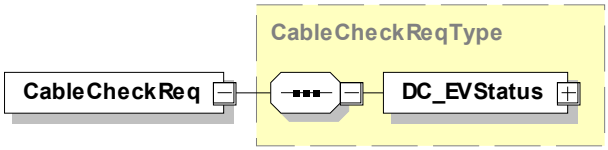


Figure 49 — Schema Diagram – CableCheckReq

[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

8.4.5.2.3 CableCheckRes

After receiving the CableCheckReq from the EVCC the SECC sends the CableCheckRes 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 in Table 104 and according to Figure 50.

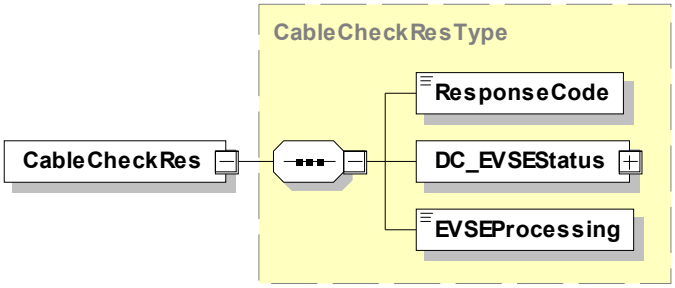


Figure 50 — Schema Diagram – CableCheckRes

[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 CableCheckReq 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	ResponseCode 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 continuing the communication session while fulfilling the performance and timeout requirements.

8.4.5.3 PreChargeReq/Res

8.4.5.3.1 PreChargeReq/Res Handling

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

8.4.5.3.2 PreChargeReq

The PreChargeReq 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 in Table 104 and according to Figure 51.

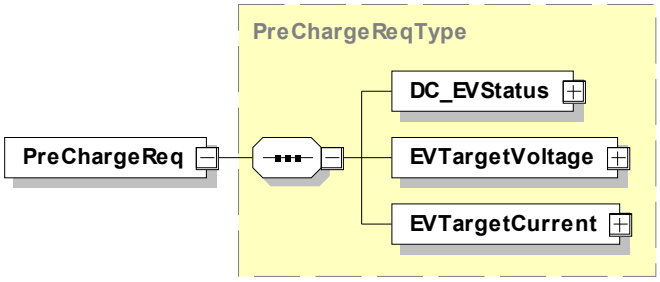


Figure 51 — Schema Diagram – PreChargeReq

[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

8.4.5.3.3 PreChargeRes

After receiving the PreChargeReq from the EVCC the SECC sends the PreChargeRes 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 in Table 104 and according to Figure 52.

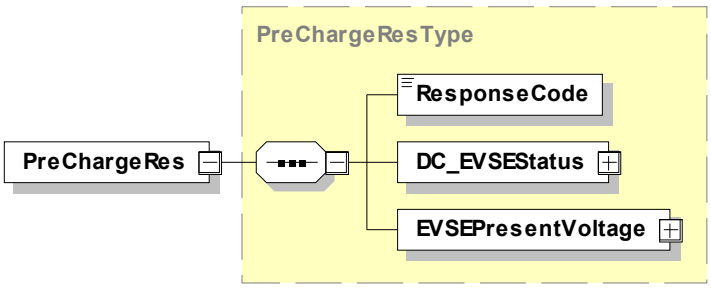


Figure 52 — Schema Diagram – PreChargeRes

[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	ResponseCode 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	Output voltage of the EVSE as defined in IEC CDV 61851-23.

8.4.5.4 CurrentDemandReq/Res

8.4.5.4.1 CurrentDemandReq/Res 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.

8.4.5.4.2 CurrentDemandReq

By sending the CurrentDemandReq 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 in Table 104 and according to Figure 53.

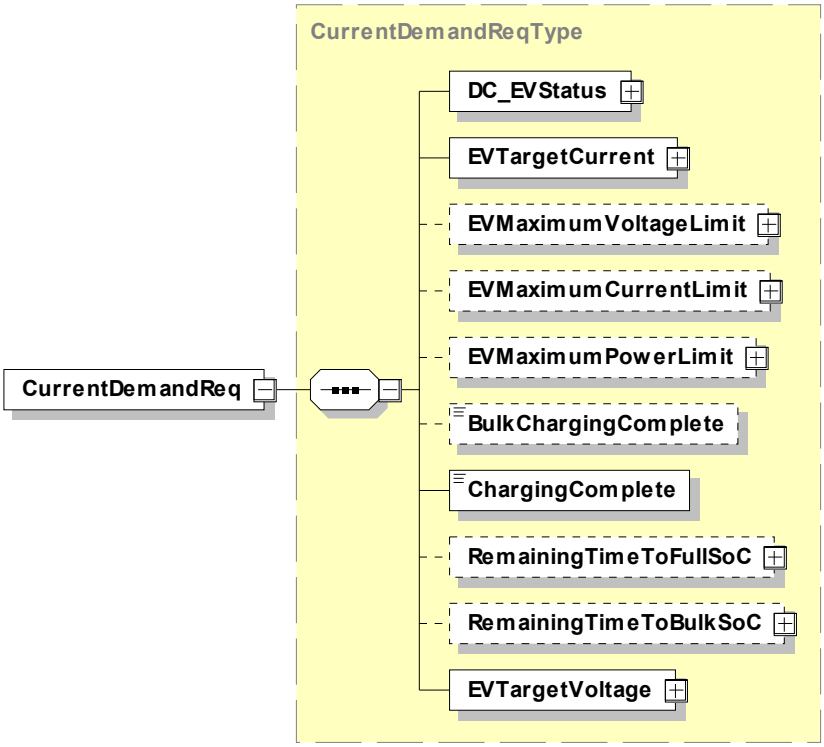


Figure 53 — Schema Diagram – CurrentDemandReq

[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
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.

8.4.5.4.3 CurrentDemandRes

After receiving the CurrentDemandReq from the EVCC the SECC sends the CurrentDemandRes 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 in Table 104 and according to Figure 54.

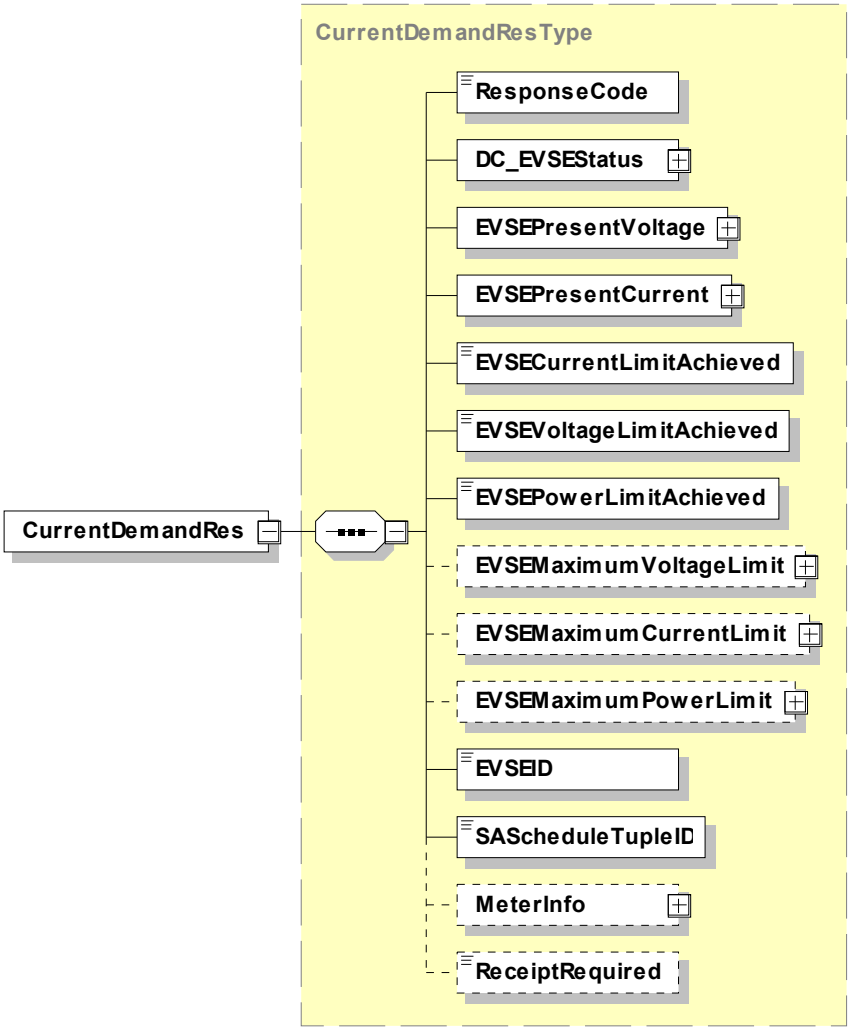


Figure 54 — Schema Diagram – CurrentDemandRes

[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	ResponseCode 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

Element Name	Type	Semantics
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
EVSEID	simpleType: evseIDType hexBinary (max length:32) refer to Annex C.6 for the type definition	Any ID that uniquely identifies the EVSE and the power outlet the vehicle is connected to. The format of this message element is defined in Annex H.2. 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	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. An SAID remains a unique identifier for one schedule throughout a charging session.
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	Optional: This element is used by the SECC to indicate that the EVCC is required to send 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 a signature. If ReceiptRequired is equal to False the EVCC is not required to send a MeteringReceiptReq message.

8.4.5.5 WeldingDetectionReq/Res

8.4.5.5.1 WeldingDetectionReq/Res Handling

The Welding Detection messages described in this subclause allow to implement the Welding Detection mechanism according to IEC CDV 61851-23.

8.4.5.5.2 WeldingDetectionReq

By sending the WeldingDetectionReq 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 in Table 104 and according to Figure 55.

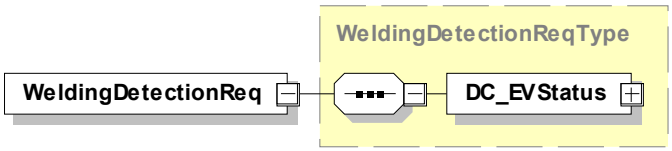


Figure 55 — Schema Diagram – WeldingDetectionReq

[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

8.4.5.5.3 Welding Detection Response

After receiving the WeldingDetectionReq 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 in Table 104 and according to Figure 56.

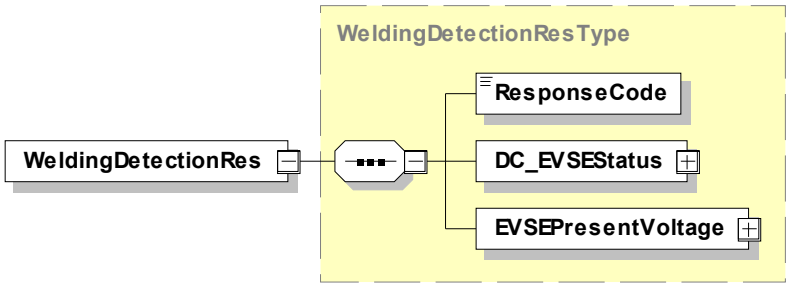


Figure 56 — Schema Diagram – WeldingDetectionRes

[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	ResponseCode 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 clause 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 ServiceType

This Type represents 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 104 and according to Figure 57.

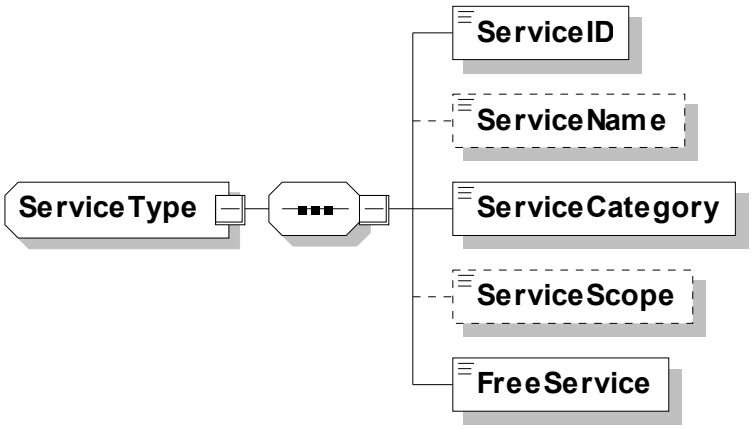


Figure 57 — Schema Diagram – ServiceType

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

Table 59 —Semantics and type definition for ServiceType

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: serviceCategoryType enumeration 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: 64) refer to Annex C.6 for the type definition	Optional: Additional information about usage of that 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.

8.5.2.2 ServiceListType

[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 104 and according to Figure 58.

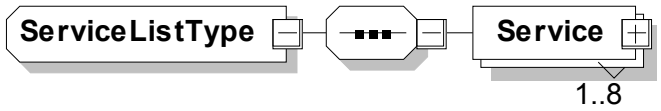


Figure 58 — Schema Diagram – ServiceListType

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

Table 60 — Semantics and type definition for ServiceListType

Element Name	Type	Semantics
Service	complexType: serviceType refer to subclause	Contains all information for identifying a service. The ServiceList includes at least one Service and may include up to 8 Services,

8.5.2.3 ChargeServiceType

EV charging specific service derived from ServiceType (refer to subclause 8.5.2.1) which contains additional information about SupportedEnergyTransferMode(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 104 and according to Figure 59 — Schema Diagram – ChargeServiceType

.

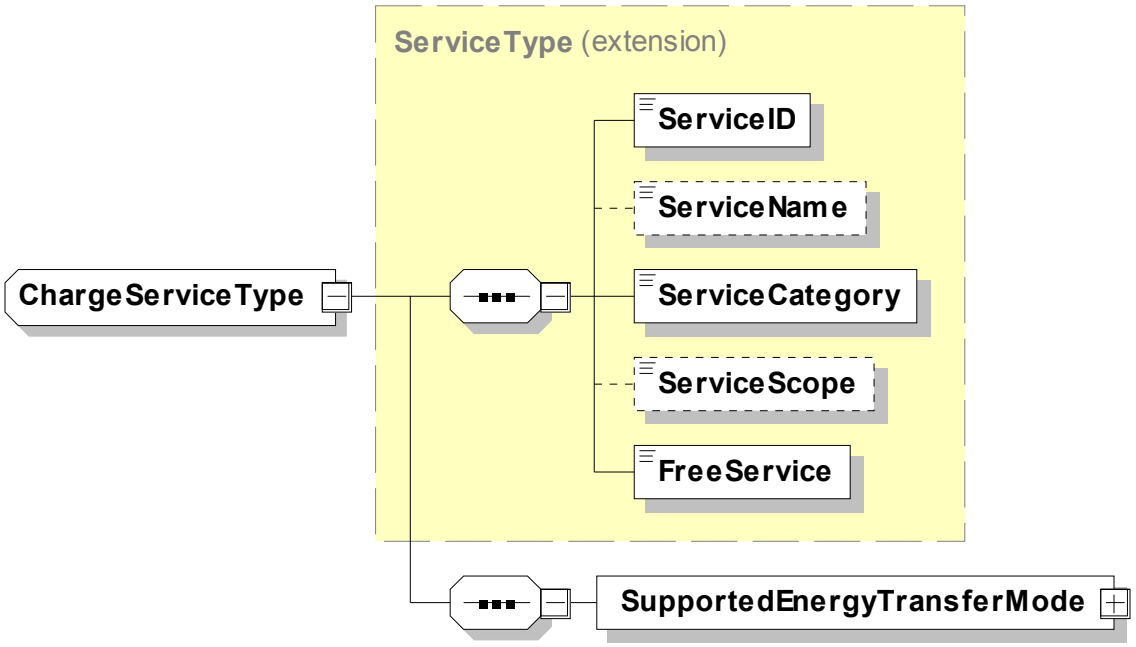


Figure 59 — Schema Diagram – ChargeServiceType

[V2G2-272] The message element shall be used as defined in Table 61.

Table 61 — Semantics and type definition for ChargeServiceType

Element Name	Type	Semantics
ServiceType (extension)	complexType: ServiceType refer to 8.5.2.1 for the type definition	This set of message elements is inherited from ServiceType. Refer to Table 59 for the semantic description of these message elements.
SupportedEnergyTransferMode	complexType: SupportedEnergyTransferModeType refer 8.5.2.4 for the type definition	Available energy transfer modes supported by the EVSE.

The definition and usage of the EnergyTransferModeType within the SupportedEnergyTransferModeType supports the connectors Type 1, 2, and 3 as defined in IEC 62196-2 and the connectors as defined in IEC 62196-3 Annex cc, dd, ee, and ff.

8.5.2.4 SupportedEnergyTransferModeType

[V2G2-757] 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 62 and according to Figure 59.

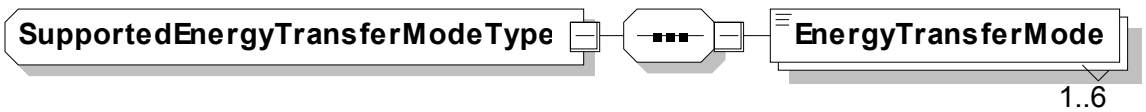


Figure 60 — Schema Diagram – SupportedEnergyTransferModeType

[V2G2-758] The message element shall be used as defined in Table 62.

Table 62 — Semantics and type definition for SupportedEnergyTransferModeType

Element Name	Type	Semantics
EnergyTransferMode	simpleType: EnergyTransferModeType enumeration refer to Annex C.6 for the type definition and Table 63.	Energy transfer mode for charging that is supported by the SECC. Refer to Table 63 for details. Max occurrence is 6.

[V2G2-759] The EVCC and the SECC shall use the EnergyTransferModeType as described in Table 63.

Table 63 — Semantics for EnergyTransferModeType

EnergyTransferMode	Offered charging service
AC_single_phase_core	AC single phase charging according to IEC 62196.
AC_three_phase_core	AC three phase charging according to IEC 62196.
DC_core	DC charging according to IEC 62196 on the core pins.
DC_extended	DC charging using the extended pins of an IEC 62196-3 Configuration EE or Configuration FF connector.
DC_combo_core	DC charging using the core pins of an IEC 62196-3 Configuration EE or Configuration FF connector.
DC_unique	DC charging using a dedicated DC coupler.

NOTE The SECC may provide multiple options for charge services. Depending on these options, the EVCC has to select one of the offered options. For example, if the EVSE offers AC_single_phase_core and DC_core, the EVCC has to select either AC_single_phase_core or DC_core because both options can technically not be supported at the same time (refer to EnergyTransferModeType, 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 104 and according to Figure 61.

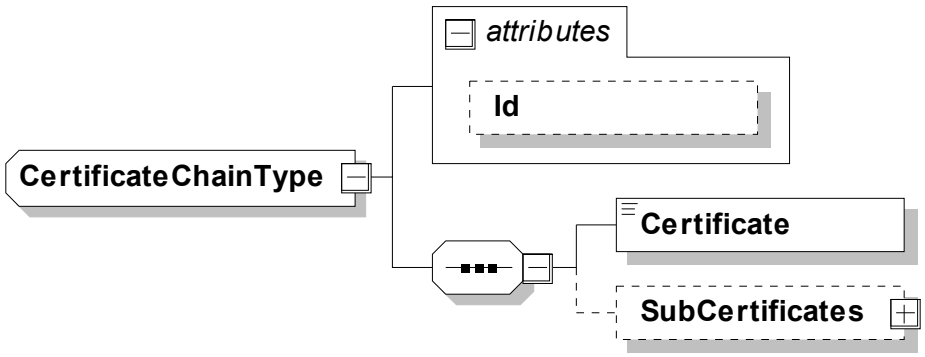


Figure 61 — Schema Diagram – CertificateChainType

[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
Id	simpleType: ID string refer to Annex C.6 for the type definition	Optional: This attribute is used for referencing the message element in the signature header when a signature needs to be applied. It is used in response messages only.
Certificate	simpleType: certificateType base64Binary (max length: 800) refer to Annex C.6 for the type definition	An x.509v3 Certificate (the "client" certificate). The certificate is DER encoded.
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 104 and according to Figure 62.

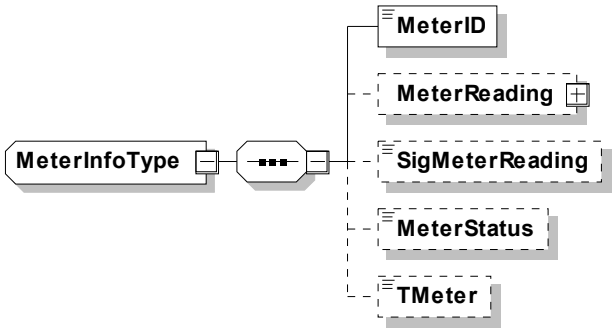


Figure 62 — Schema Diagram – MeterInfoType

[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	simpleType: unsignedLong refer to Annex C.6 for the type definition	Optional: Current meter reading in Watthours from the EVSE
SigMeterReading	simpleType: sigMeterReadingType base64Binary (max length: 64) refer to Annex C.6 for the type definition	Optional: 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: Current status of the meter. The definition of the content of the MeterStatus is out of scope of the standard. The content may be defined by the EVSE operator or utility.

Element Name	Type	Semantics
TMeter	simpleType short refer to Annex C.6 for the type definition	Optional: Timestamp of the current Meter time using the Unix Time Stamp format.

[V2G2-830] When processing the message element MeterReading the SECC and the EVCC shall apply the base unit Watt-hour (Wh).

[V2G2-831] When using the message element MeterReading the SECC and the EVCC shall apply the value range definition as defined in Table 66.

Table 66 — Value range definition of message element MeterReading

Element Name	Unit symbol	Physical unit	Minimum value	Maximum value
MeterReading	Wh	Watt-hour	0	999,999,999

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 104 and according to Figure 63.

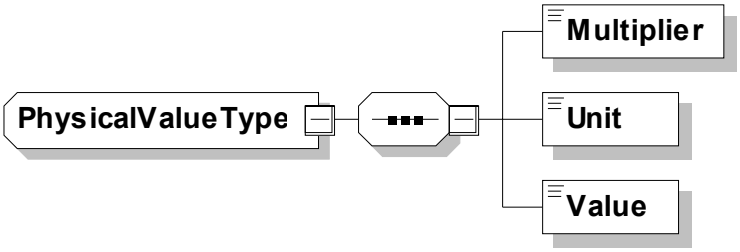


Figure 63 — Schema Diagram – PhysicalValueType

[V2G2-279] The message element shall be used as defined in Table 67.

Table 67 — 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	The Multiplier defines the exponent to base 10 (dec). The final physical value is determined by: Value * 10 ^ Multiplier [Unit]..
Unit	simpleType: unitSymbolType enumeration refer to Annex C.6 for the type definition	Unit of the value
Value	simpleType short refer to Annex C.6 for the type definition	Value which has to be multiplied

[V2G2-832] For all message elements of type PhysicalValueType the SECC and the EVCC shall apply the value range and unit definition as according to Table 68.

Table 68 — Value range and unit definition for message elements using PhysicalValueType

Element Name	unitSymbolType	Physical unit	Minimum value	Maximum value
ChargingProfileEntryMaxPower	W	Watt	0	200000
EAmount	Wh	Watt-hour	0	200000
EVEnergyCapacity	Wh	Watt-hour	0	200000
EVEnergyRequest	Wh	Watt-hour	0	200000
EVMaxCurrent	A	Ampere	0	400
EVMaximumCurrentLimit	A	Ampere	0	400
EVMaximumPowerLimit	W	Watt	0	200000
EVMaximumVoltageLimit	V	Volt	0	1000
EVMaxVoltage	V	Volt	0	1000
EVMinCurrent	A	Ampere	0	400
EVSECurrentRegulationTolerance	A	Ampere	0	400
EVSEEnergyToBeDelivered	Wh	Watt-hour	0	200000
EVSEMaxCurrent	A	Ampere	0	400
EVSEMaximumCurrentLimit	A	Ampere	0	400
EVSEMaximumPowerLimit	W	Watt	0	200000
EVSEMaximumVoltageLimit	V	Volt	0	1000
EVSENominalVoltage	V	Volt	0	1000
EVSEMinimumCurrentLimit	A	Ampere	0	400
EVSEMinimumVoltageLimit	V	Volt	0	1000
EVSEPeakCurrentRipple	A	Ampere	0	400
EVSEPresentCurrent	A	Ampere	0	400
EVSEPresentVoltage	V	Volt	0	1000
EVTargetCurrent	A	Ampere	0	400
EVTargetVoltage	V	Volt	0	1000
PMax	W	Watt	0	200000
RemainingTimeToBulkSoC	s	Seconds	0	172800
RemainingTimeToFullSoC	s	Seconds	0	172800
startValue	W	Watt	0	200000

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 element shall be implemented according to Figure 64.

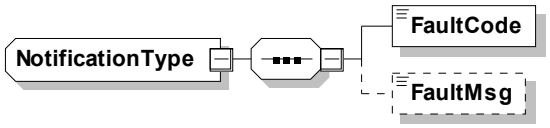


Figure 64 — Schema Diagram – NotificationType

[V2G2-281] The message element shall be used as defined in Table 69.

Table 69 — Semantics and type definition for NotificationType

Element Name	Type	Semantics
FaultCode	simpleType: faultCodeType 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: Includes fault related information which may be used for fault analysis.

8.5.2.9 PaymentOptionListType

[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 104 and according to Figure 65.

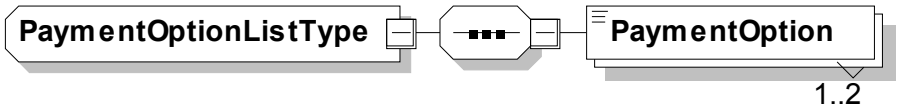


Figure 65 — Schema Diagram – PaymentOptionListType

[V2G2-283] The message element shall be used as defined in Table 70.

Table 70 — Semantics and type definition for PaymentOptionListType

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 chosen 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 104 and according to Figure 66.

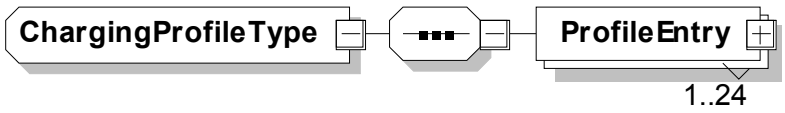


Figure 66 — Schema Diagram – ChargingProfileType

[V2G2-606] The message element shall be used as defined in Table 71.

Table 71 — Semantics and type definition for ChargingProfileType

Element Name	Type	Semantics
ProfileEntry	complexType: ProfileEntryType, refer to subclause 8.5.2.11	Element used for encapsulating an individual charging profile entry of the charge schedule. The number of ProfileEntry elements is 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 104 and according to Figure 67.

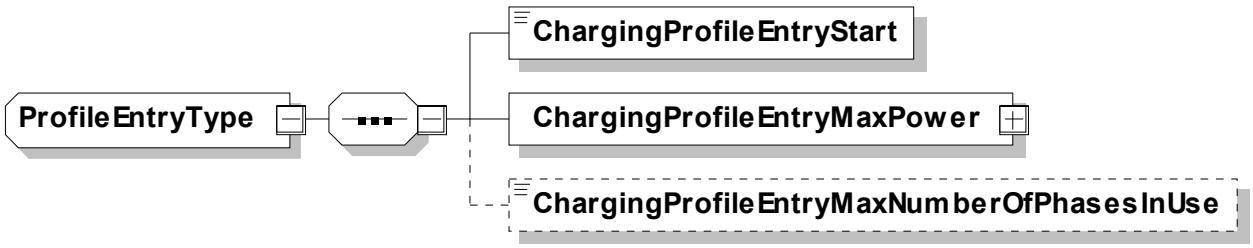


Figure 67 — Schema Diagram – ProfileEntryType

[V2G2-607] The message element shall be used as defined in Table 72.

Table 72 —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	complexType: PhysicalValueType refer to subclause 8.5.2.7	Maximum power in Watt consumed by the EV within the current charging profile entry (beginning from ChargingProfileEntryStart)
ChargingProfileEntryMaxNumberOfPhasesInUse	simpleType byte (min value: 1 max value: 3) refer to Annex C.6 for the type definition	Optional: This element is used by the EV to indicate the maximum number of phases it intends to use during the time interval defined by ChargingProfileEntryStart of this ProfileEntry.

[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 1

[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 ChargeParameterDiscoveryRes message.

[V2G2-829] If the EV is capable of determining the number of phases which will be used within an individual time interval (ChargingProfileEntryStart), the EVCC shall indicate for each time interval the maximum number of phases it intends to use during this time interval by using the parameter ChargingProfileEntryMaxNumberOfPhasesInUse.

NOTE 2 The information about the usage of phases during a certain time interval might be used by the SECC to optimize the energy distribution over the available phases.

8.5.2.12 SAScheduleListType

[V2G2-294] 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 104 and according to Figure 68.

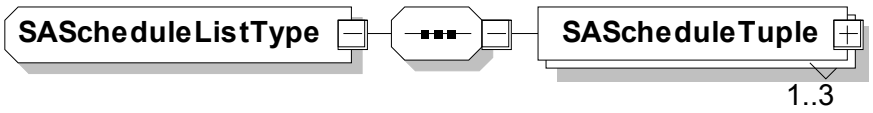


Figure 68 — Schema Diagram – SAScheduleListType

[V2G2-608] The message element shall be used as defined in Table 73.

Table 73 — 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. The number of SAScheduleTuple elements is 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 [V2G2-803] 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 104 and according to Figure 69.

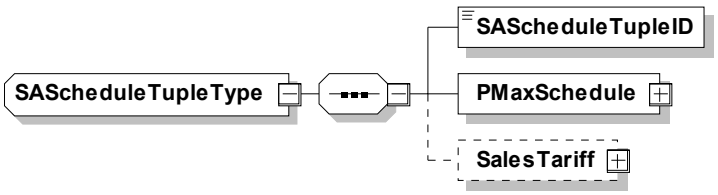


Figure 69 — Schema Diagram – SAScheduleTupleType

[V2G2-609] The message element shall be used as defined in Table 74.

Table 74 — Semantics and type definition for SAScheduleTupleType

Element Name	Type	Semantics
SAScheduleTupleID	simpleType: SAIDType unsignedByte refer to Annex C.6 for the type definition	Unique identifier within a charging session for a SAScheduleTuple element An SAID remains a unique identifier for one schedule throughout a charging session.
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

[V2G2-773] The SECC shall use the values 1 to 255 for the parameter SAScheduleTupleID.

[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 sum of the individual time intervals described in the PMaxSchedule (refer to 8.5.2.14) and SalesTariff (see 8.5.2.16) provided in the ChargeParameterDiscoveryRes message shall match the period of time indicated by the EVCC in the message element DepartureTime of the ChargeParameterDiscoveryReq message.

[V2G2-304] If the EVCC did not provide a DepartureTime Target Setting (refer to subclause 8.4.3.8.2 and 8.5.3.2) in the ChargeParameterDiscoveryReq message, the sum of the individual time intervals described in the PMaxSchedule and SalesTariff provided in the ChargeParameterDiscoveryRes message, shall be greater or equal to 24 hours..

[V2G2-305] If the number of SalesTariffEntry elements in the SalesTariff or the number of PMaxScheduleEntry elements in the PMaxSchedule provided by the secondary actor(s) are not covering the entire period of time until DepartureTime, the Target Setting EAmount (refer to subclause 8.4.3.8.2 and 8.5.3.2) has not been met and the communication session has not been finished, it is the responsibility of the EVCC to request a new element of type SAScheduleListType as soon as the last SalesTariffEntry element or the last PMaxScheduleEntry element becomes active by sending a new ChargeParameterDiscoveryReq message.

NOTE 1 The EVCC may request an update of tariff information by applying the renegotiation process at any time during the charging loop.

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

NOTE 2 The algorithm for optimizing the charging profile is out of scope of this standard.

[V2G2-307] In case of PnC, and if a Tariff Table is used by the secondary actor, the secondary actor shall sign the field SalesTariff of type SalesTariffType. In case of EIM, the secondary actor may sign this field.

[V2G2-905] In case the secondary actor uses the SalesTariff-Mechanism, the SECC shall forward this information to the EVCC using the SalesTariff field of type SalesTariffType.

NOTE 3 If the secondary actor is unaware of which authentication mode is used during EVCC-SECC communication (EIM/ PnC), it can simply always sign the 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 4 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 ChargeParameterDiscoveryRes message (see subclause 8.4.3.8.3 and 8.3.3).

[V2G2-906] If the element SalesTariff is signed, it shall be signed by the same private key that was used to issue the leaf contract certificate that the EVCC used during this connection for contract authentication (PnC).

NOTE 5 This allows the EVCC to verify the integrity of the tariff table by performing only one ECC verification. SECC/SA implementation shall use the EVCC contract certificate chain which was transmitted in the element ContractSignatureCertChain of the message PaymentDetailsReq during contract authentication to identify the private key which shall be used for tariff table signature. This requirement could be fulfilled by cooperation between SECC and SA including online communication; or the signed tariff tables may be cached.

[V2G2-907] In case of PnC, the EVCC, after receiving the SalesTariff field of type SalesTariffType, shall verify the signature using the same Sub-CA Certificate that was used to issue the Contract Certificate that the EVCC previously used for contract authentication (PnC). If this verification fails, the EVCC shall treat the SalesTariff as invalid. See **[V2G2-908]**.

NOTE 6 In case of EIM, the EVCC may ignore the signature (if it exists).

NOTE 7 The EVCC needs to perform only one ECC verification, because the signature of the tariff table was created by the same private key which was used to issue the leaf contract certificate which EVCC used during contract authentication cycle (PnC). The EVCC therefore already has the certificate for tariff table verification stored (as part of its Contract Certificate chain).

[V2G2-908] If the EVCC treats the SalesTariff as invalid, it shall ignore the SalesTariff table, i.e. the behaviour of the EVCC shall be the same as if no tariff tables were received. Furthermore, the EVCC may close the connection. It then may reopen the connection again.

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 104 and according to Figure 70.

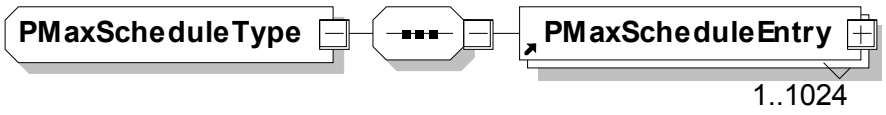


Figure 70 — Schema Diagram – PMaxScheduleType

[V2G2-610] The message element shall be used as defined in Table 75.

Table 75 — Semantics and type definition for PMaxScheduleType

Element Name	Type	Semantics
PMaxScheduleEntry	complexType: PmaxScheduleEntryType refer to subclause 8.5.2.15	List of PMaxScheduleEntry elements. The number of PMaxScheduleEntry elements is limited by the parameter MaxEntriesSAScheduleTuple (according to [V2G2-261]). The maximum value shall be 1024.

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 104 and according to Figure 71.

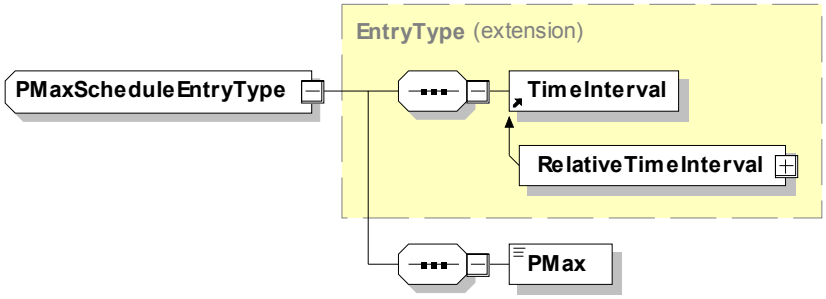


Figure 71 — Schema Diagram – PMaxScheduleEntryType

[V2G2-611] The message element shall be used as defined in Table 76.

Table 76 — Semantics and type definition for PMaxScheduleEntryType

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 PMaxScheduleEntry is valid for based upon relative times.
PMax	complexType: PhysicalValueType refer to subclause 8.5.2.7	Defines maximum amount of power for a time interval to be drawn from the EVSE power outlet the vehicle is connected to. This value represents the total power over all selected phases.

[V2G2-314] An extension of the TimeInterval element according to [V2G2-330] 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 power outlet when the element of type PMaxScheduleEntryType is active.

NOTE In case an EV changes from 3-phase charging to 1-phase charging the EV has to recalculate the Pmax.

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 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 104 and according to Figure 72.

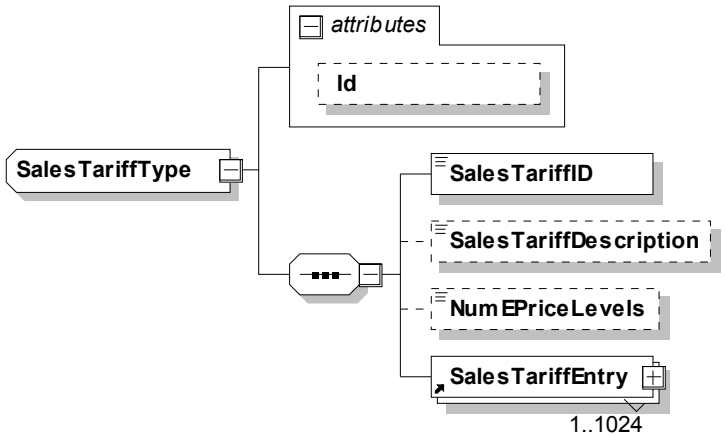


Figure 72 — Schema Diagram – SalesTariffType

[V2G2-612] The message element shall be used as defined in Table 77.

Table 77 — Semantics and type definition for SalesTariffType

Element Name	Type	Semantics
Id	simpleType: ID string refer to Annex C.6 for the type definition	Optional: This attribute is used for referencing the message element in the signature header when a signature needs to be applied.
SalesTariffID	simpleType: SAIDType short refer to Annex C.6 for the type definition	SalesTariff identifier used to identify one sales tariff. An SAID remains a unique identifier for one schedule throughout a charging session.
SalesTariffDescription	simpleType: tariffDescriptionType string (max. length: 32) refer to Annex C.6 for the type definition	Optional: A human readable title/short description of the sales tariff e.g. for HMI display purposes
NumEPriceLevels	simpleType unsignedByte refer to Annex C.6 for the type definition	Optional: Defines the overall number of distinct price levels used across all provided SalesTariff elements.
SalesTariffEntry	complexType: SalesTariffEntryType refer to subclause 8.5.2.17	Encapsulating element describing all relevant details for one time interval of the SalesTariff. The number of SalesTariffEntry elements is limited by the parameter MaxEntriesSAScheduleTuple (according to [V2G2-261]). The maximum value shall be 1024.

[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 EPriceLevel elements used across all SalesTariff elements.

NOTE 1 If several sales tariffs are used, they have to be provided from one secondary actor (SA). Therefore one signature of this SA fits for all sales tariffs.

EXAMPLE 1 If only one SalesTariff is provided as part of the SAScheduleList element, where different price levels for e.g. on-peak, mid-peak, off-peak intervals are defined, the resulting value for NumEPriceLevels element shall be set to 3.

EXAMPLE 2 If only one flat rate SalesTariff is provided as part of the SAScheduleList element with one price level at all times, the resulting value for NumEPriceLevels element shall be set to 1.

EXAMPLE 3 If two SalesTariffs are provided as part of the SAScheduleList element, where the first SalesTariff T1 has three different price levels and the second SalesTariff T2 has two different price levels and all price levels from both SalesTariffs T1 and T2 are distinct from each other, the resulting value for NumEPriceLevels element shall be set to 5.

EXAMPLE 4 If two SalesTariffs are provided as part of the SAScheduleList element, where the first SalesTariff T1 has three different price levels and the second SalesTariff T2 has two different price levels and one price level from each SalesTariffs T1 equals one price level in SalesTariff T2, the resulting value for NumEPriceLevels element shall be set to 4.

NOTE 2 Refer to Annex D.3.7 for detailed examples.

[V2G2-805] If no price information is available but an element of type SalesTariffType is still present to be able to transmit consumption cost information (CarbonDioxideEmission and RenewableGenerationPercentage only), the elements NumEPriceLevels and EPriceLevel shall not be included in the ChargeParameterDiscoveryRes message.

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 in Table 104 and according to Figure 73.

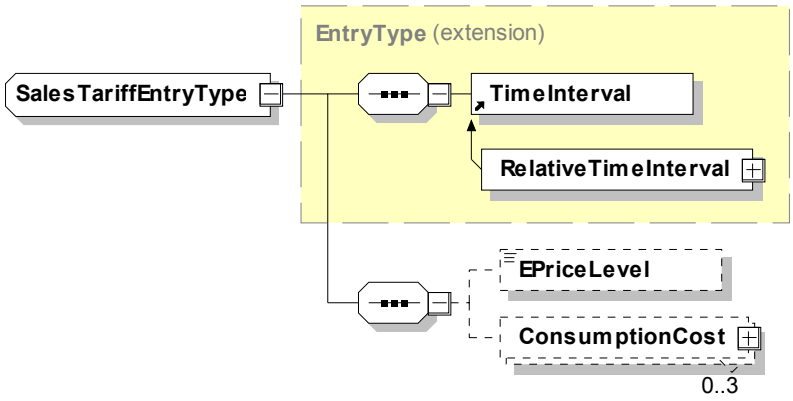


Figure 73 — Schema Diagram – SalesTariffEntryType

[V2G2-613] The message element shall be used as defined in Table 78.

Table 78 — 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	Optional: Defines the price level of this SalesTariffEntry (referring to NumEPriceLevels). Small values for the EPriceLevel represent a cheaper TariffEntry. Large values for the EPriceLevel represent a more expensive TariffEntry.
ConsumptionCost	complexType: ConsumptionCostType 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] Any extension of the TimeInterval element shall define the active period of time for the respective parent element of type SalesTariffEntryType.

[V2G2-803] The value of EPriceLevel element shall indicate which price level is used for the interval of this SalesTariffEntry.

NOTE 1 The value of EPriceLevel element only defines which price level is valid for this interval of the SalesTariff. The value of EPriceLevel element does not indicate the actual costs within the interval of this SalesTariffEntry. Applicable relative costs may optionally be defined in the ConsumptionCost element defined in subclause 8.5.2.19.NOTE The EPriceLevel element can be used by the EV for calculating a price optimized charging schedule in addition to power limits.

[V2G2-324] The valid range for the value of EPriceLevel element shall be defined as being between 0 and the value of NumEPriceLevels element (see subclause 8.5.2.16) including the boundary values.

[V2G2-802] In case the value of NumEPriceLevels is equal to "0" (and only then), the value of EPriceLevel shall be "0".

NOTE 2 The elements EPriceLevel and NumEPriceLevels are set to "0" in case the SA wants to transmit consumption costs for type RenewableGenerationPercentage and CarbonDioxideEmission only.

[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.

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 104 and according to Figure 74.

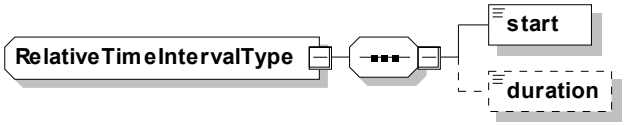


Figure 74 — Schema Diagram – RelativeTimeIntervalType

[V2G2-614] The message element shall be used as defined in Table 79.

Table 79 — 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 1 It indicates the end of the coverage time of the delivered SalesTariff or PMaxSchedule information.

NOTE 2 A relative time specification is used to avoid synchronization of clocks. While message turnaround time may take several seconds, a SECC should accept a small grace period. For example: If an EVCC starts charging earlier or ends later than defined within start element, the SECC should not end charging session with an error condition and accept this event within grace period as correct

[V2G2-833] At the point in time when a new ChargingProfile time slot begins as indicated by ChargingProfileEntryStart (n+1), the EVSE shall accept the power consumption by the EV as indicated in the ChargingProfileEntryMaxPower (n) for the previous time slot defined by ChargingProfileEntryStart (n) up to 120s longer than defined in ChargingProfileEntryStart (n+1), even if the value of the ChargingProfileEntryMaxPower (n) is higher than the ChargingProfileEntryMaxPower (n+1).

[V2G2-834] At the point in time when a new ChargingProfile time slot begins as indicated by ChargingProfileEntryStart (n+1), the EVSE shall accept the power consumption by the EV as indicated in the ChargingProfileEntryMaxPower (n+1) for the previous time slot defined by ChargingProfileEntryStart (n) up to 120s earlier than defined in ChargingProfileEntryStart (n+1), even if the value of the ChargingProfileEntryMaxPower (n+1) is higher than the ChargingProfileEntryMaxPower (n).

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 104 and according to Figure 75.

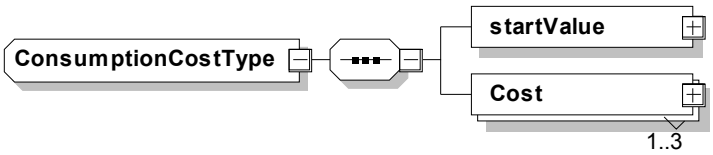


Figure 75 — Schema Diagram – ConsumptionCostType

[V2G2-615] The message element shall be used as defined in Table 80.

Table 80 — Semantics and type definition for ConsumptionCostType

Element Name	Type	Semantics
Cost	complexType: CostType refer to subclause 8.5.2.20	Encapsulating element describing all relevant cost details for this consumption block in this TariffEntry.
startValue	complexType PhysicalValueType 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] Referring to

[V2G2-328] 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 82).

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 104 and according to Figure 76.

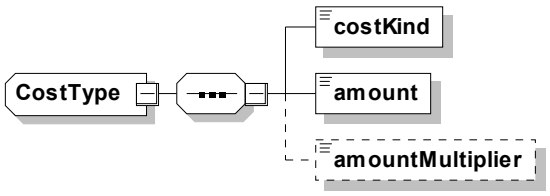


Figure 76 — Schema Diagram – CostType

[V2G2-616] The message element shall be used as defined in Table 81 and Table 82.

Table 81 — 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 kWh
amountMultiplier	simpleType unitMultiplierType (min value: -3 ; max value: 3) refer to Annex C.6	Optional: The amountMultiplier defines the exponent to base 10 (dec). The final value is determined by: amount * 10 ^ amountMultiplier
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 82 for the semantics description of individual enumeration values defined for this type).

Table 82 — Semantics for CostKindType

CostKindType	Semantics
relativePricePercentage	Relative value. Price per kWh, as percentage relative to the maximum price stated in any of all tariffs indicated to the EV.
RenewableGenerationPercentage	Relative value. Percentage of renewable generation within total generation.
CarbonDioxideEmission	Absolute value. Carbon Dioxide emissions, in grams per kWh.

[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 82) the amount element shall be defined as percentage relative to the highest cost of the same kind.

[V2G2-775] The amount element in case of a relativePricePercentage costKind shall not exceed 100%.

[V2G2-340] For absolute kinds of costs (refer to Table 82) the amount element shall be defined as absolute value of the cost.

EXAMPLE If the multiplier is 2 and the transmitted value 1, the value to be calculated is 1*10^2=100.

[V2G2-806] If the costKindType in one SalesTariff includes an absolute cost type, all other SalesTariffs included in the ChargeParameterDiscoveryRes message shall also include this absolute cost type to allow for comparison of the different SalesTariffs.

[V2G2-807] If the costKindType in one SalesTariff includes a relative cost type, all other SalesTariffs included in the ChargeParameterDiscoveryRes message shall also include this relative cost type to allow for comparison of the different SalesTariffs.

[V2G2-808] If a relative cost type is used in the SalesTariff it shall be normalized to a common base across all SalesTariff elements by the provider of the SalesTariff data.

NOTE If this precondition is not satisfied, the EVCC is not able to compare different tariffs.

[V2G2-772] If at least one SalesTarifEntry includes a consumptionCost element, all SalesTariffEntries included in all SAScheduleTuples transmitted from the SECC to the EVCC during one specific charging session, shall include consumptionCost elements with an identical structure (i.e. number of costTypes and intervals per SalesTariffEntry shall be identical).

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 104 and according to Figure 77.

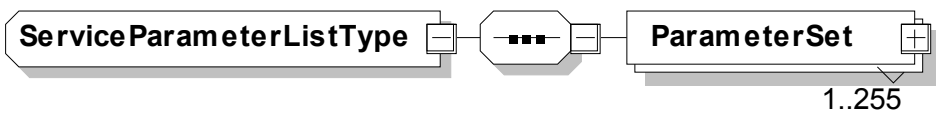


Figure 77 — Schema Diagram – ServiceParameterListType

[V2G2-344] The message element shall be used as defined in Table 83.

Table 83 —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 104 and according to Figure 78.

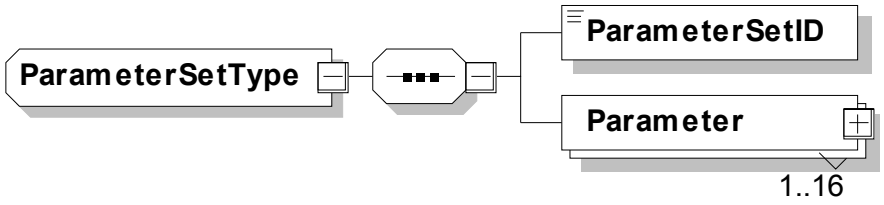


Figure 78 — Schema Diagram – ParameterSetType

[V2G2-346] The message element shall be used as defined in Table 84.

Table 84 —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 selecting a service using the PaymentServiceSelectionReq 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. The number of Parameter elements is limited to 16.

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 104 and according to Figure 79.

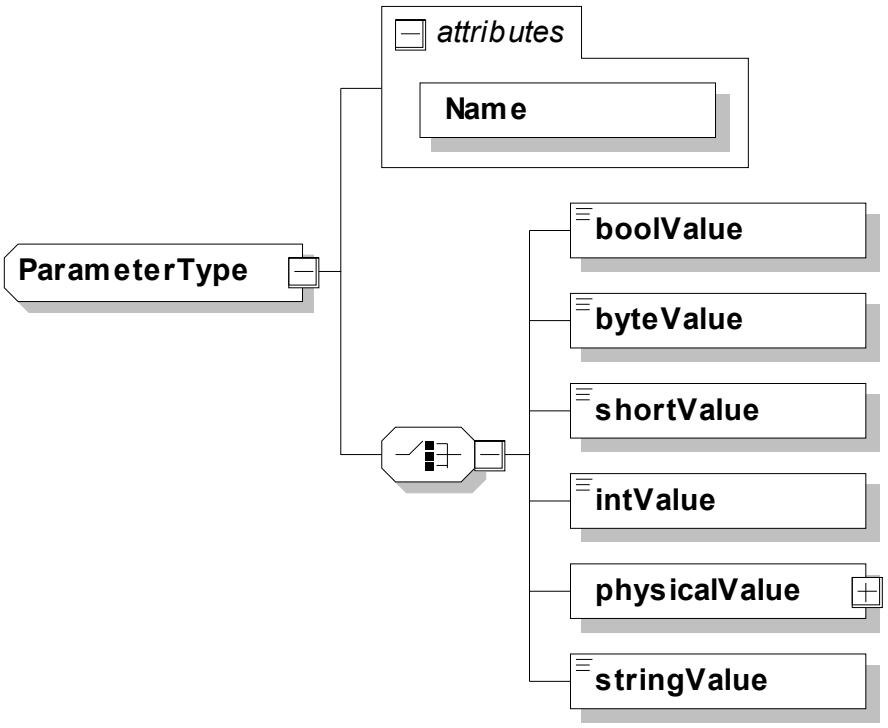


Figure 79 — Schema Diagram – ParameterType

[V2G2-348] The message element shall be used as defined in Table 85.

Table 85 —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.
One of the element: • boolValue • byteValue • shortValue • intValue • physicalValue • stringValue	simpleType: boolean (boolValue) byte (byteValue) short (shortValue) int (intValue) string (stringValue) refer to Annex C.6 for the type definition complexType: PhysicalValueType (physicalValue) refer to subclause 8.5.2.7	This element is used to indicate the value for the parameter indicated by the element Name. A choice of 6 different element types. Only one for each parameter can be selected.

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 104 and according to Figure 80.

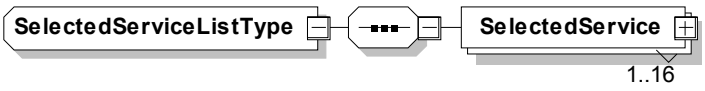


Figure 80 — Schema Diagram – SelectedServiceListType

[V2G2-350] The message element shall be used as defined in Table 86.

Table 86 —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 104 and according to Figure 81.

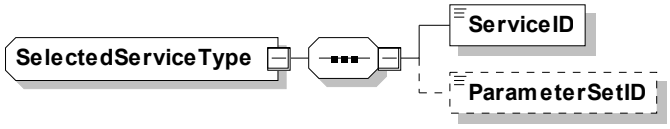


Figure 81 — Schema Diagram – SelectedServiceType

[V2G2-352] The message element shall be used as defined in Table 87.

Table 87 —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 PaymentServiceSelectionReq 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 104 and according to Figure 82.

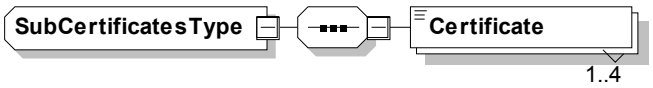


Figure 82 — Schema Diagram – SubCertificatesType

Respect that, because of [V2G2-656] and [V2G2-009], an element of type SubCertificatesType may contain 2 elements of type certificateType at maximum.

[V2G2-354] The message element shall be used as defined in Table 88.

Table 88 —Semantics and type definition for SubCertificatesType

Element Name	Type	Semantics
Certificate	simpleType: certificateType base64Binary (max length: 800) refer to Annex C.6 for the type definition	An x.509v3 certificate. The certificate is DER encoded.

[V2G2-656] The number of Certificates in the SubCertificates shall not exceed 2.

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 104 and according to Figure 83.

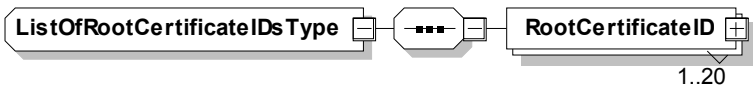


Figure 83 — Schema Diagram – ListOfRootCertificateIDsType

[V2G2-356] The message element shall be used as defined in Table 89.

Table 89 —Semantics and type definition for ListOfRootCertificateIDsType

Element Name	Type	Semantics
RootCertificateID	simpleType xmlsig:X509IssuerSerialType 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.

8.5.2.28 ContractSignatureEncryptedPrivateKeyType

[V2G2-778] 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 104 and according to Figure 84.

Figure 84 — Schema Diagram – ContractSignatureEncryptedPrivateKeyType

[V2G2-779] **T**he message element shall be used as defined in Table 90.

Table 90 —Semantics and type definition for ContractSignatureEncryptedPrivateKeyType

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

8.5.2.29 DiffieHellmanPublickeyType

[V2G2-780] 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 104 and according to Figure 85.

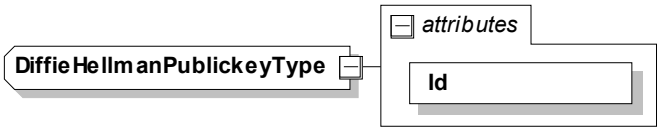


Figure 85 — Schema Diagram – DiffieHellmanPublickeyType

[V2G2-781] The message element shall be used as defined in Table 91.

Table 91 —Semantics and type definition for DiffieHellmanPublickeyType

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

8.5.2.30 EMAIDType

[V2G2-782] 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 104 and according to Figure 86.

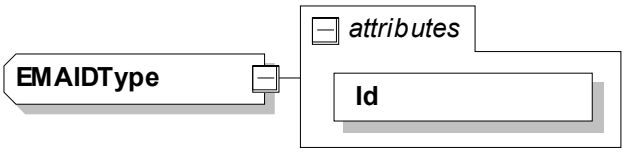


Figure 86 — Schema Diagram – EMAIDType

[V2G2-783] The message element shall be used as defined in Table 92.

Table 92 —Semantics and type definition for EMAIDType

Element Name	Type	Semantics
Id	simpleType: ID string refer to Annex C.6 for the type definition	This element is used for referencing the message element eMAID in the message header allowing to apply a signature.

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 in Table 104 and according to Figure 87.

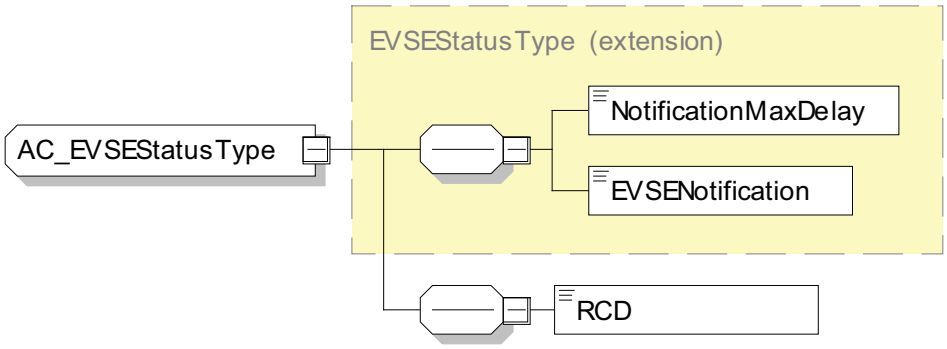


Figure 87 — Schema Diagram – AC_EVSEStatusType

[V2G2-359] The message element shall be used as defined in Table 93.

Table 93 —Semantics and type definition for AC_EVSEStatusType

Element Name	Type	Semantics
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 unsignedShort This element is inherited from EVStatusType. refer to Annex C.6 for the type definition	This value is the time in seconds from the point in time this message is sent (relative time) and expected to perform the action immediately. " 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.
EVSENotification	simpleType EVSENotificationType Enumeration This element is inherited from EVStatusType. 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 occurred 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 104 and according to Figure 88.

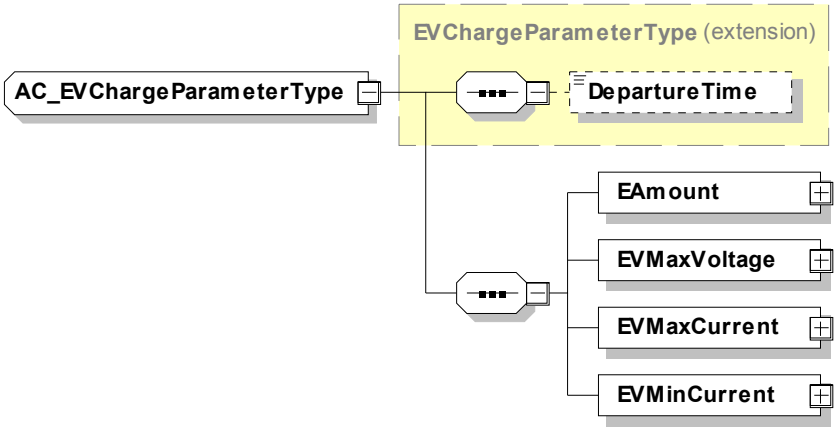


Figure 88 — Schema Diagram – AC_EVChargeParameterType

[V2G2-361] The message element shall be used as defined in Table 94.

Table 94 — Semantics and type definition for AC_EVChargeParameterType

Element Name	Type	Semantics
DepartureTime	simpleType unsignedInt This element is inherited from EVChargeParameterType refer to Annex C.6 for the type definition	Optional: 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.
EAmount	complexType PhysicalValueType refer to subclause 8.5.2.7	Amount of energy reflecting the EV's estimate how much energy is needed to fulfill the user configured charging goal for the current charging session. This might include energy for other purposes than solely charging the HV battery of an EV.
EVMaxVoltage	complexType PhysicalValueType refer to subclause 8.5.2.7	The RMS of the maximal nominal voltage the vehicle can accept, measured between one phase and neutral.
EVMaxCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	Maximum current supported by the EV per phase.
EVMinCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	EVMinCurrent 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 in Table 104 and according to Figure 89.

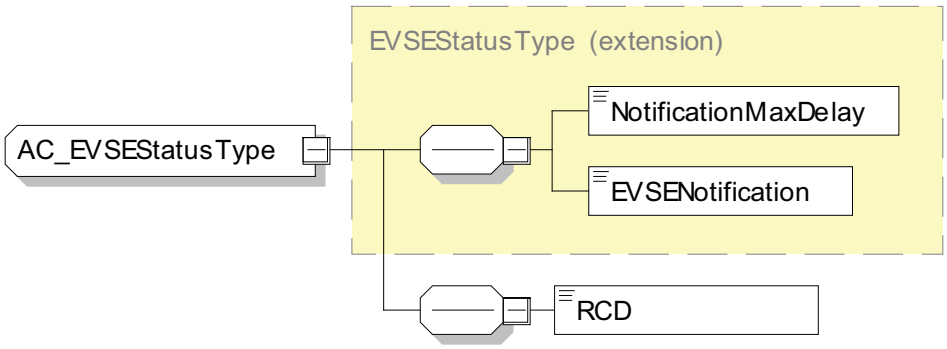


Figure 89 — Schema Diagram – AC_EVSEChargeParameterType

[V2G2-363] The message element shall be used as defined in Table 95.

Table 95 —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
EVSENominalVoltage	complexType PhysicalValueType refer to subclause 8.5.2.7	Line voltage supported by the EVSE. This is the voltage measured between one phases and neutral. If the EVSE supports multiple phase charging the EV might easily calculate the voltage between phases. This parameter is also used as reference for calculating the corresponding maximum charging current out of the PMax values in the SASchedule entities.
EVSEMaxCurrent	complexType PhysicalValueType refer to subclause 8.5.2.7	Maximum allowed line current restriction set by the EVSE per phase. If the PWM ratio is set to 5% ratio then this is the only line current restriction processed by the EVCC. Otherwise the EVCC applies the smaller current constraint from the EVSEMaxCurrent value and the PWM ratio information.

[V2G2-708] The EVSE shall calculate the PMax values in the SASchedule based on the parameter EVSENominalVoltage.

[V2G2-709] The EV shall use the value EVSENominalVoltage provided by the EVSE to calculate the maximum available charging current based on the PMax values in the element SASchedule of the ChargeParameterDiscoveryResponse and the ChargingProfileEntryMaxPower values in the element ChargingProfile Power Delivery Request.

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 104 and according to Figure 90.

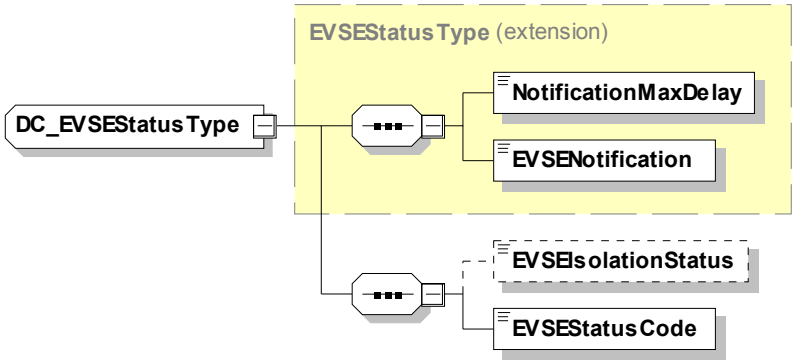


Figure 90 — Schema Diagram – DC_EVSEStatusType

[V2G2-365] The message element shall be used as defined in Table 96.

Table 96 — Semantics and type definition for DC_EVSEStatusType

Element Name	Type	Semantics
NotificationMaxDelay	simpleType unsignedShort This element is inherited from EVStatusType. refer to Annex C.6 for the type definition	This value is the time in seconds from the point in time this message is sent (relative time) and expected to perform the action immediately. " 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.
EVSENotification	simpleType EVSENotificationType Enumeration This element is inherited from EVStatusType. 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). The parameter shall be handled according to IEC CDV 61851-23
DC_EVSEStatusCode	simpleType: DC_EVSEStatusCodeType enumeration refer to Annex C.6 for the type definition	Indicates the internal state of the EVSE. Refer to Table 98 for details.

[V2G2-801] The EVCC shall use the EVSEIsolationStatus as described in Table 97.

Table 97 — Semantics and type definition for isolationLevelType

Element Name	Semantics
Invalid	An isolation test has not been carried out.
Valid	The isolation test has been carried out successfully and did not result in an isolation warning or fault.
Warning	The measured isolation resistance is below the warning level defined in IEC CDV 61851-23.
Fault	The measured isolation resistance is below the fault level defined in IEC CDV 61851-23
No_IMD	The EVSE does not contain an IMD and therefore cannot carry out an isolation test.

[V2G2-366] The SECC shall use the EVSEStatusCodes as described in Table 98 when any of the DC Message Set(s) as defined in subclause 8.6.2 has been selected.

Table 98 — 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 104 and according to Figure 91.

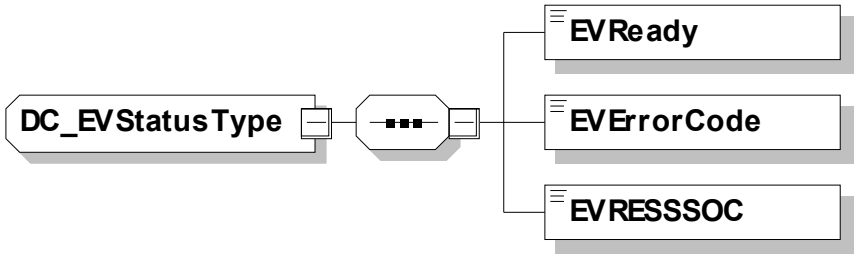


Figure 91 — Schema Diagram – DC_EVStatusType

[V2G2-368] The message element shall be used as defined in Table 99.

Table 99 — 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.
EVErrorCode	simpleType: DC_EVErrorCodeType enumeration refer to Annex C.6 for the type definition	Indicates the EV internal status.Refer to Table 100 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 100 when any of the DC Message Set(s) as defined in subclause 8.6.2 has been selected.

Table 100 — 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. Using this value is optional; as an alternative, the vehicle can use EVReady in DC_EVStatusType equal to "FALSE"
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 104 and according to Figure 92.

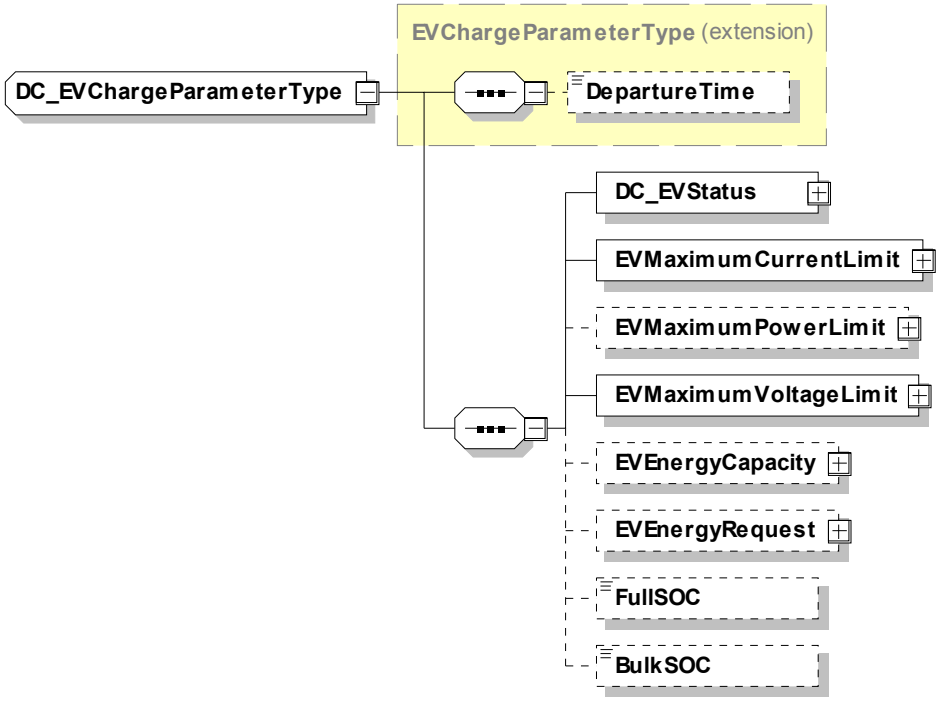


Figure 92 — Schema Diagram – DC_EVChargeParameterType

[V2G2-371] The message element shall be used as defined in Table 101.

Table 101 — Semantics and type definition for DC_EVChargeParameterType

Element Name	Type	Semantics
DepartureTime	simpleType unsignedInt This element is inherited from EVChargeParameterType refer to Annex C.6 for the type definition	Optional: 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.
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
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: Energy capacity of the EV
EVEnergyRequest	complexType PhysicalValueType refer to subclause 8.5.2.7	Optional: Amount of energy the EV requests from the EVSE.

Element Name	Type	Semantics
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 104 and according to Figure 93.

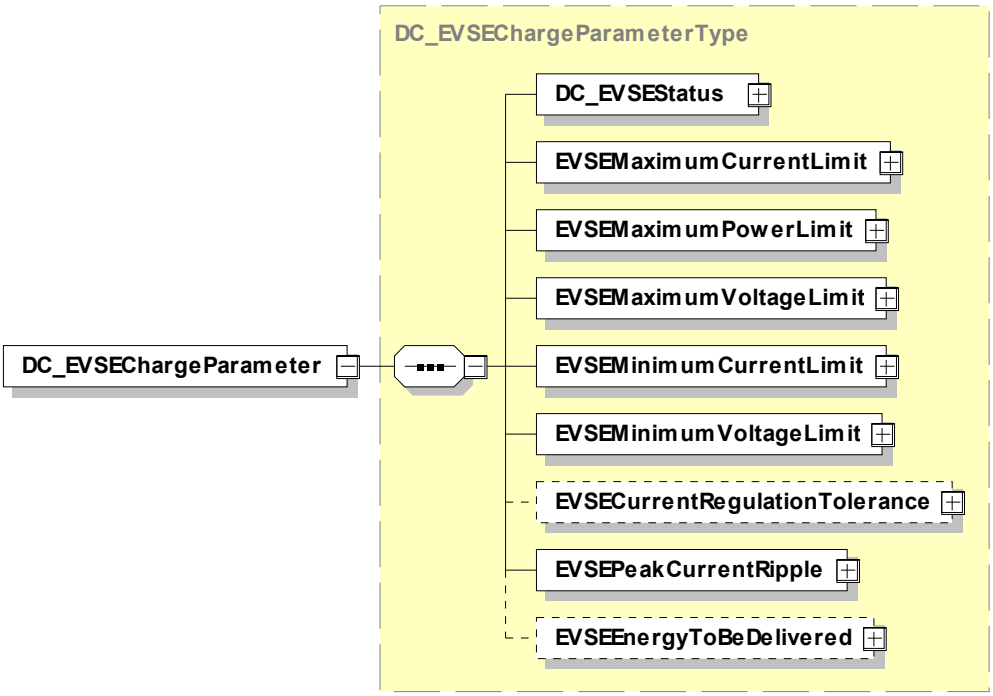


Figure 93 — Schema Diagram – DC_EVSEChargeParameterType

[V2G2-373] The message element shall be used as defined in Table 102.

Table 102 — Semantics and type definition for DC_EVSEChargeParameterType

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	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 104 and according to Figure 94.

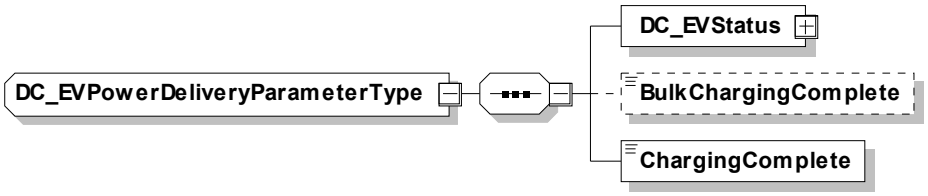


Figure 94 — Schema Diagram – DC_EVPowerDeliveryParameterType

[V2G2-375] The message element shall be used as defined in Table 103.

Table 103 — 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 these are referred to as “Identification Mode”.

Figure 95 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.

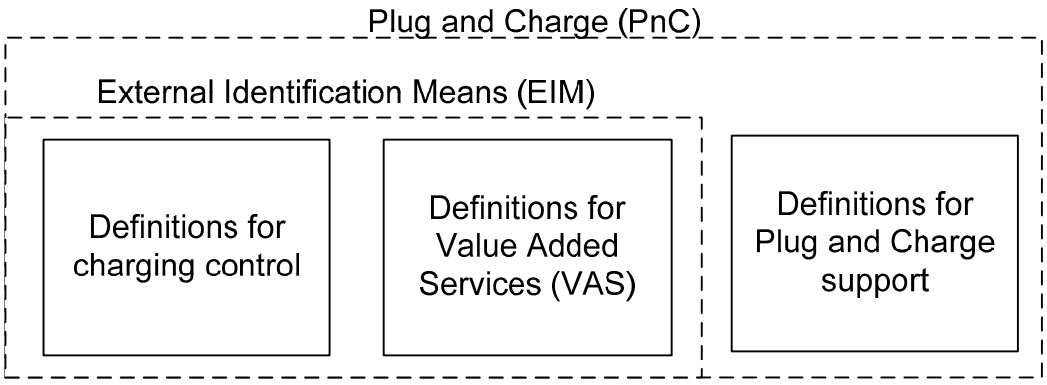


Figure 95 — Relation of Identification Modes and definitions for messages and parameter

NOTE 1 Because the main difference between the Identification Modes PnC and EIM is the support of definitions for plug and charge, an ISO 15118 implementation for a charging scenario with Identification Mode PnC can also support the same charging scenario with Identification Mode EIM.

NOTE 2 If both, an EV and an EVSE, support the Identification Modes PnC and EIM the EV is free to select the preferred identification mode.

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 shall 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 one use case it is defined as optional in the XML schema. This enables the sending of messages with only the required parameters and omits parameters that are not mandatory for a use case.

NOTE 3 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 4 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 shall 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 consists of at least one mandatory Message Set providing the basis for a specific set of charging scenarios. An Identification Mode may additionally use one or multiple additional Message Sets which allow to extend the charging scenarios.

Figure 96 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]".

This standard defines four Message Sets to be applied depending on the Identification Mode (EIM, PnC) and the charging mode (AC charging or DC charging):

— AC Charging EIM

— DC Charging EIM

— AC Charging PnC

— DC Charging PnC

Optionally, the four Message Sets can be extended by optional Message Sets defining the necessary mandatory and optional parameters and messages for the support of additional use cases. Optional Message Sets may overrule definitions of the basic Message Sets.

[V2G2-760] If an optional Message Set defines parameters already defined in a Message Set, the definition of the optional Message Set applies.

[V2G2-828] If an ISO 15118 enabled EV supporting the message set „EIM charging AC" measures a nominal duty cycle in CP State B before the PaymentServiceSelectionReq message, it shall indicate "ExternalPayment" in the parameter SelectedPaymentOption in PaymentServiceSelectionReq message as defined in subclause 8.4.3.5.

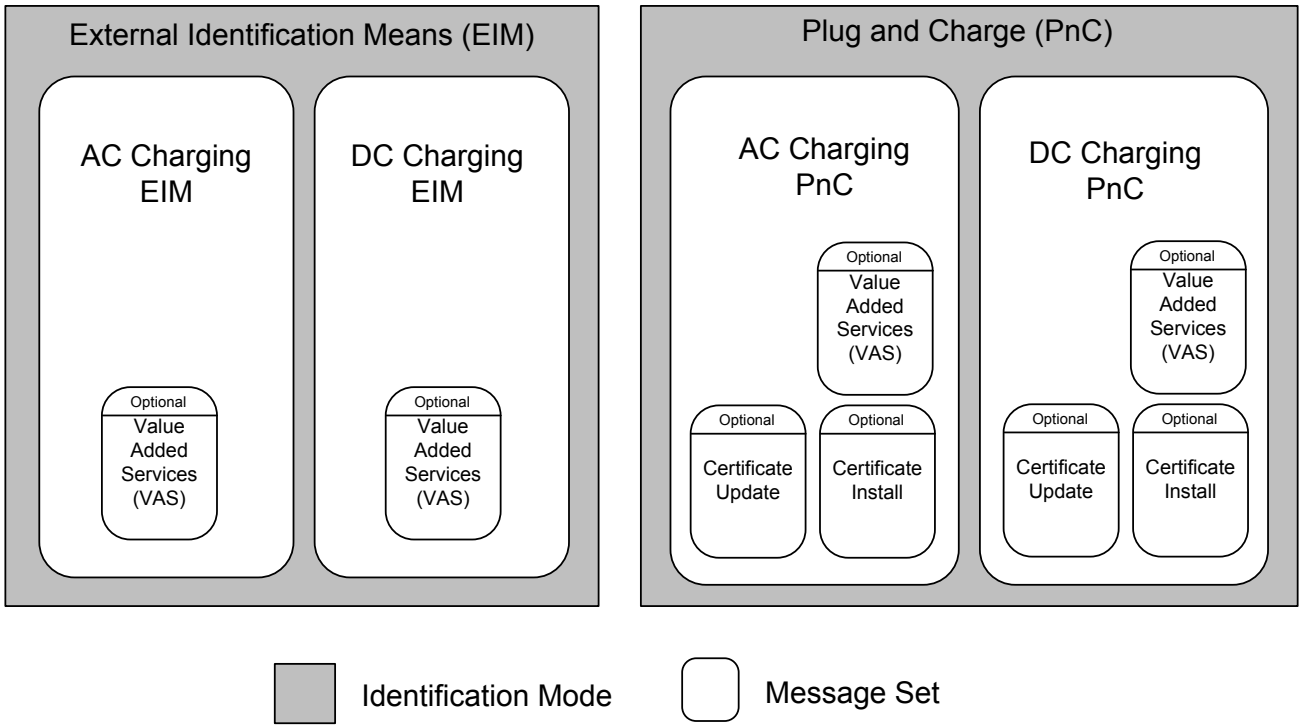


Figure 96 — Overview on Identification Modes and Message Sets

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 as shown in Figure 96. Table 104 defines all mandatory parts of the message sets as shown in Figure 96.

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] If not defined differently by requirements, 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] If not defined differently by requirements, 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 104 — Mandatory messages and message elements of Message Sets

		AC Charging EIM		DC Charging EIM		AC Charging PnC		DC Charging PnC		Option: Certificate Update		Option: Certificate Installation		Option: VAS	
		EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC	EVCC	SECC/SA	EVCC	SECC/SA	EVCC	SECC
supportedAppProtocolReq		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	-	-	-	-	-	-
supportedAppProtocolRes		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	-	-	-	-	-	-
SessionSetupReq		M	M	M	M	M	M	M	M	-	-	-	-	-	-
Header		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	SessionId	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Notification	O	M	O	M	O	M	O	M	-	-	-	-	-	-
	FaultCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	FaultMsg	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Signature (see sep. table)	-	-	-	-	M	M	M	M	-	-	-	-	-	-
Body		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	EVCCID	M	M	M	M	M	M	M	M	-	-	-	-	-	-
SessionSetupRes		M	M	M	M	M	M	M	M	-	-	-	-	-	-

Header			M	M	M	M	M	M	M	M	-	-	-	-	-	-
	SessionId		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Notification		O	M	O	M	O	M	O	M	-	-	-	-	-	-
		FaultCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
		FaultMsg	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Signature (see sep. table)		-	-	-	-	M	M	M	M	-	-	-	-	-	-
Body			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	-	-	-	-	-	-
ServiceDiscoveryReq			M	M	M	M	M	M	M	M	-	-	-	-	-	-
Header			M	M	M	M	M	M	M	M	-	-	-	-	-	-
	SessionId		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Notification		O	M	O	M	O	M	O	M	-	-	-	-	-	-
		FaultCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
		FaultMsg	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Signature (see sep. table)		-	-	-	-	M	M	M	M	-	-	-	-	-	-
Body			M	M	M	M	M	M	M	M	-	-	-	-	-	-
	ServiceScope		O	O	O	O	O	O	O	O	-	-	-	-	-	-
	ServiceCategory		O	M	O	M	O	M	O	M	-	-	-	-	-	-
ServiceDiscoveryRes			M	M	M	M	M	M	M	M	-	-	-	-	-	-
Header			M	M	M	M	M	M	M	M	-	-	-	-	-	-
	SessionId		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Notification		O	M	O	M	O	M	O	M	-	-	-	-	-	-
		FaultCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
		FaultMsg	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Signature (see sep. table)		-	-	-	-	M	M	M	M	-	-	-	-	-	-
Body			M	M	M	M	M	M	M	M	-	-	-	-	-	-
	ResponseCode		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	PaymentOptions		M	M	M	M	M	M	M	M	-	-	-	-	-	-
		PaymentOption	M	M	M	M	M	M	M	M	-	-	-	-	-	-
		PaymentOpti	M	O	M	O	M	O	M	O	-	-	-	-	-	-

		on															
	ChargeService		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	-	-	-	-	-	-
		SupportedEnergyTransferMode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
		EnergyTransferMode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	ServiceList		O	O	O	O	O	O	O	O	M	M	M	M	M	M
		Service	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	-	-	-	-	-	-
ServiceDetailReq			O	O	O	O	O	O	O	O	M	M	M	M	M	M
Header			M	M	M	M	M	M	M	M	-	-	-	-	-	-
	SessionId		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Notification		O	M	O	M	O	M	O	M	-	-	-	-	-	-
		FaultCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
		FaultMsg	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Signature (see sep. table)		M	M	M	M	M	M	M	M	-	-	-	-	-	-

	Body				M	M	M	M		M	M	M	M		-	-	-	-	-	-
		ServiceID			M	M	M	M		M	M	M	M		-	-	-	-	-	-
ServiceDetailRes					O	O	O	O		O	O	O	O		M	M	M	M	M	M
	Header				M	M	M	M		M	M	M	M		-	-	-	-	-	-
		SessionId			M	M	M	M		M	M	M	M		-	-	-	-	-	-
		Notification			O	M	O	M		O	M	O	M		-	-	-	-	-	-
			FaultCode		M	M	M	M		M	M	M	M		-	-	-	-	-	-
			FaultMsg		M	M	M	M		M	M	M	M		-	-	-	-	-	-
		Signature (see sep. table)			M	M	M	M		M	M	M	M		-	-	-	-	-	-
	Body				M	M	M	M		M	M	M	M		-	-	-	-	-	-
		ResponseCode			M	M	M	M		M	M	M	M		-	-	-	-	-	-
		ServiceID			M	M	M	M		M	M	M	M		-	-	-	-	-	-
		ServiceParameterList			M	M	M	M		M	M	M	M		-	-	-	-	-	-
			ParameterSet		M	M	M	M		M	M	M	M		-	-	-	-	-	-
				ParameterSetID	M	M	M	M		M	M	M	M		-	-	-	-	-	-
				Parameter	M	M	M	M		M	M	M	M		-	-	-	-	-	-
ServicePaymentSelectionReq					M	M	M	M		M	M	M	M		-	-	-	-	-	-
	Header				M	M	M	M		M	M	M	M		-	-	-	-	-	-
		SessionId			M	M	M	M		M	M	M	M		-	-	-	-	-	-
		Notification			O	M	O	M		O	M	O	M		-	-	-	-	-	-
			FaultCode		M	M	M	M		M	M	M	M		-	-	-	-	-	-
			FaultMsg		M	M	M	M		M	M	M	M		-	-	-	-	-	-
		Signature (see sep. table)			-	-	-	-		M	M	M	M		-	-	-	-	-	-
	Body				M	M	M	M		M	M	M	M		-	-	-	-	-	-
		SelectedPaymentOption			M	M	M	M		M	M	M	M		-	-	-	-	-	-
		SelectedServiceList			O	O	O	O		O	O	O	O		M	M	M	M	M	M
			SelectedService		M	M	M	M		M	M	M	M		-	-	-	-	-	-
				ServiceID	M	M	M	M		M	M	M	M		-	-	-	-	-	-

					ParameterSetID	M	M	M	M	M	M	M	M	-	-	-	-	-	-
ServicePaymentSelectionRes						M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Header					M	M	M	M	M	M	M	M	-	-	-	-	-	-
		SessionId				M	M	M	M	M	M	M	M	-	-	-	-	-	-
		Notification				O	M	O	M	O	M	O	M	-	-	-	-	-	-
			FaultCode			M	M	M	M	M	M	M	M	-	-	-	-	-	-
			FaultMsg			M	M	M	M	M	M	M	M	-	-	-	-	-	-
		Signature (see sep. table)				-	-	-	-	M	M	M	M	-	-	-	-	-	-
	Body					M	M	M	M	M	M	M	M	-	-	-	-	-	-
		ResponseCode				M	M	M	M	M	M	M	M	-	-	-	-	-	-
PaymentDetailsReq						-	-	-	-	M	M	M	M	-	-	-	-	-	-
	Header					-	-	-	-	M	M	M	M	-	-	-	-	-	-
		SessionId				-	-	-	-	M	M	M	M	-	-	-	-	-	-
		Notification				-	-	-	-	O	M	O	M	-	-	-	-	-	-
			FaultCode			-	-	-	-	M	M	M	M	-	-	-	-	-	-
			FaultMsg			-	-	-	-	M	M	M	M	-	-	-	-	-	-
		Signature (see sep. table)				-	-	-	-	M	M	M	M	-	-	-	-	-	-
	Body					-	-	-	-	M	M	M	M	-	-	-	-	-	-
		eMAID				-	-	-	-	M	M	M	M	-	-	-	-	-	-
		ContractSignatureCertChain				-	-	-	-	M	M	M	M	-	-	-	-	-	-
			Certificate			-	-	-	-	M	M	M	M	-	-	-	-	-	-
			SubCertificates			-	-	-	-	O	O	O	O	-	-	-	-	-	-
				Certificate		-	-	-	-	M	M	M	M	-	-	-	-	-	-
PaymentDetailsRes						-	-	-	-	M	M	M	M	-	-	-	-	-	-
	Header					-	-	-	-	M	M	M	M	-	-	-	-	-	-
		SessionId				-	-	-	-	M	M	M	M	-	-	-	-	-	-
		Notification				-	-	-	-	O	M	O	M	-	-	-	-	-	-
			FaultCode			-	-	-	-	M	M	M	M	-	-	-	-	-	-

		FaultMsg	-	-	-	-	M	M	M	M	-	-	-	-	-	-
	Signature (see sep. table)		-	-	-	-	M	M	M	M	-	-	-	-	-	-
Body			-	-	-	-	M	M	M	M	-	-	-	-	-	-
	ResponseCode		-	-	-	-	M	M	M	M	-	-	-	-	-	-
	GenChallenge		-	-	-	-	M	M	M	M	-	-	-	-	-	-
	DateTimeNow		-	-	-	-	M	M	M	M	-	-	-	-	-	-
AuthorizationReq			M	M	M	M	M	M	M	M	-	-	-	-	-	-
Header			M	M	M	M	M	M	M	M	-	-	-	-	-	-
	SessionId		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Notification		O	M	O	M	O	M	O	M	-	-	-	-	-	-
		FaultCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
		FaultMsg	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Signature (see sep. table)		-	-	-	-	M	M	M	M	-	-	-	-	-	-
Body			M	M	M	M	M	M	M	M	-	-	-	-	-	-
	GenChallenge		-	-	-	-	M	M	M	M	-	-	-	-	-	-
AuthorizationRes			M	M	M	M	M	M	M	M	-	-	-	-	-	-
Header			M	M	M	M	M	M	M	M	-	-	-	-	-	-
	SessionId		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Notification		O	M	O	M	O	M	O	M	-	-	-	-	-	-
		FaultCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
		FaultMsg	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Signature (see sep. table)		-	-	-	-	M	M	M	M	-	-	-	-	-	-
Body			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	-	-	-	-	-	-
ChargeParameterDiscoveryReq			M	M	M	M	M	M	M	M	-	-	-	-	-	-
Header			M	M	M	M	M	M	M	M	-	-	-	-	-	-
	SessionId		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Notification		O	M	O	M	O	M	O	M	-	-	-	-	-	-
		FaultCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
		FaultMsg	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Signature (see sep.		-	-	-	-	M	M	M	M	-	-	-	-	-	-

table)														
Body	M	M	M	M	M	M	M	M	-	-	-	-	-	-
MaxEntriesSAScheduleTuple	O	M	O	M	O	M	O	M	-	-	-	-	-	-
RequestedEnergyTransferMode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
AC_EVChargeParameter	M	M	-	-	M	M	-	-	-	-	-	-	-	-
DepartureTime	O	M	-	-	O	M	-	-	-	-	-	-	-	-
Eamount	M	M	-	-	M	M	-	-	-	-	-	-	-	-
EVMaxVoltage	M	M	-	-	M	M	-	-	-	-	-	-	-	-
EVMaxCurrent	M	M	-	-	M	M	-	-	-	-	-	-	-	-
EVMinCurrent	M	M	-	-	M	M	-	-	-	-	-	-	-	-
DC_EVChargeParameter	-	-	M	M	-	-	M	M	-	-	-	-	-	-
DepartureTime	-	-	O	M	-	-	O	M	-	-	-	-	-	-
DC_EVStatus	-	-	M	M	-	-	M	M	-	-	-	-	-	-
EVReady	-	-	M	M	-	-	M	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	-	-	-	-	-	-
ChargeParameterDiscoveryRes										M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Header									M	M	M	M	M	M	M	M	-	-	-	-	-	-
		SessionId								M	M	M	M	M	M	M	M	-	-	-	-	-	-
		Notification								O	M	O	M	O	M	O	M	-	-	-	-	-	-
			FaultCode							M	M	M	M	M	M	M	M	-	-	-	-	-	-
			FaultMsg							M	M	M	M	M	M	M	M	-	-	-	-	-	-
		Signature (see sep. table)								-	-	-	-	M	M	M	M	-	-	-	-	-	-
	Body									M	M	M	M	M	M	M	M	-	-	-	-	-	-
		ResponseCode								M	M	M	M	M	M	M	M	-	-	-	-	-	-
		EVSEProcessing								M	M	M	M	M	M	M	M	-	-	-	-	-	-
		SASchedule								M	O	M	O	M	O	M	O	-	-	-	-	-	-
			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	-	-	-	-	-	-
						SalesTariffI				M	M	M	M	M	M	M	M	-	-	-	-	-	-

								D															
								SalesTariff Description	O	O	O	O	O	O	O	O	-	-	-	-	-	-
								NumEPrice Levels	M	O	M	O	M	O	M	O	-	-	-	-	-	-
								SalesTariff Entry	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	O	M	O	M	O	M	O	-	-	-	-	-	-
								ConsumptionCost	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	-	-	-	-	-	-	-	-
							RCD		M	M	-	-	M	M	-	-	-	-	-	-	-	-
							NotificationMaxDelay		M	M	-	-	M	M	-	-	-	-	-	-	-	-
							EVSENotification		M	M	-	-	M	M	-	-	-	-	-	-	-	-
						EVSENominalVoltage			M	M	-	-	M	M	-	-	-	-	-	-	-	-
						EVSEMaxCurrent			M	M	-	-	M	M	-	-	-	-	-	-	-	-
					DC_EVSEChargeParameter				-	-	M	M	-	-	M	M	-	-	-	-	-	-
						DC_EVSEStatus			-	-	M	M	-	-	M	M	-	-	-	-	-	-
							EVSEIsolationStatus		-	-	M	O	-	-	M	O	-	-	-	-	-	-
							EVSE		-	-	M	M	-	-	M	M	-	-	-	-	-	-

					Status Code																		
					Notification MaxDelay	-	-	M	M	-	-	M	M	-	-	-	-	-	-				
					EVSE Notification	-	-	M	M	-	-	M	M	-	-	-	-	-	-				
				EVSEMaximumCurrentLimit		-	-	M	M	-	-	M	M	-	-	-	-	-	-				
				EVSEMaximumPowerLimit		-	-	M	M	-	-	M	M	-	-	-	-	-	-				
				EVSEMaximumVoltageLimit		-	-	M	M	-	-	M	M	-	-	-	-	-	-				
				EVSEMinimumCurrentLimit		-	-	M	M	-	-	M	M	-	-	-	-	-	-				
				EVSEMinimumVoltageLimit		-	-	M	M	-	-	M	M	-	-	-	-	-	-				
				EVSECurrentRegulationTolerance		-	-	O	O	-	-	O	O	-	-	-	-	-	-				
				EVSEPeakCurrentRipple		-	-	M	M	-	-	M	M	-	-	-	-	-	-				
				EVSEEnergyToBeDelivered		-	-	O	O	-	-	O	O	-	-	-	-	-	-				
PowerDeliveryReq						M	M	M	M	M	M	M	M	-	-	-	-	-	-				
	Header					M	M	M	M	M	M	M	M	-	-	-	-	-	-				
		SessionId				M	M	M	M	M	M	M	M	-	-	-	-	-	-				
		Notification				O	M	O	M	O	M	O	M	-	-	-	-	-	-				
			FaultCode			M	M	M	M	M	M	M	M	-	-	-	-	-	-				
			FaultMsg			M	M	M	M	M	M	M	M	-	-	-	-	-	-				
		Signature (see sep. table)				-	-	-	-	M	M	M	M	-	-	-	-	-	-				
	Body					M	M	M	M	M	M	M	M	-	-	-	-	-	-				
		ChargeProgress				M	M	M	M	M	M	M	M	-	-	-	-	-	-				

		SAScheduleTupleID			M	M	M	M	M	M	M	M	-	-	-	-	-	-
		ChargingProfile			O	M	O	M	O	M	O	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	-	-	-	-	-	-
				ChargingProfileEntryMaxNumberOfPhasesInUse	O	M	-	-	O	M	-	-	-	-	-	-	-	-
		DC_EVPowerDeliveryParameter			-	-	M	M	-	-	M	M	-	-	-	-	-	-
			DC_EVStatus		-	-	M	M	-	-	M	M	-	-	-	-	-	-
				EVReady	-	-	M	M	-	-	M	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	-	-	-	-	-	-
PowerDeliveryRes					M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Header				M	M	M	M	M	M	M	M	-	-	-	-	-	-
		SessionId			M	M	M	M	M	M	M	M	-	-	-	-	-	-
		Notification			O	M	O	M	O	M	O	M	-	-	-	-	-	-
			FaultCode		M	M	M	M	M	M	M	M	-	-	-	-	-	-
			FaultMsg		M	M	M	M	M	M	M	M	-	-	-	-	-	-
		Signature (see sep. table)			-	-	-	-	M	M	M	M	-	-	-	-	-	-

	Body					M	M	M	M		M	M	M	M		-	-	-	-	-	-
		ResponseCode				M	M	M	M		M	M	M	M		-	-	-	-	-	-
		AC_EVSEStatus				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		-	-	-	-	-	-
			EVSEIsolationStatus			-	-	M	O		-	-	M	O		-	-	-	-	-	-
			EVSEStatusCode			-	-	M	M		-	-	M	M		-	-	-	-	-	-
			NotificationMaxDelay			-	-	M	M		-	-	M	M		-	-	-	-	-	-
			EVSENotification			-	-	M	M		-	-	M	M		-	-	-	-	-	-
CertificateupdateReq						-	-	-	-		-	-	-	-		M	M	-	-	-	-
	Header					-	-	-	-		-	-	-	-		M	M	-	-	-	-
		SessionId				-	-	-	-		-	-	-	-		M	M	-	-	-	-
		Notification				-	-	-	-		-	-	-	-		M	M	-	-	-	-
			FaultCode			-	-	-	-		-	-	-	-		M	M	-	-	-	-
			FaultMsg			-	-	-	-		-	-	-	-		M	M	-	-	-	-
		Signature (see sep. table)				-	-	-	-		-	-	-	-		M	M	-	-	-	-
	Body					-	-	-	-		-	-	-	-		M	M	-	-	-	-
		ContractSignatureCertChain				-	-	-	-		-	-	-	-		M	M	-	-	-	-
			Certificate			-	-	-	-		-	-	-	-		M	M	-	-	-	-
			SubCertificates			-	-	-	-		-	-	-	-		O	O	-	-	-	-
				Certificate		-	-	-	-		-	-	-	-		M	M	-	-	-	-
		eMAID				-	-	-	-		-	-	-	-		M	M	-	-	-	-
		ListOfRootCertificateIDs				-	-	-	-		-	-	-	-		M	M	-	-	-	-
			RootCertificateID			-	-	-	-		-	-	-	-		M	M	-	-	-	-
CertificateupdateRes						-	-	-	-		-	-	-	-		O	O	-	-	-	-

	Header				-	-	-	-		-	-	-	-		M	M	-	-	-	-
		SessionId			-	-	-	-		-	-	-	-		M	M	-	-	-	-
		Notification			-	-	-	-		-	-	-	-		O	M	-	-	-	-
			FaultCode		-	-	-	-		-	-	-	-		M	M	-	-	-	-
			FaultMsg		-	-	-	-		-	-	-	-		M	M	-	-	-	-
		Signature (see sep. table)			-	-	-	-		-	-	-	-		O	M	-	-	-	-
	Body				-	-	-	-		-	-	-	-		M	M	-	-	-	-
		ResponseCode			-	-	-	-		-	-	-	-		M	M	-	-	-	-
		SAProvisioningCertificateChain			-	-	-	-		-	-	-	-		M	M	-	-	-	-
		ContractSignatureCertChain			-	-	-	-		-	-	-	-		M	M	-	-	-	-
			Certificate		-	-	-	-		-	-	-	-		M	M	-	-	-	-
			SubCertificates		-	-	-	-		-	-	-	-		O	O	-	-	-	-
				Certificate	-	-	-	-		-	-	-	-		M	M	-	-	-	-
		ContractSignatureEncryptedPrivateKey			-	-	-	-		-	-	-	-		M	M	-	-	-	-
		Dhpublickey			-	-	-	-		-	-	-	-		M	M	-	-	-	-
		eMAID			-	-	-	-		-	-	-	-		M	M	-	-	-	-
		RetryCounter			-	-	-	-		-	-	-	-		M	O	-	-	-	-
CertificateinstallationReq					-	-	-	-		-	-	-	-		-	-	M	M	-	-
	Header				-	-	-	-		-	-	-	-		-	-	M	M	-	-
		SessionId			-	-	-	-		-	-	-	-		-	-	M	M	-	-
		Notification			-	-	-	-		-	-	-	-		-	-	O	M	-	-
			FaultCode		-	-	-	-		-	-	-	-		-	-	M	M	-	-
			FaultMsg		-	-	-	-		-	-	-	-		-	-	M	M	-	-
		Signature (see sep. table)			-	-	-	-		-	-	-	-		-	-	M	M	-	-
	Body				-	-	-	-		-	-	-	-		-	-	M	M	-	-
		OEMProvisioningCert			-	-	-	-		-	-	-	-		-	-	M	M	-	-
		ListOfRootCertificateIDs			-	-	-	-		-	-	-	-		-	-	M	M	-	-

			RootCertificateID	-	-	-	-	-	-	-	-	-	-	M	M	-	-
CertificateinstallationRes				-	-	-	-	-	-	-	-	-	-	O	O	-	-
	Header			-	-	-	-	-	-	-	-	-	-	M	M	-	-
		SessionId		-	-	-	-	-	-	-	-	-	-	M	M	-	-
		Notification		-	-	-	-	-	-	-	-	-	-	M	M	-	-
			FaultCode	-	-	-	-	-	-	-	-	-	-	M	M	-	-
			FaultMsg	-	-	-	-	-	-	-	-	-	-	M	M	-	-
		Signature (see sep. table)		-	-	-	-	-	-	-	-	-	-	O	M	-	-
	Body			-	-	-	-	-	-	-	-	-	-	M	M	-	-
		ResponseCode		-	-	-	-	-	-	-	-	-	-	M	M	-	-
		SAProvisioningCertificateChain		-	-	-	-	-	-	-	-	-	-	M	M	-	-
		ContractSignatureCertChain		-	-	-	-	-	-	-	-	-	-	M	M	-	-
			Certificate	-	-	-	-	-	-	-	-	-	-	M	M	-	-
			SubCertificates	-	-	-	-	-	-	-	-	-	-	O	O	-	-
				Certificate	-	-	-	-	-	-	-	-	-	-	M	M	-	-
		ContractSignatureEncryptedPrivateKey		-	-	-	-	-	-	-	-	-	-	M	M	-	-
		Dhpublickey		-	-	-	-	-	-	-	-	-	-	M	M	-	-
		eMAID		-	-	-	-	-	-	-	-	-	-	M	M	-	-
SessionStopReq				M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Header			M	M	M	M	M	M	M	M	-	-	-	-	-	-
		SessionId		M	M	M	M	M	M	M	M	-	-	-	-	-	-
		Notification		O	M	O	M	O	M	O	M	-	-	-	-	-	-
			FaultCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
			FaultMsg	M	M	M	M	M	M	M	M	-	-	-	-	-	-
		Signature (see sep. table)		-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Body			M	M	M	M	M	M	M	M	-	-	-	-	-	-
		ChargingSession		M	M	M	M	M	M	M	M	-	-	-	-	-	-
SessionStopRes				M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Header			M	M	M	M	M	M	M	M	-	-	-	-	-	-

	SessionId	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Notification	O	M	O	M	O	M	O	M	-	-	-	-	-	-
	FaultCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	FaultMsg	M	M	M	M	M	M	M	M	-	-	-	-	-	-
	Signature (see sep. table)	-	-	-	-	M	M	M	M	-	-	-	-	-	-
Body		M	M	M	M	M	M	M	M	-	-	-	-	-	-
	ResponseCode	M	M	M	M	M	M	M	M	-	-	-	-	-	-
ChargingStatusReq		M	M	-	-	M	M	-	-	-	-	-	-	-	-
Header		M	M	-	-	M	M	-	-	-	-	-	-	-	-
	SessionId	M	M	-	-	M	M	-	-	-	-	-	-	-	-
	Notification	O	M	-	-	O	M	-	-	-	-	-	-	-	-
	FaultCode	M	M	-	-	M	M	-	-	-	-	-	-	-	-
	FaultMsg	M	M	-	-	M	M	-	-	-	-	-	-	-	-
	Signature (see sep. table)	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Body		M	M	-	-	M	M	-	-	-	-	-	-	-	-
ChargingStatusRes		M	M	-	-	M	M	-	-	-	-	-	-	-	-
Header		M	M	-	-	M	M	-	-	-	-	-	-	-	-
	SessionId	M	M	-	-	M	M	-	-	-	-	-	-	-	-
	Notification	O	M	-	-	O	M	-	-	-	-	-	-	-	-
	FaultCode	M	M	-	-	M	M	-	-	-	-	-	-	-	-
	FaultMsg	M	M	-	-	M	M	-	-	-	-	-	-	-	-
	Signature (see sep. table)	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Body		M	M	-	-	M	M	-	-	-	-	-	-	-	-
	ResponseCode	M	M	-	-	M	M	-	-	-	-	-	-	-	-
	EVSEID	M	M	-	-	M	M	-	-	-	-	-	-	-	-
	SAScheduleTupleID	M	M	-	-	M	M	-	-	-	-	-	-	-	-
	EVSEMaxCurrent	M	O	-	-	O	O	-	-	-	-	-	-	-	-
	MeterInfo	M	O	-	-	M	O	-	-	-	-	-	-	-	-
	MeterID	M	M	-	-	M	M	-	-	-	-	-	-	-	-
	MeterReading	O	O	-	-	O	O	-	-	-	-	-	-	-	-
	SigMeterReading	O	O	-	-	O	O	-	-	-	-	-	-	-	-

			MeterStatus	O	O	-	-		O	O	-	-		-	-	-	-	-	-				
			TMeter	O	O	-	-		O	O	-	-		-	-	-	-	-	-				
		ReceiptRequired		-	-	-	-		M	O	-	-		-	-	-	-	-	-				
		AC_EVSEStatus		M	M	-	-		M	M	-	-		-	-	-	-	-	-				
			RCD	M	M	-	-		M	M	-	-		-	-	-	-	-	-				
			NotificationMaxDelay	M	M	-	-		M	M	-	-		-	-	-	-	-	-				
			EVSENotification	M	M	-	-		M	M	-	-		-	-	-	-	-	-				
MeteringReceiptReq				-	-	-	-		M	M	-	-		-	-	-	-	-	-				
	Header			-	-	-	-		M	M	-	-		-	-	-	-	-	-				
		SessionId		-	-	-	-		M	M	-	-		-	-	-	-	-	-				
		Notification		-	-	-	-		O	M	-	-		-	-	-	-	-	-				
			FaultCode	-	-	-	-		M	M	-	-		-	-	-	-	-	-				
			FaultMsg	-	-	-	-		M	M	-	-		-	-	-	-	-	-				
		Signature (see sep. table)		-	-	-	-		M	M	-	-		-	-	-	-	-	-				
	Body			-	-	-	-		M	M	-	-		-	-	-	-	-	-				
		SessionID		-	-	-	-		M	M	-	-		-	-	-	-	-	-				
		SAScheduleTupleID		-	-	-	-		O	M	-	-		-	-	-	-	-	-				
		MeterInfo		-	-	-	-		M	M	-	-		-	-	-	-	-	-				
			MeterID	-	-	-	-		M	M	-	-		-	-	-	-	-	-				
			MeterReading	-	-	-	-		O	O	-	-		-	-	-	-	-	-				
			SigMeterReading	-	-	-	-		O	O	-	-		-	-	-	-	-	-				
			MeterStatus	-	-	-	-		O	O	-	-		-	-	-	-	-	-				
			Tmeter	-	-	-	-		O	O	-	-		-	-	-	-	-	-				
MeteringReceiptRes				-	-	-	-		M	M	-	-		-	-	-	-	-	-				
	Header			-	-	-	-		M	M	-	-		-	-	-	-	-	-				
		SessionId		-	-	-	-		M	M	-	-		-	-	-	-	-	-				
		Notification		-	-	-	-		O	M	-	-		-	-	-	-	-	-				
			FaultCode	-	-	-	-		M	M	-	-		-	-	-	-	-	-				
			FaultMsg	-	-	-	-		M	M	-	-		-	-	-	-	-	-				
		Signature (see sep. table)		-	-	-	-		-	-	-	-		-	-	-	-	-	-				
	Body			-	-	-	-		M	M	-	-		-	-	-	-	-	-				

		AC_EVSEStatus		-	-	-	-	M	M	-	-	-	-	-	-	-	-
			RCD	-	-	-	-	M	M	-	-	-	-	-	-	-	-
			NotificationMaxDelay	-	-	-	-	M	M	-	-	-	-	-	-	-	-
			EVSENotification	-	-	-	-	M	M	-	-	-	-	-	-	-	-
		DC_EVSEStatus		-	-	-	-	-	-	M	M	-	-	-	-	-	-
			EVSEIsolationStatus	-	-	-	-	-	-	M	O	-	-	-	-	-	-
			EVSEStatusCode	-	-	-	-	-	-	M	M	-	-	-	-	-	-
			NotificationMaxDelay	-	-	-	-	-	-	M	M	-	-	-	-	-	-
			EVSENotification	-	-	-	-	-	-	M	M	-	-	-	-	-	-
		ResponseCode		-	-	-	-	M	M	-	-	-	-	-	-	-	-
CableCheckReq				-	-	M	M	-	-	M	M	-	-	-	-	-	-
	Header			-	-	M	M	-	-	M	M	-	-	-	-	-	-
		SessionId		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Notification		-	-	O	M	-	-	O	M	-	-	-	-	-	-
			FaultCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-
			FaultMsg	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Signature (see sep. table)		-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Body			-	-	M	M	-	-	M	M	-	-	-	-	-	-
		DC_EVStatus		-	-	M	M	-	-	M	M	-	-	-	-	-	-
			EVReady	-	-	M	M	-	-	M	M	-	-	-	-	-	-
			EVRESSConditioning	-	-	O	M	-	-	O	M	-	-	-	-	-	-
			EVErrorCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-
			EVRESSSOC	-	-	M	M	-	-	M	M	-	-	-	-	-	-
CableCheckRes				-	-	M	M	-	-	M	M	-	-	-	-	-	-
	Header			-	-	M	M	-	-	M	M	-	-	-	-	-	-
		SessionId		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Notification		-	-	O	M	-	-	O	M	-	-	-	-	-	-
			FaultCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-

			FaultMsg	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Signature (see sep. table)		-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Body			-	-	M	M	-	-	M	M	-	-	-	-	-	-
		ResponseCode		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		EVSEProcessing		-	-	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	-	-	-	-	-	-
PreChargeReq				-	-	M	M	-	-	M	M	-	-	-	-	-	-
	Header			-	-	M	M	-	-	M	M	-	-	-	-	-	-
		SessionId		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Notification		-	-	O	M	-	-	O	M	-	-	-	-	-	-
			FaultCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-
			FaultMsg	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Signature (see sep. table)		-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Body			-	-	M	M	-	-	M	M	-	-	-	-	-	-
		DC_EVStatus		-	-	M	M	-	-	M	M	-	-	-	-	-	-
			EVReady	-	-	M	M	-	-	M	M	-	-	-	-	-	-
			EVRESSConditioning	-	-	O	M	-	-	O	M	-	-	-	-	-	-
			EVErrorCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-
			EVRESSSOC	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		EVTargetVoltage		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		EVTargetCurrent		-	-	M	M	-	-	M	M	-	-	-	-	-	-
PreChargeRes				-	-	M	M	-	-	M	M	-	-	-	-	-	-
	Header			-	-	M	M	-	-	M	M	-	-	-	-	-	-
		SessionId		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Notification		-	-	O	M	-	-	O	M	-	-	-	-	-	-

		FaultCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		FaultMsg	-	-	M	M	-	-	M	M	-	-	-	-	-	-
	Signature (see sep. table)		-	-	-	-	-	-	-	-	-	-	-	-	-	-
Body			-	-	M	M	-	-	M	M	-	-	-	-	-	-
	ResponseCode		-	-	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	-	-	-	-	-	-
	EVSEPresentVoltage		-	-	M	M	-	-	M	M	-	-	-	-	-	-
CurrentDemandReq			-	-	M	M	-	-	M	M	-	-	-	-	-	-
Header			-	-	M	M	-	-	M	M	-	-	-	-	-	-
	SessionId		-	-	M	M	-	-	M	M	-	-	-	-	-	-
	Notification		-	-	O	M	-	-	O	M	-	-	-	-	-	-
		FaultCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		FaultMsg	-	-	M	M	-	-	M	M	-	-	-	-	-	-
	Signature (see sep. table)		-	-	-	-	-	-	-	-	-	-	-	-	-	-
Body			-	-	M	M	-	-	M	M	-	-	-	-	-	-
	DC_EVStatus		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		EVReady	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		EVRESSConditioning	-	-	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	-	-	-	-	-	-
		RemainingTimeToFullSoC		-	-	O	M	-	-	O	M	-	-	-	-	-	-
		RemainingTimeToBulkSoC		-	-	O	M	-	-	O	M	-	-	-	-	-	-
		EVTargetVoltage		-	-	M	M	-	-	M	M	-	-	-	-	-	-
CurrentDemandRes				-	-	M	M	-	-	M	M	-	-	-	-	-	-
	Header			-	-	M	M	-	-	M	M	-	-	-	-	-	-
		SessionId		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Notification		-	-	O	M	-	-	O	M	-	-	-	-	-	-
			FaultCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-
			FaultMsg	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Signature (see sep. table)		-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Body			-	-	M	M	-	-	M	M	-	-	-	-	-	-
		ResponseCode		-	-	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	-	-	-	-	-	-
		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	-	-	-	-	-	-
		EVSEMaximumVolt		-	-	M	O	-	-	M	O	-	-	-	-	-	-

		ageLimit															
		EVSEMaximumCurrentLimit	-	-	M	O	-	-	M	O	-	-	-	-	-	-
		EVSEMaximumPowerLimit	-	-	M	O	-	-	M	O	-	-	-	-	-	-
		EVSEID	-	-	M	M	-	-	M	O	-	-	-	-	-	-
		SAScheduleTupleID	-	-	M	M	-	-	M	O	-	-	-	-	-	-
		MeterInfo	-	-	-	-	-	-	M	O	-	-	-	-	-	-
		MeterID	-	-	-	-	-	-	M	M	-	-	-	-	-	-
		MeterReading	-	-	-	-	-	-	O	O	-	-	-	-	-	-
		SigMeterReading	-	-	-	-	-	-	O	O	-	-	-	-	-	-
		MeterStatus	-	-	-	-	-	-	O	O	-	-	-	-	-	-
		Tmeter	-	-	-	-	-	-	O	O	-	-	-	-	-	-
		ReceiptRequired	-	-	-	-	-	-	M	O	-	-	-	-	-	-
WeldingDetectionReq			-	-	O	M	-	-	O	M	-	-	-	-	-	-
	Header		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		SessionId	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Notification	-	-	O	M	-	-	O	M	-	-	-	-	-	-
		FaultCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		FaultMsg	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Signature (see sep. table)	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Body		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		DC_EVStatus	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		EVReady	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		EVRESSConditioning	-	-	O	M	-	-	O	M	-	-	-	-	-	-
		EVErrorCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		EVRESSSOC	-	-	M	M	-	-	M	M	-	-	-	-	-	-
WeldingDetectionRes			-	-	O	M	-	-	O	M	-	-	-	-	-	-
	Header		-	-	M	M	-	-	M	M	-	-	-	-	-	-
		SessionId	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		Notification	-	-	O	M	-	-	O	M	-	-	-	-	-	-

		FaultCode	-	-	M	M	-	-	M	M	-	-	-	-	-	-
		FaultMsg	-	-	M	M	-	-	M	M	-	-	-	-	-	-
	Signature (see sep. table)		-	-	-	-	-	-	-	-	-	-	-	-	-	-
Body			-	-	M	M	-	-	M	M	-	-	-	-	-	-
	ResponseCode		-	-	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	-	-	-	-	-	-
	EVSEPresentVoltage		-	-	M	M	-	-	M	M	-	-	-	-	-	-

8.6.2.2 Common

[V2G2-762] If an SECC operates in the EIM Identification Mode and if the SECC chooses to send a sales tariff it shall send the sales tariff that was selected at the EVSE (e.g. using an RFID card) in the first entry (SAScheduleTuple) within the parameter SAScheduleList in the message ChargeParameterDiscoveryRes.

[V2G2-763] If an EVCC operates in the EIM Identification Mode it shall process the first entry (SAScheduleTuple) in the parameter SAScheduleList included in the message ChargeParameterDiscoveryRes only.

NOTE In PnC the parameter sales tariff might contain several tariffs allowing the EVCC to select one for internal charge schedule optimization. In EIM Identification Mode the selection of the sales tariff is implemented at the EVSE which means that the SECC transmits the selected sales tariff to the EVCC and the EV might use this information for charge schedule optimization.

8.6.2.3 AC

8.6.2.3.1 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 104, 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 104, 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 104, 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 104, column “Option: VAS”, sub column “SECC”.

8.6.2.3.2 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 104, 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 104, column “AC Charging PnC”, 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 104, 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 104, 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 104, 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 104, 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 104, 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 104, column “Certificate Update”, sub column “SECC”.

8.6.2.4 DC

8.6.2.4.1 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 104, 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 104, 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 104, 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 104, column “Option: VAS”, sub column “SECC”.

8.6.2.4.2 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 104, 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 104, 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 104, 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 104, 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 104, 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 104, 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 104, 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 104, 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 PaymentServiceSelectionReq message.

Figure 97 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 PaymentServiceSelectionReq (refer to subclause 8.4.3.5.2) and ChargeParameterDiscoveryReq (refer to subclause 8.4.3.8.2).

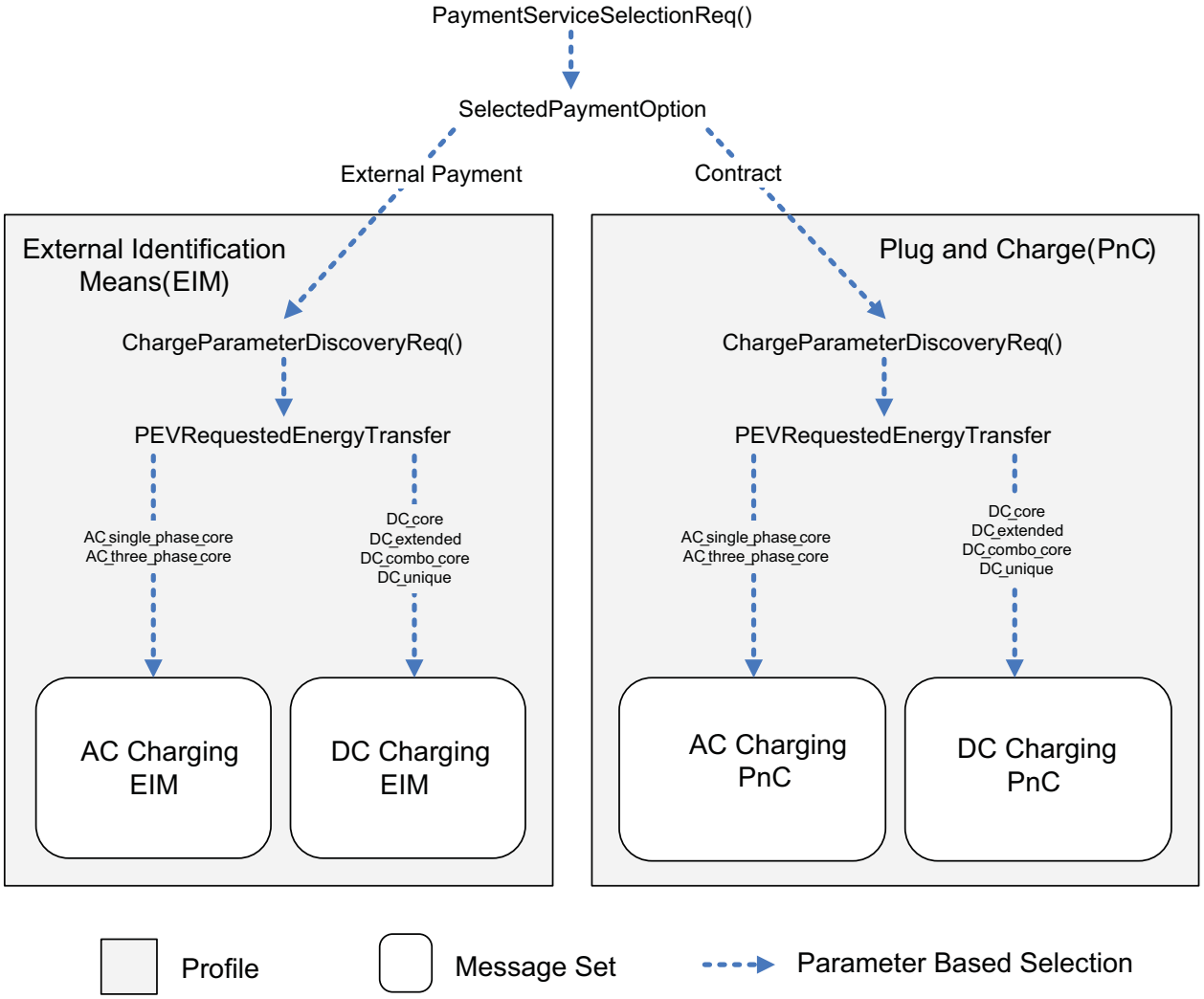


Figure 97 — Selection of Message Sets

[V2G2-402] If an EVCC sends the payment option “External Payment” in the parameter “SelectedPaymentOption” of the message “PaymentServiceSelectionReq” and sends “AC_single_phase_core” or “AC_three_phase_core” in the parameter “RequestedEnergyTransferMode” 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 “PaymentServiceSelectionReq ” and sends “DC_core” or “DC_extended” or “DC_combo_core” or “DC_unique” in the parameter “RequestedEnergyTransferMode” 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 “PaymentServiceSelectionReq” and sends “AC_single_phase_core” or “AC_three_phase_core” in the parameter “RequestedEnergyTransferMode” of the Message “ChargeParameterDiscoveryReq” it shall use the Message Set “AC Charging PnC”.

[V2G2-405] If an EVCC sends the payment option “Contract” in the parameter “SelectedPaymentOption” of the message “PaymentServiceSelectionReq” and sends “DC_core” or “DC_extended” or “DC_combo_core” or “DC_unique” in the parameter “RequestedEnergyTransferMode” 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 PnC” or "DC Charging PnC". 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 “ReceiptRequired” 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 “ReceiptRequired” 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 “TRUE” in the message “ChargingStatusRes”.

[V2G2-787] If an EVCC selected the Message Set “DC Charging PnC” as described in subclause 8.6.3.1 and an SECC requires a metering receipt from an EVCC, the SECC shall use the optional parameter “ReceiptRequired” in the message “CurrentDemandRes” and set its value to “TRUE”.

[V2G2-788] If an EVCC selected the Message Set “DC 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 use the optional parameter “Receipt Required” in the message “CurrentDemandRes” and set its value to “FALSE”.

[V2G2-789] If an EVCC selected the Message Set “DC Charging PnC” as described in subclause 8.6.3.1 it shall use the Message Set “Metering Receipt” if the SECC includes the parameter “ReceiptRequired” with value “TRUE” in the message “CurrentDemandRes”.

[V2G2-691] When operating in EIM Identification Mode, the SECC shall always set the value for the parameter ReceiptRequired to 'FALSE' when sending the message ChargingStatusRes**.**

NOTE It could be valuable for the EV to validate the amount of energy transferred. If the EV finds a divergence between the received amount of energy and the amount of energy indicated by EVSE, it may stop the charging process and make a proprietary error code entry for later analysis (Detection of "T-branches").

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 Installation”.

[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 Installation”.

8.6.3.4 Certificate Update

[V2G2-412] If an SECC offers the certificate update 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 update 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

This subclause defines the reserved ServiceID ranges and the ranges that can be used implementation specific. Table 105 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 106 and Table 107). Also this subclause includes requirements defining the usage of ServiceDiscoveryReq, ServiceDiscoveryRes, ServiceDetailsReq, ServiceDetailsRes, PaymentServiceSelectionReq and PaymentServiceSelectionRes.

NOTE 1 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 ServiceIDs in the range from 1 to 4 as defined in this standard.

[V2G2-417] The EVCC and the SECC shall conform to the ServiceID, ServiceName, ServiceCategory as defined in Table 105.

Table 105 — Definition of ServiceID, Service Category, Service Name, and Service Scope

ServiceID (unsignedshort)	ServiceName	ServiceCategory	Description
0			Reserved by ISO/IEC
1	AC_DC_Charging	EVCharging	All charging services as defined by SupportedEnergyTransferMode in subclause 8.5.2.3.
2	Certificate	ContractCertificate	Service allowing to update or install Contract Certificates.
3	InternetAccess	Internet	Service for standard protocols like HTTP, HTTPs, FTP, etc.
4	UseCaseInformation	EVSEInformation	Service enabling the exchange of use case specific information about the EVSE.
5 – 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, PaymentOptionList, 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 105. 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 105.

[V2G2-425] The SECC shall respond with a negative ResponseCode '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 106 if service details for ServiceID 2 'Certificate' is requested

Table 106 — 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 8.4.3.11.
2	stringValue = Update	Service to update a Contract Certificate in the EVCC, according to 8.4.3.10.
4 – 60000		Reserved by ISO/IEC
60001 – 65535		Implementation specific use

[V2G2-429] The ServiceParameterList shall comply with Table 107 if service details for ServiceID 3 'InternetAccess' is requested.

Table 107 — 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 107, it shall use the service names and the assigned 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 2 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 PaymentServiceSelectionReq 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 ResponseCode 'FAILED_ServiceSelectionInvalid' if the EVCC provided a not previously retrieved ServiceID, ParameterSetID pair in the PaymentServiceSelectionReq.

[V2G2-774] If the EVCC and the SECC agreed on the usage of VAS "Internet Access" (Service ID 3, AuthorizationRes with ResponseCode = OK & EVSEProcessing = Finished) the SECC shall provide the VAS for the entire charging session.

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 and then started. The value of a timer is the duration from its most recent reset and start time to the present time. The monitoring of a V2G communication message exchange 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 SessionSetupRes and the PowerDeliveryRes, 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 the following timing concepts:

— Message timing: Monitors the timing between the request and the response of a Request-Response-Pair. This allows e.g. the EVCC application to initiate the error handling if no response is sent.

— Sequence timing: Monitors the timing of subsequent Request-Response-Pairs. This allows e.g. the SECC application to initiate the error handling if an expected next request message was not sent.

— Ongoing timing: Monitors the timing in case of sending a Request-Response-Pair repeatingly based on the parameter EVSEProcessing equal to "Ongoing". This allows an e.g. the EVCC to initiate the error handling in case the SECC exceeds the processing time.

— Communication Setup timing: Monitors the time from the moment of an established data link until the Session Setup message. It allows deciding if the communication setup was successful within a defined time.

— CableCheck timing: The monitoring of the Cable Check is carried out by the EV using the V2G_EVCC_CableCheck_Timer. It is started when the EV requests the EVSE to start the Cable Check, and ends when the EVSE has finished the Cable Check, or when the V2G_EVCC_CableCheck_Timer expires.

— PreCharge timing: The monitoring of the Pre Charge is carried out by the EV using the V2G_EVCC_PreCharge_Timer. It is started when the EV starts the Pre Charging by sending the first PreChargeReq message, and ends when the Pre Charging has finished, indicated by the EV determining that the EVSE output voltage, as measured inside the EV, has sufficiently been adjusted to the EV RESS voltage, or when the V2G_EVCC_PreCharge_Timer expires.

The timers are compared to predefined time values as decision criterion. The EVCC and the SECC distinguish 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 if not defined differently by requirements. 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 108. Timeouts and Performance Times are parameterized for messages separately to describe different processing times. Table 109 defines the values for each V2G message type.

Table 108 — 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_Ongoing_Timer	Ongoing Timer in the EVCC	x	
V2G_SECC_Ongoing_Timer	Ongoing Timer in the SECC		x
V2G_EVCC_Msg_Timeout (MessageType)	Timeout for the Message Timer The value is defined depending on the parameter MessageType as defined in Table 109.	x	
V2G_SECC_Msg_Performance_Time (MessageType)	Performance Time for the Message Timer The value is defined depending on the parameter MessageType as defined in Table 109.		x
V2G_EVCC_Sequence_Performance_Time	Performance Time for the Sequence Timer as defined in Table 109.	x	

V2G_SECC_Sequence_Timeout	Timeout for the Sequence Timer as defined in Table 109.		x
V2G_EVCC_Ongoing_Timeout	Timeout for Ongoing Timer	x	
V2G_SECC_Ongoing_Performance_Time	Performance Time for Ongoing Timer		x

Table 109 — EVCC and SECC Message sequence and session timing parameter values

Name	MessageType	Value [s]
V2G_EVCC_Msg_Timeout(MessageType)	SupportedAppProtocolReq	2
	SessionSetupReq	2
	ServiceDiscoveryReq	2
	ServiceDetailReq	5
	PaymentServiceSelectionReq	2
	PaymentDetailsReq	5
	AuthorizationReq	2
	ChargeParameterDiscoveryReq	2
	ChargingStatusReq	2
	MeteringReceiptReq	2
	PowerDeliveryReq	5
	CableCheckReq	2
	PreChargeReq	2
	CurrentDemandReq	0,25
	WeldingDetectionReq	2
	SessionStopReq	2
	CertificateInstallationReq	5
	CertificateUpdateReq	5

Name	MessageType	Value [s]
V2G_SECC_Msg_Performance_Time(MessageType)	SupportedAppProtocolRes	1,5
	SessionSetupRes	1,5
	ServiceDiscoveryRes	1,5
	ServiceDetailRes	4,5
	PaymentServiceSelectionRes	1,5
	PaymentDetailsRes	4,5
	AuthorizationRes	1,5
	ChargeParameterDiscoveryRes	1,5
	ChargingStatusRes	1,5
	MeteringReceiptRes	1,5
	PowerDeliveryRes	4,5
	CableCheckRes	1,5
	PreChargeRes	1,5
	CurrentDemandRes	0,025
	WeldingDetectionRes	1,5
	SessionStopRes	1,5
	CertificateInstallationRes	4,5
	CertificateUpdateRes	4,5
V2G_EVCC_Sequence_Performance_Time	(all messages)	40
V2G_SECC_Sequence_Timeout	(all messages)	60
V2G_EVCC_Ongoing_Timeout	Response messages with parameter EVSEProcessing equal to 'Ongoing'	60
V2G_SECC_Ongoing_Performance_Time	Response messages with parameter EVSEProcessing equal to 'Ongoing'	55

Figure 98 illustrates how the Message Timers, Sequence Timers, Timeouts, and Performance Times are applied in the EVCC and the SECC.

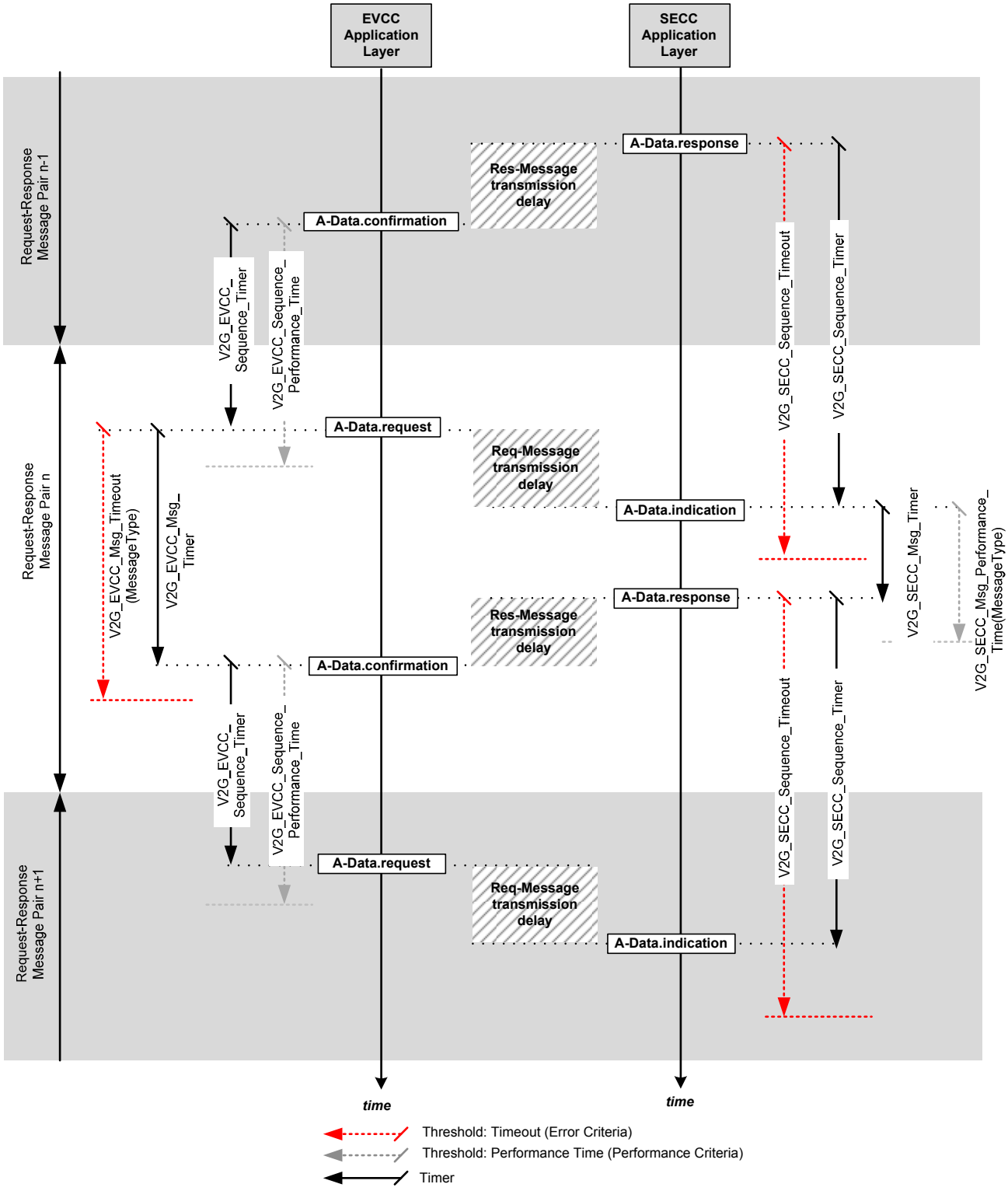


Figure 98 — Message sequence and session timing

[V2G2-434] The EVCC shall implement the EVCC specific Timeouts and Performance Times defined in Table 108 and Table 109.

[V2G2-435] The SECC shall implement the SECC specific Timeouts and Performance Times defend in Table 108 and Table 109.

8.7.2.2 EVCC Timing for Request-Response Message Pairs

[V2G2-436] The EVCC shall set the timeout V2G_EVCC_Msg_Timeout depending on the value MessageType as defined in Table 109, 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 Response-Request Message Sequence

[V2G2-441] The SECC shall set the timeout V2G_SECC_Sequence_Timeout to the value as defined in Table 109, 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 stop the V2G Communication Session..

NOTE 2 In this document receiving a request message is described by A-Data.indication. A-Data.indication (A_Msg="message name") signalizes the successful reception of a valid request message for the V2G message that is given by A_Msg where "Valid message" means that all mandatory elements are filled in so that it can be deserialized.

[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 request message is described by A-Data.indication.

[V2G2-445] The SECC 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 110. Table 111 define the values for the related performance times and the Timeouts.

Table 110 — 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_CableCheck_Timer	Cable Check Timer in the EVCC	x	
V2G_SECC_CableCheck_Timer	Cable Check Timer in the SECC		x
V2G_EVCC_PreCharge_Timer	PreCharge Timer in the EVCC	x	
V2G_SECC_PreCharge_Timer	PreCharge Timer in the SECC		x
V2G_EVCC_CommunicationSetup_Timeout	Timeout for the Communication Setup Timer as defined in Table 111.	x	
V2G_SECC_CommunicationSetup_Performance_Time	Performance Time for the Communication Setup Timer as defined in Table 111.		x
V2G_EVCC_CableCheck_Timeout	Timeout for the CableCheck Timer as defined in Table 111.	x	
V2G_SECC_CableCheck_Performance_Time	Performance Time for the CableCheck Timer as defined in Table 111.		x
V2G_EVCC_PreCharge_Timeout	Timeout for the PreCharge Timer as defined in Table 111.	x	
V2G_SECC_ PreCharge_Performance_Time	Performance Time for the PreCharge Timer as defined in Table 111.		x

[V2G2-605] The EVCC and SECC shall implement the timing parameter values defined in Table 111.

Table 111 — EVCC and SECC Message sequence and session timing parameter values

Parameter name	Value [s]	Implementation	
		EVCC	SECC
V2G_SECC_CommunicationSetup_Performance_Time	18		x
V2G_EVCC_CommunicationSetup_Timeout	20	x	
V2G_SECC_CableCheck_Performance_Time	38		x
V2G_EVCC_CableCheck_Timeout	40	x	
V2G_SECC_PreCharge_Performance_Time	5		x
V2G_EVCC_PreCharge_Timeout	7	x	

Figure 99 illustrates how the timing parameters defined in Table 110 are applied.

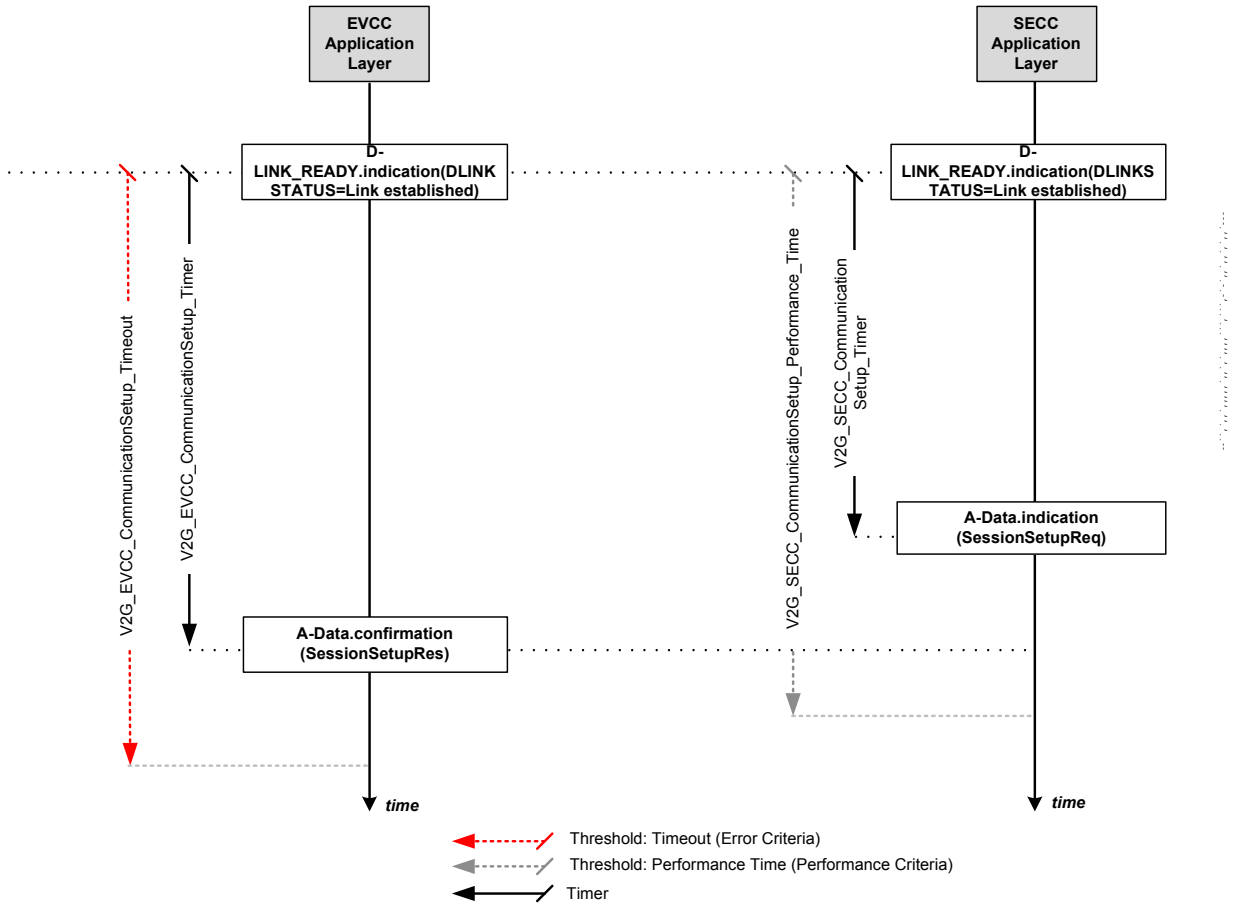


Figure 99 — CommunicationSetup timing

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 111, reset the V2G_EVCC_CommunicationSetup_Timer and start monitoring the V2G_EVCC_CommunicationSetup_Timer when a successful Data-Link establishment is indicated (D-LINK_READY.indication(DLINKSTATUS=Link established)).

[V2G2-447] The EVCC shall wait for the SessionSetupRes message.

[V2G2-448] The EVCC shall stop waiting for the SessionSetupRes message and stop monitoring the V2G_EVCC_CommunicationSetup_Timer when V2G_EVCC_CommunicationSetup_Timer is equal or larger than V2G_EVCC_CommunicationSetup_Timeout and no SessionSetupRes message was received. It shall then stop the V2G Communication Session.

NOTE 1 In this document receiving the response message “SessionSetupRes” is described by A-Data.confirmation(SessionSetupRes).

[V2G2-449] The EVCC shall stop waiting for the SessionSetupRes message and stop monitoring the V2G_EVCC_CommunicationSetup_Timer when V2G_EVCC_CommunicationSetup_Timer is smaller than V2G_EVCC_CommunicationSetup_Timeout and a SessionSetupRes message was received. It shall then process the response message as defined in subclause 8.8.

NOTE 2 In this document receiving the response message “SessionSetupRes” is described by A-Data.confirmation(SessionSetupRes).

8.7.3.3 SECC Timing for communication session setup

[V2G2-714] The SECC shall set the timeout V2G_SECC_CommunicationSetup_Performance_Time to the value as defined in Table 110, reset the V2G_SECC_CommunicationSetup_Timer and start monitoring the V2G_SECC_CommunicationSetup_Timer when a successful Data-Link establishment is indicated (D-LINK_READY.indication(DLINKSTATUS=Link established).

[V2G2-715] The SECC shall wait for the SessionSetupReq message.

[V2G2-716] The SECC shall stop waiting for SessionSetupReq and stop monitoring the V2G_SECC_CommunicationSetup_Timer when V2G_SECC_CommunicationSetup_Timer is equal or larger than V2G_SECC_CommunicationSetup_Performance_Time and no SessionSetupRes message was sent. It shall then apply [V2G2-034].

NOTE 1 In this document sending the response message “SessionSetupRes” is described by A-Data.response(SessionSetupRes).

8.7.3.4 EVCC Timing for EVSEProcessing parameter

[V2G2-710] If the EVCC receives a V2G response message with parameter EVSEProcessing equal to 'Ongoing' for the first time in a response message it shall start the timer V2G_EVCC_Ongoing_Timer and wait for parameter EVSEProcessing equal to 'Finished'.

[V2G2-711] If **[V2G2-710]** applies, the EVCC shall stop the V2G Communication Session when V2G_EVCC_Ongoing_Timer is equal or larger than V2G_EVCC_Ongoing_Timeout and no parameter EVSEProcessing equal to 'Finished' has been received.

8.7.3.5 SECC Timing for EVSEProcessing parameter

[V2G2-712] If the SECC sends a V2G response message with parameter EVSEProcessing equal to 'Ongoing' for the first time in a response message it shall start the timer V2G_SECC_Ongoing_Timer.

[V2G2-713] If **[V2G2-712]** applies, the SECC shall try to send ResponseCode equal to "FAILED" when V2G_SECC_Ongoing_Timer is equal or larger than V2G_SECC_Ongoing_Performance_Time and no parameter EVSEProcessing equal to 'Finished' has been sent. The SECC shall stop the V2G Communication Session.

8.7.3.6 EVCC Timing for cable check

[V2G2-700] The EVCC shall set the timeout V2G_EVCC_CableCheck_Timeout to the value as defined in Table 111, reset the V2G_EVCC_CableCheck_Timer and start monitoring the V2G_EVCC_CableCheck_Timer when sending the message CableCheckReq for the first time in a charging session.

NOTE 1 In this document, sending a request message is described by A-Data.request.

[V2G2-701] The EVCC shall wait for the Cable Check of the EVSE to finish indicated by the reception of a CableCheckRes with ResponseCode equal to “OK” and EVSEProcessing equal to “Finished”.

NOTE 2 In this document, receiving a response message is described by A-Data.confirmation.

[V2G2-702] The EVCC shall stop waiting for the Cable Check of the EVSE to finish and stop monitoring the V2G_EVCC_CableCheck_Timer when V2G_EVCC_CableCheck_Timer is equal or larger than V2G_EVCC_CableCheck_Timeout. It shall then apply the error handling as defined in subclause 8.8.

[V2G2-703] The EVCC shall stop waiting for the Cable Check of the EVSE to finish and stop monitoring the V2G_EVCC_CableCheck_Timer if V2G_EVCC_CableCheck_Timer is smaller than

V2G_EVCC_CableCheck_Timeout and a CableCheckRes message with ResponseCode equal to “OK” and EVSEProcessing equal to “Finished” 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.

8.7.3.7 EVCC Timing for pre charging

[V2G2-704] The EVCC shall set the timeout V2G_EVCC_PreCharge_Timeout to the value as defined in Table 111, reset the V2G_EVCC_PreCharge_Timer and start monitoring the V2G_EVCC_PreCharge_Timer when sending the message PreChargeReq for the first time in a charging session.

NOTE In this document, sending a request message is described by A-Data.request.

[V2G2-705] The EVCC shall wait for the Pre Charging to finish, indicated by the EV determining that the EVSE output voltage, as measured inside the EV, has sufficiently been adjusted to the EV RESS voltage.

[V2G2-706] The EVCC shall stop waiting for the Pre Charging to finish and stop monitoring the V2G_EVCC_PreCharge_Timer when V2G_EVCC_PreCharge_Timer is equal or larger than V2G_EVCC_PreCharge_Timeout. It shall then apply the error handling as defined in subclause 8.8.

[V2G2-707] The EVCC shall stop monitoring the V2G_EVCC_PreCharge_Timer when V2G_EVCC_PreCharge_Timer is smaller than V2G_EVCC_PreCharge_Timeout and Pre Charging has finished, indicated by the EV determining that the EVSE output voltage, as measured inside the EV, has sufficiently been adjusted to the EV RESS voltage. It shall then process the response message as defined in subclause 8.8.

8.7.4 V2G message synchronization with IEC 61851-1 signalling

8.7.4.1 Overview

ISO 15118 based charging control extends the IEC 61861-1 signaled charging. For this, the messaging on application layer is synchronised with the CP States defined in IEC 61851-1.

ISO 15118 based messaging is able to manage the AC and DC charging process for a complete charging session from the beginning to the end in 5% duty cycle case.

For AC, ISO 15118 also allows to start charging based on IEC 61851-1 (Basic Charging, BC) and switch to ISO 15118 based charging control (High Level Communication Control, HLC-C) later.

In this subclauses, terms and definitions in requirements are applied as defined in *Part 3* and IEC 61851-1.

Figure 100 shows an example for AC and DC charging with BC and HLC-C in relation to the phases Data Link Setup, V2G Setup, and V2G Charging Loop during a V2G Communication Session. The figure also shows an example for the most important entry and exit conditions for the phases. In case of AC a BC phase can be applied. In general, BC before HLC-C requires a nominal Duty Cycle instead of a 5% Duty Cycle.

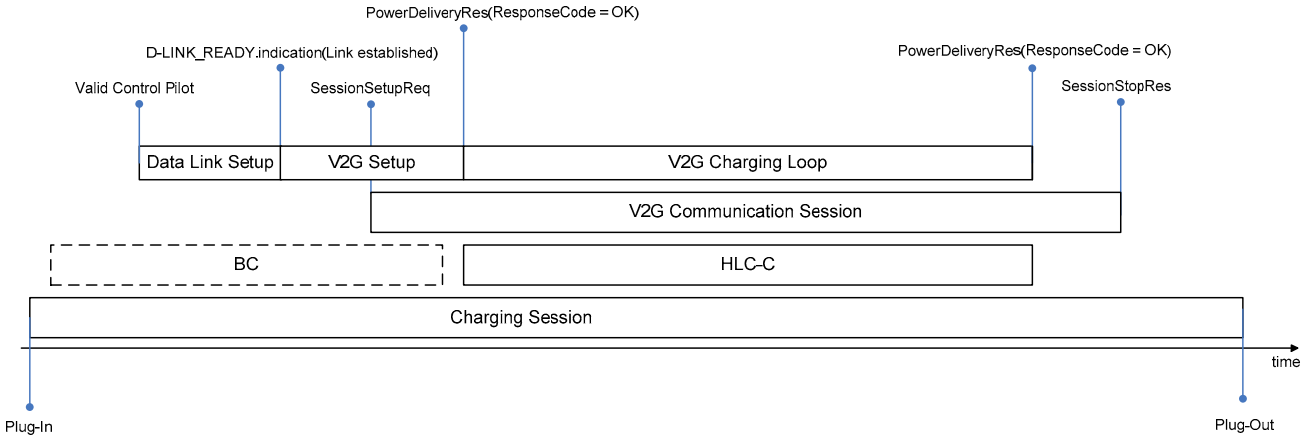


Figure 100 — AC Example for BC and HLC-C charging in relation to a V2G Communication Session

The V2G Charging Loop is defined as V2G messaging phase for controlling the charging process by ISO 15118 in normal operation. The charging phase under control of ISO 15118 is defined as High Level Controlled Charging (HLC-C).

The entry and exit conditions for the V2G Charging Loop are as follows:

— Entry Condition: PowerDeliveryRes message with parameter ResponseCode equal to OK after PowerDeliveryReq with ChargeProgress equals "Start".

— Exit Condition: PowerDeliveryRes with parameter ResponseCode equal to OK after PowerDeliveryReq with ChargeProgress equals "Stop".

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 requirements as defined in subclause 8.7.4.2, 8.7.4.3, and 8.7.4.4.

8.7.4.2 Common requirements

The following requirements apply for the EV:

[V2G2-836] The EVCC shall use the message PowerDeliveryReq with parameter ChargeProgress set to “Start” to start HLC-C based charging.

[V2G2-837] If **[V2G2-836]** applies and the PowerDeliveryRes message has been received with ResponseCode set to OK, the EV shall apply the Charging Limits according to the V2G messaging.

[V2G2-838] The EV shall stop charging (no current drawn by EV) before sending the message PowerDeliveryReq with the parameter ChargeProgress set to “Stop”.

[V2G2-839] The EVCC shall use the message PowerDeliveryReq with parameter ChargeProgress set to “Stop” to stop HLC-C based charging.

[V2G2-840] The EVCC shall use the message PowerDeliveryReq with parameter ChargeProgress set to “Renegotiation” to initiate a renegotiation messaging.

[V2G2-841] The EVCC shall not send the message PowerDeliveryReq with parameter ChargeProgress set to “Renegotiation” before sending PowerDeliveryReq with parameter ChargeProgress set to “Start. (see **[V2G2-837]**).

[V2G2-842] If **[V2G2-841]** applies the EVCC shall set the parameter ChargeProgress to “Start” in the next following message PowerDeliveryReq to apply the negotiated charging limits after a renegotiation.

In General, ISO 15118 is based on the requirements as defined in IEC 61851-1.

[V2G2-843] An ISO 15118 enabled EV shall conform to IEC 61851-1.

[V2G2-844] During HLC-C, the EV shall apply all charging limits as negotiated by ISO 15118 in addition to the limits defined by IEC 61851-1.

NOTE For DC charging only ISO 15118 limits apply.

[V2G2-845] If the Message Set “EIM charging AC” or “EIM charging DC” is selected and the parameter EVSEProcessing is set to „Ongoing_WaitingForCustomerInteraction“ in AuthorizationRes the EV shall resent the AuthorizationReq message.

[V2G2-731] If **[V2G2-508]** or **[V2G2-728]** applies, the EVCC can establishing a new or resumed V2G Communication Session by applying **[V2G2-014]**.

[V2G2-737] If the EVCC measures CP State E or F it shall stop the V2G Communication Session and continue with **[V2G2-728]**.

The following requirements apply for the EVSE:

[V2G2-850] ISO 15118 enabled EVSE shall conform to IEC 61851-1.

[V2G2-854] If identification mode EIM has been selected by the parameter SelectedPaymentOption equal to "External Payment" in message ServicePaymentSelectReq and no positive EIM information is available an SECC shall set the parameter EVSEProcessing to „Ongoing_WaitingForCustomerInteraction“ in AuthorizationRes.

[V2G2-855] If identification mode PnC has been selected by the parameter SelectedPaymentOption equal to "Contract" in message ServicePaymentSelectReq and no positive PnC information is available an SECC shall set the parameter EVSEProcessing to „Ongoing“ in AuthorizationRes.

[V2G2-856] If an SECC positive authorization information is available the SECC shall set the parameter EVSEProcessing to „finished“ in AuthorizationRes.

NOTE2 **[V2G2-856]** applies the same behaviour for positive authorization for identification mode EIM and PnC.

[V2G2-733] If **[V2G2-571]** or **[V2G2-539]** or **[V2G2-729]** applies, the SECC can allow to establish a new or resumed V2G Communication Session by applying **[V2G2-027]**.

8.7.4.3 AC specific requirements

The following requirements apply for the EV:

Before entering HLC-C in PWM 5% case, the EV shall signal CP State B.

[V2G2-930] The EV shall measure a PWM of 5% or nominal duty cycle before sending the message ChargeParameterDiscoveryReq.

[V2G2-846] In case of PWM is 5% or nominal PWM without BC the EV shall signal CP State B before sending the first PowerDeliveryReq with ChargeProgress equals "Start" within V2G Communication SessionPowerDeliveryReq.

[V2G2-847] The EV shall signal CP State C or D no later than 250ms after sending the first PowerDeliveryReq with ChargeProgress equals "Start" within V2G Communication SessionPowerDeliveryReq.

[V2G2-848] The EV shall signal CP State B no later than 250ms after sending PowerDeliveryReq with ChargeProgress equals "Stop".

[V2G2-849] If **[V2G2-847]** applies and no error has been identified, the EV shall not change the CP State until **[V2G2-848]** applies.

NOTE 1 In case of renegotiation the Contactor stays closed to allow charging based on the existing charging limits during renegotiation.

The following requirements apply for the EVSE:

The EVSE will provide a PWM of 5 % to signal to the EV that ISO 15118 is required for charging. With a PWM of 10 – 95 % and EIM, ISO/IEC15118 can be supported in parallel to IEC 61851-1.

[V2G2-931] The EVSE shall signal a PWM of 5 % or nominal duty cycle after sending the message AuthorizationRes.

[V2G2-851] If no energy is available an EVSE shall apply a duty cycle of 100 %.

[V2G2-852] If no positive authorization information from an EIM has been received the EVSE shall apply PWM equal to 5 %.

[V2G2-853] If an EVSE receives positive authorization information from an EIM the EVSE shall apply a nominal duty cycle.

The SECC waits for positive authorization and availability of energy before entering the Charging Loop.

[V2G2-857] If **[V2G2-856]** applies and the EVSE PWM signal equals 5 % the EVSE shall not change the PWM value until the end of a V2G communication session.

NOTE 2 If the EVSE cannot provide energy (EVSE signals a PWM equal to 100 %) the EVSE can provide the time of availability of energy in the parameter SASchedule in the ChargeParameterDiscoveryRes message. The EV then has the opportunity to start with a PowerDeliveryReq with parameter ChargeProgress equal to "Stop" to start with pausing.

The EVSE Contactor must be closed before sending the PowerDeliverRes message.

[V2G2-858] After receiving a PowerDeliveryReq with parameter ChargeProgress equal to "Start" the SECC shall monitor the Contactor status and start the V2G_SECC_Msg_Performance_Timer.

[V2G2-859] In case of High-level communication based charging, the SECC shall not close the Contactor before receiving PowerDeliveryReq(ChargeProgress = Start).

[V2G2-860] If no error is detected, the SECC shall close the Contactor no later than 3s after measuring CP State C or D.

[V2G2-861] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 109, if the SECC measures a closed Contactor.

[V2G2-862] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = FAILED_ContactorError” if V2G_SECC_Msg_Timer is equal or larger than V2G_SECC_Msg_Performance of 'PowerDelivery' according to Table 109. and SECC measures opened Contactor.

[V2G2-863] After receiving a PowerDeliveryReq with parameter ChargeProgress equal to "Stop", the SECC shall monitor the Contactor status and start the V2G_SECC_Msg_Performance_Timer.

[V2G2-864] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 109, if the SECC measures an open Contactor.

[V2G2-865] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = FAILED_ContactorError” if V2G_SECC_Msg_Timer is equal or larger than V2G_SECC_Msg_Performance of 'PowerDelivery' according to Table 109 and SECC measures a closed Contactor.

[V2G2-866] If **[V2G2-862]** or **[V2G2-865]** applies, the SECC shall set the PWM to 100 % no later than 250 ms after sending PowerDeliveryRes.

NOTE 3 ISO 15118 does not require a specific implementation for controlling the Contactor in an EVSE. Control of the power Contactor depends on the EVSE architecture.

NOTE 4 For AC, the latest point in the messaging sequence for providing energy (closing Contactor, Voltage at EV inlet) is before sending the PowerDeliveryRes message by the SECC. If a CP State C or D was signalled earlier, the EVSE may provide energy earlier. E.g. if the EVSE and the EV support BC before HLC-C the EVSE will signal a PWM between 10 % and 96 %, the EV will signal CP State C or D, and the EVSE close the Contactor and provide energy as defined IEC 61851-1. In any case the timing limits as defined in IEC 61851-1 must be fulfilled.

8.7.4.4 DC specific requirements

The following requirements apply for the EV:

[V2G2-912] After receiving a ChargeParameterDiscoveryRes message and before sending a CableCheckReq message, the EVCC shall change to CP State C or D as defined in IEC 61851-1.

[V2G2-913] After sending a PowerDeliveryReq message with ChargeProgress equal to “STOP” and receiving the corresponding PowerDeliveryRes message, the EVCC shall change to CP State B as defined in IEC 61851-1 before sending the next Request Message.

[V2G2-914] If **[V2G2-912]** applies and no error has been identified, the EV shall not change the CP State until **[V2G2-913]** applies.

The following requirements apply for the EVSE:

[V2G2-915] The EVSE shall apply a CP duty cycle of 5 % from start of Data Link Setup until end of V2G Communication Session.

The SECC waits for positive authorization and availability of energy before entering the Charging Loop.

[V2G2-916] After receiving a CableCheckReq message, the SECC shall monitor the CP State and start the V2G_SECC_Msg_Performance_Timer.

[V2G2-917] The SECC shall respond with CableCheckRes containing “ResponseCode = "OK" within V2G_SECC_Msg_Performance_Time according to Table 109, if the SECC measures a CP State C or D.

[V2G2-918] The SECC shall respond with CableCheckRes containing “ResponseCode = "FAILED" within V2G_SECC_Msg_Performance_Time according to Table 109, if the SECC measures no CP State C or D.

[V2G2-919] After sending a PowerDeliveryRes message with parameter ResponseCode set to "OK" in response to a PowerDeliveryReq with parameter ChargeProgress set to “Stop”, the SECC shall try to measure CP state B.

[V2G2-920] After receiving a WeldingDetectionReq message or a SessionStopReq message, the SECC shall wait for a maximum of V2G_SECC_Msg_Performance_Time to measure CP state B.

[V2G2-921] After receiving a WeldingDetectionReq message or a SessionStopReq message, the SECC shall send the corresponding Response message with parameter ResponseCode set to 'OK' within V2G_SECC_Msg_Performance_Time according to Table 103, if CP State B was measured.

[V2G2-922] After receiving a WeldingDetectionReq message or a SessionStopReq message, the SECC shall send the corresponding Response message with parameter ResponseCode set to 'FAILED' within V2G_SECC_Msg_Performance_Time according to Table 103, if CP State B was not measured.

NOTE If the EVSE cannot provide energy (EVSE signals a PWM equal to 100 %) the EVSE can provide the time of availability of energy in the parameter SASchedule in the ChargeParameterDiscoveryRes message. The EV then has the opportunity to start with a PowerDeliveryReq with parameter ChargeProgress equal to "Stop" to start with pausing.

8.8 Message sequencing and error handling

8.8.1 Overview

In a V2G Communication Session the EVCC and the SECC exchange Request-Response Message Pairs based on a predefined sequence. In this document this is referenced as Request-Response Message Sequence. These Request-Response Message Sequences allow both sides to synchronize the process in any situation and to control the communication between two V2G Entities.

The basic error handling concept in ISO 15118 is based on application level timers for Request-Response Message Sequences. This enables a V2G Entity to manage any processing error and communication error on application level after waiting for the expected behaviour until a specified timeout. Therefore, a V2G Entity can decide on a successful processing after waiting for the positive result until a timeout. In case there is no error in the communication layers below the application layer, the application has the option to terminate the TCP communication in case of application error. A terminated TCP connection before a SessionStopRes is always interpreted as an error by the EVCC or the SECC.

Besides the timeout concept defined in this document, the standard does not limit additional error detection mechanisms as long as such mechanisms do not lead to incompatibility.

Within a Charging Session, an EVCC can establish a new V2G Communication Session after an error by applying the V2G communication state processing as described in subclause 7.4. In this case, the EVCC and the SECC start the communication in the same way as for the first V2G session setup.

In general, the processing time of a request message in an SECC is limited by the performance requirements defined in subclause 8.7.2. If an SECC requires more processing time it may apply the parameter EVSEProcessing (set to Ongoing) to avoid a timeout at the EVCC by explicitly indicating that the processing is not finished yet and the EVCC has to wait for finishing the processing before proceeding with the message sequence. The EVCC will then repeat the request message periodically until the EVSE signals the finishing of the processing by the parameter EVSEProcessing (set to Finished).

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 ResponseCode included in the Response Message of the SECC. Depending on the value in the ResponseCode 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 ResponseCode 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.

— FAILED:
Any Value starting with “FAILED” or “FAILED_” indicates a negative response. Detailed information may be provided by FAILED_<additional info>. This information may be used to differentiate the reaction on the negative response.

The processing of the remaining parameters in a response message in the EVCC depends on the parameter ResponseCode. If the ResponseCode contains a value starting with OK indicating that no error was detected, the EVCC can process other parameters of the response message mandatorily or optionally as defined in subclause 8.6. If the ResponseCode contains a value starting with FAILED indicating that an error was detected, the EVCC shall ignore other parameters of the response message.

As long as the message content of the received request message is valid and the request message is accepted in the current SECC state (sequencing), the ResponseCode is set to OK. This is also true for response messages with EVSEProcessing set to Ongoing. With the ResponseCode set to OK the SECC indicates that the request message is accepted and valid, but cannot be finally answered at this moment and more time is needed to send the valid values in the response message. When the valid values are sent in the response message, the SECC indicates this with the parameter EVSEProcessing (set to Finished) and the EVCC can proceed with the sequence as expected for the first request message before receiving the parameter EVSEProvessing (set to Ongoing). If the SECC sets the ResponseCode to FAIL it identified an error in processing or an invalid message and the standard error handling applies.

8.8.3 ResponseCode 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.

[V2G2-734] If the ResponseCode contains a value starting with OK, the EVCC shall process the other parameters of the response message as defined in subclause 8.8.

[V2G2-735] If the ResponseCode contains a value starting with FAILED, the EVCC shall ignore the other parameters of the response message.

[V2G2-736] If the SECC sends a ResponseCode containing a value starting with FAILED all mandatory parameters shall be filled in with arbitrary XSD conform values. Additionally, all values shall be chosen in a way that results in a minimal size of the response message.

In general each response message can contain two types of ResponseCode values '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 any request message except SessionSetupReq is not equal to the SessionID value stored for the currently active V2G Communication Session.

[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 'PaymentServiceSelectionRes' shall contain the ResponseCode 'FAILED_PaymentSelectionInvalid' if the SelectedPaymentOption contained in the PaymentServiceSelectionReq message was not part of the offered PaymentOptionList of ServiceDiscoveryRes.

[V2G2-467] The message 'PaymentServiceSelectionRes' shall contain the ResponseCode 'FAILED_ServiceSelectionInvalid' if the SelectedServiceList contained in the PaymentServiceSelectionReq message contains a ServiceID which was not contained in the offered ServiceList of ServiceDiscoveryRes.

[V2G2-804] The message 'PaymentServiceSelectionRes' shall contain the ResponseCode 'FAILED_NoChargeServiceSelected' if the SelectedServiceList contained in the PaymentServiceSelectionReq message does not contain a ServiceID value equal to 1 which was offered by the SECC with the parameter ChargeService in the 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 Table 109.

[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 109.

[V2G2-472] The message 'CertificateUpdateRes' shall contain the ResponseCode FAILED_ContractCanceled if the EMAID 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 expired.

[V2G2-824] The message 'PaymentDetailsRes' shall contain the ResponseCode 'FAILED_CertificateExpired' if the Contract Certificate contained in the PaymentDetailsReq message in attribute ContractSignatureCertChain is not yet valid.

[V2G2-475] The message 'AuthorizationRes' shall contain the ResponseCode 'FAILED_ChallengeInvalid' if the challenge response contained in the AuthorizationReq message in attribute GenChallenge is not valid versus the provided GenChallenge in PaymentDetailsRes.

[V2G2-476] The message 'ChargeParameterDiscoveryRes' shall contain the ResponseCode 'FAILED_WrongEnergyTransferMode' if the content of attribute 'RequestedEnergyTransferMode' 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.

[V2G2-690] If the SECC is capable of comparing the validity of a contractCertificate with the current time, the SECC shall set the responseCode for the PaymentServiceSelectionRes to OK_CertificateExpiresSoon if the received certificate will expire in 21 days or fewer.

[V2G2-693] The message AuthorizationRes shall contain the ResponseCode 'FAILED_CertificateRevoked' if the SECC matches the ContractCertificate with a CRL and the ContractCertificate is marked as revoked.

[V2G2-694] The message CertificateInstallationRes shall contain the ResponseCode 'FAILED_CertificateRevoked' if the SECC matches the OEM Provisioning Certificate with a CRL and the ContractCertificate is marked as revoked.

[V2G2-695] The message CertificateUpdateRes shall contain the ResponseCode 'FAILED_CertificateRevoked' if the SECC matches the ContractCertificate with a CRL and the ContractCertificate is marked as revoked.

NOTE ResponseCodes that are not defined in this chapter can be used implementation specific.

Table 112 shows an overview on the ResponseCode values and the messages as defined before.

Table 112 — Overview on application of ResponseCodes

ResponseCode (Enumeration)	V2G application layer protocol handshake messages	V2G application layer messages												
	supportedAppProtocolRes	SessionSetupRes	ServiceDiscoveryRes	ServiceDetailRes	ServicePaymentSelectionRes	PaymentDetailsRes	AuthorizationRes	ChargeParameterDiscoveryRes	PowerDeliveryRes	ChargingStatusRes	MeteringReceiptRes	CertificateupdateRes	CertificateinstallationRes	SessionStopRes
OK	x	x	x	x	x	x	x	x	x	x	x	x	x	x
OK_CertificateExpiresSoon					x									
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_CertificateRevoked						x						x	x	
FAILED_NoCertificate Available						x						x	x	
FAILED_CertChainError												x		
FAILED_ContractCanceled												x		
FAILED_ChallengeInvalid							x							
FAILED_WrongEnergy TransferMode								x						
FAILED_WrongCharge Parameter								x						
FAILED_ChargingProfile Invalid									x					
FAILED_TariffSelection Invalid									x					
FAILED_PowerDelivery NotApplied									x					
FAILED_MeteringSignature NotValid											x			

FAILED_NoChargeServiceSelected					x									
FAILED_ContactorError									x					
FAILED_CertificateNotAllowedAtThisEVSE						x								

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 1 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] The EVCC shall calculate a charging profile that does not violate the parameter limits of a specific SAScheduleTuple (represented by a particular TupleID) included in SASchedules, which has been received in the latest ChargeParameterDiscoveryRes message.

[V2G2-674] If the EVCC sends a 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 currently applied parameter ChargingProfile (same SAScheduleTupleID as before the renegotiation) or to send a new parameter ChargingProfile that does not violate the parameter limits of a specific SAScheduleTuple (represented by a particular TupleID) included in SASchedules, which has been received in the latest ChargeParameterDiscoveryRes message.

[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 a 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 2 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 3 The decision of an EV for accepting or rejecting an EVSE initiated renegotiation 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 behaviour defining all valid Request-Response Message Sequences for AC is shown in Figure 101 and for DC is shown in Figure 102.

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 1 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 109.

[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 = FAILED' of 'SessionSetupRes' according to Table 109.

[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 = FAILED' of ' ServiceDiscoveryRes' according to Table 109.

[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.

[V2G2-490] After receiving the ServiceDiscoveryRes, the EVCC shall send a PaymentServiceSelectionReq, 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 = FAILED' of 'ServiceDetailRes' according to Table 109.

[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 = FAILED' of 'PaymentServiceSelectionRes' according to Table 109.

[V2G2-509] After receiving the PaymentServiceSelectionRes, the EVCC shall send a AuthorizationReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-493] After receiving the ServiceDetailRes, the EVCC shall send a PaymentServiceSelectionReq 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 PaymentServiceSelectionRes, 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” or “DC Charging PnC” is selected and the EVCC does not intend to use the Message Sets “Certificate Install” or “Certificate Update”.

[V2G2-496] After receiving the PaymentServiceSelectionRes, the EVCC shall send a CertifiacteInstallationReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the ServiceDetailRes indicates that CertificateInstallation is available and the EVCC wants to use the Message Set “Certificate Installation".

[V2G2-497] After receiving the PaymentServiceSelectionRes, the EVCC shall send a CertifiacteUpdateReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the ServiceDetailRes indicates that CertificateUpdate 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 = FAILED' of 'CertifiacteInstallationRes' according to Table 109.

[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 109.

[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 = FAILED' of 'PaymentDetailsRes' according to Table 109.

[V2G2-503] After receiving the PaymentDetailsRes, the EVCC shall send a AuthorizationReq 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 = FAILED' of 'AuthorizationRes' according to Table 109.

[V2G2-505] After receiving the AuthorizationRes 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 = FAILED' of 'ChargeParameterDiscoveryRes' according to Table 109.

[V2G2-799] The EVCC shall stop the V2G Communication Session when V2G_EVCC_Msg_Timer is equal or larger than V2G_EVCC_Msg_Timeout or 'ResponseCode = FAILED' of 'PowerDeliveryRes' according to Table 109.

[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 = FAILED' of 'SessionStopRes' according to Table 109.

[V2G2-508] After receiving the SessionStopRes, the EVCC shall terminate the communication by applying **[V2G2-025]**.

[V2G2-728] After the EVCC stopped the V2G Communication Session with an error it shall terminate the communication by applying **[V2G2-025]**.

The EVCC may renegotiate the charging schedule as follows:

[V2G2-683] If an EVCC initiated a 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 AuthorizationRes, the EVCC shall send an empty AuthorizationReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time as long as the parameter EVSEProcessing is equal to 'Ongoing'.

NOTE 2 The EV only sends a AuthorizationReq message containing a Signature, Id and GenChallenge included as the first AuthorizationReq message. Consecutive messages that are only required when a AuthorizationRes message was received with EVSEProcessing set to Ongoing are sent without this information.

[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 = FAILED' of 'ChargingStatusRes' according to Table 109.

[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 ChargProgress is set to’Stop’, 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.

[V2G2-516] 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 the Parameter “ReceiptRequired" 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 = FAILED' of 'MeteringReceiptRes' according to Table 109.

[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 ChargeProgress set to ’Stop’, 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 ChargeProgress is set to ‘Stop’ 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 ChargeProgress set to ’Renegotiate’, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-522] An EV can decide 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 ChargeProgress set to ’Renegotiate’, 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 1 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 EV initiated renegotiation but triggered by the EVSE as described in [V2G2-689].

NOTE 2 The decision for initiating renegotiation on EV side is up to the EV. E.g. if a ChargingProfile, once accepted by the SECC, has to be changed by the demand of user’s convenience or request from grid utilities, EVCC can trigger a renegotiation process that exchanges the SASchedule and ChargingProfile again and make a new agreement with SECC.

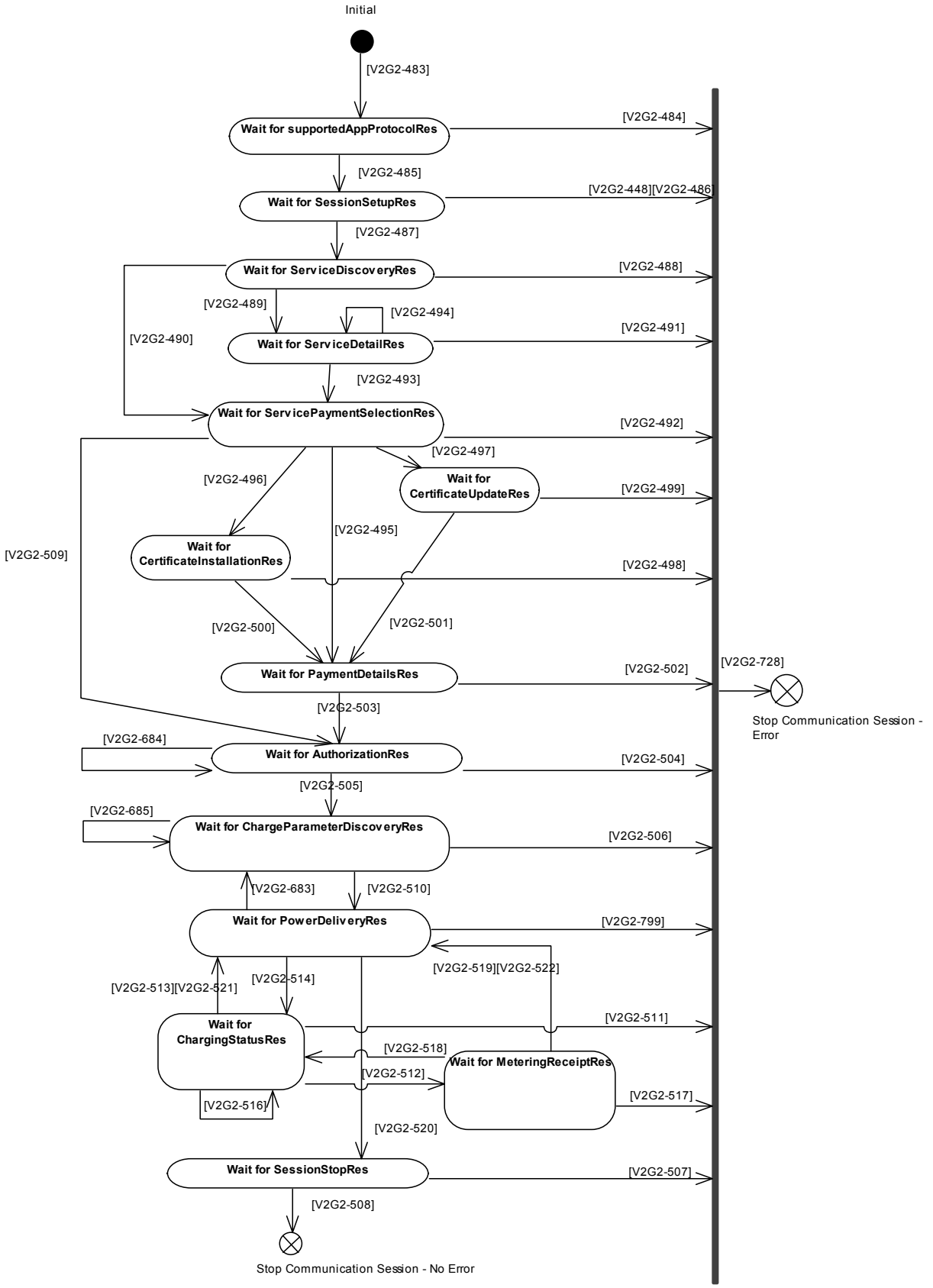


Figure 101 — EVCC Communication states for AC V2G messaging

8.8.4.2.3 DC specific requirements

[V2G2-880] All DC_EVSEStatusCodes in Table 98 for which there are no explicit requirements in this document are for information purpose only. They shall not influence the EVSE charging process.

NOTE 1 EVSE_EmergencyShutdown" may be used by the SECC to inform the EVCC that the EVSE is in the process of, or has just executed, an emergency shutdown. However, to force the EV to participate in the emergency shutdown, the SECC does not use this value, but instead uses the CPL in accordance with IEC 61851-1. In all other cases, the expected value for DC_EVSEStatusCode is "EVSE_Ready" or "EVSE_IsolationMonitoringActive", even though they do not influence the EV charging process.

[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 = FAILED' of 'CableCeckRes' according to Table 109.

[V2G2-617] After receiving the CableCheckRes, the EVCC shall send a new 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 = FAILED' of ' PreChargeReq ' according to Table 109.

[V2G2-527] After receiving the CurrentDemandRes, the EVCC shall send a PowerDeliveryReq with parameter ChargeProgress is set to 'Stop', 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 2 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.

[V2G2-530] 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 as a response to a previous PowerDeliveryReq message with ChargeProgress equal to "Stop", the EVCC shall send a SessionStopReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time if the EV does not wish to perform a Welding Detection.

[V2G2-531] After receiving the CurrentDemandRes with the parameter "ReceiptRequired" set to value "FALSE", 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.

[V2G2-790] After receiving the CurrentDemandRes, the EVCC shall send a MeteringReceiptReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the parameter “ReceiptRequired" in the CurrentDemandRes message is set to value "TRUE”.

[V2G2-791] 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 toTable 109 — EVCC and SECC Message sequence and session timing parameter values.

[V2G2-792] After receiving the MeteringReceiptRes, the EVCC shall send a CurrentDemandReq, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time., if the charge process is continued.

[V2G2-793] After receiving the MeteringReceiptRes, the EVCC shall send a PowerDeliveryReq with parameter ChargeProgress set to ’Stop’, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the charge process is stopped by the EV.

[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 = FAILED' of 'CurrentDemandRes' according to Table 109.

[V2G2-533] After receiving the PowerDeliveryRes as a response to a previous PowerDeliveryReq message with ChargeProgress equal to “Stop”, the EVCC shall send a WeldingDetectionReq while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time, if the EV wishes to perform a Welding Detection.

[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 = FAILED' of 'WeldingDetectionRes' according to Table 109.

[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.

The EVCC may renegotiate the charging schedule as follows:

[V2G2-686] An EV can decide to perform a renegotiation for adapting its charge schedule as follows. After CurrentDemandRes has been received and the EVCC intends to renegotiate, it shall initiate a renegotiation by sending a PowerDeliveryReq with parameter ChargeProgress set to ’Renegotiate’, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

[V2G2-794] An EV can decide to perform a renegotiation for adapting its charge schedule as follows. An EV can decide to perform a renegotiation for adapting its charge schedule as follows. After a MeteringReceiptRes has been received, the EVCC shall initiate a renegotiation by sending a PowerDeliveryReq with parameter ChargeProgress set to ’Renegotiate’, while V2G_EVCC_Sequence_Timer is smaller than V2G_EVCC_Sequence_Performance_Time.

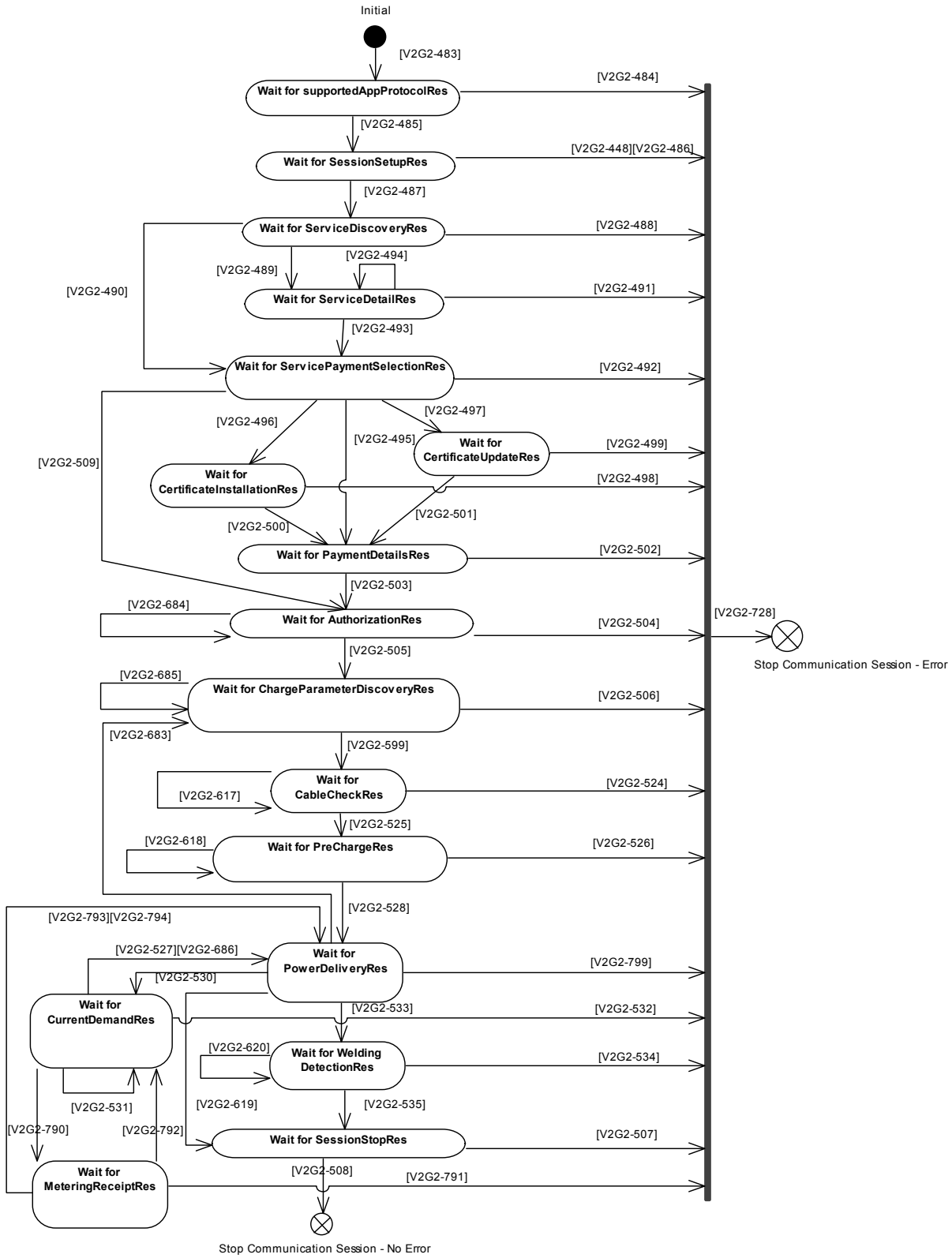


Figure 102 — EVCC Communication states for DC V2G messaging

8.8.4.3 SECC

The SECC behavior defining all valid Request-Response Message Sequences for AC is shown in Figure 103 and for DC in Figure 104.

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 109, 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 109.

[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 109, if request message was received that the SECC does not expect in the wait state.

[V2G2-539] After the SECC sent a "ResponseCode = FAILED" it shall terminate the communication by applying **[V2G2-034]**.

[V2G2-729] After the SECC stopped the V2G Communication Session it shall terminate the communication by applying **[V2G2-034]**.

[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 109. The allowed next request shall be ServiceSetupReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[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 109. The allowed next request shall be ServiceDiscoveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[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 109, if the processing of the information is passed successfully. The allowed next request shall be ServiceDetailReq and PaymentServiceSelectionReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-546] The SECC shall respond with ServiceDiscoveryRes containing “ResponseCode=FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109, if the processing of the information is successfully passed. The allowed next request shall be ServiceDetailReq and

PaymentServiceSelectionReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-549] The SECC shall respond with ServiceDetailRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is not successful.

[V2G2-550] After receiving a PaymentServiceSelectionReq, the SECC shall process the received information.

[V2G2-551] The SECC shall respond with PaymentServiceSelectionRescontaining “ResponseCode = OK' within V2G_SECC_Msg_Performance_Time according to Table 109, 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 AuthorizationReq if Message Set “AC Charging EIM” is selected. V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-552] The SECC shall respond with PaymentServiceSelectionRescontaining “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109, 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 109.

[V2G2-555] The SECC shall respond with CertificateInstallationRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109, 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 109.

[V2G2-558] The SECC shall respond with CertificateUpdateRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109.

[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 109, if the processing of the information is successfully passed. The allowed next request shall be AuthorizationReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-561] The SECC shall respond with PaymentDetailsRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109.

[V2G2-562] After receiving a AuthorizationReq, the SECC shall process the received information.

[V2G2-563] The SECC shall respond with AuthorizationRes containing “ResponseCode = OK” and "EVSEProcessing=Finished" within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is successfully passed and the authorization is

finished. The allowed next request shall be ChargeParameterDiscoveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-564] The SECC shall respond with AuthorizationRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is not successful.

[V2G2-566] The SECC shall respond with ChargeParameterDiscoveryRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109, if the processing of the information is successfully passed and the request contained ChargeProgress set to ’Stop’. The allowed next request shall be SessionStopReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-569] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is not successful.

[V2G2-812] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is successfully passed and the request contained ChargeProgress set to ’Renegotiate’ and no PowerDeliveryReq with ChargeProgress set to ’Start’ as been received before.

[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 109 and shall terminate the communication by applying **[V2G2-034]**, 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 = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is not successful.

[V2G2-687] The SECC shall respond with AuthorizationRes containing “ResponseCode = OK” and "EVSEProcessing=Ongoing" within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is successfully passed and the authorization is ongoing. The allowed next request shall be AuthorizationReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[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 109, 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 109.

The EVCC may renegotiate the charging schedule as follows:

[V2G2-813] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is successfully passed and the request contained ChargeProgress set to

'Renegotiate'. The allowed next request shall be ChargeParameterDiscoveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

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 109, 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 109.

[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 109, 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 the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-576] The SECC shall respond with PowerDeliveryRes containing "ResponseCode = OK" within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is successfully passed and the request contained ChargeProgress set to 'Start'. The allowed next request shall be ChargingStatusReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-577] The SECC shall respond with ChargingStatusRes containing "ResponseCode = OK" within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109.

[V2G2-578] The SECC shall respond with ChargingStatusRes containing "ResponseCode = FAILED" within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109, if the processing of the information is successfully passed. The allowed next request shall be ChargingStatusReq, PowerDeliveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-581] The SECC shall respond with MeteringReceiptRes containing "ResponseCode = FAILED" within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is not successful.

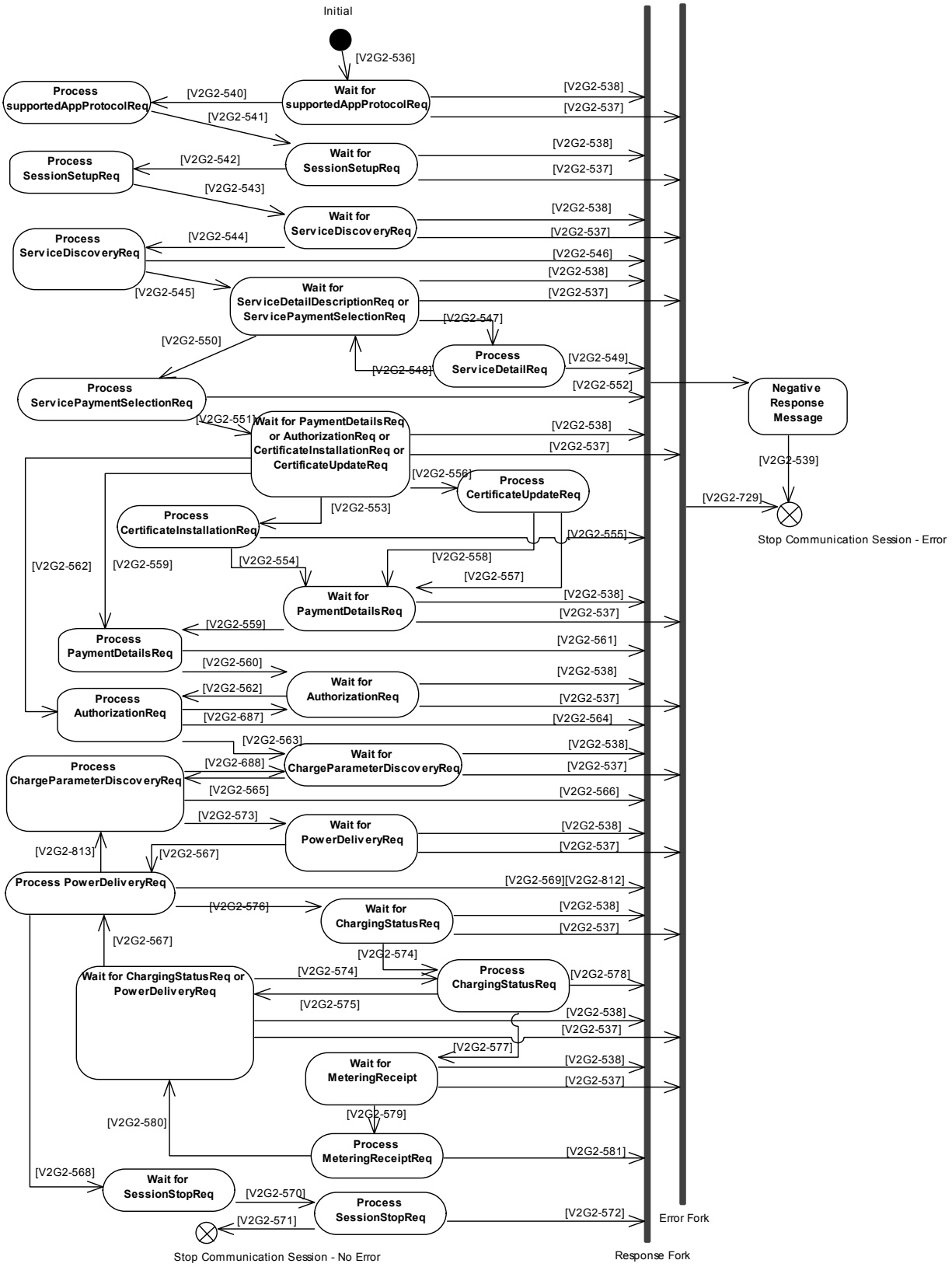


Figure 103 — SECC Communication states for AC V2G messaging

8.8.4.3.3 DC specific requirements

[V2G2-881] All EVErrorCodes in Table 104 for which there are no explicit requirements in this document are for information purpose only. They may be used for information to the customer, but they shall not influence the EVSE charging process.

NOTE 1 This means that the EVSE shall not change its behavior based on the value of EVErrorCode. If the EV detects a situation that requires a termination of the charging process, the EV will use other means to execute this shutdown, e.g. ramp down the current request, or, in case an emergency shutdown is required, change to CP State B in accordance with IEC 61851-1. In all other cases, the expected value for EVErrorCode is “NO_ERROR”.

[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 109, 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 109.

[V2G2-583] After receiving a CableCheckReq, the SECC shall process the received information.

[V2G2-584] The SECC shall respond with CableCheckRes containing “ResponseCode = OK” and "EVSEProcessing=Finished" within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is successfully passed and cable check is finished. The allowed next request shall be PreChargeReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-621] The SECC shall respond with CableCheckRes containing “ResponseCode = OK” and "EVSEProcessing=Ongoing" within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109.

[V2G2-585] The SECC shall respond with CableCheckRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109, 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 109.

[V2G2-588] The SECC shall respond with PreChargeRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109, if the processing of the information is successfully passed and the request’s parameter ChargeProgress is set to ‘Start’. The allowed next request shall be CurrentDemmandReq and PowerDeliveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-601] The SECC shall respond with PowerDeliveryRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is successfully passed and the request‘s parameter ChargeProgress is set to ‘Stop’.

The allowed next request shall be ChargeParameterDiscoveryReq and WeldingDetectionReq and SessionStopReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[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 109, 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 CurrentDemandReq and PowerDeliveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-595] The SECC shall respond with CurrentDemandRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is not successful.

[V2G2-795] The SECC shall respond with CurrentDemandRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109.

[V2G2-796] After receiving a MeteringReceiptReq, the SECC shall process the received information.

[V2G2-797] The SECC shall respond with MeteringReceiptRes containing “ResponseCode = OK” within V2G_SECC_Msg_Performance_Time according to Table 109, if the processing of the information is successfully passed. The allowed next request shall be CurrentDemandReq, PowerDeliveryReq and ChargeParameterDiscoveryReq and the V2G_SECC_Sequence_Timeout is set according to Table 109.

[V2G2-798] The SECC shall respond with MeteringReceiptRes containing “ResponseCode = FAIL” within V2G_SECC_Msg_Performance_Time according to Table 109, 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 109, 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 109.

[V2G2-598] The SECC shall respond with WeldingDetectionyRes containing “ResponseCode = FAILED” within V2G_SECC_Msg_Performance_Time according to Table 109 if the processing of the information is not successful.

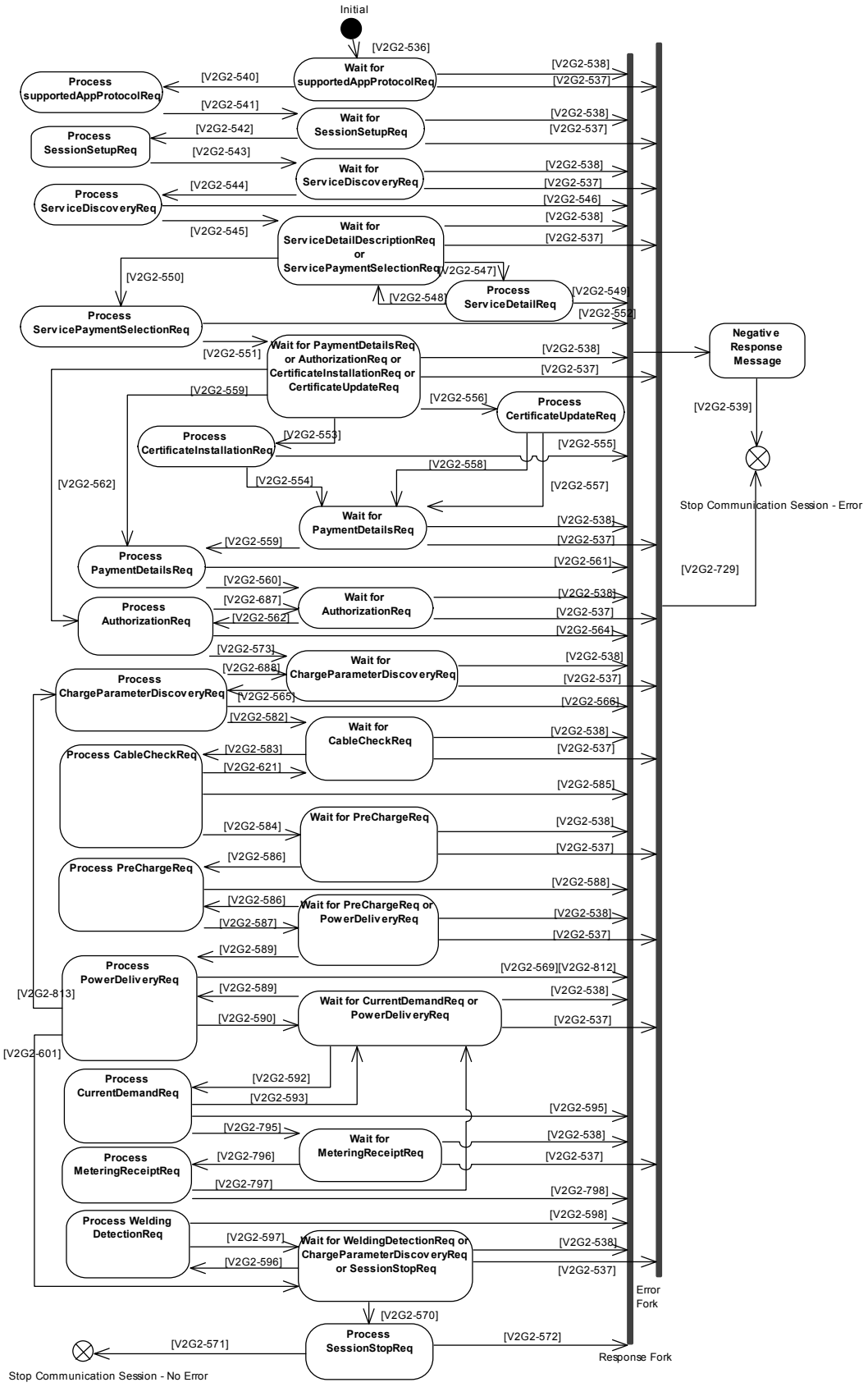


Figure 104 — SECC Communication states for DC V2G messaging

8.9 Request-Response Message Sequence examples

8.9.1 AC

8.9.1.1 EIM

Figure 105 depicts an example for a Request-Response Message Sequence for the EIM Identification Mode without any errors including optional messages.

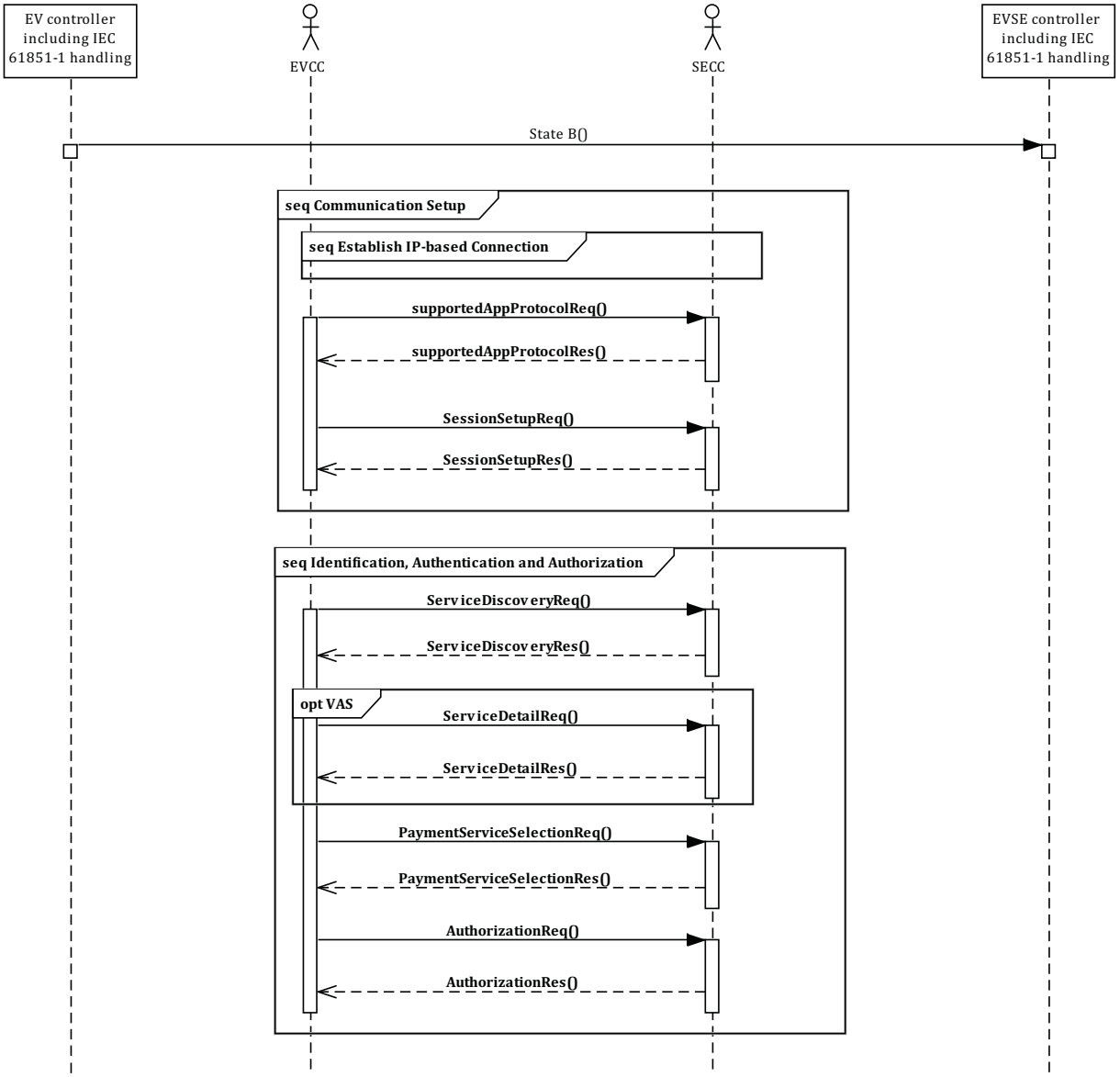


Figure 105 — Overview AC Request-Response Message Sequence EIM Identification Mode (1 of 2)

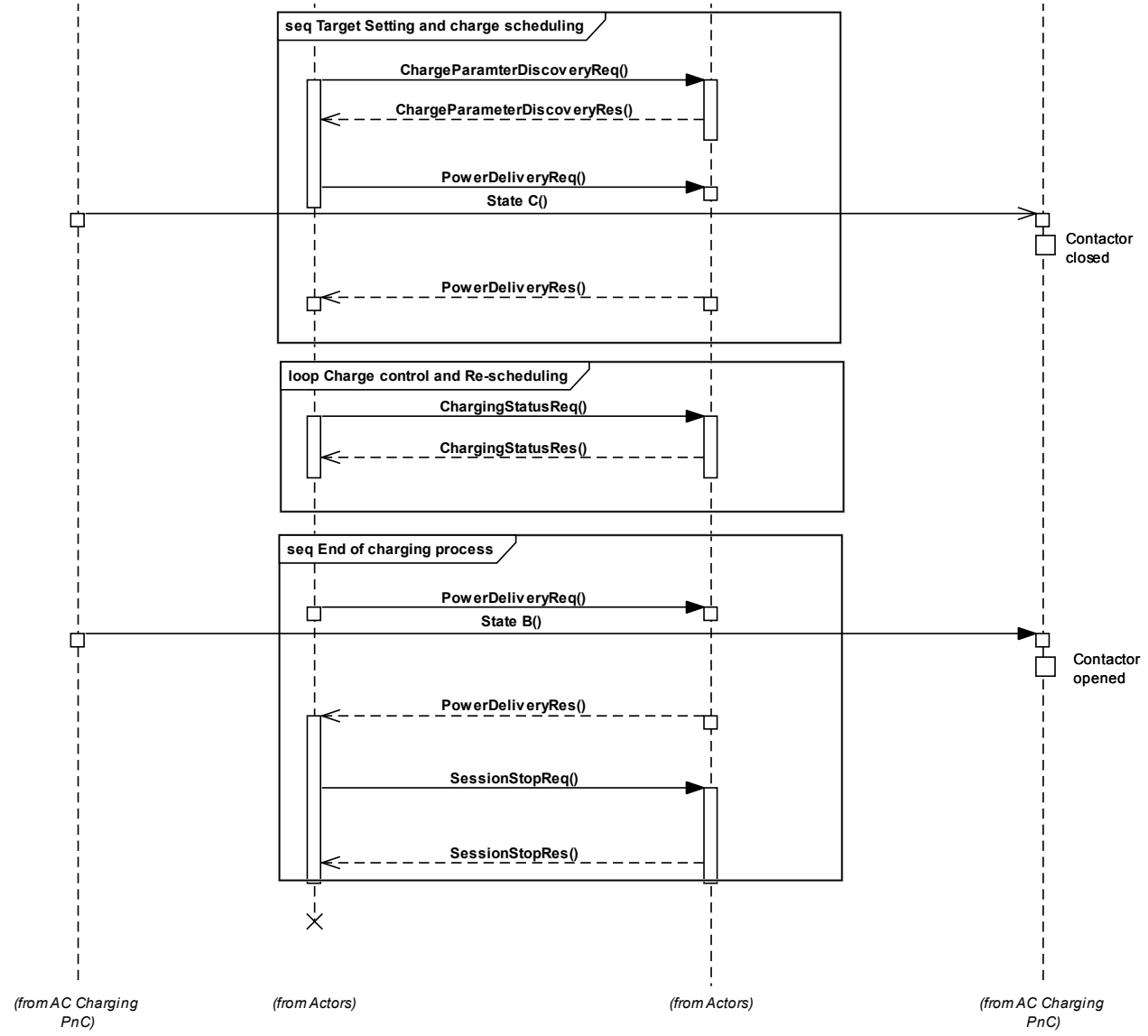


Figure 105 — Overview AC Request-Response Message Sequence EIM Identification Mode (2 of 2)

8.9.1.2 PnC

Figure 106 depicts an example for a Request-Response Message Sequence for the PnC Identification Mode without any errors including optional messages.

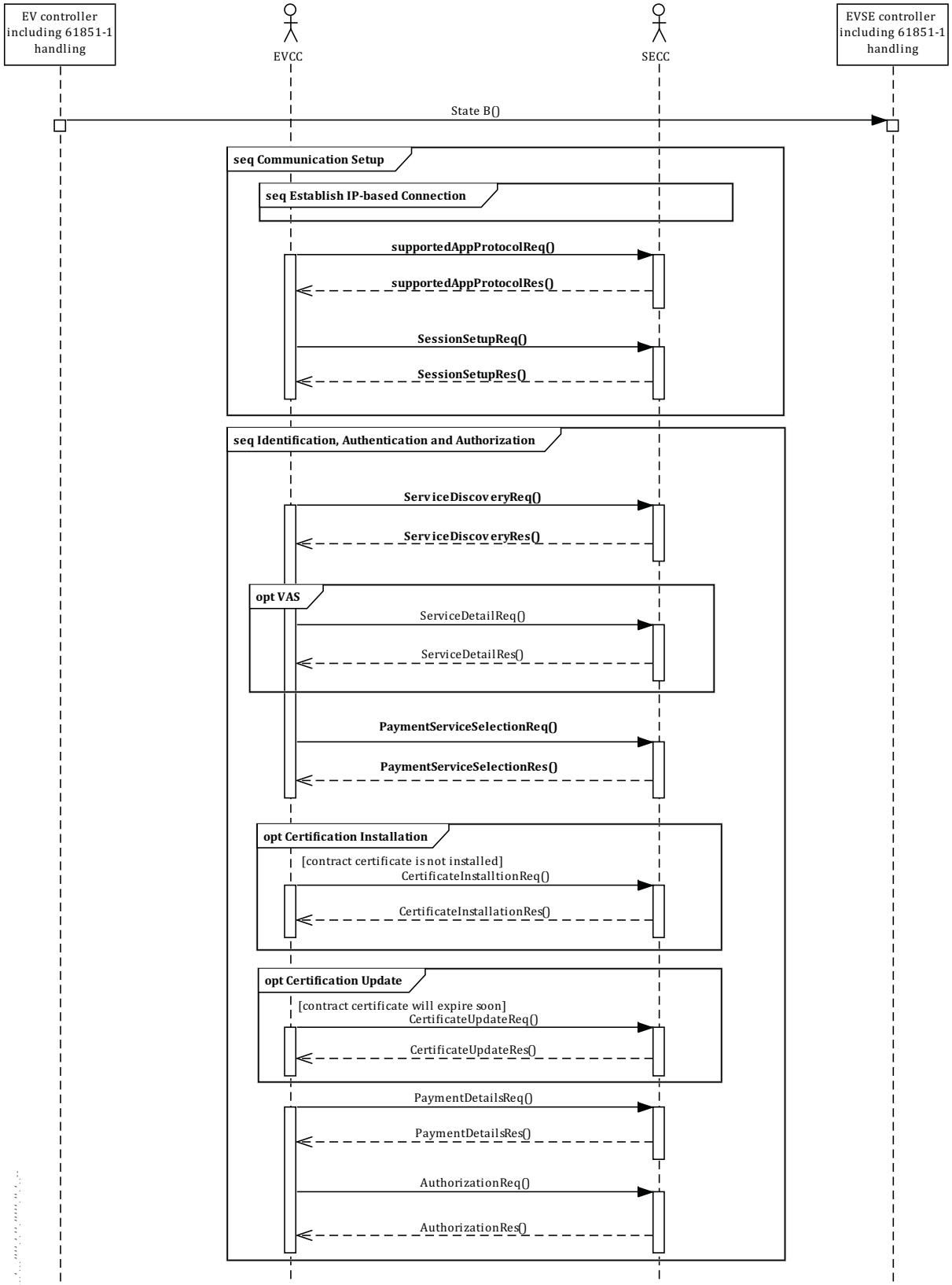


Figure 106 — Overview AC Request-Response Message Sequence PnC Identification Mode (1 of 2)

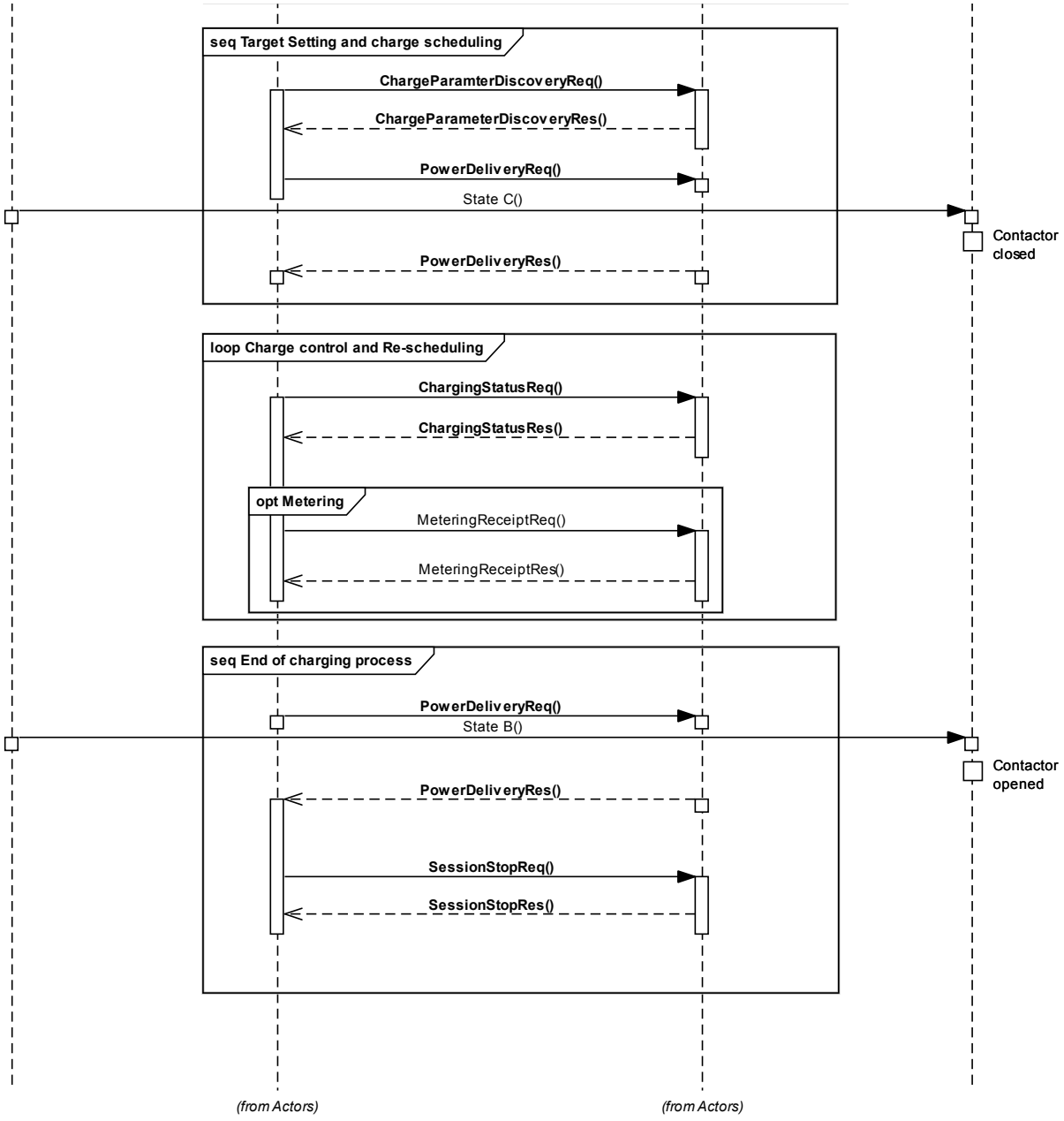


Figure 106 — Overview AC Request-Response Message Sequence PnC Identification Mode (2 of 2)

8.9.2 DC

8.9.2.1 EIM

Figure 107 depicts an example for a Request-Response Message Sequence in EIM Identification Mode without any errors including optional messages.

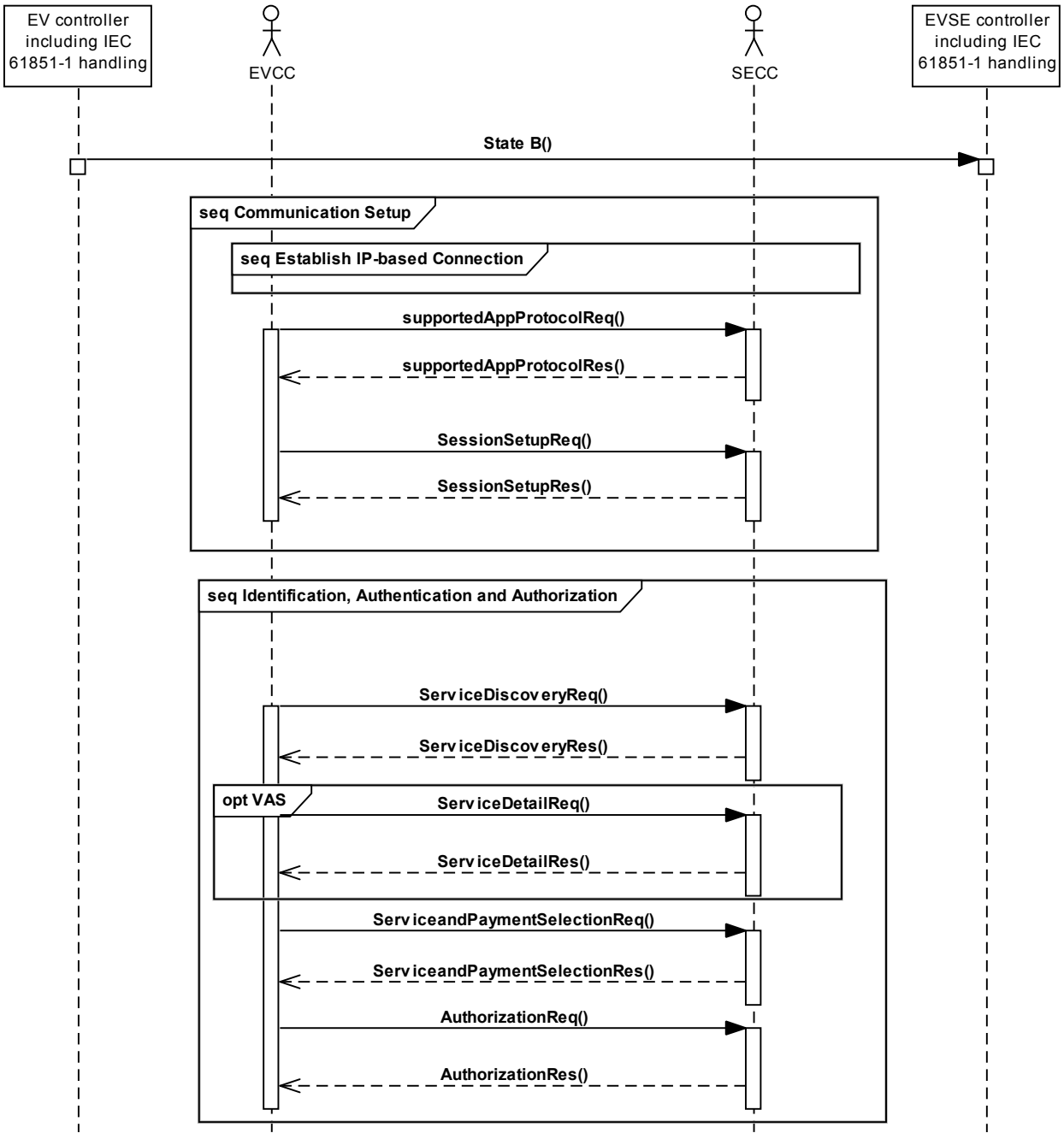


Figure 107 — Overview DC Request-Response Message Sequence EIM Identification Mode (1 of 2)

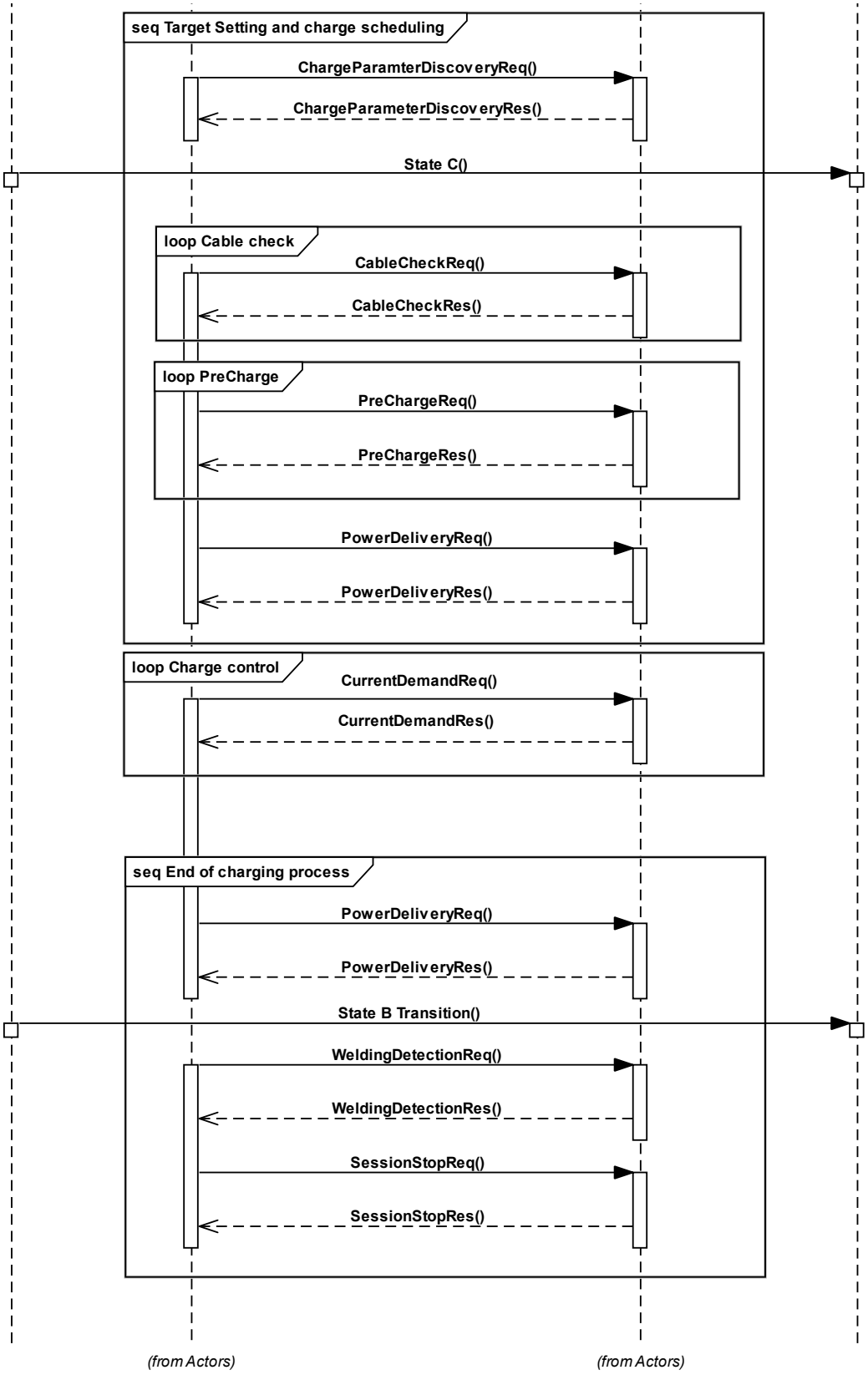


Figure 107 — Overview DC Request-Response Message Sequence EIM Identification Mode (2 of 2)

8.9.2.2 PnC

Figure 108 depicts an example for a Request-Response Message Sequence in PnC Identification Mode without any errors including optional messages.

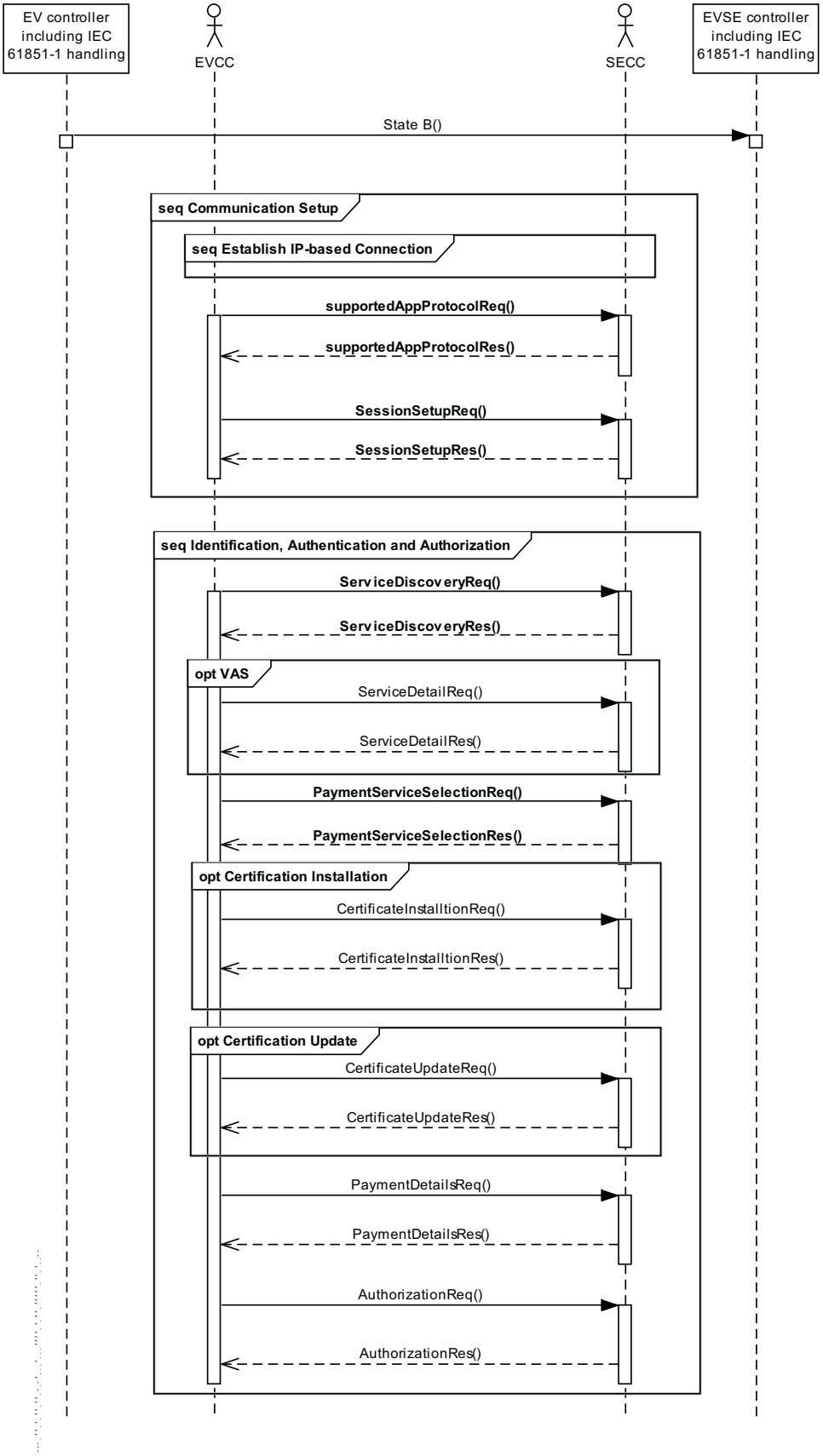


Figure 108 — Overview DC Request-Response Message Sequence PnC Identification Mode (1 of 2)

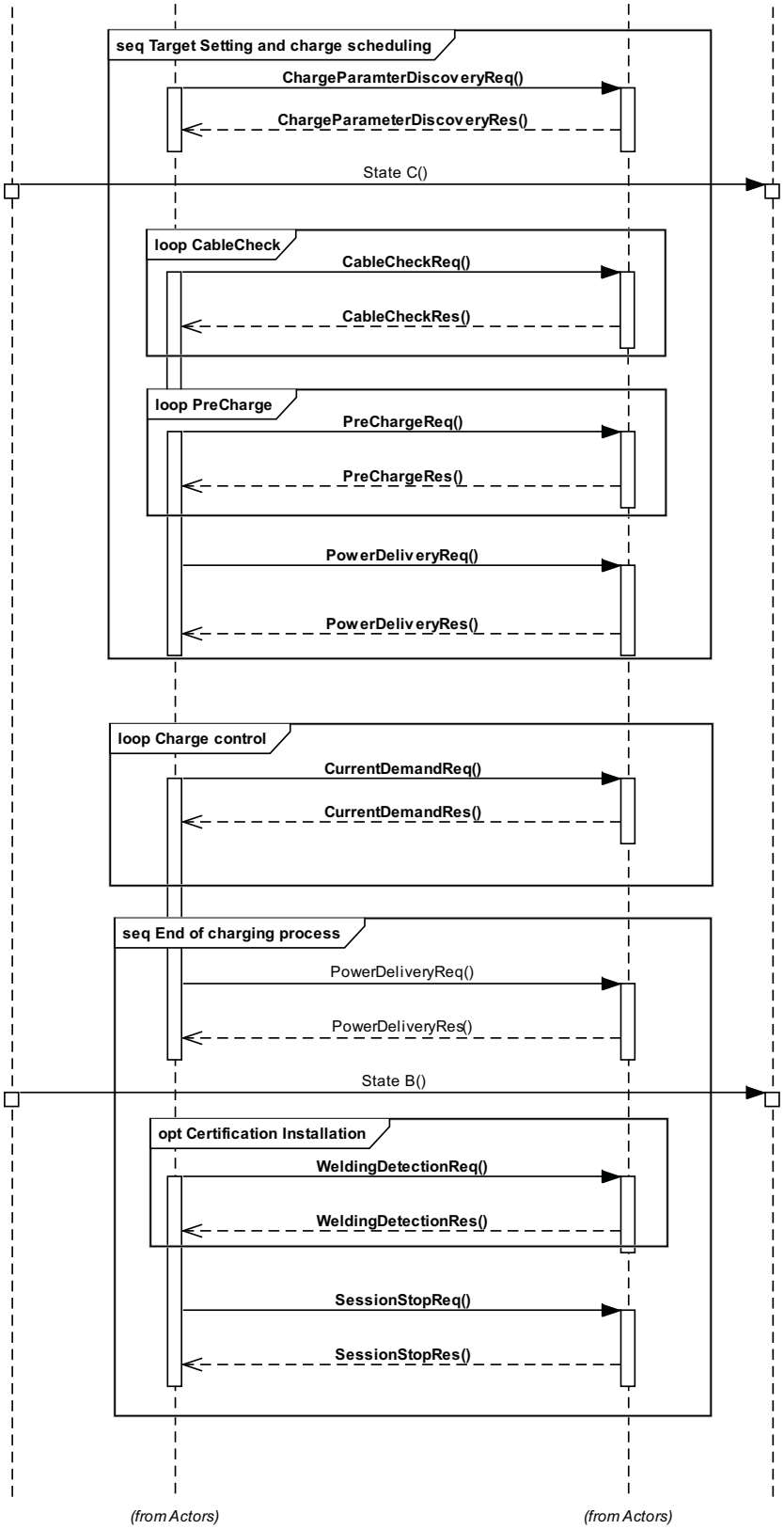


Figure 108 — Overview DC Request-Response Message Sequence PnC Identification Mode (2 of 2)

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

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	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	-	-	-
Header		B1	B1	B1	B1	-	-	-
	SessionId	B1	B1	B1	B1	-	-	-
	Notification	B1	B1	B1	B1	-	-	-
	FaultCode	B1	B1	B1	B1	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
		FaultMsg	B1	B1	B1	B1	-	-	-
	Signature (see sep. table)		B1	B1	B1	B1	-	-	-
Body			B1	B1	B1	B1	-	-	-
	EVCCID		B1	B1	B1	B1	-	-	-
SessionSetupRes			B1	B1	B1	B1	-	-	-
Header			B1	B1	B1	B1	-	-	-
	SessionId		B1	B1	B1	B1	-	-	-
	Notification		B1	B1	B1	B1	-	-	-
		FaultCode	B1	B1	B1	B1	-	-	-
		FaultMsg	B1	B1	B1	B1	-	-	-
	Signature (see sep. table)		-	-	B1	B1	-	-	-
Body			B1	B1	B1	B1	-	-	-
	ResponseCode		B1	B1	B1	B1	-	-	-
	EVSEID		B1	B1	B1	B1	-	-	-
	DateTimeNow		B1	B1	B1	B1	-	-	-
ServiceDiscoveryReq			D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
Header			D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	SessionId		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	Notification		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
		FaultCode	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
		FaultMsg	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	Signature (see sep.		D3,	D3,	D1,	D1,	-	-	-

				AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
		table)		D4	D4	D2	D2			
	Body			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	-	-	-
	Header			D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
		SessionId		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
		Notification		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
			FaultCode	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
			FaultMsg	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
		Signature (see sep. table)		-	-	D1, D2	D1, D2	-	-	-
	Body			D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
		ResponseCode		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
		PaymentOptions		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
			PaymentOption	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
			PaymentOption	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
		ChargeService		D3,	D3,	D1,	D1,	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
		D4	D4	D2	D2			
	ServiceID	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	ServiceName	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	ServiceCategory	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	ServiceScope	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	FreeService	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	SupportedEnergyTransferMode	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	EnergyTransferMode	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
ServiceList		-	-	-	-	C1	C2	G1, G2
	Service	-	-	-	-	C1	C2	G1, G2
	ServiceID	-	-	-	-	C1	C2	G1, G2
	ServiceName	-	-	-	-	C1	C2	G1, G2
	ServiceCategory	-	-	-	-	C1	C2	G1, G2
	ServiceScope	-	-	-	-	C1	C2	G1, G2
	FreeService	-	-	-	-	C1	C2	G1, G2
ServiceDetailReq		-	-	-	-	C1	C2	G1, G2

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
Header		-	-	-	-	C1	C2	G1, G2
SessionId		-	-	-	-	C1	C2	G1, G2
Notification		-	-	-	-	C1	C2	G1, G2
FaultCode		-	-	-	-	C1	C2	G1, G2
FaultMsg		-	-	-	-	C1	C2	G1, G2
Signature (see sep. table)		-	-	-	-	C1	C2	G1, G2
Body		-	-	-	-	C1	C2	G1, G2
ServiceID		-	-	-	-	C1	C2	G1, G2
ServiceDetailRes		-	-	-	-	C1	C2	G1, G2
Header		-	-	-	-	C1	C2	G1, G2
SessionId		-	-	-	-	C1	C2	G1, G2
Notification		-	-	-	-	C1	C2	G1, G2
FaultCode		-	-	-	-	C1	C2	G1, G2
FaultMsg		-	-	-	-	C1	C2	G1, G2
Signature (see sep. table)		-	-	-	-	C1	C2	G1, G2
Body		-	-	-	-	C1	C2	G1, G2

				AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
	ResponseCode			-	-	-	-	C1	C2	G1, G2
	ServiceID			-	-	-	-	C1	C2	G1, G2
	ServiceParameterList			-	-	-	-	C1	C2	G1, G2
		Parameter Set		-	-	-	-	C1	C2	G1, G2
			ParameterSet ID	-	-	-	-	C1	C2	G1, G2
			Parameter	-	-	-	-	C1	C2	G1, G2
ServicePaymentSelectionReq				D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
Header				D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
	SessionId			D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
	Notification			D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
		FaultCode		D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
		FaultMsg		D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
	Signature (see sep. table)			-	-	D1, D2	D1, D2	C1	C2	G1, G2
Body				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
		SelectedSe		D3,	D3,	D1,	D1,	C1	C2	G1,

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
rvice		D4	D4	D2	D2			G2
	ServiceID	D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
	ParameterSet ID	D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
ServicePaymentSelectionRes		D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
Header		D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
SessionId		D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
Notification		D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
	FaultCode	D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
	FaultMsg	D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
Signature (see sep. table)		D3, D4	D3, D4	D1, D2	D1, D2	C1	C2	G1, G2
Body		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	-	-	-
Header		-	-	D1, D2	D1, D2	-	-	-
SessionId		-	-	D1, D2	D1, D2	-	-	-
Notification		-	-	D1, D2	D1, D2	-	-	-
	FaultCode	-	-	D1,	D1,	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
				D2	D2			
	FaultMsg	-	-	D1, D2	D1, D2	-	-	-
	Signature (see sep. table)	-	-	D1, D2	D1, D2	-	-	-
Body		-	-	D1, D2	D1, D2	-	-	-
	eMAID	-	-	D1, D2	D1, D2	-	-	-
	ContractSignatureCertChain	-	-	D1, D2	D1, D2	-	-	-
	Certificate	-	-	D1, D2	D1, D2	-	-	-
	SubCertificates	-	-	D1, D2	D1, D2	-	-	-
	Certificate	-	-	D1, D2	D1, D2	-	-	-
PaymentDetailsRes		-	-	D1, D2	D1, D2	-	-	-
Header		-	-	D1, D2	D1, D2	-	-	-
	SessionId	-	-	D1, D2	D1, D2	-	-	-
	Notification	-	-	D1, D2	D1, D2	-	-	-
	FaultCode	-	-	D1, D2	D1, D2	-	-	-
	FaultMsg	-	-	D1, D2	D1, D2	-	-	-
	Signature (see sep. table)	-	-	D1, D2	D1, D2	-	-	-
Body		-	-	D1, D2	D1, D2	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
ResponseCode		-	-	D1, D2	D1, D2	-	-	-
GenChallenge		-	-	D1, D2	D1, D2	-	-	-
DateTimeNow		-	-	D1, D2	D1, D2	-	-	-
AuthorizationReq		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
Header		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
SessionId		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
Notification		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	FaultCode	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	FaultMsg	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
Signature (see sep. table)		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
Body		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
GenChallenge		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
AuthorizationRes		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
Header		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
SessionId		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
Notification		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
		FaultCode	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
		FaultMsg	D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
	Signature (see sep. table)		D3, D4	D3, D4	D1, D2	D1, D2	-	-	-
Body			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	-	-	-
ChargeParameterDiscoveryReq			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Header			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	SessionId		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	Notification		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
		FaultCode	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
		FaultMsg	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	Signature (see sep. table)		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
Body			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	MaxEntriesSAScheduleTuple		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	RequestedEnergyTransferMode		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	AC_EVChargeParameter		E1, E2	-	E3, E5, F3	-	-	-	-
		DepartureTime	E1, E2	-	E3, E5, F3	-	-	-	-
		Eamount	E1, E2	-	E3, E5, F3	-	-	-	-
		EVMaxVoltage	E1, E2	-	E3, E5, F3	-	-	-	-
		EVMaxCurrent	E1, E2	-	E3, E5, F3	-	-	-	-
		EVMinCurrent	E1, E2	-	E3, E5, F3	-	-	-	-
	DC_EVChargeParameter		-	E4	-	E4, F3	-	-	-
		DepartureTime	-	E4	-	E4, F3	-	-	-
		DC_EVStatus	-	E4	-	E4, F3	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
		EVReady	-	E4	-	E4, F3	-	-	-
		EVRESSConditioning	-	E4	-	E4, F3	-	-	-
		EVErrorCode	-	E4	-	E4, F3	-	-	-
		EVRESSSOC	-	E4	-	E4, F3	-	-	-
	EVMaximumCurrentLimit		-	E4	-	E4, F3	-	-	-
	EVMaximumPowerLimit		-	E4	-	E4, F3	-	-	-
	EVMaximumVoltageLimit		-	E4	-	E4, F3	-	-	-
	EVEnergyCapacity		-	E4	-	E4, F3	-	-	-
	EVEnergyRequest		-	E4	-	E4, F3	-	-	-
	FullSOC		-	E4	-	E4, F3	-	-	-
	BulkSOC		-	E4	-	E4, F3	-	-	-
ChargeParameterDiscoveryRes			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Header			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
SessionId			E1, E2	E4	E3, E5,	E4, F3	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
				F3				
Notification		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
FaultCode		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
FaultMsg		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Signature (see sep. table)		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Body		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	-	-	-
SASchedule		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
SAScheduleTuple		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	SAScheduleTupleID	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	PMaxSchedule	E1, E2	E4	E3, E5,	E4, F3	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
					F3				
		PMaxScheduleID	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
		PMAaxScheduleEntry	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
		RelativeTimeInterval	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
		start	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
		duration	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
		Pmax	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	SalesTariff		-	-	E3, E5, F3	E4, F3	-	-	-
		SalesTariffID	-	-	E3, E5, F3	E4, F3	-	-	-
		SalesTariffDescription	-	-	E3, E5, F3	E4, F3	-	-	-
		NumEPriceLevels	-	-	E3, E5, F3	E4, F3	-	-	-
		SalesTariffEntry	-	-	E3, E5,	E4, F3	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
				F3				
	RelativeTimeInterval	-	-	E3, E5, F3	E4, F3	-	-	-
	start	-	-	E3, E5, F3	E4, F3	-	-	-
	duration	-	-	E3, E5, F3	E4, F3	-	-	-
	EPrice Level	-	-	E3, E5, F3	E4, F3	-	-	-
	ConsumptionCost	-	-	E3, E5, F3	E4, F3	-	-	-
	Start Value	-	-	E3, E5, F3	E4, F3	-	-	-
	Cost	-	-	E3, E5, F3	E4, F3	-	-	-
AC_EVSEChargeParameter		E1, E2	-	E3, E5, F3	-	-	-	-
	AC_EVSEStatus	E1, E2	-	E3, E5, F3	-	-	-	-
	RCD	E1, E2	-	E3, E5, F3	-	-	-	-
	NotificationMaxDelay	E1, E2	-	E3, E5,	-	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
					F3				
		EVSENotification	E1, E2	-	E3, E5, F3	-	-	-	-
	EVSENominalVoltage		E1, E2	-	E3, E5, F3	-	-	-	-
	EVSEMaxCurrent		E1, E2	-	E3, E5, F3	-	-	-	-
DC_EVSEChargeParameter			-	E4	-	E4, F3	-	-	-
	DC_EVSEStatus		-	E4	-	E4, F3	-	-	-
		EVSEIsolationStatus	-	E4	-	E4, F3	-	-	-
		EVSEStatusCode	-	E4	-	E4, F3	-	-	-
		NotificationMaxDelay	-	E4	-	E4, F3	-	-	-
		EVSENotification	-	E4	-	E4, F3	-	-	-
	EVSEMaximumCurrentLimit		-	E4	-	E4, F3	-	-	-
	EVSEMaximumPowerLimit		-	E4	-	E4, F3	-	-	-
	EVSEMaximumVoltageLimit		-	E4	-	E4, F3	-	-	-
	EVSEMinimumCurre		-	E4	-	E4, F3	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
	ntLimit							
	EVSEMinimumVoltageLimit	-	E4	-	E4, F3	-	-	-
	EVSECurrentRegulationTolerance	-	E4	-	E4, F3	-	-	-
	EVSEPeakCurrentRipple	-	E4	-	E4, F3	-	-	-
	EVSEEnergyToBeDelivered	-	E4	-	E4, F3	-	-	-
PowerDeliveryReq		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Header		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
SessionId		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Notification		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	FaultCode	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	FaultMsg	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Signature (see sep. table)		E1, E2	E4	E3, E5,	E4, F3	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
					F3				
Body			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	ChargeProgress		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	SAScheduleTupleID		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	ChargingProfile		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
		ProfileEntry	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
			ChargingProfileEntryStart	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
			ChargingProfileEntryMaxPower	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
			ChargingProfileEntryMaxNumberOfPhasesInUse	E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	DC_EVPowerDeliveryParameter			-	E4	-	E4, F3	-	-	-
		DC_EVStatus		-	E4	-	E4, F3	-	-	-
			EVReady	-	E4	-	E4, F3	-	-	-
			EVRESSConditioning	-	E4	-	E4, F3	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
		EVErrorCode	-	E4	-	E4, F3	-	-	-
		EVRESSSOC	-	E4	-	E4, F3	-	-	-
	BulkChargingComplete		-	E4	-	E4, F3	-	-	-
	ChargingComplete		-	E4	-	E4, F3	-	-	-
PowerDeliveryRes			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Header			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
SessionId			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Notification			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	FaultCode		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	FaultMsg		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Signature (see sep. table)			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
Body			E1, E2	E4	E3, E5, F3	E4, F3	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
	ResponseCode		E1, E2	E4	E3, E5, F3	E4, F3	-	-	-
	AC_EVSEStatus		E1, E2	-	E3, E5, F3	-	-	-	-
		RCD	E1, E2	-	E3, E5, F3	-	-	-	-
		Notification MaxDelay	E1, E2	-	E3, E5, F3	-	-	-	-
		EVSENotification	E1, E2	-	E3, E5, F3	-	-	-	-
	DC_EVSEStatus		-	E4	-	E4, F3	-	-	-
		EVSEIsolationStatus	-	E4	-	E4, F3	-	-	-
		EVSEStatusCode	-	E4	-	E4, F3	-	-	-
		Notification MaxDelay	-	E4	-	E4, F3	-	-	-
		EVSENotification	-	E4	-	E4, F3	-	-	-
CertificateupdateReq			-	-	-	-	C1	-	-
Header			-	-	-	-	C1	-	-
	SessionId		-	-	-	-	C1	-	-
	Notification		-	-	-	-	C1	-	-
		FaultCode	-	-	-	-	C1	-	-
		FaultMsg	-	-	-	-	C1	-	-
	Signature (see sep.		-	-	-	-	C1	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
		table)							
	Body		-	-	-	-	C1	-	-
		ContractSignatureCertChain	-	-	-	-	C1	-	-
		Certificate	-	-	-	-	C1	-	-
		SubCertificates	-	-	-	-	C1	-	-
		Certificate	-	-	-	-	C1	-	-
		eMAID	-	-	-	-	C1	-	-
		ListOfRootCertificateIDs	-	-	-	-	C1	-	-
		RootCertificateID	-	-	-	-	C1	-	-
CertificateupdateRes			-	-	-	-	C1	-	-
	Header		-	-	-	-	C1	-	-
		SessionId	-	-	-	-	C1	-	-
		Notification	-	-	-	-	C1	-	-
		FaultCode	-	-	-	-	C1	-	-
		FaultMsg	-	-	-	-	C1	-	-
		Signature (see sep. table)	-	-	-	-	C1	-	-
	Body		-	-	-	-	C1	-	-
		ResponseCode	-	-	-	-	C1	-	-
		SAProvisioningCertificateChain					C1	-	-
		ContractSignatureCertChain	-	-	-	-	C1	-	-
		Certificate	-	-	-	-	C1	-	-
		SubCertificates	-	-	-	-	C1	-	-

				AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
			Certificate	-	-	-	-	C1	-	-
		ContractSignatureEncryptedPrivateKey		-	-	-	-	C1	-	-
		Dhpublickey		-	-	-	-	C1	-	-
		eMAID		-	-	-	-	C1	-	-
		RetryCounter		-	-	-	-	C1	-	-
CertificateinstallationReq				-	-	-	-	-	C2	-
	Header			-	-	-	-	-	C2	-
		SessionId		-	-	-	-	-	C2	-
		Notification		-	-	-	-	-	C2	-
			FaultCode	-	-	-	-	-	C2	-
			FaultMsg	-	-	-	-	-	C2	-
		Signature (see sep. table)		-	-	-	-	-	C2	-
	Body			-	-	-	-	-	C2	-
		OEMProvisioningCert		-	-	-	-	-	C2	-
		ListOfRootCertificateIDs		-	-	-	-	-	C2	-
			RootCertificateID	-	-	-	-	-	C2	-
CertificateinstallationRes				-	-	-	-	-	C2	-
	Header			-	-	-	-	-	C2	-
		SessionId		-	-	-	-	-	C2	-
		Notification		-	-	-	-	-	C2	-
			FaultCode	-	-	-	-	-	C2	-
			FaultMsg	-	-	-	-	-	C2	-
		Signature (see sep. table)		-	-	-	-	-	C2	-
	Body			-	-	-	-	-	C2	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
	ResponseCode		-	-	-	-	-	C2	-
	SAProvisioningCertificateChain						-	C2	-
	ContractSignatureCertChain		-	-	-	-	-	C2	-
		Certificate	-	-	-	-	-	C2	-
		SubCertificates	-	-	-	-	-	C2	-
			Certificate	-	-	-	-	-	C2	-
	ContractSignatureEncryptedPrivateKey		-	-	-	-	-	C2	-
	Dhpublickey		-	-	-	-	-	C2	-
	eMAID		-	-	-	-	-	-	-
SessionStopReq			H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
Header			H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
	SessionId		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
	Notification		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
		FaultCode	H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
		FaultMsg	H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
	Signature (see sep. table)		-	-	-	-	-	-	-
Body			H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
	ChargingSession		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
SessionStopRes		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
Header		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
SessionId		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
Notification		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
FaultCode		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
FaultMsg		H1, F3	H1, F3	H1, F3	H1, F3	-	-	-
Signature (see sep. table)		-	-	H1, F3	H1, F3	-	-	-
Body		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	-	-	-	-
Header		E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
SessionId		E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
Notification		E1, E2, F0	-	E3, E5, F0,	-	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
				F1				
	FaultCode	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	FaultMsg	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	Signature (see sep. table)	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
Body		E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
ChargingStatusRes		E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
Header		E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	SessionId	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	Notification	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
	FaultCode	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	FaultMsg	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	Signature (see sep. table)	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
Body		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,	-	E3,	-	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
		E2, F0		E5, F0, F1				
	MeterID	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	MeterReading	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	SigMeterReading	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	MeterStatus	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	TMeter	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
ReceiptRequired		E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
AC_EVSEStatus		E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
	RCD	E1, E2,	-	E3, E5,	-	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
			F0		F0, F1				
		Notification MaxDelay	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
		EVSENotification	E1, E2, F0	-	E3, E5, F0, F1	-	-	-	-
MeteringReceiptReq			-	-	F1	F1	-	-	-
	Header		-	-	F1	F1	-	-	-
		SessionId	-	-	F1	F1	-	-	-
		Notification	-	-	F1	F1	-	-	-
		FaultCode	-	-	F1	F1	-	-	-
		FaultMsg	-	-	F1	F1	-	-	-
		Signature (see sep. table)	-	-	F1	F1	-	-	-
	Body		-	-	F1	F1	-	-	-
		SessionID	-	-	F1	F1	-	-	-
		SAScheduleTupleID	-	-	F1	F1	-	-	-
		MeterInfo	-	-	F1	F1	-	-	-
		MeterID	-	-	F1	F1	-	-	-
		MeterReading	-	-	F1	F1	-	-	-
		SigMeterReading	-	-	F1	F1	-	-	-
		MeterStatus	-	-	F1	F1	-	-	-
		Tmeter	-	-	F1	F1	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
MeteringReceiptRes		-	-	F1	F1	-	-	-
Header		-	-	F1	F1	-	-	-
SessionId		-	-	F1	F1	-	-	-
Notification		-	-	F1	F1	-	-	-
	FaultCode	-	-	F1	F1	-	-	-
	FaultMsg	-	-	F1	F1	-	-	-
Signature (see sep. table)		-	-	-	-	-	-	-
Body		-	-	F1	F1	-	-	-
AC_EVSEStatus		-	-	F1	F1	-	-	-
	RCD	-	-	F1	F1	-	-	-
	NotificationMaxDelay	-	-	F1	F1	-	-	-
	EVSENotification	-	-	F1	F1	-	-	-
DC_EVSEStatus		-	-	F1	F1	-	-	-
	EVSEIsolationStatus	-	-	F1	F1	-	-	-
	EVSEStatusCode	-	-	F1	F1	-	-	-
	NotificationMaxDelay	-	-	F1	F1	-	-	-
	EVSENotification	-	-	F1	F1	-	-	-
ResponseCode		-	-	F1	F1	-	-	-
CableCheckReq		-	E4	-	E4	-	-	-
Header		-	E4	-	E4	-	-	-
SessionId		-	E4	-	E4	-	-	-
Notification		-	E4	-	E4	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
		FaultCode	-	E4	-	E4	-	-	-
		FaultMsg	-	E4	-	E4	-	-	-
	Signature (see sep. table)		-	E4	-	E4	-	-	-
Body			-	E4	-	E4	-	-	-
	DC_EVStatus		-	E4	-	E4	-	-	-
		EVReady	-	E4	-	E4	-	-	-
		EVRESSConditioning	-	E4	-	E4	-	-	-
		EVErrorCode	-	E4	-	E4	-	-	-
		EVRESSSOC	-	E4	-	E4	-	-	-
CableCheckRes			-	E4	-	E4	-	-	-
Header			-	E4	-	E4	-	-	-
	SessionId		-	E4	-	E4	-	-	-
	Notification		-	E4	-	E4	-	-	-
		FaultCode	-	E4	-	E4	-	-	-
		FaultMsg	-	E4	-	E4	-	-	-
	Signature (see sep. table)		-	E4	-	E4	-	-	-
Body			-	E4	-	E4	-	-	-
	ResponseCode		-	E4	-	E4	-	-	-
	EVSEProcessing		-	E4	-	E4	-	-	-
	DC_EVSEStatus		-	E4	-	E4	-	-	-
		EVSEIsolationStatus	-	E4	-	E4	-	-	-
		EVSEStatusCode	-	E4	-	E4	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
		Notification MaxDelay	-	E4	-	E4	-	-	-
		EVSENotification	-	E4	-	E4	-	-	-
PreChargeReq			-	E4	-	E4	-	-	-
Header			-	E4	-	E4	-	-	-
	SessionId		-	E4	-	E4	-	-	-
	Notification		-	E4	-	E4	-	-	-
		FaultCode	-	E4	-	E4	-	-	-
		FaultMsg	-	E4	-	E4	-	-	-
	Signature (see sep. table)		-	E4	-	E4	-	-	-
Body			-	E4	-	E4	-	-	-
	DC_EVStatus		-	E4	-	E4	-	-	-
		EVReady	-	E4	-	E4	-	-	-
		EVRESSConiditioning	-	E4	-	E4	-	-	-
		EVErrorCode	-	E4	-	E4	-	-	-
		EVRESSSOC	-	E4	-	E4	-	-	-
	EVTargetVoltage		-	E4	-	E4	-	-	-
	EVTargetCurrent		-	E4	-	E4	-	-	-
PreChargeRes			-	E4	-	E4	-	-	-
Header			-	E4	-	E4	-	-	-
	SessionId		-	E4	-	E4	-	-	-
	Notification		-	E4	-	E4	-	-	-
		FaultCode	-	E4	-	E4	-	-	-
		FaultMsg	-	E4	-	E4	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
	Signature (see sep. table)		-	E4	-	E4	-	-	-
Body			-	E4	-	E4	-	-	-
	ResponseCode		-	E4	-	E4	-	-	-
	DC_EVSEStatus		-	E4	-	E4	-	-	-
		EVSEIsolationStatus	-	E4	-	E4	-	-	-
		EVSEStatusCode	-	E4	-	E4	-	-	-
		NotificationMaxDelay	-	E4	-	E4	-	-	-
		EVSENotification	-	E4	-	E4	-	-	-
	EVSEPresentVoltage		-	E4	-	E4	-	-	-
CurrentDemandReq			-	E4	-	E4	-	-	-
Header			-	E4	-	E4	-	-	-
	SessionId		-	E4	-	E4	-	-	-
	Notification		-	E4	-	E4	-	-	-
		FaultCode	-	E4	-	E4	-	-	-
		FaultMsg	-	E4	-	E4	-	-	-
	Signature (see sep. table)		-	E4	-	E4	-	-	-
Body			-	E4	-	E4	-	-	-
	DC_EVStatus		-	E4	-	E4	-	-	-
		EVReady	-	E4	-	E4	-	-	-
		EVRESSConiditioning	-	E4	-	E4	-	-	-
		EVErrorCode	-	E4	-	E4	-	-	-
		EVRESSS	-	E4	-	E4	-	-	-

				AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
			OC							
		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	-	-	-
	Header			-	E4	-	E4	-	-	-
		SessionId		-	E4	-	E4	-	-	-
		Notification		-	E4	-	E4	-	-	-
			FaultCode	-	E4	-	E4	-	-	-
			FaultMsg	-	E4	-	E4	-	-	-
		Signature (see sep. table)		-	-	-	-	-	-	-
	Body			-	E4	-	E4	-	-	-
		ResponseCode		-	E4	-	E4	-	-	-
		DC_EVSEStatus		-	E4	-	E4	-	-	-
			EVSEIsolationStatus	-	E4	-	E4	-	-	-
			EVSEStatusCode	-	E4	-	E4	-	-	-
			NotificationMaxDelay	-	E4	-	E4	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
	EVSENotification	-	E4	-	E4	-	-	-
EVSEPresentVoltage		-	E4	-	E4	-	-	-
EVSEPresentCurrent		-	E4	-	E4	-	-	-
EVSECurrentLimitAchieved		-	E4	-	E4	-	-	-
EVSEVoltageLimitAchieved		-	E4	-	E4	-	-	-
EVSEPowerLimitAchieved		-	E4	-	E4	-	-	-
EVSEMaximumVoltageLimit		-	E4	-	E4	-	-	-
EVSEMaximumCurrentLimit		-	E4	-	E4	-	-	-
EVSEMaximumPowerLimit		-	E4	-	E4	-	-	-
EVSEID		-	-	-	E4	-	-	-
SAScheduleTupleID		-	-	-	E4	-	-	-
MeterInfo		-	-	-	E4	-	-	-
	MeterID	-	-	-	E4	-	-	-
	MeterReading	-	-	-	E4	-	-	-
	SigMeterReading	-	-	-	E4	-	-	-
	MeterStatus	-	-	-	E4	-	-	-
	Tmeter	-	-	-	E4	-	-	-
ReceiptRequired		-	-	-	E4	-	-	-
WeldingDetectionReq		-	E4	-	E4	-	-	-
Header		-	E4	-	E4	-	-	-

			AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
	SessionId		-	E4	-	E4	-	-	-
	Notification		-	E4	-	E4	-	-	-
		FaultCode	-	E4	-	E4	-	-	-
		FaultMsg	-	E4	-	E4	-	-	-
	Signature (see sep. table)		-	-	-	-	-	-	-
Body			-	E4	-	E4	-	-	-
	DC_EVStatus		-	E4	-	E4	-	-	-
		EVReady	-	E4	-	E4	-	-	-
		EVRESSConditioning	-	E4	-	E4	-	-	-
		EVErrorCode	-	E4	-	E4	-	-	-
		EVRESSSOC	-	E4	-	E4	-	-	-
WeldingDetectionRes			-	E4	-	E4	-	-	-
Header			-	E4	-	E4	-	-	-
	SessionId		-	E4	-	E4	-	-	-
	Notification		-	E4	-	E4	-	-	-
		FaultCode	-	E4	-	E4	-	-	-
		FaultMsg	-	E4	-	E4	-	-	-
	Signature (see sep. table)		-	-	-	-	-	-	-
Body			-	E4	-	E4	-	-	-
	ResponseCode		-	E4	-	E4	-	-	-
	DC_EVSEStatus		-	E4	-	E4	-	-	-
		EVSEIsolationStatus	-	E4	-	E4	-	-	-
		EVSEStatu	-	E4	-	E4	-	-	-

		AC Charging EIM	DC Charging EIM	AC Charging PnC	DC Charging PnC	Option: Certificate Update	Option: Certificate Installation	Option: VAS
	sCode							
	NotificationMaxDelay	-	E4	-	E4	-	-	-
	EVSENotification	-	E4	-	E4	-	-	-
EVSEPresentVoltage		-	E4	-	E4	-	-	-

Annex B
(informative)

Mapping of ISO 15118 message element names to SAE J2847/2 terms

B.1 SAE J2847/2 status codes

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 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 Error Code

Enumeration	Definition in SAE J2847/2 (Vehicle Error Code)	Definition in ISO 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)
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 ResponseCode calues
0xF	No Data	DC_EVStatus: DC_EVErrorCode (Value = NoData)

NOTE 1 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 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 2 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 Requested Energy Transfer Type)	Definition in ISO 15118 (RequestedEnergyTransferMode)
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
0x08 – 0x0E	Reserved for future use	Reserved_8.. Reserved_E
0x0F	Undetermined	Undetermined

B.3 SAE J2847/2 signals

Table B. 4 defines the name mapping for the SAE J2847/2 signals to ISO 15118 message elements.

Table B. 4 — Mapping of naming for SAE J2847/2 signals to ISO 15118 message elements

SAE J2847/2 signal definition	ISO 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 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 Error 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

— "xmldsig-core-schema": Defines the W3C schema for XML signatures

Figure C.1 shows the dependency graph for all five XML Schema documents.

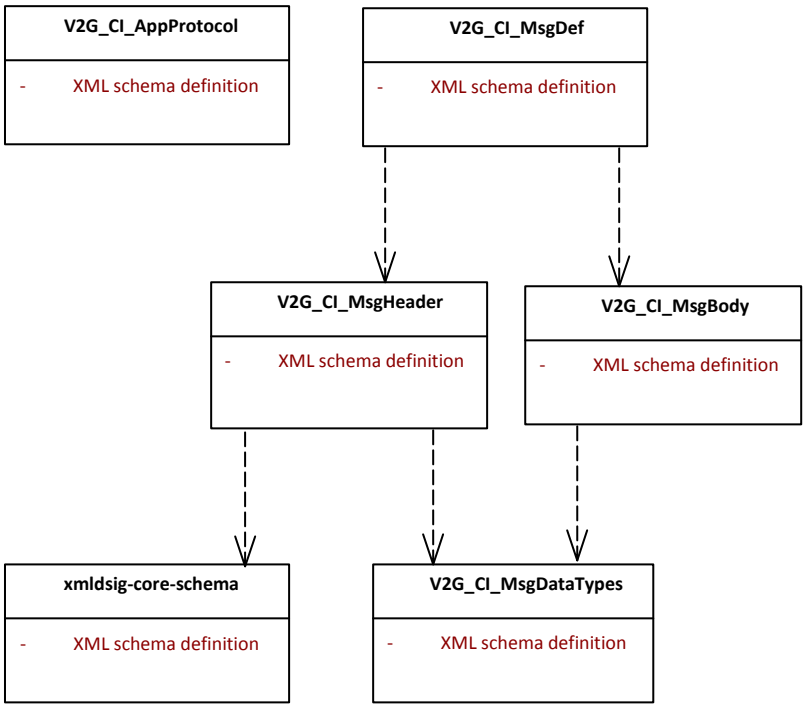


Figure C.1 — Dependency Chart of the V2G CI XML Schema Definitions

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"
xmlns="urn:iso:15118:2:2013:MsgDef" xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_b="urn:iso:15118:2:2013:MsgBody" targetNamespace="urn:iso:15118:2:2013:MsgDef"
elementFormDefault="qualified" attributeFormDefault="qualified" version="15118 2.0">
	<!-- attributeFormDefault="unqualified" -->
	<xs:import namespace="urn:iso:15118:2:2013:MsgHeader" schemaLocation="V2G_CI_MsgHeader.xsd"/>
	<xs:import namespace="urn:iso:15118:2:2013:MsgBody" schemaLocation="V2G_CI_MsgBody.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="v2gci_b:BodyType"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

C.4 V2G_CI_MsgHeader.xsd

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
targetNamespace="urn:iso:15118:2:2013:MsgHeader"
elementFormDefault="qualified" attributeFormDefault="qualified">
<xs:import namespace="urn:iso:15118:2:2013: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"
xmlns="urn:iso:15118:2:2013:MsgBody"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
targetNamespace="urn:iso:15118:2:2013:MsgBody"
elementFormDefault="qualified" attributeFormDefault="qualified">
    <xs:import namespace="urn:iso:15118:2:2013:MsgDataTypes" schemaLocation="V2G_CI_MsgDataTypes.xsd"/>
    <!-- Body -->
    <xs:complexType name="BodyType">
        <xs:sequence>
            <xs:element ref="BodyElement" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="BodyElement" type="BodyBaseType" abstract="true"/>
    <xs:complexType name="BodyBaseType" abstract="true"/>
    <!-- ..... -->
    <!-- Common Messages (AC/DC) -->
    <!-- ..... -->
    <!--          -->
    <!-- Session Setup -->
    <!--          -->
    <xs:element name="SessionSetupReq" type="SessionSetupReqType" substitutionGroup="BodyElement"/>
    <xs:complexType name="SessionSetupReqType">
        <xs:complexContent>
            <xs:extension base="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="BodyElement"/>
    <xs:complexType name="SessionSetupResType">
        <xs:complexContent>
            <xs:extension base="BodyBaseType">
                <xs:sequence>
                    <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                    <xs:element name="EVSEID" type="v2gci_t:evseIDType"/>
                    <xs:element name="EVSETimeStamp" type="xs:long" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
```

```
            </xs:complexContent>
        </xs:complexType>
        <!--                -->
        <!-- Service Discovery -->
        <!--                -->
        <xs:element name="ServiceDiscoveryReq" type="ServiceDiscoveryReqType" substitutionGroup="BodyElement"/>
        <xs:complexType name="ServiceDiscoveryReqType">
            <xs:complexContent>
                <xs:extension base="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="BodyElement"/>
        <xs:complexType name="ServiceDiscoveryResType">
            <xs:complexContent>
                <xs:extension base="BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="PaymentOptionList" type="v2gci_t:PaymentOptionListType"/>
                        <xs:element name="ChargeService" type="v2gci_t:ChargeServiceType"/>
                        <!--<xs:element name="ServiceList" type="v2gci_t:ServiceTagListType" minOccurs="0"/> -->
                        <xs:element name="ServiceList" type="v2gci_t:ServiceListType" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                -->
        <!-- Service Detail    -->
        <!--                -->
        <xs:element name="ServiceDetailReq" type="ServiceDetailReqType" substitutionGroup="BodyElement"/>
        <xs:complexType name="ServiceDetailReqType">
            <xs:complexContent>
                <xs:extension base="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="BodyElement"/>
        <xs:complexType name="ServiceDetailResType">
            <xs:complexContent>
                <xs:extension base="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="PaymentServiceSelectionReq" type="PaymentServiceSelectionReqType"
substitutionGroup="BodyElement"/>
        <xs:complexType name="PaymentServiceSelectionReqType">
            <xs:complexContent>
                <xs:extension base="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="PaymentServiceSelectionRes" type="PaymentServiceSelectionResType"
substitutionGroup="BodyElement"/>
        <xs:complexType name="PaymentServiceSelectionResType">
            <xs:complexContent>
                <xs:extension base="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="BodyElement"/>
        <xs:complexType name="PaymentDetailsReqType">
            <xs:complexContent>
                <xs:extension base="BodyBaseType">
                    <xs:sequence>
                        <xs:element name="eMAID" type="v2gci_t:eMAIDType"/>
                        <xs:element name="ContractSignatureCertChain" type="v2gci_t:CertificateChainType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="PaymentDetailsRes" type="PaymentDetailsResType" substitutionGroup="BodyElement"/>
        <xs:complexType name="PaymentDetailsResType">
            <xs:complexContent>
                <xs:extension base="BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="GenChallenge" type="v2gci_t:genChallengeType"/>
                        <xs:element name="EVSETimeStamp" type="xs:long"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                    -->
        <!-- Authorization-->
        <!--                    -->
        <xs:element name="AuthorizationReq" type="AuthorizationReqType" substitutionGroup="BodyElement"/>
        <xs:complexType name="AuthorizationReqType">
            <xs:complexContent>
                <xs:extension base="BodyBaseType">
                    <xs:sequence>
                        <xs:element name="GenChallenge" type="v2gci_t:genChallengeType" minOccurs="0"/>
                    </xs:sequence>
                    <xs:attribute name="Id" type="xs:ID"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="AuthorizationRes" type="AuthorizationResType" substitutionGroup="BodyElement"/>
        <xs:complexType name="AuthorizationResType">
            <xs:complexContent>
                <xs:extension base="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="BodyElement"/>
        <xs:complexType name="ChargeParameterDiscoveryReqType">
            <xs:complexContent>
                <xs:extension base="BodyBaseType">
                    <xs:sequence>
                        <xs:element name="MaxEntriesSAScheduleTuple" type="xs:unsignedShort" minOccurs="0"/>
                        <!-- new -->
                        <xs:element name="RequestedEnergyTransferMode" type="v2gci_t:EnergyTransferModeType"/>
                        <xs:element ref="v2gci_t:EVChargeParameter"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
```

```
	<xs:element name="ChargeParameterDiscoveryRes" type="ChargeParameterDiscoveryResType"
substitutionGroup="BodyElement"/>
	<xs:complexType name="ChargeParameterDiscoveryResType">
		<xs:complexContent>
			<xs:extension base="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" minOccurs="0"/>
					<xs:element ref="v2gci_t:EVSEChargeParameter"/>
				</xs:sequence>
			</xs:extension>
		</xs:complexContent>
	</xs:complexType>
	<!--                    -->
	<!-- Power Delivery     -->
	<!--                    -->
	<xs:element name="PowerDeliveryReq" type="PowerDeliveryReqType" substitutionGroup="BodyElement"/>
	<xs:complexType name="PowerDeliveryReqType">
		<xs:complexContent>
			<xs:extension base="BodyBaseType">
				<xs:sequence>
					<xs:element name="ChargeProgress" type="v2gci_t:chargeProgressType"/>
					<xs:element name="SAScheduleTupleID" type="v2gci_t:SAIDType"/>
					<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="BodyElement"/>
	<xs:complexType name="PowerDeliveryResType">
		<xs:complexContent>
			<xs:extension base="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>
	<!--                    -->
	<!-- Metering Receipt   -->
	<!--                    -->
	<xs:element name="MeteringReceiptReq" type="MeteringReceiptReqType" substitutionGroup="BodyElement"/>
	<xs:complexType name="MeteringReceiptReqType">
		<xs:complexContent>
			<xs:extension base="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:ID"/>
			</xs:extension>
		</xs:complexContent>
	</xs:complexType>
	<xs:element name="MeteringReceiptRes" type="MeteringReceiptResType" substitutionGroup="BodyElement"/>
	<xs:complexType name="MeteringReceiptResType">
		<xs:complexContent>
			<xs:extension base="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>
	<!--                    -->
	<!-- SessionStop        -->
	<!--                    -->
	<xs:element name="SessionStopReq" type="SessionStopReqType" substitutionGroup="BodyElement"/>
	<xs:complexType name="SessionStopReqType">
		<xs:complexContent>
			<xs:extension base="BodyBaseType">
```

```
                    <xs:sequence>
                        <xs:element name="ChargingSession" type="v2gci_t:chargingSessionType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="SessionStopRes" type="SessionStopResType" substitutionGroup="BodyElement"/>
        <xs:complexType name="SessionStopResType">
            <xs:complexContent>
                <xs:extension base="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="BodyElement"/>
        <xs:complexType name="CertificateUpdateReqType">
            <xs:complexContent>
                <xs:extension base="BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ContractSignatureCertChain" type="v2gci_t:CertificateChainType"/>
                        <xs:element name="eMAID" type="v2gci_t:eMAIDType"/>
                        <xs:element name="ListOfRootCertificateIDs" type="v2gci_t:ListOfRootCertificateIDsType"/>
                    </xs:sequence>
                    <xs:attribute name="Id" type="xs:ID" use="required"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="CertificateUpdateRes" type="CertificateUpdateResType" substitutionGroup="BodyElement"/>
        <xs:complexType name="CertificateUpdateResType">
            <xs:complexContent>
                <xs:extension base="BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="SAProvisioningCertificateChain" type="v2gci_t:CertificateChainType"/>
                        <xs:element name="ContractSignatureCertChain" type="v2gci_t:CertificateChainType"/>
                        <xs:element name="ContractSignatureEncryptedPrivateKey"
type="v2gci_t:ContractSignatureEncryptedPrivateKeyType"/>
                        <xs:element name="DHpublickey" type="v2gci_t:DiffieHellmanPublickeyType"/>
                        <xs:element name="eMAID" type="v2gci_t:EMAIDType"/>
                        <xs:element name="RetryCounter" type="xs:short" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--                    -->
        <!-- Certificate Installation     -->
        <!--                    -->
        <xs:element name="CertificateInstallationReq" type="CertificateInstallationReqType" substitutionGroup="BodyElement"/>
        <xs:complexType name="CertificateInstallationReqType">
            <xs:complexContent>
                <xs:extension base="BodyBaseType">
                    <xs:sequence>
                        <xs:element name="OEMProvisioningCert" type="v2gci_t:certificateType"/>
                        <xs:element name="ListOfRootCertificateIDs" type="v2gci_t:ListOfRootCertificateIDsType"/>
                    </xs:sequence>
                    <xs:attribute name="Id" type="xs:ID" use="required"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="CertificateInstallationRes" type="CertificateInstallationResType" substitutionGroup="BodyElement"/>
        <xs:complexType name="CertificateInstallationResType">
            <xs:complexContent>
                <xs:extension base="BodyBaseType">
                    <xs:sequence>
                        <xs:element name="ResponseCode" type="v2gci_t:responseCodeType"/>
                        <xs:element name="SAProvisioningCertificateChain" type="v2gci_t:CertificateChainType"/>
                        <!-- new -->
                        <xs:element name="ContractSignatureCertChain" type="v2gci_t:CertificateChainType"/>
                        <xs:element name="ContractSignatureEncryptedPrivateKey"
```

```
type="v2gci_t:ContractSignatureEncryptedPrivateKeyType"/>
                        <xs:element name="DHpublickey" type="v2gci_t:DiffieHellmanPublickeyType"/>
                        <xs:element name="eMAID" type="v2gci_t:EMAIDType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!-- ..... -->
        <!-- AC-Messages          -->
        <!-- ..... -->
        <!--                      -->
        <!-- Charging Status      -->
        <!--                      -->
        <xs:element name="ChargingStatusReq" type="ChargingStatusReqType" substitutionGroup="BodyElement"/>
        <xs:complexType name="ChargingStatusReqType">
            <xs:complexContent>
                <xs:extension base="BodyBaseType">
                    <xs:sequence/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="ChargingStatusRes" type="ChargingStatusResType" substitutionGroup="BodyElement"/>
        <xs:complexType name="ChargingStatusResType">
            <xs:complexContent>
                <xs:extension base="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" minOccurs="0"/>
                        <xs:element name="AC_EVSEStatus" type="v2gci_t:AC_EVSEStatusType"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!-- ..... -->
        <!-- DC-Messages          -->
        <!-- ..... -->
        <!--            -->
        <!-- Cable Check  -->
        <!--            -->
        <xs:element name="CableCheckReq" type="CableCheckReqType" substitutionGroup="BodyElement"/>
        <xs:complexType name="CableCheckReqType">
            <xs:complexContent>
                <xs:extension base="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="BodyElement"/>
        <xs:complexType name="CableCheckResType">
            <xs:complexContent>
                <xs:extension base="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="BodyElement"/>
        <xs:complexType name="PreChargeReqType">
            <xs:complexContent>
                <xs:extension base="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="BodyElement"/>
    <xs:complexType name="PreChargeResType">
        <xs:complexContent>
            <xs:extension base="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="BodyElement"/>
    <xs:complexType name="CurrentDemandReqType">
        <xs:complexContent>
            <xs:extension base="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="BodyElement"/>
    <xs:complexType name="CurrentDemandResType">
        <xs:complexContent>
            <xs:extension base="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:element name="EVSEID" type="v2gci_t:evseIDType"/>
                    <xs:element name="SAScheduleTupleID" type="v2gci_t:SAIDType"/>
                    <xs:element name="MeterInfo" type="v2gci_t:MeterInfoType" minOccurs="0"/>
                    <xs:element name="ReceiptRequired" type="xs:boolean" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <!--                -->
    <!-- Welding Detection -->
    <!--                -->
    <xs:element name="WeldingDetectionReq" type="WeldingDetectionReqType" substitutionGroup="BodyElement"/>
    <xs:complexType name="WeldingDetectionReqType">
        <xs:complexContent>
            <xs:extension base="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="BodyElement"/>
        <xs:complexType name="WeldingDetectionResType">
            <xs:complexContent>
                <xs:extension base="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"
xmlns="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
targetNamespace="urn:iso:15118:2:2013:MsgDataTypes"
elementFormDefault="qualified" attributeFormDefault="qualified">
    <xs:import namespace="http://www.w3.org/2000/09/xmldsig#" schemaLocation="xmldsig-core-schema.xsd"/>
    <!-- ================== -->
    <!-- Complex types      -->
    <!-- ================== -->
    <!--                    -->
    <!-- service-related types -->
    <!--                    -->
    <xs:complexType name="ServiceType">
        <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:element name="FreeService" type="xs:boolean"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ServiceListType">
        <xs:sequence>
            <xs:element name="Service" type="ServiceType" maxOccurs="8"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="SelectedServiceListType">
        <xs:sequence>
            <xs:element name="SelectedService" type="SelectedServiceType" maxOccurs="16"/>
        </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="255"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ParameterSetType">
        <xs:sequence>
            <xs:element name="ParameterSetID" type="xs:short"/>
            <xs:element name="Parameter" type="ParameterType" maxOccurs="16"/>
        </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:complexType>
<xs:complexType name="ChargeServiceType">
        <xs:complexContent>
                <xs:extension base="ServiceType">
                        <xs:sequence>
                                <xs:element name="SupportedEnergyTransferMode" type="SupportedEnergyTransferModeType"/>
                        </xs:sequence>
                </xs:extension>
        </xs:complexContent>
</xs:complexType>
<xs:complexType name="SupportedEnergyTransferModeType">
        <xs:sequence>
                <xs:element name="EnergyTransferMode" type="EnergyTransferModeType" maxOccurs="6"/>
        </xs:sequence>
</xs:complexType>
<!--                                    -->
<!-- security related types -->
<!--                                    -->
<xs:complexType name="ContractSignatureEncryptedPrivateKeyType">
        <xs:simpleContent>
                <xs:extension base="privateKeyType">
                        <xs:attribute name="Id" type="xs:ID" use="required"/>
                </xs:extension>
        </xs:simpleContent>
</xs:complexType>
<xs:complexType name="DiffieHellmanPublickeyType">
        <xs:simpleContent>
                <xs:extension base="dHpublickeyType">
                        <xs:attribute name="Id" type="xs:ID" use="required"/>
                </xs:extension>
        </xs:simpleContent>
</xs:complexType>
<xs:complexType name="EMAIDType">
        <xs:simpleContent>
                <xs:extension base="eMAIDType">
                        <xs:attribute name="Id" type="xs:ID" use="required"/>
                </xs:extension>
        </xs:simpleContent>
</xs:complexType>
<xs:complexType name="CertificateChainType">
        <xs:sequence>
                <xs:element name="Certificate" type="certificateType"/>
                <xs:element name="SubCertificates" type="SubCertificatesType" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="Id" type="xs:ID"/>
</xs:complexType>
<xs:complexType name="SubCertificatesType">
        <xs:sequence>
                <xs:element name="Certificate" type="certificateType" maxOccurs="4"/>
        </xs:sequence>
</xs:complexType>
<xs:complexType name="ListOfRootCertificateIDsType">
        <xs:sequence>
                <xs:element name="RootCertificateID" type="xmlsig:X509IssuerSerialType" maxOccurs="20"/>
        </xs:sequence>
</xs:complexType>
<!--                                    -->
<!-- metering related types -->
<!--                                    -->
<xs:complexType name="MeterInfoType">
        <xs:sequence>
                <xs:element name="MeterID" type="meterIDType"/>
                <xs:element name="MeterReading" type="xs:unsignedLong" 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>
<!--                                    -->
<!-- Physical value type   -->
<!--                                    -->
<xs:complexType name="PhysicalValueType">
```

```
        <xs:sequence>
            <xs:element name="Multiplier" type="unitMultiplierType"/>
            <xs:element name="Unit" type="unitSymbolType"/>
            <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" abstract="true"/>
<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="3"/>
                </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" minOccurs="0"/>
            <xs:element ref="SalesTariffEntry" maxOccurs="1024"/>
        </xs:sequence>
        <xs:attribute name="Id" type="xs:ID"/>
</xs:complexType>
<xs:complexType name="PMaxScheduleType">
        <xs:sequence>
            <xs:element ref="PMaxScheduleEntry" maxOccurs="1024"/>
        </xs:sequence>
</xs:complexType>
<xs:element name="Entry" type="EntryType" abstract="true"/>
<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" minOccurs="0"/>
                    <xs:element name="ConsumptionCost" type="ConsumptionCostType" minOccurs="0" maxOccurs="3"/>
                </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="PhysicalValueType"/>
                </xs:sequence>
```

```xml
			</xs:extension>
		</xs:complexContent>
	</xs:complexType>
	<xs:complexType name="IntervalType" abstract="true"/>
	<xs:element name="TimeInterval" type="IntervalType" abstract="true"/>
	<xs:element name="RelativeTimeInterval" type="RelativeTimeIntervalType" substitutionGroup="TimeInterval"/>
	<xs:complexType name="RelativeTimeIntervalType">
		<xs:complexContent>
			<xs:extension base="IntervalType">
				<xs:sequence>
					<xs:element name="start">
						<xs:simpleType>
							<xs:restriction base="xs:unsignedInt">
								<xs:minInclusive value="0"/>
								<xs:maxInclusive value="16777214"/>
							</xs:restriction>
						</xs:simpleType>
					</xs:element>
					<xs:element name="duration" minOccurs="0">
						<xs:simpleType>
							<xs:restriction base="xs:unsignedInt">
								<xs:minInclusive value="0"/>
								<xs:maxInclusive value="86400"/>
							</xs:restriction>
						</xs:simpleType>
					</xs:element>
				</xs:sequence>
			</xs:extension>
		</xs:complexContent>
	</xs:complexType>
	<xs:complexType name="ConsumptionCostType">
		<xs:sequence>
			<xs:element name="startValue" type="PhysicalValueType"/>
			<xs:element name="Cost" type="CostType" maxOccurs="3"/>
		</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:sequence>
			<xs:element name="NotificationMaxDelay" type="xs:unsignedShort"/>
			<xs:element name="EVSENotification" type="EVSENotificationType"/>
		</xs:sequence>
	</xs:complexType>
	<xs:element name="EVSEStatus" type="EVSEStatusType" abstract="true"/>
	<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="RCD" type="xs:boolean"/>
				</xs:sequence>
			</xs:extension>
		</xs:complexContent>
	</xs:complexType>
	<xs:complexType name="EVStatusType" abstract="true"/>
	<xs:element name="EVStatus" type="EVStatusType" abstract="true"/>
	<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: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="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:sequence>
                <xs:element name="DepartureTime" type="xs:unsignedInt" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
        <xs:element name="EVChargeParameter" type="EVChargeParameterType" abstract="true"/>
        <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="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" abstract="true"/>
        <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="EVSENominalVoltage" type="PhysicalValueType"/>
                        <xs:element name="EVSEMaxCurrent" 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"/>
                        <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" abstract="true"/>
        <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="ProfileEntry" type="ProfileEntryType" maxOccurs="24"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="ProfileEntryType">
            <xs:sequence>
                <xs:element name="ChargingProfileEntryStart" type="xs:unsignedInt"/>
                <!-- <xs:element name="ChargingProfileEntryMaxPower" type="PMaxType"/> -->
                <xs:element name="ChargingProfileEntryMaxPower" type="PhysicalValueType"/>
                <!-- 2013-06-20 VR: added per DE 101 and AT 5 -->
                <xs:element name="ChargingProfileEntryMaxNumberOfPhasesInUse" type="maxNumPhasesType" minOccurs="0"/>
                <!-- end add-->
            </xs:sequence>
        </xs:complexType>
        <!-- ====================  -->
        <!-- Simple types          -->
        <!-- ====================  -->
        <!--                       -->
        <!-- General Types -->
        <!--                       -->
        <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:enumeration value="Ongoing_WaitingForCustomerInteraction"/>
            </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>
<xs:simpleType name="chargeProgressType">
	<xs:restriction base="xs:string">
		<xs:enumeration value="Start"/>
		<xs:enumeration value="Stop"/>
		<xs:enumeration value="Renegotiate"/>
	</xs:restriction>
</xs:simpleType>
<xs:simpleType name="chargingSessionType">
	<xs:restriction base="xs:string">
		<xs:enumeration value="Terminate"/>
		<xs:enumeration value="Pause"/>
	</xs:restriction>
</xs:simpleType>
<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="64"/>
	</xs:restriction>
</xs:simpleType>
<xs:simpleType name="maxNumPhasesType">
	<xs:restriction base="xs:byte">
		<xs:minInclusive value="1"/>
		<xs:maxInclusive value="3"/>
	</xs:restriction>
</xs:simpleType>
<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:simpleType name="meterStatusType">
	<xs:restriction base="xs:short"/>
</xs:simpleType>
<!--                              -->
<!-- EnergyTransferType    -->
<!--                              -->
<xs:simpleType name="EnergyTransferModeType">
	<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:base64Binary">
```

```
            <xs:length value="16"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="certificateType">
        <xs:restriction base="xs:base64Binary">
            <xs:maxLength value="800"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="dHpublickeyType">
        <xs:restriction base="xs:base64Binary">
            <xs:maxLength value="65"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="privateKeyType">
        <xs:restriction base="xs:base64Binary">
            <xs:maxLength value="48"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="sigMeterReadingType">
        <xs:restriction base="xs:base64Binary">
            <xs:maxLength value="64"/>
        </xs:restriction>
    </xs:simpleType>
    <!--                                    -->
    <!-- Identification Numbers    -->
    <!--                                    -->
    <xs:simpleType name="sessionIDType">
        <xs:restriction base="xs:hexBinary">
            <xs:maxLength value="8"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="evccIDType">
        <xs:restriction base="xs:hexBinary">
            <xs:maxLength value="6"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="evseIDType">
        <xs:restriction base="xs:string">
            <xs:minLength value="7"/>
            <xs:maxLength value="37"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="serviceIDType">
        <xs:restriction base="xs:unsignedShort"/>
    </xs:simpleType>
    <xs:simpleType name="eMAIDType">
        <xs:restriction base="xs:string">
            <xs:minLength value="14"/>
            <xs:maxLength value="15"/>
        </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:unsignedByte">
            <xs:minInclusive value="1"/>
            <xs:maxInclusive value="255"/>
        </xs:restriction>
    </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="PaymentOptionListType">
        <xs:sequence>
            <xs:element name="PaymentOption" type="paymentOptionType" minOccurs="1" maxOccurs="2"/>
        </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_MeteringSignatureNotValid"/>
            <xs:enumeration value="FAILED_NoChargeServiceSelected"/>
            <xs:enumeration value="FAILED_WrongEnergyTransferMode"/>
            <xs:enumeration value="FAILED_ContactorError"/>
            <xs:enumeration value="FAILED_CertificateNotAllowedAtThisEVSE"/>
            <xs:enumeration value="FAILED_CertificateRevoked"/>
        </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="V">
                    <xs:annotation>
                        <xs:documentation>Voltage in Volt</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="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="Valid"/>
                <xs:enumeration value="Warning"/>
                <xs:enumeration value="Fault"/>
                <xs:enumeration value="No_IMD"/>
            </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#" version="0.1" elementFormDefault="qualified">

<!-- Basic Types Defined for Signatures -->

<simpleType name="CryptoBinary">
  <restriction base="base64Binary">
  </restriction>
</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" minOccurs="0" type="ds:HMACOutputLengthType"/>
      <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 processContents="lax" namespace="##other"/>
    <!-- (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"/> <!-- add a grep facet -->
  <attribute name="Encoding" type="anyURI" use="optional"/>
</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 XML instances shown is this subclause provide an example how the selection of VAS can be implemented. The following assumption 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:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_b="urn:iso:15118:2:2013:MsgBody"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
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:ServiceDiscoveryRes>
            <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
            <v2gci_b:PaymentOptionList>
                <v2gci_t:PaymentOption>Contract</v2gci_t:PaymentOption>
                <v2gci_t:PaymentOption>ExternalPayment</v2gci_t:PaymentOption>
            </v2gci_b:PaymentOptionList>
            <v2gci_b:ChargeService>
                <v2gci_t:ServiceID>1</v2gci_t:ServiceID>
                <v2gci_t:ServiceName>AC_DC_Charging</v2gci_t:ServiceName>
                <v2gci_t:ServiceCategory>EVCharging</v2gci_t:ServiceCategory>
                <v2gci_t:FreeService>true</v2gci_t:FreeService>
                <v2gci_t:SupportedEnergyTransferMode>
                    <v2gci_t:EnergyTransferMode>AC_three_phase_core</v2gci_t:EnergyTransferMode>
                    <v2gci_t:EnergyTransferMode>DC_extended</v2gci_t:EnergyTransferMode>
                </v2gci_t:SupportedEnergyTransferMode>
            </v2gci_b:ChargeService>
            <v2gci_b:ServiceList>
                <v2gci_t:Service>
                    <v2gci_t:ServiceID>3</v2gci_t:ServiceID>
                    <v2gci_t:ServiceName>Fast Internet</v2gci_t:ServiceName>
                    <v2gci_t:ServiceCategory>Internet</v2gci_t:ServiceCategory>
                    <v2gci_t:FreeService>false</v2gci_t:FreeService>
                </v2gci_t:Service>
                <v2gci_t:Service>
                    <v2gci_t:ServiceID>2</v2gci_t:ServiceID>
                    <v2gci_t:ServiceName>Certificate</v2gci_t:ServiceName>
                    <v2gci_t:ServiceCategory>ContractCertificate</v2gci_t:ServiceCategory>
                    <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 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:15118:2:2013:MsgHeader"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013:MsgBody"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
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:ServiceDetailReq>
                  <v2gci_b:ServiceID>2</v2gci_b:ServiceID>
            </v2gci_b:ServiceDetailReq>
      </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 7 – ServiceDetailsReq message

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013:MsgBody"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
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:ServiceDetailRes>
                  <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
                  <v2gci_b:ServiceID>2</v2gci_b:ServiceID>
                  <v2gci_b:ServiceParameterList>
                        <v2gci_t:ParameterSet>
                              <v2gci_t:ParameterSetID>3</v2gci_t:ParameterSetID>
                              <v2gci_t:Parameter v2gci_t:Name="Protocol">
                                    <v2gci_t:stringValue>HTTP</v2gci_t:stringValue>
                              </v2gci_t:Parameter>
                              <v2gci_t:Parameter v2gci_t:Name="Port">
                                    <v2gci_t:intValue>80</v2gci_t:intValue>
                              </v2gci_t:Parameter>
                        </v2gci_t:ParameterSet>
                        <v2gci_t:ParameterSet>
                              <v2gci_t:ParameterSetID>4</v2gci_t:ParameterSetID>
                              <v2gci_t:Parameter v2gci_t:Name="Protocol">
                                    <v2gci_t:stringValue>HTTPS</v2gci_t:stringValue>
                              </v2gci_t:Parameter>
                              <v2gci_t:Parameter v2gci_t:Name="Port">
                                    <v2gci_t:intValue>443</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 8 – ServiceDetailsRes message

The response message shows that the SECC offers internet access via HTTP using port 80 and via HTTPS using port 443, therefore the EVCC is not allowed to use FTP (Parameter Set ID = 1 or 2) but to use HTTP or HTTPS.

Now, the EVCC request the desired services using the PaymentServiceSelectionReq

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013:MsgBody"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
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:PaymentServiceSelectionReq>
                  <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>3</v2gci_t:ServiceID>
                              <!-- internet service -->
                              <v2gci_t:ParameterSetID>3</v2gci_t:ParameterSetID>
                        </v2gci_t:SelectedService>
                  </v2gci_b:SelectedServiceList>
            </v2gci_b:PaymentServiceSelectionReq>
      </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 9 – PaymentServiceSelectionReq message

D.2 EXI encoded message examples

The following subclause shows 3 examples (SessionSetupRes, ChargeParameterDiscoveryReq, and CurrentDemandRequest) of plain XML V2G message instances and the equivalent EXI format representation.

D.2.1 SessionSetupRes message

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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>FRA23E45B78C</v2gci_b:EVSEID>
            </v2gci_b:SessionSetupRes>
      </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 10 – Plain XML representation of a SessionSetupRes message

```
80 98 02 0C 0C 4C 8C CD 0D 4D 8D D1 E0 00 39 19 49 04 C8 CD 14 D0 D5 08 DC E1 0C 80
```

V2G message example 11 – EXI data stream representation of the SessionSetupRes message

D.2.2 ChargeParameterDiscoveryReq message (AC-based)

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
```

```
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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:RequestedEnergyTransferMode>AC_single_phase_core</v2gci_b:RequestedEnergyTransferMode>
            <v2gci_t:AC_EVChargeParameter>
                <v2gci_t:DepartureTime>100</v2gci_t:DepartureTime>
                <v2gci_t:EAmount>
                    <v2gci_t:Multiplier>3</v2gci_t:Multiplier>
                    <v2gci_t:Unit>Wh</v2gci_t:Unit>
                    <v2gci_t:Value>18</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>230</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>32</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>0</v2gci_t:Value>
                </v2gci_t:EVMinCurrent>
            </v2gci_t:AC_EVChargeParameter>
        </v2gci_b:ChargeParameterDiscoveryReq>
    </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 12 – Plain XML representation of a ChargeParameterDiscoveryReq message (AC-based)

```
80 98 02 0C 0C 4C 8C CD 0D 4D 8D D0 94 00 64 0C 30 09 01 88 1C C0 20 61 81 00 18 60 00 00
```

V2G message example 13 – EXI data stream representation of the ChargeParameterDiscoveryReq message

D.2.3 CurrentDemandReq message

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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>60</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>450</v2gci_t:Value>
                </v2gci_b:EVTargetVoltage>
            </v2gci_b:CurrentDemandReq>
        </v2gci_d:Body>
    </v2gci_d:V2G_Message>
```

V2G message example 14 – Plain XML representation of a CurrentDemandReq message

```
80 98 02 0C 0C 4C 8C CD 0D 4D 8D D0 D1 00 1B 81 86 07 84 10 C4 0C 20 30 00
```

V2G message example 15 – EXI data stream representation of the CurrentDemandReq message

D.3 Schedules and tariff information

D.3.1 Overview

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 15118 V2G CI (see D.3.2)

— "Time Of Use"-based SalesTariff including constant value for GridSchedule (see D.3.3)

— "Time Of Use"-based SalesTariff with dynamic GridSchedule (see D.3.4)

— "Consumption"-based SalesTariff with constant value for GridSchedule (see D.3.5)

— Multiple SalesTariffs with different Demand Limits in GridSchedule (see D.3.6)

— Time of Use-based SalesTariffs with relativePricePercentage (see D.3.7)

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.3.2 Dynamic GridSchedule w/o SalesTariff over ISO 15118 V2G CI

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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:ChargeParameterDiscoveryRes>
              <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
              <v2gci_b:EVSEProcessing>Finished</v2gci_b:EVSEProcessing>
              <v2gci_t:SAScheduleList>
                  <v2gci_t:SAScheduleTuple>
                      <v2gci_t:SAScheduleTupleID>55</v2gci_t:SAScheduleTupleID>
                      <v2gci_t:PMaxSchedule>
                          <v2gci_t:PMaxScheduleEntry>
                              <v2gci_t:RelativeTimeInterval>
                                  <v2gci_t:start>0</v2gci_t:start>
                              </v2gci_t:RelativeTimeInterval>
                              <v2gci_t:PMax>
                                  <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                  <v2gci_t:Unit>W</v2gci_t:Unit>
                                  <v2gci_t:Value>9600</v2gci_t:Value>
```

```
                                    </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>
                                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                        <v2gci_t:Unit>W</v2gci_t:Unit>
                                        <v2gci_t:Value>22000</v2gci_t:Value>
                                    </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>
                                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                        <v2gci_t:Unit>W</v2gci_t:Unit>
                                        <v2gci_t:Value>9600</v2gci_t:Value>
                                    </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>
                                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                        <v2gci_t:Unit>W</v2gci_t:Unit>
                                        <v2gci_t:Value>22000</v2gci_t:Value>
                                    </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>
                                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                        <v2gci_t:Unit>W</v2gci_t:Unit>
                                        <v2gci_t:Value>12000</v2gci_t:Value>
                                    </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:NotificationMaxDelay>0</v2gci_t:NotificationMaxDelay>
                            <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
                            <v2gci_t:RCD>true</v2gci_t:RCD>
                        </v2gci_t:AC_EVSEStatus>
                        <v2gci_t:EVSENominalVoltage>
                            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                            <v2gci_t:Unit>V</v2gci_t:Unit>
                            <v2gci_t:Value>230</v2gci_t:Value>
                        </v2gci_t:EVSENominalVoltage>
                        <v2gci_t:EVSEMaxCurrent>
                            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                            <v2gci_t:Unit>A</v2gci_t:Unit>
                            <v2gci_t:Value>50</v2gci_t:Value>
                        </v2gci_t:EVSEMaxCurrent>
                    </v2gci_t:AC_EVSEChargeParameter>
                </v2gci_b:ChargeParameterDiscoveryRes>
            </v2gci_d:Body>
        </v2gci_d:V2G_Message>
```

V2G message example 16 – ChargeParameterDiscovery Response example for GridSchedule w/o SalesTariff

D.3.3 “Time Of Use”-based SalesTariff with constant value for GridSchedule

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
```

```
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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:ChargeParameterDiscoveryRes>
            <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
            <v2gci_b:EVSEProcessing>Finished</v2gci_b:EVSEProcessing>
            <v2gci_t:SAScheduleList>
                  <v2gci_t:SAScheduleTuple>
                        <v2gci_t:SAScheduleTupleID>55</v2gci_t:SAScheduleTupleID>
                        <v2gci_t:PMaxSchedule>
                              <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>
                                          <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                          <v2gci_t:Unit>W</v2gci_t:Unit>
                                          <v2gci_t:Value>9600</v2gci_t:Value>
                                    </v2gci_t:PMax>
                               </v2gci_t:PMaxScheduleEntry>
                        </v2gci_t:PMaxSchedule>
                        <v2gci_t:SalesTariff v2gci_t:Id="ID001">
                              <v2gci_t:SalesTariffID>67</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:NotificationMaxDelay>0</v2gci_t:NotificationMaxDelay>
                        <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
                        <v2gci_t:RCD>false</v2gci_t:RCD>
                  </v2gci_t:AC_EVSEStatus>
                  <v2gci_t:EVSENominalVoltage>
                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
```

```
                    <v2gci_t:Unit>V</v2gci_t:Unit>
                    <v2gci_t:Value>230</v2gci_t:Value>
                </v2gci_t:EVSENominalVoltage>
                <v2gci_t:EVSEMaxCurrent>
                    <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                    <v2gci_t:Unit>A</v2gci_t:Unit>
                    <v2gci_t:Value>50</v2gci_t:Value>
                </v2gci_t:EVSEMaxCurrent>
            </v2gci_t:AC_EVSEChargeParameter>
        </v2gci_b:ChargeParameterDiscoveryRes>
</v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 17 – ChargeParameterDiscovery Response example for “Time of Use”-based SalesTariff with constant value for GridSchedule

D.3.4 “Time Of Use”-based SalesTariff with dynamic GridSchedule

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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:ChargeParameterDiscoveryRes>
            <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
            <v2gci_b:EVSEProcessing>Finished</v2gci_b:EVSEProcessing>
            <v2gci_t:SAScheduleList>
                <v2gci_t:SAScheduleTuple>
                    <v2gci_t:SAScheduleTupleID>55</v2gci_t:SAScheduleTupleID>
                    <v2gci_t:PMaxSchedule>
                        <v2gci_t:PMaxScheduleEntry>
                            <v2gci_t:RelativeTimeInterval>
                                <v2gci_t:start>0</v2gci_t:start>
                            </v2gci_t:RelativeTimeInterval>
                            <v2gci_t:PMax>
                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                <v2gci_t:Value>9600</v2gci_t:Value>
                            </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>
                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                <v2gci_t:Value>22000</v2gci_t:Value>
                            </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>
                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                <v2gci_t:Value>9600</v2gci_t:Value>
                            </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>
                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                <v2gci_t:Value>22000</v2gci_t:Value>
```

```
                                    </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>
                                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                        <v2gci_t:Unit>W</v2gci_t:Unit>
                                        <v2gci_t:Value>12000</v2gci_t:Value>
                                    </v2gci_t:PMax>
                                </v2gci_t:PMaxScheduleEntry>
                            </v2gci_t:PMaxSchedule>
                            <v2gci_t:SalesTariff v2gci_t:Id="ID001">
                                <!-- used by XML Signature Framework for signature assignment in msg header -->
                                <v2gci_t:SalesTariffID>65</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:NotificationMaxDelay>0</v2gci_t:NotificationMaxDelay>
                            <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
                            <v2gci_t:RCD>false</v2gci_t:RCD>
                        </v2gci_t:AC_EVSEStatus>
                        <v2gci_t:EVSENominalVoltage>
                            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                            <v2gci_t:Unit>V</v2gci_t:Unit>
                            <v2gci_t:Value>230</v2gci_t:Value>
                        </v2gci_t:EVSENominalVoltage>
                        <v2gci_t:EVSEMaxCurrent>
                            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                            <v2gci_t:Unit>A</v2gci_t:Unit>
                            <v2gci_t:Value>50</v2gci_t:Value>
                        </v2gci_t:EVSEMaxCurrent>
                    </v2gci_t:AC_EVSEChargeParameter>
                </v2gci_b:ChargeParameterDiscoveryRes>
            </v2gci_d:Body>
        </v2gci_d:V2G_Message>
```

V2G message example 18 – ChargeParameterDiscovery Response example for "Time of Use"-based SalesTariff with dynamic GridSchedule

D.3.5 "Consumption"-based SalesTariff with constant value for GridSchedule

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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:ChargeParameterDiscoveryRes>
            <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
            <v2gci_b:EVSEProcessing>Finished</v2gci_b:EVSEProcessing>
            <v2gci_t:SAScheduleList>
                <v2gci_t:SAScheduleTuple>
                    <v2gci_t:SAScheduleTupleID>67</v2gci_t:SAScheduleTupleID>
                    <v2gci_t:PMaxSchedule>
                        <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>
                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                <v2gci_t:Value>9600</v2gci_t:Value>
                            </v2gci_t:PMax>
                        </v2gci_t:PMaxScheduleEntry>
                    </v2gci_t:PMaxSchedule>
                    <v2gci_t:SalesTariff v2gci_t:Id="ID001">
                        <v2gci_t:SalesTariffID>45</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>
                                    <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                    <v2gci_t:Unit>W</v2gci_t:Unit>
                                    <v2gci_t:Value>0</v2gci_t:Value>
                                </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>
                                    <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                    <v2gci_t:Unit>W</v2gci_t:Unit>
                                    <v2gci_t:Value>10000</v2gci_t:Value>
                                </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>
                                    <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                    <v2gci_t:Unit>W</v2gci_t:Unit>
                                    <v2gci_t:Value>20000</v2gci_t:Value>
```

```
                                    </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:NotificationMaxDelay>0</v2gci_t:NotificationMaxDelay>
                        <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
                        <v2gci_t:RCD>false</v2gci_t:RCD>
                    </v2gci_t:AC_EVSEStatus>
                    <v2gci_t:EVSENominalVoltage>
                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                        <v2gci_t:Unit>V</v2gci_t:Unit>
                        <v2gci_t:Value>230</v2gci_t:Value>
                    </v2gci_t:EVSENominalVoltage>
                    <v2gci_t:EVSEMaxCurrent>
                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                        <v2gci_t:Unit>A</v2gci_t:Unit>
                        <v2gci_t:Value>50</v2gci_t:Value>
                    </v2gci_t:EVSEMaxCurrent>
                </v2gci_t:AC_EVSEChargeParameter>
            </v2gci_b:ChargeParameterDiscoveryRes>
        </v2gci_d:Body>
    </v2gci_d:V2G_Message>
```

V2G message example 19 – ChargeParameterDiscovery Response example for “Consumption”-based SalesTariff with constant value for GridSchedule

D.3.6 Multiple SalesTariffs with different Demand Limits in GridSchedule

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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:ChargeParameterDiscoveryRes>
            <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
            <v2gci_b:EVSEProcessing>Finished</v2gci_b:EVSEProcessing>
            <v2gci_t:SAScheduleList>
                <v2gci_t:SAScheduleTuple>
                    <v2gci_t:SAScheduleTupleID>55</v2gci_t:SAScheduleTupleID>
                    <v2gci_t:PMaxSchedule>
                        <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>
                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                <v2gci_t:Value>22000</v2gci_t:Value>
                            </v2gci_t:PMax>
                        </v2gci_t:PMaxScheduleEntry>
                    </v2gci_t:PMaxSchedule>
                    <v2gci_t:SalesTariff v2gci_t:Id="ID001">
                        <!-- used by XML Signature Framework for signature assignment in msg header -->
                        <v2gci_t:SalesTariffID>67</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>56</v2gci_t:SAScheduleTupleID>
                            <v2gci_t:PMaxSchedule>
                                <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>
                                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                        <v2gci_t:Unit>V</v2gci_t:Unit>
                                        <v2gci_t:Value>11000</v2gci_t:Value>
                                    </v2gci_t:PMax>
                                </v2gci_t:PMaxScheduleEntry>
                            </v2gci_t:PMaxSchedule>
                            <v2gci_t:SalesTariff v2gci_t:Id="ID002">
                                <!-- used by XML Signature Framework for signature assignment in msg header -->
                                <v2gci_t:SalesTariffID>68</v2gci_t:SalesTariffID>
                                <v2gci_t:SalesTariffDescription>SalesTariffDescription2</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:NotificationMaxDelay>0</v2gci_t:NotificationMaxDelay>
                            <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
                            <v2gci_t:RCD>false</v2gci_t:RCD>
                        </v2gci_t:AC_EVSEStatus>
                        <v2gci_t:EVSENominalVoltage>
                            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                            <v2gci_t:Unit>V</v2gci_t:Unit>
                            <v2gci_t:Value>230</v2gci_t:Value>
                        </v2gci_t:EVSENominalVoltage>
                        <v2gci_t:EVSEMaxCurrent>
                            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                            <v2gci_t:Unit>A</v2gci_t:Unit>
                            <v2gci_t:Value>50</v2gci_t:Value>
                        </v2gci_t:EVSEMaxCurrent>
                    </v2gci_t:AC_EVSEChargeParameter>
                </v2gci_b:ChargeParameterDiscoveryRes>
            </v2gci_d:Body>
        </v2gci_d:V2G_Message>
```

V2G message example 20 – ChargeParameterDiscoveryRes example for Multiple SalesTariffs with different Demand Limits in GridSchedule

D.3.7 Time of Use-based SalesTariffs including relativePricePercentage

EXAMPLE 1 Time of use-based SalesTariff T1 with optional ConsumptionCosts of costKind relativePricePercentage

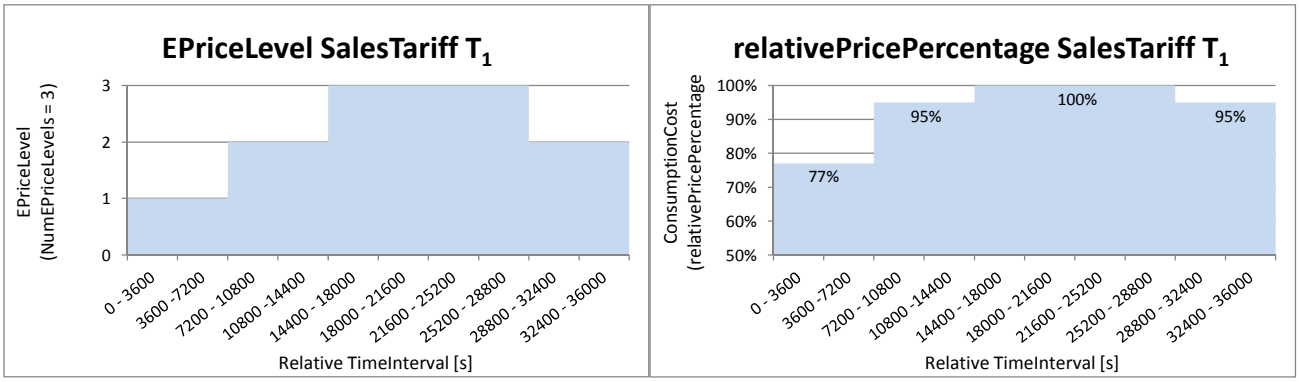


Figure D.1 — Examples for EpriceLevel and relativePricePercentage SalesTariff 1

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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:ChargeParameterDiscoveryRes>
            <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
            <v2gci_b:EVSEProcessing>Finished</v2gci_b:EVSEProcessing>
            <v2gci_t:SAScheduleList>
                <v2gci_t:SAScheduleTuple>
                    <v2gci_t:SAScheduleTupleID>55</v2gci_t:SAScheduleTupleID>
                    <v2gci_t:PMaxSchedule>
                        <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>
                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                <v2gci_t:Value>22000</v2gci_t:Value>
                            </v2gci_t:PMax>
                        </v2gci_t:PMaxScheduleEntry>
                    </v2gci_t:PMaxSchedule>
                    <v2gci_t:SalesTariff v2gci_t:Id="ID001">
                        <!-- used by XML Signature Framework for signature assignment in msg header -->
                        <v2gci_t:SalesTariffID>67</v2gci_t:SalesTariffID>
                        <v2gci_t:SalesTariffDescription>SalesTariff T1</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:ConsumptionCost>
                                <v2gci_t:startValue>
                                    <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                    <v2gci_t:Unit>W</v2gci_t:Unit>
                                    <v2gci_t:Value>0</v2gci_t:Value>
                                </v2gci_t:startValue>
                                <v2gci_t:Cost>
                                    <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
                                    <v2gci_t:amount>77</v2gci_t:amount>
                                    <v2gci_t:amountMultiplier>0</v2gci_t:amountMultiplier>
                                </v2gci_t:Cost>
                            </v2gci_t:ConsumptionCost>
                        </v2gci_t:SalesTariffEntry>
                        <v2gci_t:SalesTariffEntry>
                            <v2gci_t:RelativeTimeInterval>
```

```
                                    <v2gci_t:start>7200</v2gci_t:start>
                                </v2gci_t:RelativeTimeInterval>
                                <v2gci_t:EPriceLevel>2</v2gci_t:EPriceLevel>
                                <v2gci_t:ConsumptionCost>
                                    <v2gci_t:startValue>
                                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                        <v2gci_t:Unit>W</v2gci_t:Unit>
                                        <v2gci_t:Value>0</v2gci_t:Value>
                                    </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:SalesTariffEntry>
                            <v2gci_t:SalesTariffEntry>
                                <v2gci_t:RelativeTimeInterval>
                                    <v2gci_t:start>14400</v2gci_t:start>
                                </v2gci_t:RelativeTimeInterval>
                                <v2gci_t:EPriceLevel>3</v2gci_t:EPriceLevel>
                                <v2gci_t:ConsumptionCost>
                                    <v2gci_t:startValue>
                                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                        <v2gci_t:Unit>W</v2gci_t:Unit>
                                        <v2gci_t:Value>0</v2gci_t:Value>
                                    </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:SalesTariffEntry>
                                <v2gci_t:RelativeTimeInterval>
                                    <v2gci_t:start>28800</v2gci_t:start>
                                    <v2gci_t:duration>7200</v2gci_t:duration>
                                </v2gci_t:RelativeTimeInterval>
                                <v2gci_t:EPriceLevel>2</v2gci_t:EPriceLevel>
                                <v2gci_t:ConsumptionCost>
                                    <v2gci_t:startValue>
                                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                        <v2gci_t:Unit>W</v2gci_t:Unit>
                                        <v2gci_t:Value>0</v2gci_t:Value>
                                    </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:SalesTariffEntry>
                        </v2gci_t:SalesTariff>
                    </v2gci_t:SAScheduleTuple>
                </v2gci_t:SAScheduleList>
                <v2gci_t:AC_EVSEChargeParameter>
                    <v2gci_t:AC_EVSEStatus>
                        <v2gci_t:NotificationMaxDelay>0</v2gci_t:NotificationMaxDelay>
                        <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
                        <v2gci_t:RCD>false</v2gci_t:RCD>
                    </v2gci_t:AC_EVSEStatus>
                    <v2gci_t:EVSENominalVoltage>
                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                        <v2gci_t:Unit>V</v2gci_t:Unit>
                        <v2gci_t:Value>230</v2gci_t:Value>
                    </v2gci_t:EVSENominalVoltage>
                    <v2gci_t:EVSEMaxCurrent>
                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                        <v2gci_t:Unit>A</v2gci_t:Unit>
                        <v2gci_t:Value>50</v2gci_t:Value>
                    </v2gci_t:EVSEMaxCurrent>
                </v2gci_t:AC_EVSEChargeParameter>
            </v2gci_b:ChargeParameterDiscoveryRes>
        </v2gci_d:Body>
```

</v2gci_d:V2G_Message>

V2G message example 21 – Charge Parameter Discovery Response example for Time-of-Use based SalesTariff T1 with relativePricePercentage

EXAMPLE 2 Time of use-based SalesTariff T2 with optional ConsumptionCosts of costKind relativePricePercentage

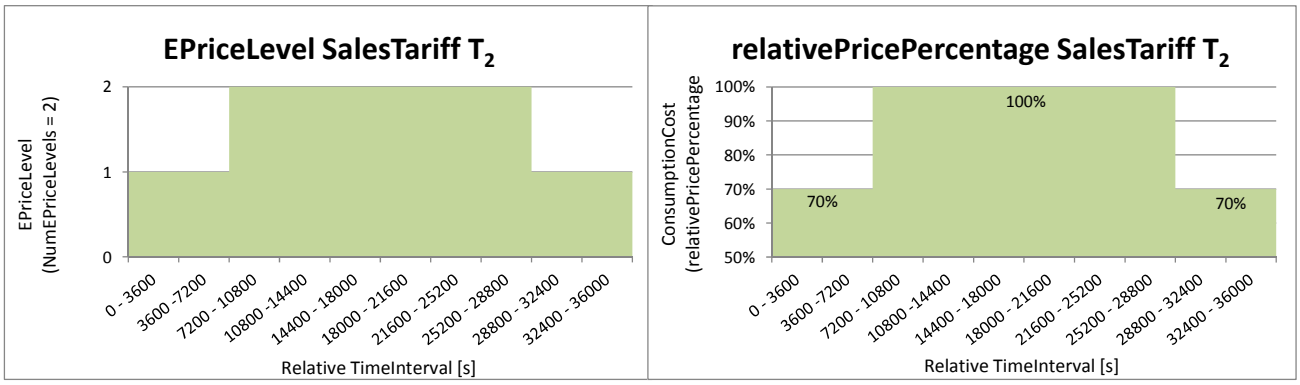


Figure D.2 — Examples for EpriceLevel and relativePricePercentage SalesTariff 2

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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:ChargeParameterDiscoveryRes>
            <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
            <v2gci_b:EVSEProcessing>Finished</v2gci_b:EVSEProcessing>
            <v2gci_t:SAScheduleList>
                <v2gci_t:SAScheduleTuple>
                    <v2gci_t:SAScheduleTupleID>55</v2gci_t:SAScheduleTupleID>
                    <v2gci_t:PMaxSchedule>
                        <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>
                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                <v2gci_t:Value>22000</v2gci_t:Value>
                            </v2gci_t:PMax>
                        </v2gci_t:PMaxScheduleEntry>
                    </v2gci_t:PMaxSchedule>
                    <v2gci_t:SalesTariff v2gci_t:Id="ID001">
                        <!-- used by XML Signature Framework for signature assignment in msg header -->
                        <v2gci_t:SalesTariffID>67</v2gci_t:SalesTariffID>
                        <v2gci_t:SalesTariffDescription>SalesTariff 2</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:RelativeTimeInterval>
                            <v2gci_t:EPriceLevel>1</v2gci_t:EPriceLevel>
                            <v2gci_t:ConsumptionCost>
                                <v2gci_t:startValue>
                                    <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                    <v2gci_t:Unit>W</v2gci_t:Unit>
                                    <v2gci_t:Value>0</v2gci_t:Value>
                                </v2gci_t:startValue>
                                <v2gci_t:Cost>
                                    <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
                                    <v2gci_t:amount>70</v2gci_t:amount>
                                    <v2gci_t:amountMultiplier>0</v2gci_t:amountMultiplier>
                                </v2gci_t:Cost>
                            </v2gci_t:ConsumptionCost>
                        </v2gci_t:SalesTariffEntry>
```

```
                    <v2gci_t:SalesTariffEntry>
                        <v2gci_t:RelativeTimeInterval>
                            <v2gci_t:start>7200</v2gci_t:start>
                        </v2gci_t:RelativeTimeInterval>
                        <v2gci_t:EPriceLevel>2</v2gci_t:EPriceLevel>
                        <v2gci_t:ConsumptionCost>
                            <v2gci_t:startValue>
                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                <v2gci_t:Value>0</v2gci_t:Value>
                            </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:SalesTariffEntry>
                        <v2gci_t:RelativeTimeInterval>
                            <v2gci_t:start>28000</v2gci_t:start>
                            <v2gci_t:duration>7200</v2gci_t:duration>
                        </v2gci_t:RelativeTimeInterval>
                        <v2gci_t:EPriceLevel>1</v2gci_t:EPriceLevel>
                        <v2gci_t:ConsumptionCost>
                            <v2gci_t:startValue>
                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                <v2gci_t:Value>0</v2gci_t:Value>
                            </v2gci_t:startValue>
                            <v2gci_t:Cost>
                                <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
                                <v2gci_t:amount>70</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:NotificationMaxDelay>0</v2gci_t:NotificationMaxDelay>
                <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
                <v2gci_t:RCD>false</v2gci_t:RCD>
            </v2gci_t:AC_EVSEStatus>
            <v2gci_t:EVSENominalVoltage>
                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                <v2gci_t:Unit>V</v2gci_t:Unit>
                <v2gci_t:Value>230</v2gci_t:Value>
            </v2gci_t:EVSENominalVoltage>
            <v2gci_t:EVSEMaxCurrent>
                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                <v2gci_t:Unit>A</v2gci_t:Unit>
                <v2gci_t:Value>50</v2gci_t:Value>
            </v2gci_t:EVSEMaxCurrent>
        </v2gci_t:AC_EVSEChargeParameter>
    </v2gci_b:ChargeParameterDiscoveryRes>
</v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 22 – Charge Parameter Discovery Response example for Time-of-Use based SalesTariff T2 with relativePricePercentage

EXAMPLE 3 In this example the exact same SalesTariffs T1 and T2 from the previous two examples are now provided as two tariffs within one ChargeParameterDiscovery Req/Res pattern. Note, the different assignment of EPriceLevels for T1 and T2 as well as relativePricePercentage for T1.

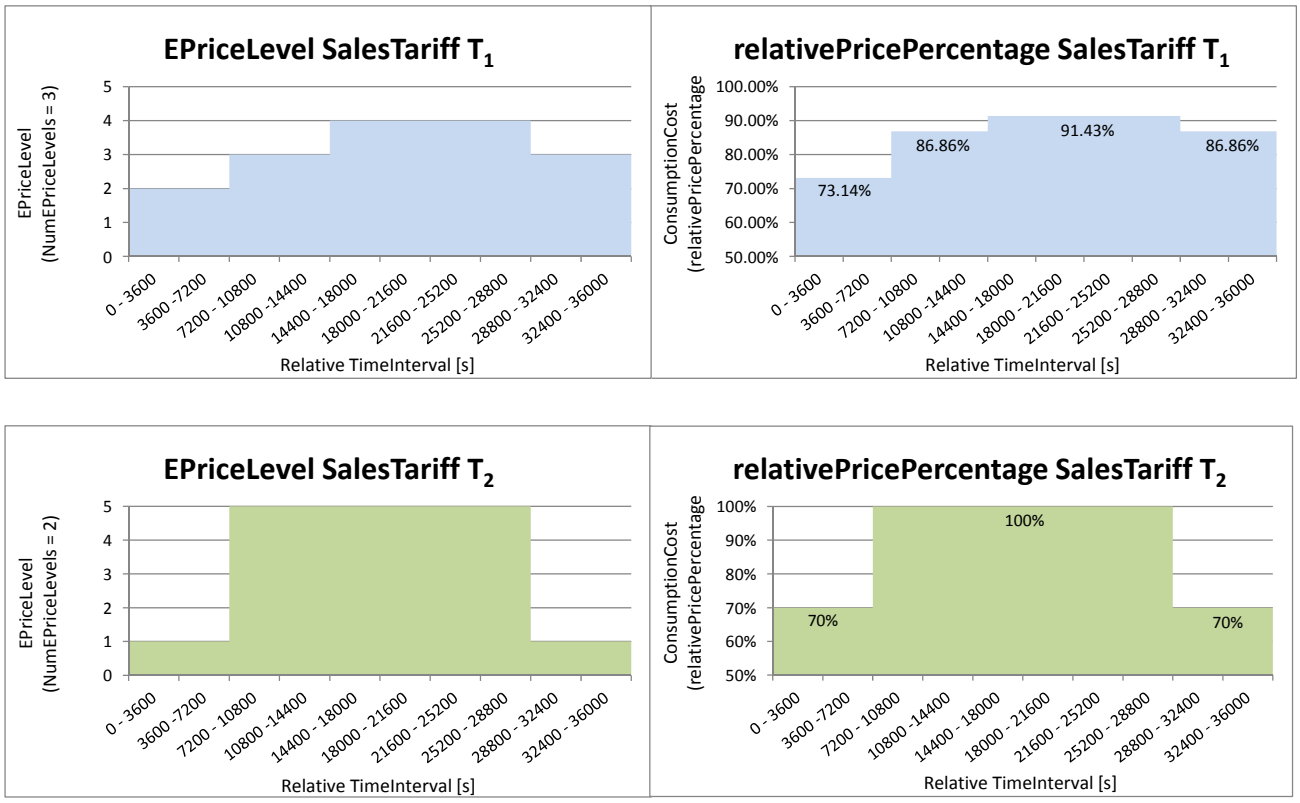


Figure D.3 — Examples for EpriceLevel and relativePricePercentage SalesTariff 3

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013: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:ChargeParameterDiscoveryRes>
                  <v2gci_b:ResponseCode>OK</v2gci_b:ResponseCode>
                  <v2gci_b:EVSEProcessing>Finished</v2gci_b:EVSEProcessing>
                  <v2gci_t:SAScheduleList>
                        <v2gci_t:SAScheduleTuple>
                              <v2gci_t:SAScheduleTupleID>55</v2gci_t:SAScheduleTupleID>
                              <v2gci_t:PMaxSchedule>
                                    <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>
                                                <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                                                <v2gci_t:Unit>W</v2gci_t:Unit>
                                                <v2gci_t:Value>22000</v2gci_t:Value>
                                          </v2gci_t:PMax>
                                    </v2gci_t:PMaxScheduleEntry>
                              </v2gci_t:PMaxSchedule>
                              <v2gci_t:SalesTariff v2gci_t:Id="ID001">
```

```
<!-- used by XML Signature Framework for signature assignment in msg header -->
<v2gci_t:SalesTariffID>67</v2gci_t:SalesTariffID>
<v2gci_t:SalesTariffDescription>SalesTariff T1</v2gci_t:SalesTariffDescription>
<v2gci_t:NumEPriceLevels>5</v2gci_t:NumEPriceLevels>
<v2gci_t:SalesTariffEntry>
    <v2gci_t:RelativeTimeInterval>
        <v2gci_t:start>0</v2gci_t:start>
    </v2gci_t:RelativeTimeInterval>
    <v2gci_t:EPriceLevel>2</v2gci_t:EPriceLevel>
    <v2gci_t:ConsumptionCost>
        <v2gci_t:startValue>
            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
            <v2gci_t:Unit>W</v2gci_t:Unit>
            <v2gci_t:Value>0</v2gci_t:Value>
        </v2gci_t:startValue>
        <v2gci_t:Cost>
            <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
            <v2gci_t:amount>7314</v2gci_t:amount>
            <v2gci_t:amountMultiplier>-2</v2gci_t:amountMultiplier>
        </v2gci_t:Cost>
    </v2gci_t:ConsumptionCost>
</v2gci_t:SalesTariffEntry>
<v2gci_t:SalesTariffEntry>
    <v2gci_t:RelativeTimeInterval>
        <v2gci_t:start>7200</v2gci_t:start>
    </v2gci_t:RelativeTimeInterval>
    <v2gci_t:EPriceLevel>3</v2gci_t:EPriceLevel>
    <v2gci_t:ConsumptionCost>
        <v2gci_t:startValue>
            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
            <v2gci_t:Unit>W</v2gci_t:Unit>
            <v2gci_t:Value>0</v2gci_t:Value>
        </v2gci_t:startValue>
        <v2gci_t:Cost>
            <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
            <v2gci_t:amount>8686</v2gci_t:amount>
            <v2gci_t:amountMultiplier>-2</v2gci_t:amountMultiplier>
        </v2gci_t:Cost>
    </v2gci_t:ConsumptionCost>
</v2gci_t:SalesTariffEntry>
<v2gci_t:SalesTariffEntry>
    <v2gci_t:RelativeTimeInterval>
        <v2gci_t:start>14400</v2gci_t:start>
    </v2gci_t:RelativeTimeInterval>
    <v2gci_t:EPriceLevel>4</v2gci_t:EPriceLevel>
    <v2gci_t:ConsumptionCost>
        <v2gci_t:startValue>
            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
            <v2gci_t:Unit>W</v2gci_t:Unit>
            <v2gci_t:Value>0</v2gci_t:Value>
        </v2gci_t:startValue>
        <v2gci_t:Cost>
            <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
            <v2gci_t:amount>9143</v2gci_t:amount>
            <v2gci_t:amountMultiplier>-2</v2gci_t:amountMultiplier>
        </v2gci_t:Cost>
    </v2gci_t:ConsumptionCost>
</v2gci_t:SalesTariffEntry>
<v2gci_t:SalesTariffEntry>
    <v2gci_t:RelativeTimeInterval>
        <v2gci_t:start>28000</v2gci_t:start>
        <v2gci_t:duration>7200</v2gci_t:duration>
    </v2gci_t:RelativeTimeInterval>
    <v2gci_t:EPriceLevel>3</v2gci_t:EPriceLevel>
    <v2gci_t:ConsumptionCost>
        <v2gci_t:startValue>
            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
            <v2gci_t:Unit>W</v2gci_t:Unit>
            <v2gci_t:Value>0</v2gci_t:Value>
        </v2gci_t:startValue>
        <v2gci_t:Cost>
            <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
            <v2gci_t:amount>8686</v2gci_t:amount>
            <v2gci_t:amountMultiplier>-2</v2gci_t:amountMultiplier>
```

```
                        </v2gci_t:Cost>
                    </v2gci_t:ConsumptionCost>
                </v2gci_t:SalesTariffEntry>
            </v2gci_t:SalesTariff>
        </v2gci_t:SAScheduleTuple>
        <v2gci_t:SAScheduleTuple>
            <v2gci_t:SAScheduleTupleID>84</v2gci_t:SAScheduleTupleID>
            <v2gci_t:PMaxSchedule>
                <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>
                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                        <v2gci_t:Unit>W</v2gci_t:Unit>
                        <v2gci_t:Value>22000</v2gci_t:Value>
                    </v2gci_t:PMax>
                </v2gci_t:PMaxScheduleEntry>
            </v2gci_t:PMaxSchedule>
            <v2gci_t:SalesTariff v2gci_t:Id="ID002">
                <!-- used by XML Signature Framework for signature assignment in msg header -->
                <v2gci_t:SalesTariffID>68</v2gci_t:SalesTariffID>
                <v2gci_t:SalesTariffDescription>SalesTariff 2</v2gci_t:SalesTariffDescription>
                <v2gci_t:NumEPriceLevels>5</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:ConsumptionCost>
                        <v2gci_t:startValue>
                            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                            <v2gci_t:Unit>W</v2gci_t:Unit>
                            <v2gci_t:Value>0</v2gci_t:Value>
                        </v2gci_t:startValue>
                        <v2gci_t:Cost>
                            <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
                            <v2gci_t:amount>70</v2gci_t:amount>
                            <v2gci_t:amountMultiplier>0</v2gci_t:amountMultiplier>
                        </v2gci_t:Cost>
                    </v2gci_t:ConsumptionCost>
                </v2gci_t:SalesTariffEntry>
                <v2gci_t:SalesTariffEntry>
                    <v2gci_t:RelativeTimeInterval>
                        <v2gci_t:start>7200</v2gci_t:start>
                    </v2gci_t:RelativeTimeInterval>
                    <v2gci_t:EPriceLevel>5</v2gci_t:EPriceLevel>
                    <v2gci_t:ConsumptionCost>
                        <v2gci_t:startValue>
                            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                            <v2gci_t:Unit>W</v2gci_t:Unit>
                            <v2gci_t:Value>0</v2gci_t:Value>
                        </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:SalesTariffEntry>
                    <v2gci_t:RelativeTimeInterval>
                        <v2gci_t:start>28000</v2gci_t:start>
                        <v2gci_t:duration>7200</v2gci_t:duration>
                    </v2gci_t:RelativeTimeInterval>
                    <v2gci_t:EPriceLevel>1</v2gci_t:EPriceLevel>
                    <v2gci_t:ConsumptionCost>
                        <v2gci_t:startValue>
                            <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                            <v2gci_t:Unit>W</v2gci_t:Unit>
                            <v2gci_t:Value>0</v2gci_t:Value>
                        </v2gci_t:startValue>
                        <v2gci_t:Cost>
                            <v2gci_t:costKind>relativePricePercentage</v2gci_t:costKind>
```

```
                                        <v2gci_t:amount>70</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:NotificationMaxDelay>0</v2gci_t:NotificationMaxDelay>
                        <v2gci_t:EVSENotification>None</v2gci_t:EVSENotification>
                        <v2gci_t:RCD>false</v2gci_t:RCD>
                    </v2gci_t:AC_EVSEStatus>
                    <v2gci_t:EVSENominalVoltage>
                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                        <v2gci_t:Unit>V</v2gci_t:Unit>
                        <v2gci_t:Value>230</v2gci_t:Value>
                    </v2gci_t:EVSENominalVoltage>
                    <v2gci_t:EVSEMaxCurrent>
                        <v2gci_t:Multiplier>0</v2gci_t:Multiplier>
                        <v2gci_t:Unit>A</v2gci_t:Unit>
                        <v2gci_t:Value>50</v2gci_t:Value>
                    </v2gci_t:EVSEMaxCurrent>
                </v2gci_t:AC_EVSEChargeParameter>
            </v2gci_b:ChargeParameterDiscoveryRes>
        </v2gci_d:Body>
    </v2gci_d:V2G_Message>
```

V2G message example 23 – Charge Parameter Discovery Response example for Time-of-Use based SalesTariffs T1 and T2 with relativePricePercentage

Annex E
(informative)

Application of certificates

E.1 General information

E.1.1 Overview

This Annex provides an overview about the application of certificates and the demands, which need to be fulfilled covered by the certificate concept implemented in this standard. It also explains the certificate parameters like validity, size, chain length and others. This annex does not constitute further requirements.

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 certificates. The corresponding private keys are in possession of the respective Root CAs.

— Mobility Operator Root Certificate: This kind of certificate is used to sign (via a chain of Sub-CAs) contract certificates.

— Contract Certificate: This kind of certificate is used in the Plug and Charge use case (i.e. contract-based charging) to represent a contract between a vehicle and a secondary actor (the Mobility Operator). It is stored in a EVCC along with the corresponding private key. The EVCC uses it to prove the existence of the corresponding contract to the EVSE. Contract Certificates are derived from a Mobility Operator Root Certificate.

— SECC Certificate: This kind of certificate is used to authenticate the SECC to the EVCC. The corresponding private key is in possession of the SECC. SECC Certificates are derived from the V2G Root Certificates mentioned above.

— Private Operate Root Certificate: A Private Operate Root Certificate is very similar to a Mobility Operator Root Certificate. The only difference is that it is organizationally handled differently (described in E.2). Its purpose is to facilitate setting up private charging infrastructure for cars that are under immediate control of the infrastructure owner.

— OEM Provisioning Certificate: This kind of certificate is individual for each vehicle (installed e.g. at vehicle production) and used to verify the identity of a vehicle at the beginning of the provisioning process (see also 0 for Certificate Provisioning).

— OEM Root Certificate: Such an OEM root certificate is used to sign OEM provisioning certificates. Each OEM may create a (top level) root certificate and distribute it to the secondary actors. The root certificate of an OEM is not part of the global PKI; i.e., it is not necessarily signed by a V2G Root Certificate.

In the following we explain the general demands and restrictions caused by the OEMs (refer to clause E.1.2) and the secondary actors (refer to clause E.1.3) 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.1.4). Finally, we give a visual overview of the resulting certificate structure and usage (refer to clause E.1.5).

E.1.2 Demands of the OEM

An OEM typically has the following general demands that result from his desire to keep control units (here the SECC) from becoming very expensive. For the same reasons, manual treatment of a control unit (e.g. in a garage) 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 garage. To enable certificate installation only at vehicle production, a certificate has to be “static”. That means, the certificate needs a very long validity. Since vehicles are used for 20 years or longer, the certificate needs an even longer validity. Such static certificates for instance may be the root certificates (of the PKI) that are stored in the vehicle.

R2 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 has to 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 garage). 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-demands: (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.1.3 Demands of the Secondary Actors

A secondary actor resp. the organization that manages its PKI (Public Key Infrastructure) typically has the general demands 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 needs to have the possibility to create its certificates itself. For this purpose, it needs 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 needed (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 should be possible that an EVSE stays offline during a whole charging procedure.

NOTE Demand 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 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, demand 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.1.4 Rationale for Decisions in this Standard

Based on the demands presented in clause E.1.2 and E.1.3 we now present and explain the decisions made in this standard. Since some of the demands result in conflictive goals, some of the decisions are a compromise between the demands of the OEMs and the demands of the secondary actors.

D1 Size of a single certificate: Because of demand R3a, certificate size is limited. According to [V2G2-010] the size of a certificate in DER encoded form is 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 demand R3b complete certificates incl. their chains have to be small. Demand 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 demand R3c, a small number of root certificates is preferred. Demand 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 demand R1 the (static) root certificates (installed already at vehicle production time) need to have an almost infinite validity. Respecting the lifetime of a vehicle, [V2G2-012] requests for any point in time there is 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 is 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 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 CertificateInstallationReq/Res. [V2G2-228] and [V2G2-231] requests that the SECC has to support the messages CertificateUpdateReq / 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 demand 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. demand 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 is 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 garage (cf. demand 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: Mechanisms to revoke SECC Certificates are not specified but instead, SECC certificates are short term certificates.

E.1.5 Overview of the resulting certificate structure

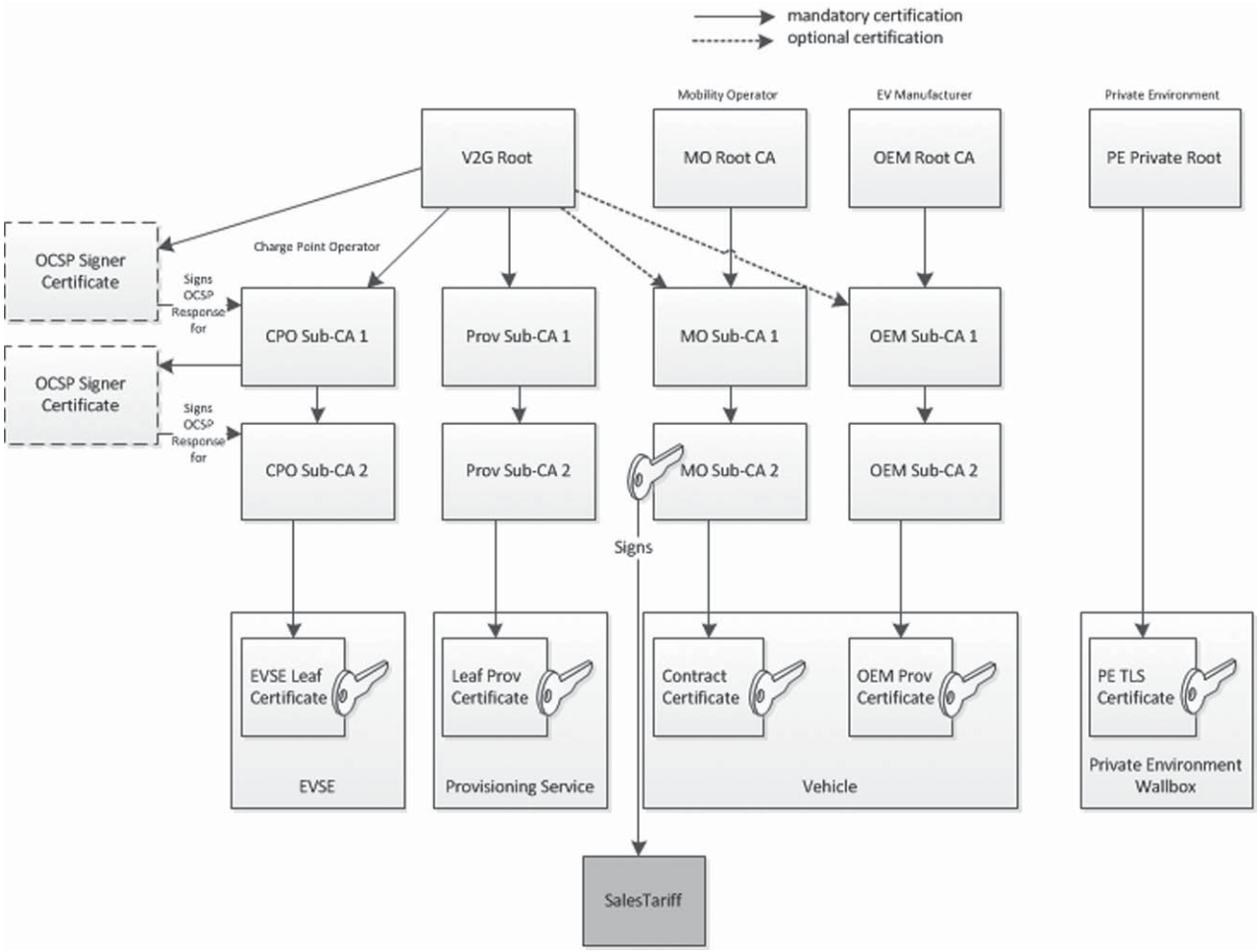


Figure E.1 — Overview certificate structure

Figure E.1 provides a visual overview of the resulting certificate structure and relevant validity periods.

As one can see, the OEM Provisioning Certificates are independent from the PKI of the secondary actors below the (global) root certificates. 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 0. It is, however, allowed to reuse a V2G Root as a Mobility Operator Root Certificate or an OEM Root Certificate (suggested with dotted lines)

All certificate chains have a maximum length of 3; i.e. including the root certificate 4 certificates are involved.

The certificate chain of an SECC 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).

The certificate chain of a Contract Certificate is transmitted into the EVCC without a Root CA. This limits the transmission to 3 certificates, but this also means, that the vehicle cannot verify its own Contract Certificate.

E.2 provides an example of simplified certificate management in private environments.

E.2 Simplified certificate management in Private Environment

E.2.1 Overview (motivation)

The main use cases of this standard imply that the EVSE is a public charging station. However, 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 special characteristics have to be considered:

— There is no connection to a backend (except e.g. a setup phase upon installation)

— Maintenance needs to be minimized, i.e. on regular operation there is no human intervention necessary at all.

— No secondary actor is involved. In particular, a wall-box does not need to communicate any bills to a backend, because no special accounting for energy is done.

This means, OCSP and short lived certificates cannot be applied here. 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). In order to explicitly enable this use case regardless, this standard provides tailored exceptions.

E.2.2 Solution for private environments

E.2.2.1 General

For each local environment, a separate individual Private Operator Root CA is created. Each wall-box in this local environment, is outfitted with a SECC Certificate (signed by the Private Operator Root Certificate of this local environment).

For example, these wall-boxes are delivered to the customer, together with the corresponding Private Operator Root Certificate. This Private Operator Root Certificate is installed into each vehicle that shall be able to charge in this private environment, e.g. using the pairing mechanism or other, OEM proprietary mechanisms.

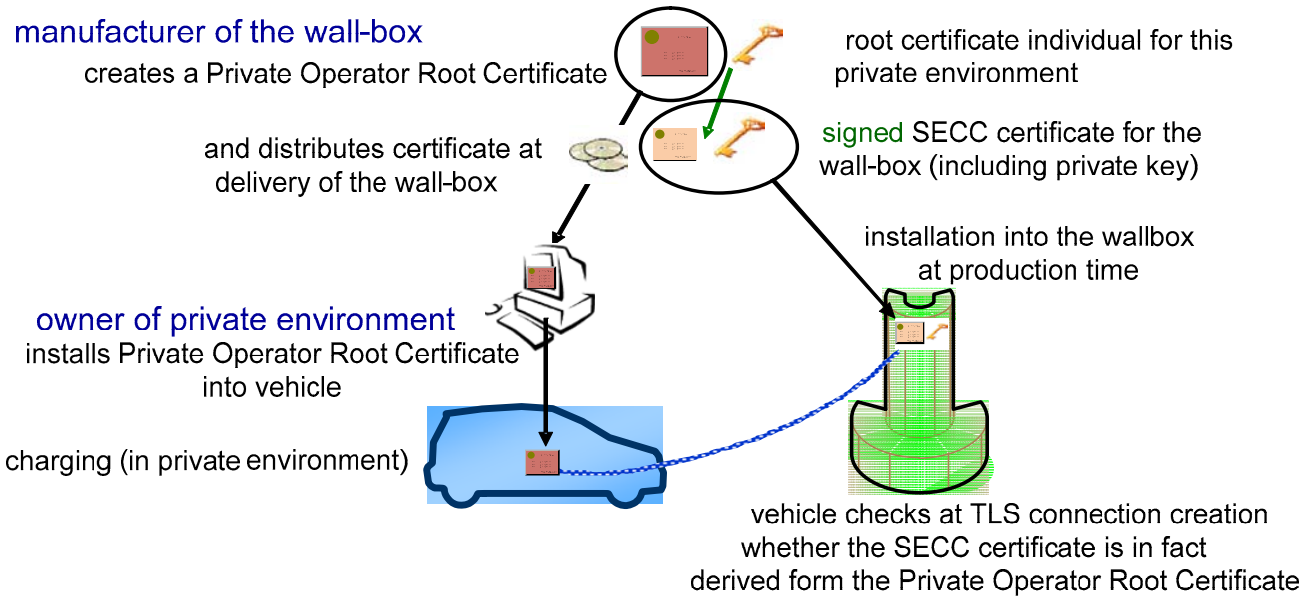


Figure E.2 — Charging in private environments

The EVCC stores one Private Operator Root Certificate for each private environment that it wants to use. Compared to the regular V2G certificate structure, the private structure does not allow Sub-CAs, and therefore all leaf certificates are directly signed by the Private Operator Root Certificate.

In order to avoid frequent exchange of certificates in the wall-box, their SECC Certificates for TLS uses a special validity policy:

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, in contrast to the shortened validity times of SECC 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 that 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 belong to this individual private environment. It is the responsibility of the operator of this private environment to ensure that the SECC Certificates cannot be easily stolen from the wall-boxes, as well as to perform appropriate actions for the case that a leaf certificate was stolen nevertheless (refer to subclause E.2.2.4).

E.2.2.2 Installation of a Private Root Certificate into a Vehicle

The operator of the local environment possesses the Private Operator Root Certificate that belongs to the corresponding wall-boxes (e.g. the certificate is included in the wall-box delivery). 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 can be performed in different ways.

This standard allows for an easy pairing mechanism: The owner first activates a the pairing mode of the EVCC, e.g. by pushing a button or selecting the appropriate menu item. This mode will expire after a specified time. While this mode is active, he connects the vehicle to the EVSE, and the certificate is automatically accepted and installed into the EVCC.

Certificate installation could also be done by using any other communication channel to the vehicle the OEM offers for this purpose. This communication channels may include an OEM online service, a diagnostic interface of the vehicle used at a garage, an USB interface contained in the vehicle, etc.

E.2.2.3 Charging in a Private Environment

From the perspective of the vehicle, charging in a private environment is exactly the same as in a non-private 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 from one of the root certificates stored in the vehicle. Thereby, V2G Root Certificates as well as Private Operator Root Certificates are recognized.

E.2.2.4 Compromised Certificate of a Wall-box

An SECC Certificate of a private wall-box can get 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 safe). 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 disable the Private Operator Root Certificate in all vehicles of the private environment. To reinstate the charging functionality, the operator will then have to deploy a completely new certificate structure starting from a new Private Operator Root Certificate.

As the description of the process shows, two aspects are crucial in this scenario:

a) The SECC Certificates in the wall-boxes are either secured safely or 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 is 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.

E.3 Use of OEM Provisioning Certificates

E.3.1 Introduction

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 situations 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 Certificates, 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 impractical, since such a manual procedure in a garage causes high costs. Other solutions, like installation via a customer web page, also cause effort, can be 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 CertificateInstallationReq/Res. 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 E.3 and described in the following:

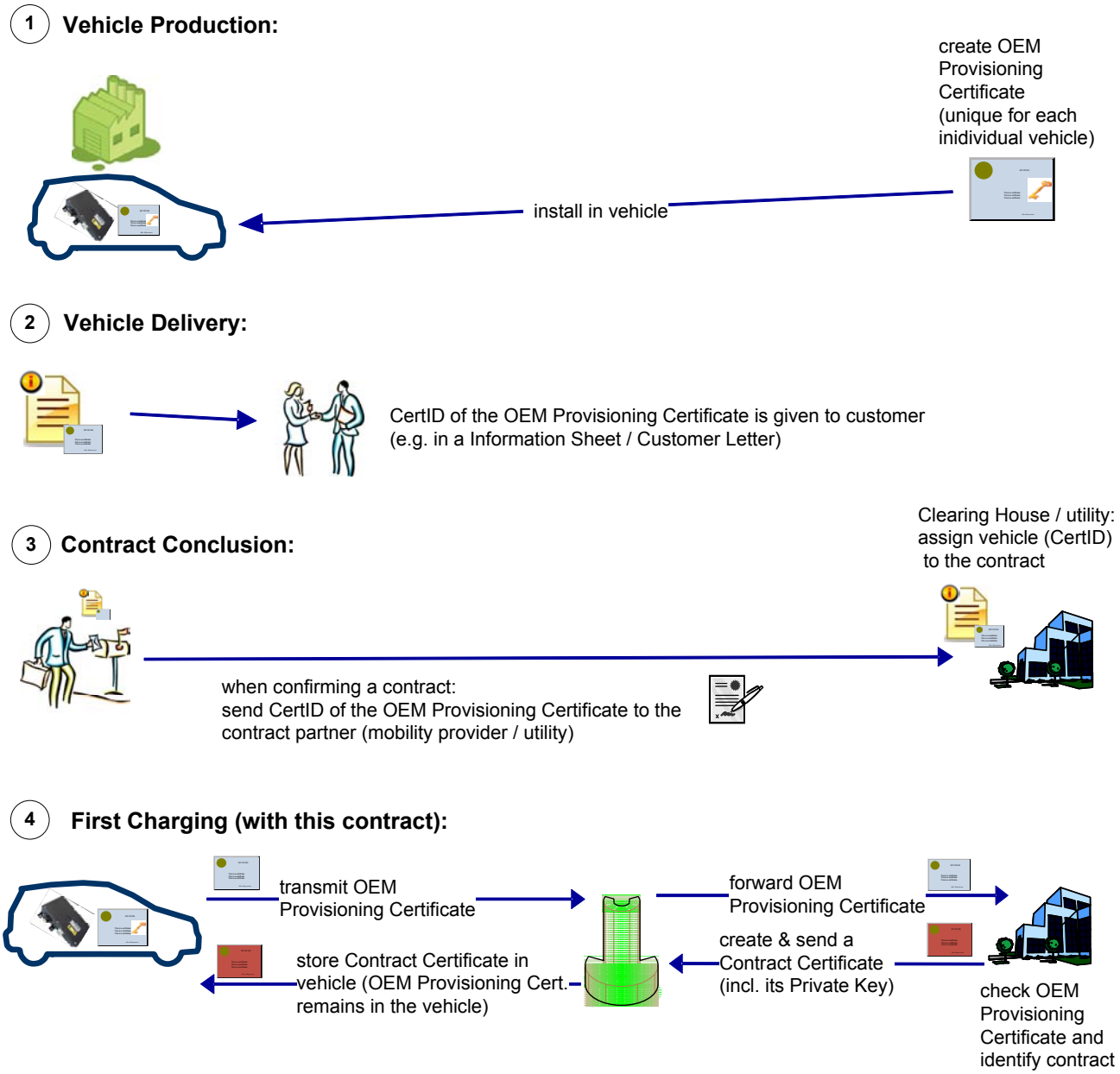


Figure E.3 — Activities required for OEM Certificate Provisioning

E.3.2 Processes

E.3.2.1 Vehicle Production

At vehicle production time, unique OEM Provisioning Certificates and the corresponding private keys are installed in each vehicle (refer to Figure E.3 – Step 1). Each OEM Provisioning Certificate contains a unique ID (for short: CertID) and is derived from the OEM Provisioning Root Certificate of the issuing OEM.

E.3.2.2 Vehicle Hand-Over

At or before vehicle hand-over, the customer is informed about his/her CertID (refer to Figure E.3 – Step 2) e.g. by distributing information sheets, integrating the CertID in the vehicle documentation, offering online access to the CertID, etc.

E.3.2.3 Contract Conclusion

At conclusion of an energy contract (refer to Figure E.3 – 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).

E.3.2.4 Certificate installation

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 CertificateInstallationReq (refer to Figure E.3 – 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 (i) checks whether the OEM provisioning certificate is authentic (i.e. valid) by using the root certificate of the OEM and (ii) verifies the certificate and checks whether a contract for this provisioning certificate is registered. 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 (including the certificate chain necessary for validation) and the corresponding encrypted private key are sent via the SECC to the vehicle by using the message CertificateInstallationRes (see below). 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; i.e. whenever the vehicle does not possess a valid Contract Certificate.

E.3.2.5 Certificate Update

If the current Contract Certificate is not expired yet, it is used in the messages CertificateUpdateReq / Response in order to renew that Contract Certificate. This avoids manual effort for exchanging an expired Contract Certificate as well. The certificate update messages may even be used by vehicles which do not possess an OEM Provisioning Certificate at all or do not realize the provisioning procedure. However, if the Contract Certificate is expired (e.g. because the vehicle was not charged for a longer time at a public / online charging station) the provisioning procedure described above is needed to refresh the Contract Certificate (or manual effort becomes necessary).

E.3.2.6 Component replacement

When in case of a vehicle repair, the component containing the Contract Certificate needs to 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 was stored solely inside the defect component and is lost. In such a case, the customer (or the garage) announces the replacement of the CertID to all relying parties. Then a current Contract Certificate can be installed using the standard certificate installation procedure (See E.3.2.4). That means that even after a vehicle repair no manual treatment of Contract Certificates becomes necessary.

E.4 Security appliances and their associated certificates

This subclause provides an overview over the different security use cases.

Table E.1 — Security use cases and their associated certificates

		Relevant Credentials		
Security Appliances	**Comment**	**EVCC (EV)**	**SECC (EVSE)**	**Secondary Actor**
Simplified Certificate management in Private Environment Please refer to Annex E.2	**DC/AC Charging with simplified key management** • TLS Authentication of SECC • TLS based Confidential Data Exchange between EV and SECC (Service Discovery, Power Delivery, Metering) • TLS based Integrity of Data between EV and EVSE (Service Discovery, Power Delivery, Metering) TLS: AES used for Stream / Session Encryption, ECDH used for Key Exchange	**PE Private Root CA Certificate**	**PE TLS Key Pair** (PE TLS **Private Key** and **Certificate**) certified by PE Private Root PE Private Root CA Certificate (only to communicate it to the EV)	**PE Private Root CA Key Pair**
DC/AC Charging with EIM (External Identification Means) • transport security, • payment at EVSE	• TLS Authentication of SECC • TLS based Confidential Data Exchange between EV and EVSE (Service Discovery, Power Delivery, Metering) • TLS based Integrity of Data between EV and EVSE (Service Discovery, Power Delivery, Metering) TLS: AES used for Stream / Session Encryption, ECDH used for Key Exchange	V2G **Root Certificates**	**SECC Leaf Key Pair** (up to 10 generations depending on the aging V2G RootKey) (SECC Leaf **Private Key** and **Certificate** for TLS) certified by a Sub-CA which was certified by a V2G Root CA All **CPO Sub-CA Certificates** of the certificate chain below the SECC Leaf Certificate **OCSP Response** Ticket(s) for the corresponding CPO-Sub-CA	none

		Relevant Credentials		
Security Appliances	**Comment**	**EVCC (EV)**	**SECC (EVSE)**	**Secondary Actor**
DC/AC Charging with Plug & Charge • DC/AC Charging with transport security • contracts based payment, with Meter Status Receipt • individual / dynamic tariff tables for customer	• TLS Authentication of SECC • Contract Authentication on Application Layer • TLS based Confidential Data Exchange between EV and EVSE (Service Discovery, Power Delivery, Metering) • TLS based Integrity of Data between EV and EVSE (Service Discovery, Power Delivery, Metering) • Meter Receipt Signing based on Contract Key Pair Appl.Layer: AES for Data Encryption, ECDH used for Key Exchange, ECC for Digital Signature for Tariff Signing TLS: AES used for Stream / Session Encryption, ECDH used for Key Exchange	V2G **Root Certificates** **Contract Key Pair** (Contract **Private Key** and **Certificate** for Contract based Payment) certified by a CA which can be certified by a V2G Root CA. All **MO Sub-CA Certificates** of the certificate chain below the Contract Certificate	**SECC Leaf Key Pair** (up to 10 generations depending on the aging V2G RootKey) (SECC Leaf **Private Key** and **Certificate** for TLS) certified by a Sub-CA which was certified by a V2G Root CA All **CPO Sub-CA Certificates** of the certificate chain below the SECC Leaf Certificate All **MO Root CA Certificates** of the MOs the CPO has contract agreements with. **OCSP Response** Ticket(s) for the corresponding CPO-Sub-CA V2G **Root Certificate** (for considered region)	V2G **Root Certificates** **MO Root CA Certificate** (if not below a V2G Root) **All MO-Sub-CA Certificate** of the certificate chain below the MO-Sub-CA Key Pair issuing Contract Certificates **MO Sub-CA Key Pair (**SA P**rivate Key** and **Certificate)** for Contract Certificate Signing

		Relevant Credentials		
Security Appliances	**Comment**	**EVCC (EV)**	**SECC (EVSE)**	**Secondary Actor**
DC/AC Charging with Plug & Charge with dynamic tariffs, Contract Key updates and Provisioning • DC/AC Charging with transport security • contracts based payment, with Meter Status Receipt • individual / dynamic tariff tables for customer • Contract Certificate Update and Provisioning	• TLS Authentication of SECC • Confidential Data Exchange between EV and EVSE (Service Discovery, Power Delivery, Metering) • Integrity of Data between EV and EVSE (Service Discovery, Power Delivery, Metering) • Integrity of tariff data based on digital signature • Integrity of Contract Keys / Certs based on digital signature • Confidentiality of private key of new Contract Certificates based on encryption Appl.Layer: AES for Data Encryption, ECDH used for Key Exchange, ECC for Digital Signature for Tariff Signing and Contract Provisioning TLS: AES used for Stream / Session Encryption, ECDH used for Key Exchange	**V2G Root Certificates** **Contract Key Pair** (Contract Private Key and Certificate for Contract based Payment) certified by a CA which can be certified by a V2G Root CA. All **MO Sub-CA Certificates** of the certificate chain below the Contract Certificate **OEM Prov Key Pair** (OEM Prov **Private Key** and **Certificate** for Contract Provisioning)	**SECC Leaf Key Pair** (up to 10 generations depending on the aging V2G RootKey) (SECC Leaf **Private Key** and **Certificate** for TLS) certified by a Sub-CA which was certified by a V2G Root CA All **CPO Sub-CA Certificates** of the certificate chain below the SECC Leaf Certificate All **MO Root CA Certificates** of the MOs the CPO has contract agreements with. **OCSP Response** Ticket(s) for the corresponding CPO-Sub-CA V2G **Root Certificate** (for considered region)	V2G **Root Certificates** **MO Root CA Certificate** (if not below a V2G Root) All **MO-Sub-CA Certificates** of the certificate chain below the MO-Sub-CA Key Pair issuing Contract Certificates **MO Sub-CA Key Pair** (SA Private Key and Certificate) for Contract certificate and Tariff Signing **Contract Key Pair** to be delivered to the car All **Prov-Sub-CA Certificates** of the certificate terminating by a V2G Root **Leaf Certificate Prov Key Pair** (Leaf Prov Private Key and Certificate for Contract installation/update) certified by a Sub-CA which was certified by a V2G Root CA **OEM Root CA Certificates** (if not below a V2G Root) All **OEM-Sub-CA Certificate** of the certificate chain below the OEM Prov Certificate **OEM Prov Certificate** for Encryption of Contract Key Pair

Annex F
(normative)

Certificate profiles

[V2G2-884] Each certificate used in this standard shall comply to the appropriate profile specified in this annex i.e. in Table F.1, Table F.2, Table F.3, Table F.4, Table F.5, or Table F.6.

Certificate profiles are presented in tables, divided into several thematic clusters each containing profiles for specific certificate types. In each cluster, the root certificate shall be a V2G Root certificate, if not specified otherwise.

Cells that are shaded in grey have identical requirements in all certificate profiles.

A cell's content has the following meaning:

x	Required
(x)	Optional
-	Must not be present
c	This extension is **critical**, see IETF RFC 5280. If a implementation recognizes that a "critical" extension is present, but the implementation cannot interpret the extension, the implementation has to reject the certificate.
nc	This extension is **non-critical**, see IETF RFC 5280. If a implementation recognizes that a "non-critical" extension is present, but the implementation cannot interpret the extension, the extension can be ignored. Therefore, all optional fields should also be "non-critical". Quote from RFC 5280: "A certificate-using system MUST reject the certificate if it encounters a critical extension it does not recognize or a critical extension that contains information that it cannot process. A non-critical extension MAY be ignored if it is not recognized, but MUST be processed if it is recognized."
PCID	Provisioning Certificate ID (max. 17)
CPID	Charge Point ID
PEID	Private Environment ID

NOTE 1 The following OIDs apply: { iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) 2 } and {iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3) prime(1) prime256v1(7) }

NOTE 2 Online Certificate Status Protocol (OCSP), defined in IETF RFC 6960 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. This means, 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 guaranteed 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 IETF RFC 6960.

Table F.1 — V2G Root Certificate

Cluster:		V2G Root
Name:		V2G Root
Typ:		Root
tbsCertificate	Version	2 (X.509v3)
	SerialNumber	Integer
	Signature	ecdsa-with-SHA256
Issuer	Country	(x)
	Organization	x
	Organization Unit	(x)
	Common Name	x
Validity		40 years
Subject	Country	(x)
	Organization	x
	Organization Unit	(x)
	Common Name	x
	Domain Component	"V2G"
SubjectPublicKeyInfo	Public Key	x
	Cryptographic Algorithm	id-ecPublicKey
	Parameters	ECParameters (namedCurve secp256r1)
Extensions	AuthorityKeyIdentifier	(x) / nc
	SubjectKeyIdentifier	(x) / nc
	KeyUsage	c
	digitalSignature	-
	nonRepudiation (contentCommitment)	-
	keyEncipherment	-
	dataEncipherment	0
	keyAgreement	-
	keyCertSign	(x)
	cRLSign	(x)
	encipherOnly	-
	decipherOnly	-
	ExtendedKeyUsage	-
	CertificatePolicies	(x) / nc
	BasicConstraints	c
	CA	true
	PathLength	-
	CRLDistributionPoints	(x) / nc
	Authority Information Access (OCSP)	(x) / nc id-ad-ocsp / location of the OCSP responder
Signature Value	Cryptographic Algorithm	ecdsa-with-SHA256
		Octect-String

Table F.2 — Charge Point Operator Certificates

Cluster:		Charge Point Operator (CPO)			OCSP
Name:		CPO Sub 1	CPO Sub 2	SECC Cert	OCSP Cert
Typ:		Sub	Sub	Leaf	Leaf
tbsCertificate	Version	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)
	SerialNumber	Integer	Integer	Integer	Integer
	Signature	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Issuer	Country	(x)	(x)	(x)	(x)
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	x	x
Validity		4 years	1-2 years	2-3 month	up to 1 year
Subject	Country	(x)	(x)	x	-
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	CPID	x
	Domain Component	(x)	(x)	"CPO"	(x)
SubjectPublic KeyInfo	Public Key	x	x	x	x
	Cryptographic Algorithm	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey
	Parameters	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)
Extensions	AuthorityKeyIdentifier	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	SubjectKeyIdentifier	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	KeyUsage	c	c	c	c
	digitalSignature	-	-	x	-
	nonRepudiation (contentCommitment)	-	-	-	-
	keyEncipherment	-	-	-	-
	dataEncipherment	0	0	0	0
	keyAgreement	-	-	-	-
	keyCertSign	(x)	(x)	-	-
	cRLSign	(x)	(x)	-	-
	encipherOnly	-	-	-	-
	decipherOnly	-	-	-	-
	ExtendedKeyUsage	-	-	-	1.3.6.1.5.5.7.3.9 - OCSPSigning
	CertificatePolicies	-	-	-	-
	BasicConstraints	c	c	c	c
	CA	true	true	false	false
	PathLength	1	0	-	-
	CRLDistributionPoints	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	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
	Cryptographic Algorithm	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Signature Value		Octect-String	Octect-String	Octect-String	Octect-String

Table F.3 — Certificate Installation Service Certificates (Provisioning)

Cluster:		Certificate Installation Service (Prov)		
Name:		Prov Sub 1	Prov Sub 2	Prov Service
Typ:		Sub	Sub	Leaf
tbsCertificate	Version	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)
	SerialNumber	Integer	Integer	Integer
	Signature	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Issuer	Country	(x)	(x)	(x)
	Organization	x	x	x
	Organization Unit	(x)	(x)	(x)
	Common Name	x	x	x
Validity		4 years	1-2 years	2-3 months
Subject	Country	(x)	(x)	x
	Organization	x	x	x
	Organization Unit	(x)	(x)	(x)
	Common Name	x	x	x
	Domain Component	(x)	(x)	"CPS"
SubjectPublicKeyInfo	Public Key	x	x	x
	Cryptographic Algorithm	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey
	Parameters	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)
Extensions	AuthorityKeyIdentifier	(x) / nc	(x) / nc	(x) / nc
	SubjectKeyIdentifier	(x) / nc	(x) / nc	(x) / nc
	KeyUsage	c	c	c
	digitalSignature	-	-	x
	nonRepudiation (contentCommitment)	-	-	-
	keyEncipherment	-	-	-
	dataEncipherment	0	0	0
	keyAgreement	-	-	-
	keyCertSign	(x)	(x)	-
	cRLSign	(x)	(x)	-
	encipherOnly	-	-	-
	decipherOnly	-	-	-
	ExtendedKeyUsage	-	-	-
	CertificatePolicies	-	-	-
	BasicConstraints	c	c	c
	CA	true	true	false
	PathLength	1	0	-
	CRLDistributionPoints	(x) / nc	(x) / nc	(x) / nc
	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
	Cryptographic Algorithm	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Signature Value		Octect-String	Octect-String	Octect-String

Table F.4 — Mobility Operator Certificates

Cluster:		Mobility Operator			
Name:		MO Root CA	MO Sub 1	MO Sub 2	Contract Cert
Typ:		Root	Sub	Sub/Leaf	Leaf
tbsCertificate	Version	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)
	SerialNumber	Integer	Integer	Integer	Integer
	Signature	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Issuer	Country	(x)	(x)	(x)	(x)
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	x	x
Validity		[up to MO]	[up to MO]	[up to MO]	4 weeks - 2 years
Subject	Country	(x)	(x)	(x)	-
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	x	EMAID
	Domain Component	(x)	(x)	(x)	(x)
SubjectPublic KeyInfo	Public Key	x	x	x	x
	Cryptographic Algorithm	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey
	Parameters	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)
Extensions	AuthorityKeyIdentifier	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	SubjectKeyIdentifier	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	KeyUsage	c	c	c	c
	digitalSignature	-	-	x	x
	nonRepudiation (contentCommitment)	-	-	x	x
	keyEncipherment	-	-	-	x
	dataEncipherment	0	0	0	0
	keyAgreement	-	-	-	x
	keyCertSign	(x)	(x)	(x)	-
	cRLSign	(x)	(x)	(x)	-
	encipherOnly	-	-	-	-
	decipherOnly	-	-	-	-
	ExtendedKeyUsage	-	-	-	-
	CertificatePolicies	(x) / nc	-	-	-
	BasicConstraints	c	c	c	c
	CA	true	true	true	false
	PathLength	-	1	0	-
	CRLDistributionPoints	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	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
	Cryptographic Algorithm	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Signature Value		Octect-String	Octect-String	Octect-String	Octect-String

NOTE 1 The Mobility Operator Sub-CA 2 has two purposes: First, it certifies Contract Certificates. Second, it signs tariff information destined for the owners of those Contract Certificates.

NOTE 2 The ID of the Provisioning Certificate (CertID) is used by the Mobility Operator to identify the contract that belongs to this EV. This is possible because the customer has given the CertID (CN and O, see below) to the Mobility Operator when creating the contract. For this purpose, CertID should be a short string that is unique to the OEM that created the Provisioning Certificate. Furthermore it is contained in the Provisioning Certificate.

[V2G2-932] CertID shall be an alphanumeric String with at maximum length of 18 characters (i.e. A..Z, a..z, 0..9).

[V2G2-933] CertID shall be contained in the subject field of the Provisioning Certificate as follows:

- The CertID itself (see [V2G2-932]) shall be the value of the Common Name (CN) of the Distinguished Name (DN) .

- The name of the OEM shall be encoded in the field Organization (O) using a unique identifier chosen by the OEM, to identify this OEM. Together with CN it builds a unique name.

- The X.500 name in the subject field shall not contain any further values.

Table F.5 — OEM Provisioning Certificates

Cluster:		OEM			
Name:		OEM Root CA	OEM SUB 1	OEM SUB 2	OEM Prov. Cert
Typ:		Root	Sub	Sub	Leafs
tbsCertificate	Version	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)
	SerialNumber	Integer	Integer	Integer	Integer
	Signature	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Issuer	Country	(x)	(x)	(x)	(x)
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	x	x
Validity		[up to OEM]	[up to OEM]	[up to OEM]	[up to OEM]
Subject	Country	(x)	(x)	(x)	-
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	x	PCID
	Domain Component	(x)	(x)	(x)	"OEM"
SubjectPublic KeyInfo	Public Key	x	x	x	x
	Cryptographic Algorithm	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey
	Parameters	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)
Extensions	AuthorityKeyIdentifier	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	SubjectKeyIdentifier	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	KeyUsage	c	c	c	c
	digitalSignature	-	-	-	x
	nonRepudiation (contentCommitment)	-	-	-	-
	keyEncipherment	-	-	-	-
	dataEncipherment	0	0	0	0
	keyAgreement	-	-	-	x
	keyCertSign	(x)	(x)	(x)	-
	cRLSign	(x)	(x)	(x)	-
	encipherOnly	-	-	-	-
	decipherOnly	-	-	-	-
	ExtendedKeyUsage	-	-	-	-
	CertificatePolicies	(x) / nc	-	-	-
	BasicConstraints	c	c	c	c
	CA	true	true	true	false
	PathLength	-	1	0	-
	CRLDistributionPoints	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	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
	Cryptographic Algorithm	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Signature Value		Octect-String	Octect-String	Octect-String	Octect-String

Table F.6 — Certificates for a Private Environment

Cluster:		Wallbox / Private Environment	
Name:		PE Root	Priv. Env. SECC Cert.
Typ:		Root	Leaf
tbsCertificate	Version	2 (X.509v3)	2 (X.509v3)
	SerialNumber	Integer	Integer
	Signature	ecdsa-with-SHA256	ecdsa-with-SHA256
Issuer	Country	(x)	(x)
	Organization	x	x
	Organization Unit	(x)	(x)
	Common Name	x	x
Validity		40 years	2 - 20 years
Subject	Country	(x)	x
	Organization	x	x
	Organization Unit	(x)	(x)
	Common Name	x	PEID
	Domain Component	(x)	(x)
SubjectPublic KeyInfo	Public Key	x	x
	Cryptographic Algorithm	id-ecPublicKey	id-ecPublicKey
	Parameters	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)
Extensions	AuthorityKeyIdentifier	(x) / nc	(x) / nc
	SubjectKeyIdentifier	(x) / nc	(x) / nc
	KeyUsage	c	c
	digitalSignature	-	x
	nonRepudiation (contentCommitment)	-	-
	keyEncipherment	-	-
	dataEncipherment	0	0
	keyAgreement	-	-
	keyCertSign	(x)	-
	cRLSign	(x)	-
	encipherOnly	-	-
	decipherOnly	-	-
	ExtendedKeyUsage	-	-
	CertificatePolicies	-	-
	BasicConstraints	c	c
	CA	true	false
	PathLength	0	-
	CRLDistributionPoints	(x) / nc	(x) / nc
	Authority Information Access (OCSP)	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder
	Cryptographic Algorithm	ecdsa-with-SHA256	ecdsa-with-SHA256
Signature Value		Octect-String	Octect-String

Annex G
(informative)

Encryption for the Distribution of Secret Keys

G.1 Overview

In the ISO 15118, Contract Certificates are created and then distributed from an SA to the EVCC. In order to minimize roundtrips and communication processing time, and ease organizational processes, it was decided, that the SA (and not the EVCC) generates the key pair for the certificate, then the corresponding certificate itself, and finally distributes the certificate and the private key to the EVCC. Since the private key must be kept confidential, appropriate mechanisms must be used.

To this end, the ISO 15118 uses an ephemeral-static Diffie-Hellman key agreement protocol to derive a session key where the receiver's keys are static. The SA then uses this session key to transmit the private key in encrypted form to the EVCC.

Since ECDSA, a variant of the DSA algorithm based on elliptic curves, is already used in the ISO 15118, the same elliptic curve as for ECDSA is also used for the ephemeral-static Diffie-Hellman protocol. Furthermore, the key pair used by EVCC for ECDSA is also used as the static key pair for the key agreement protocol.

G.2 Ephemeral-static Diffie-Hellman key agreement

The Diffie-Hellman key agreement protocol describes how a shared secret can be established between two entities using asymmetric cryptographic algorithms over an insecure channel. Both parties can then calculate the shared secret from which a session key is derived by a key derivation function. In order to meet the aforementioned goals regarding roundtrips and communication processing time, the ephemeral-static Diffie-Hellman protocol was chosen.

In the ephemeral-static variant of the Diffie-Hellman protocol, the public key of the receiver (EVCC) does not change, i.e. is static, and is already known to the sender (SA). The sender, however, still uses an ephemeral public key that is transmitted to the receiver. In this manner, both sender and receiver can derive the same session key without a reply message from the receiver. Only one communication message is required (see Figure G.1).

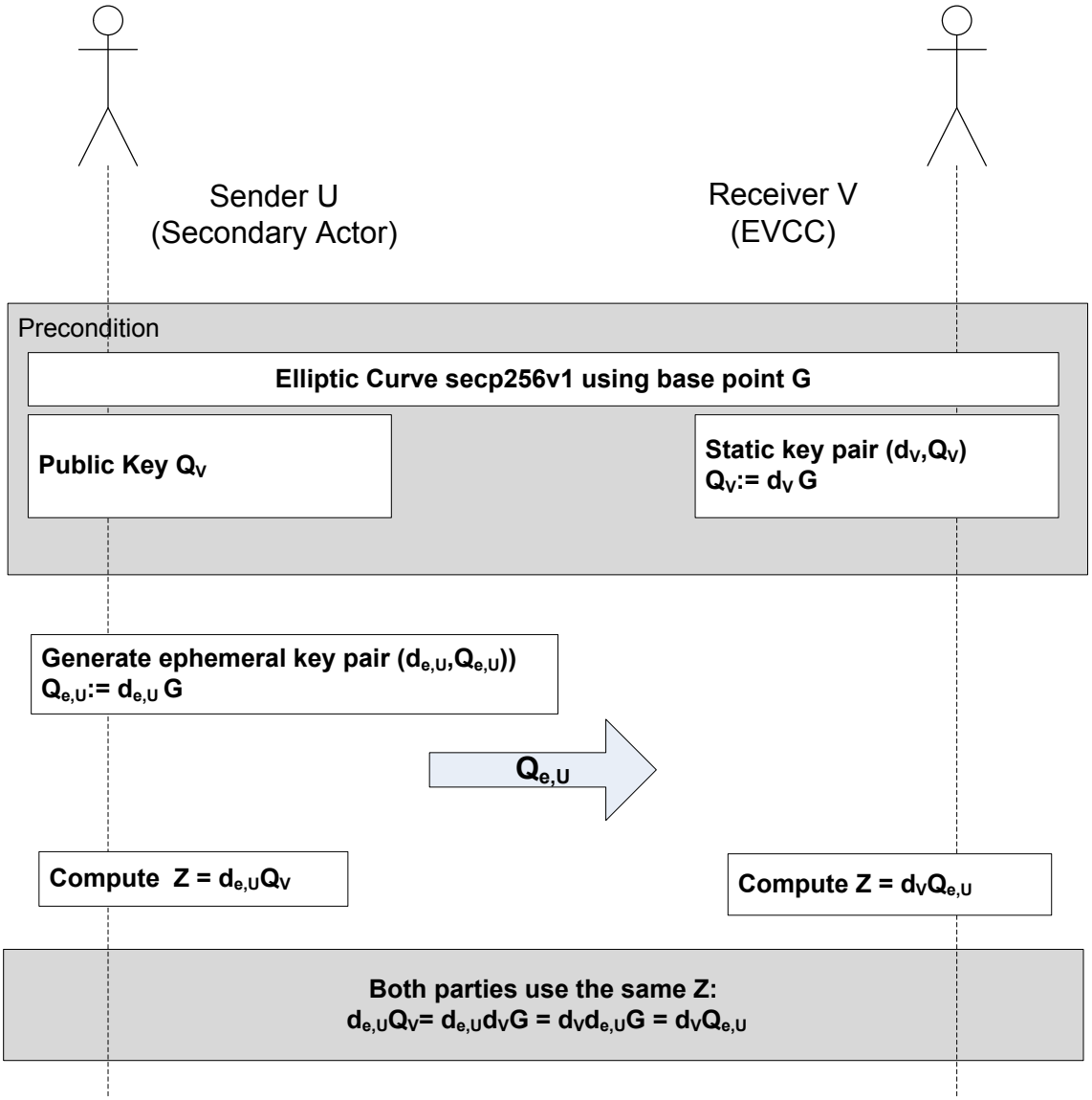


Figure G.1 — Ephemeral-static Diffie-Hellman key agreement (informational depiction, do not use for implementation)

G.3 Key pairs

The EVCC reuses the key pair that it also uses for ECDSA signatures, i.e. the key pair whose public key is stored in its current Contract Certificate respectively OEM Provisioning Certificate (depending on the message type). In both cases, the certificate and thus its public key is already known to the SA. The SA can therefore pre-compute new certificates and private keys for the EVCC and encrypt these with a session key without any message from the EVCC. The next time the EVCC connects to the SA (via the SECC), the pre-computed encrypted message can be retrieved and decrypted after only a single message from the SA, without any cryptographic processing on the side of the SA, and without any further message from the EVCC.

The SA uses ephemeral keys and hence the session keys will be different for each certificate and private key distribution. Note however that the ephemeral-static Diffie-Hellman protocol does not provide perfect forward secrecy, i.e. if the static private key of the EVCC is leaked to an adversary, then this adversary can decrypt any past and future communication between SA and EVCC, and also generate signatures, as the key for signatures is the same as the encryption key.

Annex H
(normative)

Specification of Identifiers

H.1 e-Mobility Account Identifier (EMAID)

H.1.1 EMAID Syntax

NOTE The e-Mobility Account Identifier (EMAID) is the corresponding data field to Contract ID, which is defined within ISO 15118-1. It may be renamed to Contract ID in future versions of this standard, because it is not describing an account ID but a Contract ID. The data part "eMA Instance" may also be renamed to Contract Number within future versions of this standard.

The EMAID shall match the following structure (the notation corresponds to the augmented Backus-Naur Form (ABNF) as defined in RFC 5234):

```
<EMAID> = <Country Code> <S> <Provider ID> <S> <eMA Instance> <S> <Check Digit>

<Country Code> = 2 ALPHA

   ; two character country code according to ISO 3166-1 (Alpha-2-Code)

<Provider ID> = 3 (ALPHA / DIGIT)

   ; three alphanumeric characters, defined and listed by eMI3 group

<eMA Instance> = 9 (ALPHA / DIGIT)

   ; nine alphanumeric characters

<Check Digit> = *1 (ALPHA / DIGIT)

   ; Optional but highly recommended, see subclause H.1.3 for its computation

ALPHA = %x41-5A / %x61-7A

   ; according to IETF RFC 5234(7-Bit ASCII)

DIGIT = %x30-39

   ; according to IETF RFC 5234 (7-Bit ASCII)

<S> = *1 ( "-" )

   ; optional separator
```

An example for a valid EMAID therefore is “DE8AA1A2B3C4D59” or with dashes “DE-8AA-1A2B3C4D5-9”.

H.1.2 EMAID Semantics

The <EMAID> shall be interpreted case insensitive, i.e. “DE8AA1A2B3C4D59” is exactly the same ID as “de8aA1A2b3C4d59”. A hyphen (“-“) can be used between the elements <Country Code>, <Provider ID>, <eMA Instance> and <Check Digit> in communication with users of EV or EVSE to allow for better reading,

spelling and typing. An example for such an illustration is “DE-8AA-1A2B3C4D5-9”. If such an illustration is chosen, the separators shall be set at all three places. Each <EMAID> has a length of at least fourteen and at most fifteen characters excluding the optional stars or seventeen respectively eighteen characters including the optional separators. While the <Provider ID> shall be assigned by a central issuing authority (e.g. eMI3 group), each provider with an assigned <Provider ID> can chose the <eMA Instance> within the above mentioned rules freely.

H.1.3 Calculation of the check digit

An alphanumeric character is calculated with the first 14 character of the <EMAID> and attached at the 15th position of the <EMAID>.

H.2 Electric Vehicle Supply Equipment ID (EVSEID)

H.2.1 EVSEID Syntax

The EVSEID shall match the following structure (the notation corresponds to the augmented Backus-Naur Form (ABNF) as defined in IETF RFC 5234):

```
<EVSEID> = <Country Code> <S> <EVSE Operator ID> <S> <ID Type> <Power Outlet ID>

<Country Code> = 2 ALPHA

   ; two character country code according to The following documents, in whole
or in part, are normatively referenced in this document and are indispensable for
its application. For dated references, only the edition cited applies. For
undated references, the latest edition of the referenced document (including any
amendments) applies.

ISO 3166-1 (Alpha-2-Code)

<EVSE Operator ID> = 3 (ALPHA / DIGIT)

   ; three alphanumeric characters, defined and listed by eMI3 group

<ID Type> = “E”

   ; one character “E” indicating that this ID represents an “EVSE”

<Power Outlet ID> = (ALPHA / DIGIT) *30 (ALPHA / DIGIT / <S>)

   ; sequence of alphanumeric characters or separators, start with alphanumeric
character

ALPHA = %x41-5A / %x61-7A

   ; according to IETF RFC 5234 (7-Bit ASCII)

DIGIT = %x30-39

   ; according to IETF RFC 5234 (7-Bit ASCII)

<S> = *1 ( “*” )
```

```
; optional separator
```

An example for a valid EVSEID is “FR*A23*E45B*78C” with “FR” indicating France, “A23” representing a particular EVSE Operator, “E” indicating that it is of type “EVSE” and “45B*78C” representing one of its power outlets.

NOTE In contrast to the EMAID, no check digit is specified for the EVSEID in this document.

H.2.2 EVSEID Semantics

Each <EVSEID> has a variable length with at least seven characters (two characters <Country Code>, three characters <EVSE Operator ID>, one character <ID Type>, one character <Power Outlet ID>) and at most thirty-seven characters (two characters <Country Code>, three characters <EVSE Operator ID>, one character <ID Type>, thirty-one characters <Power Outlet ID>). While the <EVSE Operator ID> shall be assigned by a central issuing authority, each operator with an assigned <EVSE Operator ID> can chose the <Power Outlet ID> within the above mentioned rules freely.

NOTE In future versions of this standard EVSEID may be interpreted case insensitive to avoid misspellings.

Annex I
(informative)

Message sequencing for renegotiation

I.1 Overview

This chapter gives message sequence examples to make the comprehension of the renegotiation mechanism easier.

In ISO 15118-2, the renegotiation mechanism is used to change charge target settings and scheduling during a charging loop. Renegotiation can be initiated either by the EVCC or SECC.

To initiate a renegotiation by the SECC, the SECC has to set EVSENotification in EVSEStatus to 'ReNegotiation'. If so, the EVCC shall initiate a renegotiation in a limited time (see also **[V2G2-680]**).

To initiate a renegotiation by the EVCC, the EVCC can decide to perform a renegotiation by sending PowerDeliveryReq with parameter ChargeProgress set to ‘Renegotiate’ (see also **[V2G2-522]**, **[V2G2-686]** and **[V2G2-683]**. The EVCC may renegotiate the charging schedule as follows:

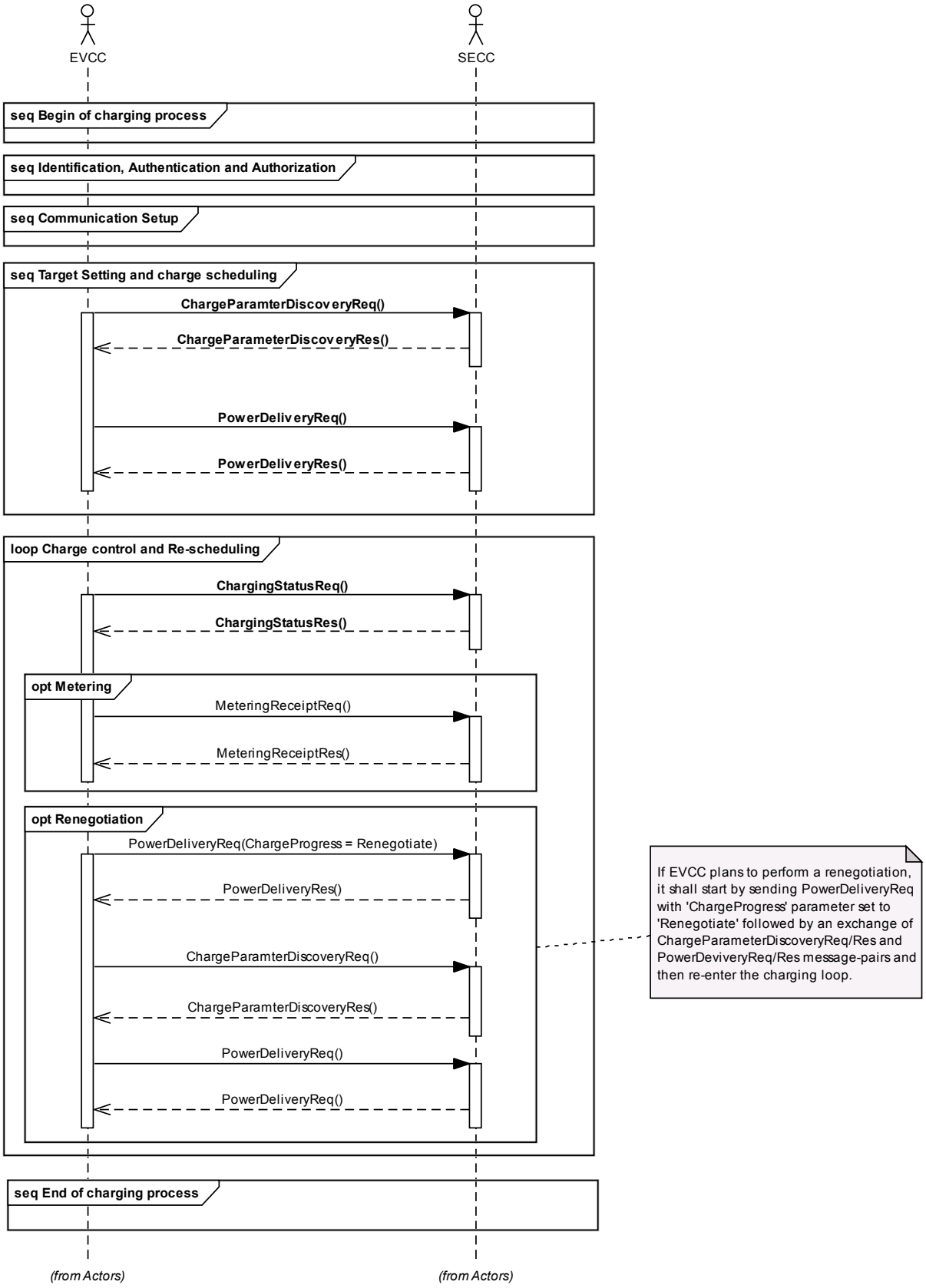


Figure I.1 —Sequence diagram for EVCC initiated renegotiation

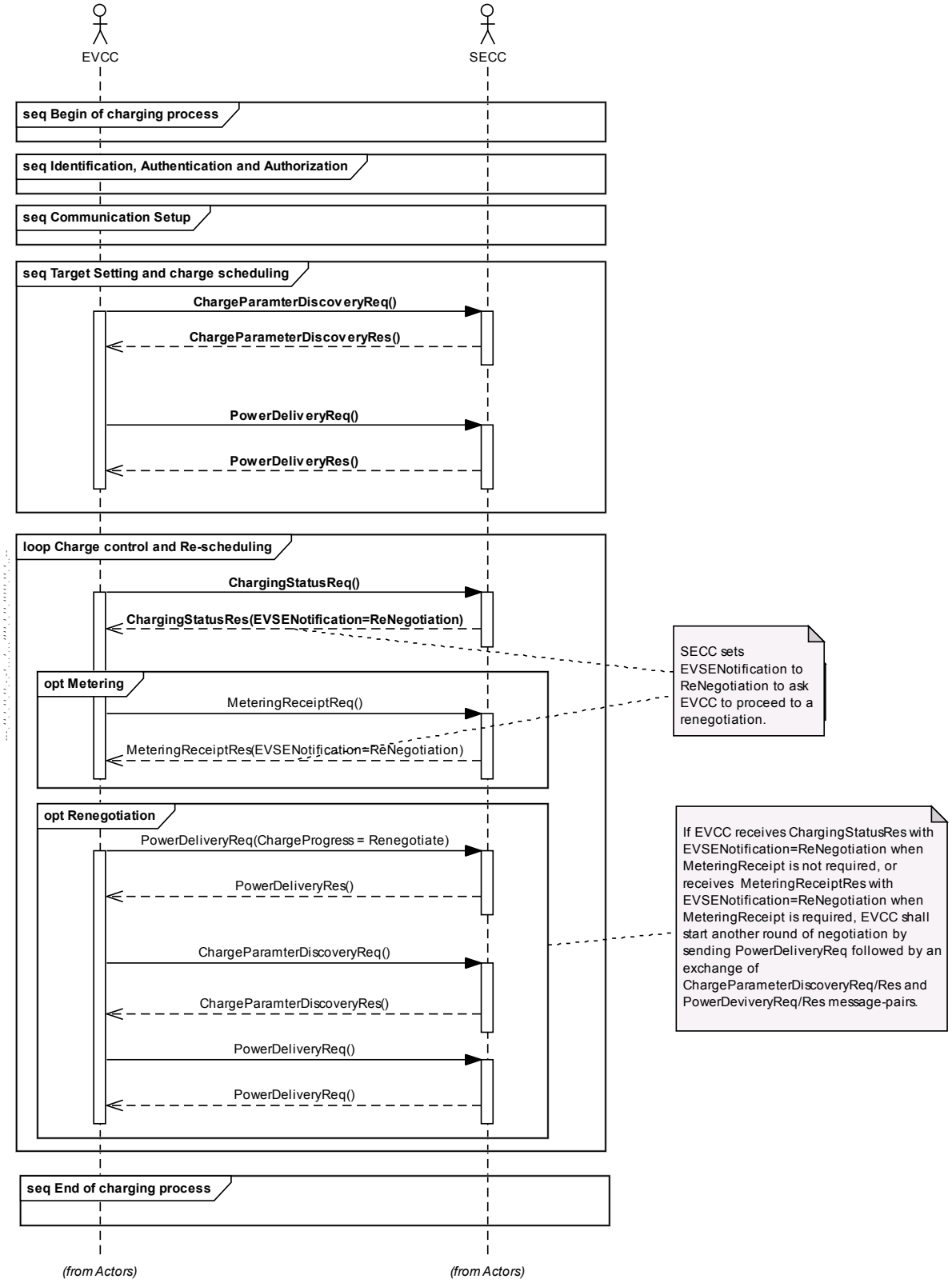


Figure I.2 —Sequence diagram for SECC initiated renegotiation

I.2 Renegotiation after resuming a V2G Communication Session

The renegotiation process is only allowed after a first successful negotiation.

If one of EVCC or SECC wants to perform a renegotiation after resuming a V2G Communication Session, they first have to go through the V2G messaging until PowerDeliveryReq/Res message pair and apply the corresponding renegotiation mechanism as described in this standard and in the two examples provided above.

Annex J
(informative)

Overview on XML Signatures

J.1 Overview

In the ISO 15118, XML signatures are used to sign e.g. sales tariffs, meter readings or complete message bodies. This chapter gives and short introduction to XML signatures. A complete example using valid data, keys and signature values is given to allow a complete understanding, how XML signatures are used for ISO 15118.

J.2 Signature generation

Signing data using XML signatures requires two main steps: reference generation and signature generation. Reference generation includes all steps that are required to add some XML data that should be signed to the SignedInfo XML element. Signature generation then calculates the signature value.

All EXI, hash and signature values given in this example are valid and can be checked using the following key pair:

— Public Key (raw x,y data):

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	43	E4	FC	4C	CB	64	39	04	27	9C	7A	5E	65	76	B3	23
00000010	E5	5E	C7	9F	F0	E5	A4	05	6E	33	40	84	CB	C3	36	FF
00000020	46	E4	4C	1A	DD	F6	91	62	E5	19	2C	2A	83	FC	2B	CA
00000030	9D	8F	46	EC	F4	B7	80	67	C2	47	6F	6B	3F	34	60	0E

— Private Key (raw x data):

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	B9	13	49	63	F5	1C	44	14	73	84	35	05	7F	97	BB	F1
00000010	01	0C	AB	CB	8D	BD	E9	C5	D4	81	38	39	6A	A9	4E	9D

The following example shows the complete workflow that is required to generate a valid signature for a AuthorizationReq message with the following content:

```
<v2gci_b:AuthorizationReq  v2gci_b:Id="ID1">
      <v2gci_b:GenChallenge>U29tZSBSYW5kb20gRGF0YQ==</v2gci_b:GenChallenge>
</v2gci_b:AuthorizationReq>
```

The following XML fragment can be used as a basic template for ISO 15118 when generating XML Signatures. Required algorithm identifiers and the minimum required elements are already added. A valid signature only requires extension of the <Reference>-Element and the concrete digest and signature values.

```
<xmlsig:Signature>
      <xmlsig:SignedInfo>
            <xmlsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/canonical-exi/"/>
            <xmlsig:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256"/>
            <xmlsig:Reference URI="#ID1">
                  <xmlsig:Transforms>
                        <xmlsig:Transform Algorithm="http://www.w3.org/TR/canonical-exi/"/>
                  </xmlsig:Transforms>
```

```
                <xmlsig:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                <xmlsig:DigestValue></xmlsig:DigestValue>
            </xmlsig:Reference>
        </xmlsig:SignedInfo>
        <xmlsig:SignatureValue></xmlsig:SignatureValue>
</xmlsig:Signature>
```

The first step generating the signature is to add the Reference XML element to the data that should be signed. For this, the URI attribute is set to the ID of the element and the data is EXI transformed (Reference transformation) using the EXI schema-informed fragment grammar based on the V2G_CI_MsgDef schema.

The resulting EXI stream for the given AuthorizationReq is

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000   80 04 01 52 51 0C 40 82 9B 7B 6B 29 02 93 0B 73
00000010   23 7B 69 02 23 0B A3 09 E8
```

The next step is to hash the EXI stream using SHA256. This hash value is then base64 encoded and added together with the URI to the SignedInfos Reference element as DigestValue (marked red).

```
<xmlsig:Signature>
    <xmlsig:SignedInfo>
        <xmlsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/canonical-exi/"/>
        <xmlsig:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256"/>
        <xmlsig:Reference URI="#ID1">
            <xmlsig:Transforms>
                <xmlsig:Transform Algorithm="http://www.w3.org/TR/canonical-exi/"/>
            </xmlsig:Transforms>
            <xmlsig:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
            <xmlsig:DigestValue>0bXgPQBlvuVrMXmERTBR61TKGPwOCRYXT4s8d6mPSqk=</xmlsig:DigestValue>
        </xmlsig:Reference>
    </xmlsig:SignedInfo>
    <xmlsig:SignatureValue></xmlsig:SignatureValue>
</xmlsig:Signature>
```

NOTE 1 If required, this step is repeated for all references that should be added to the signature (e.g. for multiple Sales Tariffs).

NOTE 2 In contrast to textual XML in the presentation of this example, EXI encodes binary data natively, so the base64 encoding step will not be applied there.

After reference generation the next step is the signature generation. For this step the original data is not needed anymore. XML signatures only signs the SignedInfo element. Therefore this element needs to be EXI encoded using the schema-informed fragment grammar based on the XMLdsig schema (Canonicalization). The resulting EXI stream is the following:

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	80	81	12	B4	3A	3A	38	1D	17	97	BB	BB	BB	97	3B	99
00000010	97	37	B9	33	97	AA	29	17	B1	B0	B7	37	B7	34	B1	B0
00000020	B6	16	B2	BC	34	97	A1	AB	43	A3	A3	81	D1	79	7B	BB
00000030	BB	B9	73	B9	99	73	7B	93	39	79	91	81	81	89	79	81
00000040	A1	7B	C3	6B	63	23	9B	4B	39	6B	6B	7B	93	29	1B	2B
00000050	1B	23	9B	09	6B	9B	43	09	91	A9	B2	20	62	34	94	43
00000060	10	25	68	74	74	70	3A	2F	2F	77	77	77	2E	77	33	2E
00000070	6F	72	67	2F	54	52	2F	63	61	6E	6F	6E	69	63	61	6C
00000080	2D	65	78	69	2F	48	52	D0	E8	E8	E0	74	5E	5E	EE	EE
00000090	EE	5C	EE	66	5C	DE	E4	CE	5E	64	60	60	62	5E	60	68
000000A0	5E	F0	DA	D8	CA	DC	C6	46	E6	D0	C2	64	6A	6C	84	1A
000000B0	36	BC	07	A0	0C	B7	DC	AD	66	2F	30	88	A6	0A	3D	6A
000000C0	99	43	1F	81	C1	22	C2	E9	F1	67	8E	F5	31	E9	55	23
000000D0	70															

This stream is then hashed using SHA256. The resulting hash value is:

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	A4	E9	03	E1	82	43	04	1B	55	4E	11	64	7E	10	1E	D2
00000010	5F	C9	F2	15	2A	F4	67	40	14	FE	2A	DE	AC	1E	1C	F7

This hash value is used as input for the signature algorithm (ECDSA with secp256r1 curve). Signature calculation is done using the private key. The resulting signature value is then base64 encoded and added as SignatureValue to the Signature XML data (marked red):

```
<xmlsig:Signature>
    <xmlsig:SignedInfo>
        <xmlsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/canonical-exi/"/>
        <xmlsig:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256"/>
        <xmlsig:Reference URI="#ID1">
            <xmlsig:Transforms>
                <xmlsig:Transform Algorithm="http://www.w3.org/TR/canonical-exi/"/>
            </xmlsig:Transforms>
            <xmlsig:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
            <xmlsig:DigestValue>0bXgPQBlvuVrMXmERTBR61TKGPwOCRYXT4s8d6mPSqk=</xmlsig:DigestValue>
        </xmlsig:Reference>
    </xmlsig:SignedInfo>
    <xmlsig:SignatureValue>TI8gwUALpnYGqkgRVyovGtPBUInZVCA2NDC7JrSdsQTwjfqL+AVeY6S3Wo
                           0xaSBvqNVDCLpY8FZrlrr2ks5ZUA==</xmlsig:SignatureValue>
</xmlsig:Signature>
```

Putting all together results in the following V2G message that will be transmitted from the EVCC to the SECC:

```
<?xml version="1.0" encoding="UTF-8"?>
<v2gci_d:V2G_Message xmlns:v2gci_h="urn:iso:15118:2:2013:MsgHeader"
xmlns:v2gci_d="urn:iso:15118:2:2013:MsgDef"
xmlns:v2gci_t="urn:iso:15118:2:2013:MsgDataTypes"
xmlns:xmlsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:v2gci_b="urn:iso:15118:2:2013:MsgBody"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <v2gci_d:Header>
        <v2gci_h:SessionID>9988AABBCCDDEEFF</v2gci_h:SessionID>
        <xmlsig:Signature>
            <xmlsig:SignedInfo>
                <xmlsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/canonical-exi/"/>
                <xmlsig:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256"/>
                <xmlsig:Reference URI="#ID1">
```

```
                        <xmlsig:Transforms>
                            <xmlsig:Transform Algorithm="http://www.w3.org/TR/canonical-exi/"/>
                        </xmlsig:Transforms>
                        <xmlsig:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                        <xmlsig:DigestValue>0bXgPQBlvuVrMXmERTBR61TKGPwO
                                            CRYXT4s8d6mPSqk=</xmlsig:DigestValue>
                    </xmlsig:Reference>
                </xmlsig:SignedInfo>
                <xmlsig:SignatureValue>TI8gwUALpnYGqkgRVyovGtPBUInZVCA2NDC7JrSdsQTwjfqL+AVeY6S3
                                       Wo0xaSBvqNVDCLpY8FZrlrr2ks5ZUA==</xmlsig:SignatureValue>
            </xmlsig:Signature>
        </v2gci_d:Header>
        <v2gci_d:Body>
            <v2gci_b:AuthorizationReq v2gci_b:Id="ID1">
                <v2gci_b:GenChallenge>U29tZSBSYW5kb20gRGF0YQ==</v2gci_b:GenChallenge>
            </v2gci_b:AuthorizationReq>
        </v2gci_d:Body>
</v2gci_d:V2G_Message>
```

V2G message example 24 – Signature Generation for AuthorizationReq message

J.3 Signature generation for secondary actors

For sales tariffs the signature generation needs to be done by the secondary actor. Therefore some information about the charging session needs to be transmitted from the SECC to the SA before generating the final ChargeParameterDiscoveryRes message. This information includes necessarily the used schema version and optionally eMAID, required amount of energy, departure time, list of EVs root certificates and - if needed - some more EVSE data that can be used for sales tariff generation. The SA has to get the XSD files for the used schema version. During signature generation these files can be used together with a generic EXI generator, e.g. EXIficient, to create the EXI fragment grammar streams during <Reference> transformation and <SignedInfo> canonicalization.

The sales tariffs are created as XML data that is valid according to the used schema version. The detached XML signature is generated by the SA (see [V2G2-307]) as described in the workflow above. The used certificate shall be derived from an EV root certificate and this key information may be added to the signature as well.

The XML data (sales tariffs and signature) is transmitted in an arbitrary way to the SECC. The ChargeParameterDiscoveryRes that includes the received sales tariffs and the corresponding signature is then generated by the SECC, EXI encoded and transmitted to the EVCC (see [V2G2-308] and [V2G2-309]).

NOTE One signature value is used to sign all sales tariffs. In this scenario multiple <Reference> elements will be included in the <SignedInfo> element.

J.4 Signature validation

Validation of XML signatures requires also two main steps, reference validation and signature validation. During reference validation first the canonicalization of the SignedInfo element is calculated. This means that the SignedInfo element is EXI encoded using the schema-informed fragment grammar based on the XMLdsig schema.

Then the DigestValue element for each Reference element is checked. This requires URI de-referencing as the first step. Here the original signed data needs to be returned and transformed using the EXI schema-informed fragment grammar based on the V2G_CI_MsgDef schema. The generated EXI stream is than hashed using SHA256 and compared against the received DigestValue element.

If reference validation passes for all references the signature value is validated. The canonicalization/EXI stream that represents the SignedInfo element is hashed using SHA256 and based on this value the received SignatureValue is validated using the public key.

Annex K
(informative)

Summary of requirements

This Annex lists in Table K.1 the requirements defined in this standard to allow for easy requirement management. It is also used for maintaining a change history concerning 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][V2G2-885][V2G2-926][V2G2-925][V2G2-886][V2G2-887][V2G2-910]
7.3.3 Number of root certificates and root validity, certificate depth and size	[V2G2-008][V2G2-009][V2G2-010][V2G2-011][V2G2-012] [V2G2-877][V2G2-878][V2G2-867][V2G2-869][V2G2-882][V2G2-927][V2G2-883][V2G2-868][V2G2-911]
7.3.4 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][V2G2-643][V2G2-644]
7.4 V2G communication states and data link handling	[V2G2-014][V2G2-016][V2G2-017][V2G2-018][V2G2-019][V2G2-020][V2G2-021][V2G2-022][V2G2-023][V2G2-024][V2G2-025][V2G2-026][V2G2-027][V2G2-029][V2G2-030][V2G2-031][V2G2-032][V2G2-033][V2G2-034][V2G2-645][V2G2-646][V2G2-717][V2G2-718][V2G2-719][V2G2-720][V2G2-721][V2G2-722][V2G2-723][V2G2-724][V2G2-725][V2G2-726][V2G2-727]
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][V2G2-049]
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] [V2G2-070] [V2G2-649][V2G2-650][V2G2-651][V2G2-870][V2G2-876][V2G2-871][V2G2-872][V2G2-923][V2G2-924][V2G2-873][V2G2-874][V2G2-875][V2G2-810][V2G2-811]

Subclause including requirement(s)	List of requirement numbers
7.7.3.4 Transport Layer Security Credentials and Cipher Suites	[V2G2-071][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] [V2G2-092] [V2G2-094] [V2G2-096][V2G2-800]
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]
7.9.2.2 Application layer credentials and cipher suites	[V2G2-103][V2G2-104]
7.9.2.3 Contract Certificates as XML signature credentials	[V2G2-108]
7.9.2.4.2 XML Signature mechanism	[V2G2-117] [V2G2-119][V2G2-121] [V2G2-764][V2G2-765][V2G2-766][V2G2-767][V2G2-768][V2G2-769][V2G2-770][V2G2-771][V2G2-909]
7.9.2.4.3 Encryption mechanism	[V2G2-121][V2G2-122][V2G2-814][V2G2-815][V2G2-816][V2G2-817][V2G2-818][V2G2-819][V2G2-820][V2G2-821][V2G2-822][V2G2-823]
7.9.2.4.4 Random Number Generation	[V2G2-835]
7.9.2.4.5 Application of security mechanisms to XML message	[V2G2-652][V2G2-653]
7.10.1.1 General Information (SDP)	[V2G2-123]
7.10.1.2 Supported ports (SDP)	[V2G2-124][V2G2-125][V2G2-126]
7.10.1.3.1 Structure	[V2G2-127][V2G2-128][V2G2-129][V2G2-130][V2G2-131][V2G2-132]
7.10.1.3.2 Header Processing	[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]
7.10.1.5 SECC Discovery Response Message	[V2G2-143][V2G2-144][V2G2-145][V2G2-146][V2G2-147] [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][V2G2-163][V2G2-164]
8.1 General information and definitions	[V2G2-809]
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 description supportedAppProtocol messages	[V2G2-178]

Subclause including requirement(s)	List of requirement numbers
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]
8.4.2 Session Handling	[V2G2-739][V2G2-740][V2G2-741][V2G2-742][V2G2-743][V2G2-744][V2G2-745][V2G2-746][V2G2-747][V2G2-748][V2G2-749][V2G2-750][V2G2-751][V2G2-752][V2G2-753][V2G2-754][V2G2-755][V2G2-756]
8.4.3.2 SessionSetup	[V2G2-188][V2G2-189][V2G2-190][V2G2-191] [V2G2-192] [V2G2-879]
8.4.3.3 ServiceDiscovery	[V2G2-193][V2G2-194][V2G2-195][V2G2-196]
8.4.3.4 ServiceDetail	[V2G2-197][V2G2-198][V2G2-199][V2G2-200]
8.4.3.5 PaymentServiceSelection	[V2G2-201][V2G2-202][V2G2-203][V2G2-204]
8.4.3.6 PaymentDetails	[V2G2-205][V2G2-206][V2G2-208][V2G2-209] [V2G2-825][V2G2-826][V2G2-898][V2G2-899]
8.4.3.7 Authorization	[V2G2-210][V2G2-211][V2G2-212][V2G2-213] [V2G2-697][V2G2-698] [V2G2-900][V2G2-901]
8.4.3.8 ChargeParameterDiscovery	[V2G2-214][V2G2-215][V2G2-216] [V2G2-218][V2G2-219][V2G2-220][V2G2-761][V2G2-784][V2G2-785][V2G2-786]
8.4.3.9 PowerDelivery	[V2G2-221][V2G2-222][V2G2-223][V2G2-224][V2G2-225][V2G2-226][V2G2-777]
8.4.3.10 CertificateUpdate	[V2G2-227][V2G2-228][V2G2-229][V2G2-230][V2G2-231][V2G2-232][V2G2-928] [V2G2-233][V2G2-696][V2G2-888][V2G2-889][V2G2-827][V2G2-890][V2G2-891][V2G2-892]
8.4.3.11 CertificateInstallation	[V2G2-234][V2G2-235][V2G2-236][V2G2-237][V2G2-238][V2G2-648] [V2G2-893][V2G2-894] [V2G2-895][V2G2-896][V2G2-897]
8.4.3.12 SessionStop	[V2G2-239][V2G2-240][V2G2-241][V2G2-738]
8.4.3.13 MeteringReceiptReq/Res	[V2G2-245][V2G2-246][V2G2-247][V2G2-248][V2G2-902] [V2G2-903][V2G2-904] [V2G2-776]
8.4.4.2 ChargingStatus	[V2G2-242][V2G2-243][V2G2-244]
8.4.5.2 CableCheck	[V2G2-249][V2G2-250][V2G2-251][V2G2-252]
8.4.5.3 PreCharg	[V2G2-253][V2G2-254][V2G2-255][V2G2-256]
8.4.5.4 CurrentDemand	[V2G2-257][V2G2-258][V2G2-259][V2G2-260]
8.4.5.5 WeldingDetection	[V2G2-261][V2G2-262][V2G2-263][V2G2-264]
8.5.2.1 ServiceType	[V2G2-265] [V2G2-266]
8.5.2.2 ServiceListType	[V2G2-267][V2G2-268]

Subclause including requirement(s)	List of requirement numbers
8.5.2.3 ChargeServiceType	[V2G2-271][V2G2-272]
8.5.2.4 SupportedEnergyTransferModeType	[V2G2-757][V2G2-758] [V2G2-759]
8.5.2.5 CertificateChainType	[V2G2-274][V2G2-275]
8.5.2.6 MeterInfoType	[V2G2-276][V2G2-277] [V2G2-830][V2G2-831]
8.5.2.7 PhysicalValueType	[V2G2-278][V2G2-279] [V2G2-832]
8.5.2.8 NotificationType	[V2G2-280][V2G2-281]
8.5.2.9 PaymentOptionListType	[V2G2-282][V2G2-283]
8.5.2.10 ChargingProfileType	[V2G2-284] [V2G2-606]
8.5.2.11 ProfileEntryType	[V2G2-288] [V2G2-289][V2G2-290][V2G2-291][V2G2-292][V2G2-293] [V2G2-607][V2G2-829]
8.5.2.12 SAScheduleListType	[V2G2-294] [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] [V2G2-773][V2G2-905][V2G2-906][V2G2-907][V2G2-908]
8.5.2.14 PMaxScheduleType	[V2G2-310][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][V2G2-612][V2G2-805]
8.5.2.17 SalesTariffEntryType	[V2G2-321] [V2G2-322] [V2G2-324][V2G2-325][V2G2-613][V2G2-803][V2G2-802]
8.5.2.18 RelativeTimeIntervalType	[V2G2-327] [V2G2-328][V2G2-329][V2G2-330][V2G2-331][V2G2-614][V2G2-833][V2G2-834]
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][V2G2-616][V2G2-772][V2G2-775][V2G2-806][V2G2-807][V2G2-808]
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]

Subclause including requirement(s)	List of requirement numbers
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]
8.5.2.28 ContractSignatureEncryptedPrivateKeyType	[V2G2-778][V2G2-779]
8.5.2.29 DiffieHellmanPublickeyType	[V2G2-780][V2G2-781]
8.5.2.30 EMAIDType	[V2G2-782][V2G2-783]
8.5.3.1 AC_EVSEStatusType	[V2G2-358][V2G2-359]
8.5.3.2 AC_EVChargeParameterType	[V2G2-360][V2G2-361]
8.5.3.3 AC_EVSEChargeParameterType	[V2G2-362][V2G2-363] [V2G2-708][V2G2-709]
8.5.4.1 DC_EVSEStatusType	[V2G2-364][V2G2-365] [V2G2-366] [V2G2-801]
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]
8.6.1 Overview	[V2G2-760][V2G2-828]
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 Common	[V2G2-762][V2G2-763]
8.6.2.3 AC (Supported Message Set(s))	[V2G2-376][V2G2-377][V2G2-378][V2G2-379][V2G2-380][V2G2-381][V2G2-384][V2G2-385][V2G2-386][V2G2-387][V2G2-388][V2G2-389]
8.6.2.4 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] [V2G2-691] [V2G2-787][V2G2-788][V2G2-789]
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]

Subclause including requirement(s)	List of requirement numbers
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][V2G2-774]
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 Response-Request Message Sequence	[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 SECC Timing for communication session setup	[V2G2-714][V2G2-715][V2G2-716]
8.7.3.4 EVCC Timing for EVSEProcessing parameter	[V2G2-710][V2G2-711]
8.7.3.5 SECC Timing for EVSEProcessing parameter	[V2G2-712][V2G2-713]
8.7.3.6 EVCC Timing for cable check	[V2G2-700][V2G2-701][V2G2-702][V2G2-703]
8.7.3.7 EVCC Timing for pre charging	[V2G2-704][V2G2-705][V2G2-706][V2G2-707]
8.7.4.2 Common requirements (Message synchronization with IEC 61851-1 signalling)	[V2G2-836][V2G2-837][V2G2-838][V2G2-839][V2G2-840][V2G2-841][V2G2-842][V2G2-843][V2G2-844][V2G2-845][V2G2-850][V2G2-854][V2G2-855][V2G2-856] [V2G2-731][V2G2-737] [V2G2-733]
8.7.4.3 AC specific requirements	[V2G2-846][V2G2-847][V2G2-848][V2G2-849][V2G2-851][V2G2-852][V2G2-853][V2G2-857][V2G2-858][V2G2-859][V2G2-860][V2G2-861][V2G2-862][V2G2-863][V2G2-864][V2G2-865][V2G2-866][V2G2-930][V2G2-931]
8.7.4.4 DC specific requirements	[V2G2-912][V2G2-913][V2G2-914][V2G2-915][V2G2-916][V2G2-917][V2G2-918][V2G2-919][V2G2-920][V2G2-921][V2G2-922]
Error! Reference source not found. Common requirements (ResponseCode Handling – Message Sequencing)	[V2G2-457][V2G2-458][V2G2-459][V2G2-460][V2G2-461][V2G2-462][V2G2-463][V2G2-464][V2G2-465] [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][V2G2-690][V2G2-693][V2G2-694][V2G2-695][V2G2-734][V2G2-735][V2G2-736][V2G2-804][V2G2-824]
8.8.4.1 General (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)	[V2G2-482][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] [V2G2-510][V2G2-511][V2G2-512][V2G2-513][V2G2-514] [V2G2-516][V2G2-517][V2G2-518][V2G2-519][V2G2-520][V2G2-521][V2G2-522][V2G2-524][V2G2-525][V2G2-526][V2G2-527][V2G2-528] [V2G2-530][V2G2-531][V2G2-532][V2G2-533][V2G2-534][V2G2-535][V2G2-617][V2G2-618][V2G2-619][V2G2-620][V2G2-683][V2G2-684][V2G2-685][V2G2-686][V2G2-689][V2G2-728][V2G2-790][V2G2-791][V2G2-792][V2G2-793] [V2G2-799][V2G2-880][V2G2-599]
8.8.4.3 SECC (Message Sequencing)	[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] [V2G2-566][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] [V2G2-592][V2G2-593][V2G2-595][V2G2-596][V2G2-597][V2G2-598][V2G2-601][V2G2-621][V2G2-687][V2G2-688][V2G2-729][V2G2-812][V2G2-813][V2G2-881][V2G2-795][V2G2-796][V2G2-797][V2G2-798]
Annex F (normative) Certificate profiles	[V2G2-923][V2G2-933]

Bibliography

[1] IETF RFC 5871, IANA Allocation Guidelines for the IPv6 Routing Header (May 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 XML, Extensible Markup Language (XML) 1.0 (Fifth Edition) (November 2008)

[8] W3C XMLSchema 0, XML Schema Part 0: Primer Second Edition (October 2004)

[9] W3C XMLSchema 1, XML Schema Part 1: Structures Second Edition (October 2004)

[10] W3C XMLSchema 2, XML Schema Part 2: Datatypes Second Edition (October 2004)

[11] ANSI X9.62, Public Key Cryptography For The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA) (2005)

[12] ISO 15118-1, Road vehicles – Vehicle to grid communication interface – Part 1: General information and use-case definition

[13] NIST SP 800-90 A, Recommendation for Random Number Generation Using Deterministic Random Bit Generators (January 2012)

[14] BSI AIS 20/AIS 31: A proposal for: Functionality classes for random number generators (2011)

[15] W3C EXI Profile, *Efficient XML Interchange (EXI) Profile, W3C Candidate Recommendation* (July 2013)

[16] IETF RFC 1323, TCP Extensions for High Performance (May 1992)

[17] IETF RFC 1624, Computation of the Internet Checksum via Incremental Update (May 1994)

[18] IETF RFC 2018, TCP Selective Acknowledgment Options (October 1996)

[19] IETF RFC 5482, TCP User Timeout Option (March 2009)

[20] IETF RFC 5681, TCP Congestion Control (September 2009)

[21] IETF RFC 6298, Computing TCP's Retransmission Timer (June 2011)

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

[23] IETF RFC 1630, Universal Resource Identifiers in WWW (June 1994)

ICS 43.120

Price based on 342 pages