Open Charge Point Protocol

Interface description between Charge Point and Central System

Document Identifier	010.030.007
Document Version	1.2 ()
Document Status	FINAL
Document Release Date	21-2-2011

Version management by Logica.

Version History

Version	Date	Author	Description
1.0	19.10.2010	Franc Buve	Final version approved by e-laad.nl. Identical to version 0.12.
1.1	17.11.2010	Franc Buve	Removed change marks.
1.2	21.02.2011	Franc Buve	#302: Added lengths to string types. #171: typographic error fixed.

Contents

1 Scope

This document defines the protocol used between a VAS and CiMS. If the protocol requires a certain action or response from one side or the other, than this will be stated in this document.

The specification does not define the communication technology. Any technology will do, as long as it supports TCP/IP connectivity.

2 Terminology and Conventions

Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

Definitions

This section contains the terminology that this used throughout this document.

CiMS

Charge point interactive Management System: the central system that manages charge points and has the information for authorizing users for using its charge points.

VAS

The Value Added Services supplier, i.e. the external party using the VAS interface.

Charge Point

The Charge Point is the physical system where an electric vehicle can be charged.

Abbreviations

EV Electrical Vehicle

FTP(S) File Transport Protocol (Secure)

HTTP(S) HyperText Transport Protocol (Secure)

ICCID Integrated Circuit Card Identifier

IMSI International Mobile Subscription Identify

PDU Protocol Data Unit

SOAP Simple Object Access Protocol

SSL Secure Socket Layer

TLS Transport Layer Security

URL Uniform Resource Locator

WSDL Web Service Definition Language

References

- [RFC2119] "Key words for use in RFCs to Indicate Requirement Levels". S. Bradner. March 1997.http://www.ietf.org/rfc/rfc2119.txt
- [SOAP] "SOAP Version 1.2 Part 0: Primer (Second Edition)". 27 April 2007. http://www.w3.org/TR/2007/REC-soap12-part0-20070427/
- [WS-ADDR] "Web Services Addressing 1.0", http://www.w3.org/2005/08/addressing
- [SOAP-SCEN] "SOAP Version 1.2 Usage Scenarios", http://www.w3.org/TR/xmlp-scenarios/#S23

3 Introduction

This section is informative.

This specification is still in a draft status and subject to change.

This document describes a standard open protocol for communication between charge points and a central system. The control unit inside a charge point is referred to as a Charge Box; the central system as Central System.

Some basic concepts are explained in the sections below in this introductory chapter. Chapter Service Operations describes the operations supported by the protocol. The exact messages and their parameters are detailed in chapter Messages and data types are described in chapter Types.

The protocol is designed to be implemented with SOAP over HTTP. See chapter Binding to Transport Protocol for more details.

3.1 General views of operation

The following figures describe the general views of the operations between Charge Box and Central System for two cases: 1) a Charge Box requesting authentication of a card and sending charge transaction status, 2) Central System requesting a Charge Box to update its firmware. The arrow labels in the following figures indicate the PDUs exchanged during the invocations of the operations. These PDUs are defined in detail in section Messages.

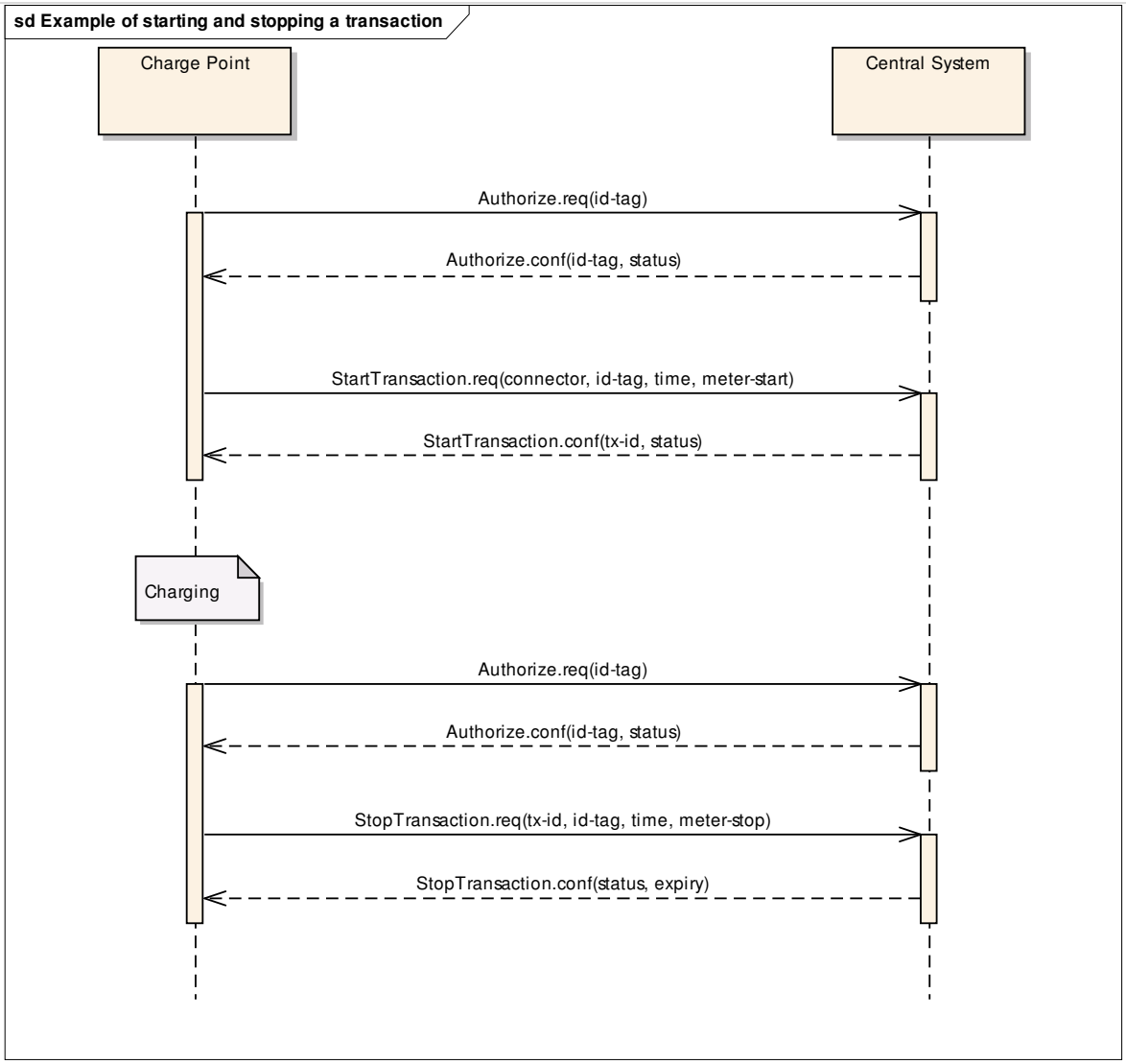


Figure: Example of starting and stopping a transaction

When a Charge Box needs to charge an electric vehicle, it needs to authenticate the user first before the charging can be started. If the user is authorized the Charge Box informs the Central System that it has started with charging.

When charging is done and a user wishes to unplug the electric vehicle from the charge point, the Charge Box needs to verify that the user is either the one that initiated the charging or that the user is in the same group and thus allowed to terminate the charging. Once authorized, the Charge Box informs the Central System that the charging has been stopped.

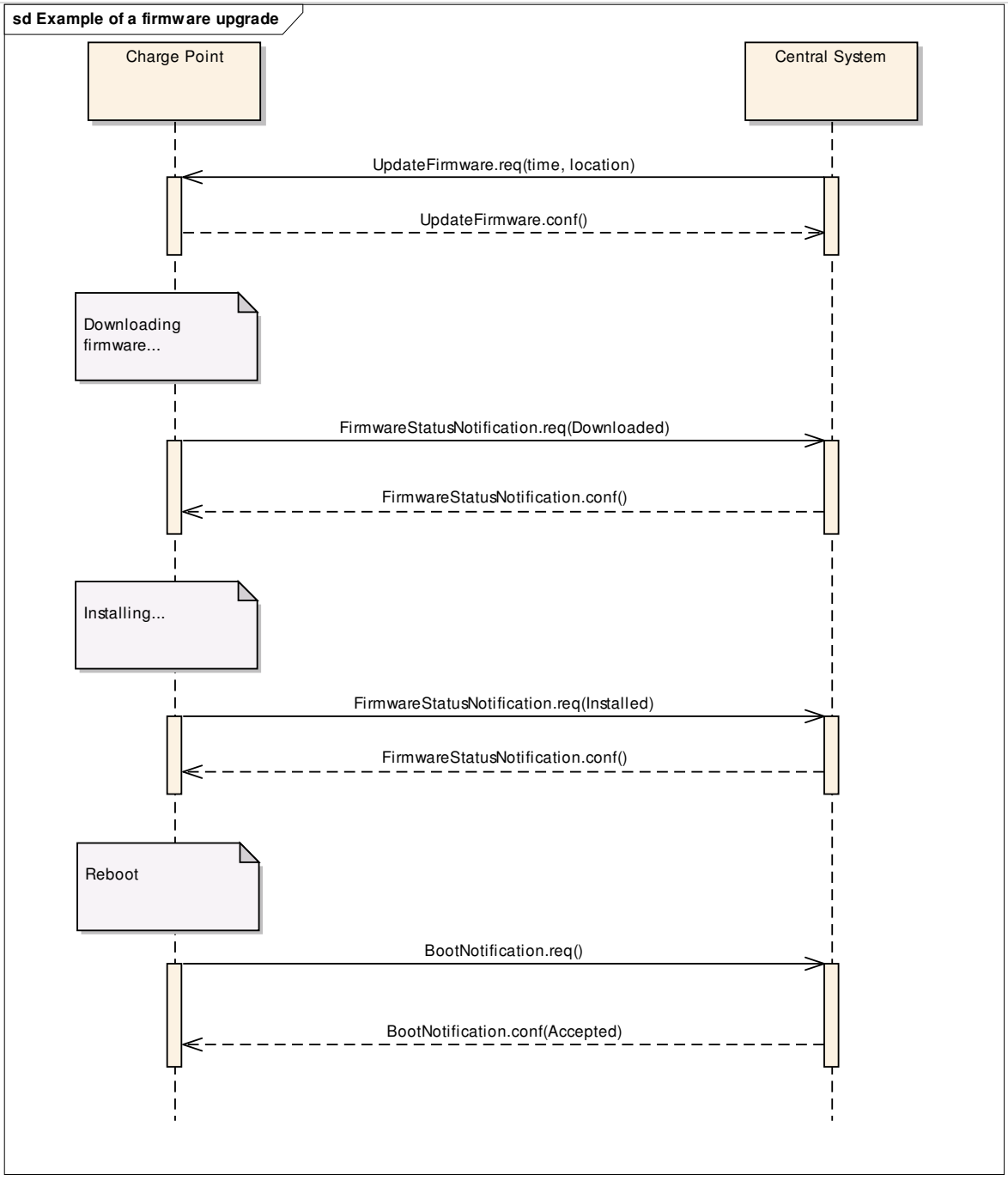


Figure: Example of a firmware upgrade

When a Charge Box needs to be updated with new firmware, then the Central System informs at which time the Charge Box can start downloading the new firmware. Once the Charge Box has downloaded and installed the firmware, it will

notify the Central System of the status for each step. Most likely the Charge Box needs to reboot, if so in that case, the Charge Box will sent a boot notification containing the new information about the installed firmware.

3.2 Off-line behaviour

Data communication between Charge Box and Central System will in most cases be done by means of wireless communication (e.g. GPRS, UMTS). In the event of unavailability of the communications or even of the Central System, the Charge Box is designed to operate stand-alone. It will accept identifiers known in its local white list. Charge Box will queue start and stop requests that need to be sent to the Central System and transmit these requests in order as soon as the connection to the Central System is restored. In such cases the Central System will not be aware that these are queued requests – it will process these as any other requests.

This off-line behaviour of Charge Box is the reason, that Central System always has to accept a 'start transaction' request. It cannot refuse it, because the transaction may already have taken place off-line.

3.3 Local white list

In order to support above-mentioned off-line behaviour Charge Box may implement a local white list to cache authorized id-tags. In response to an authorization request Central System will return parent-id and expiry-date. Charge Box should remove id-tags from its white list when they expire. In every communication that includes an id-tag (authorization, start and stop transaction) Central System will check validity of the id-tag an return a status value to Charge Box indicating validity of the id-tag. If the id-tag is no longer valid, then Charge Box should remove the id-tag from its local white list.

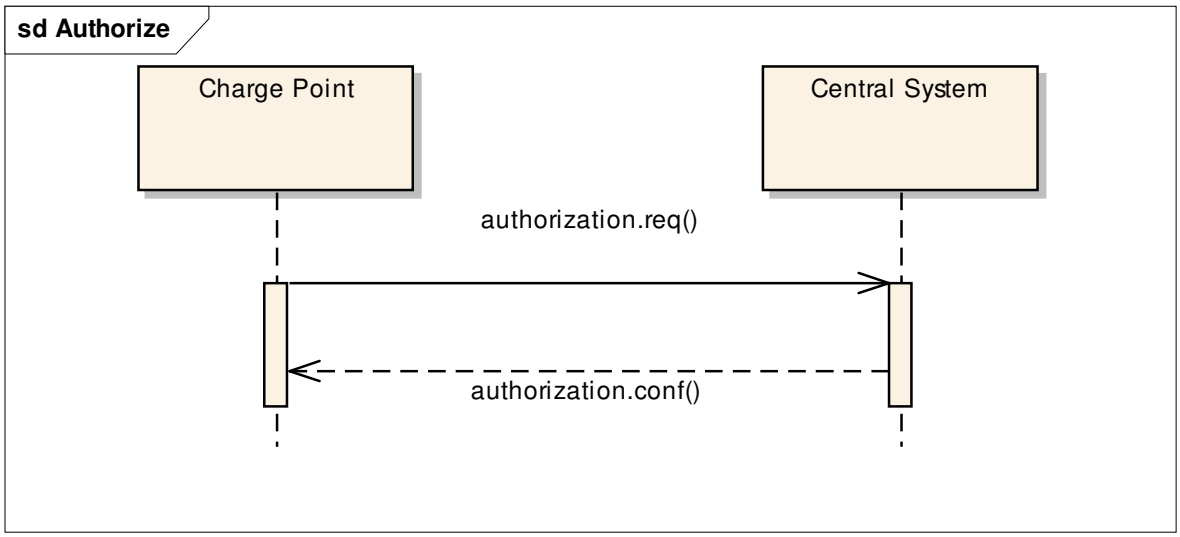
3.4 Parent id-tag

Central System has the ability to group id-tags, thus allowing one id-tag in a group to start a transaction and another id-tag in the same group to stop the transaction. Id-tags are grouped by appointing one id-tag in the group to be the parent. The value of parent-id-tag of every id-tag (including the parent) in the group is set to

the value of this ‘parent’ id-tag. Two id-tags are considered to be in the same group when their parent-id-tags match.

4 Operations Initiated by Charge Point

4.1 Authorize

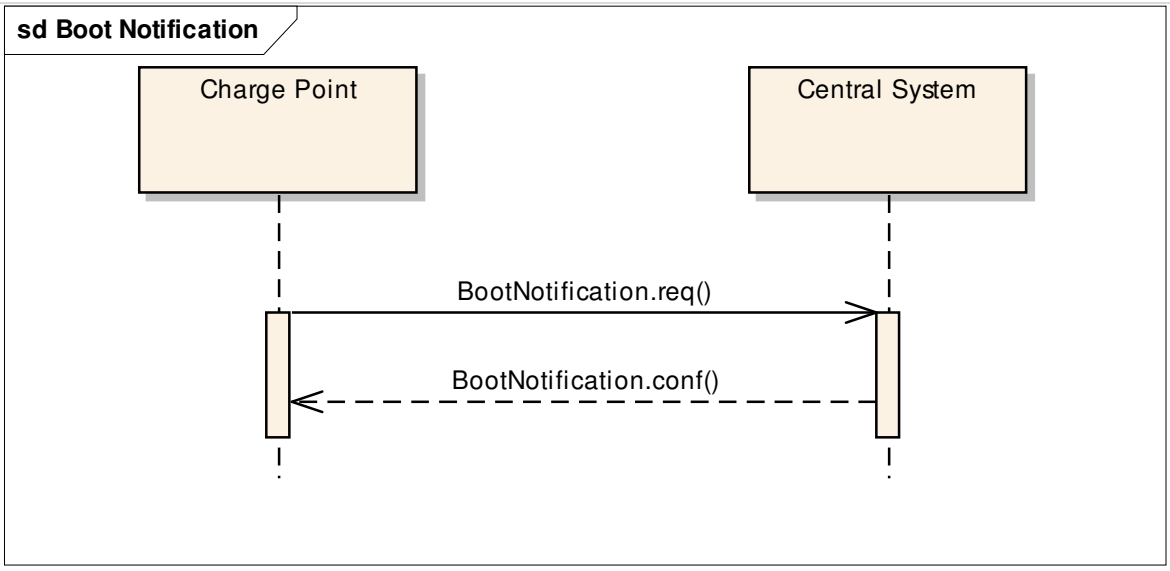


Before the owner of an electric vehicle can start or stop charging, the Charge Box needs to be able to authorize the operation. Only after authorization will Charge Box unlock the connector. Note: Stopping a charge transaction needs only to be authorized when the user stopping the transaction is different than the user that started the transaction.

A Charge Box MAY cache previously authorized identifiers in a white list and MAY use these to authorize a user. If the identifier is not present in the local white list, than the Charge Box SHALL send a Authorize.req PDU to the Central System for requesting authorization. If the identifier is present in the local white list, then Charge Box SHOULD not send an authorization reguest to the Central System.

Upon receipt of an Authorize.req PDU, the Central System SHALL respond with an Authorize.conf PDU. This response PDU SHALL indicate, whether or not the id-tag is accepted by the Central System. If the Central System accepts the id-tag then the response PDU MAY include a parent-id-tag and MUST include an authorization status value indicating acceptance or a reason for rejection.

4.2 Boot Notification



After start-up a Charge Box sends a notification to the Central System with information about its configuration (e.g. version, vendor, etc.). The Central System will only accept Charge Boxes that are registered with the Central System.

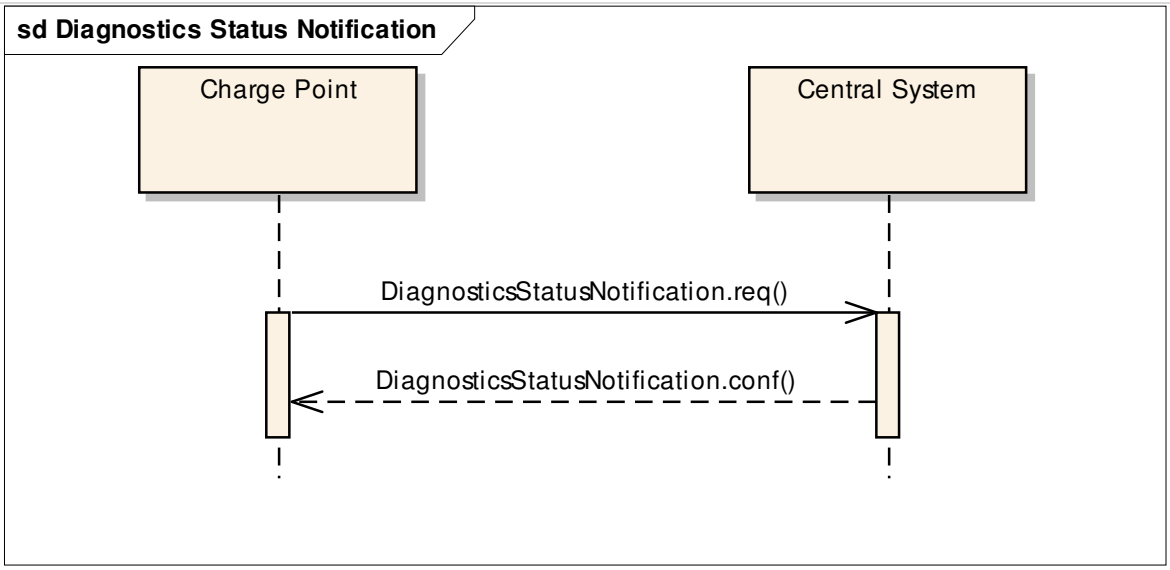
The Charge Box SHALL send a BootNotification.req PDU each time it (re-)boots.

Upon receipt of a BootNotification.req PDU, the Central System SHALL respond with a BootNotification.conf. The response PDU SHALL indicate whether the Central System accepts the Charge Box. When the Central System is willing to accept the Charge Box, then the response PDU SHALL contain the Central System's current time and heartbeat interval.

Upon receipt of a BootNotification.conf PDU, the Charge Box SHALL keep sending a BootNotification.req PDU when the Central System doesn't accept it. The Charge Box SHOULD use an acceptable interval for resending a BootNotification.req PDU, to avoid flooding the Central System.

The Charge Box SHALL NOT send any other request other then BootNotifcation.req until the Central System accepts the Charge Box.

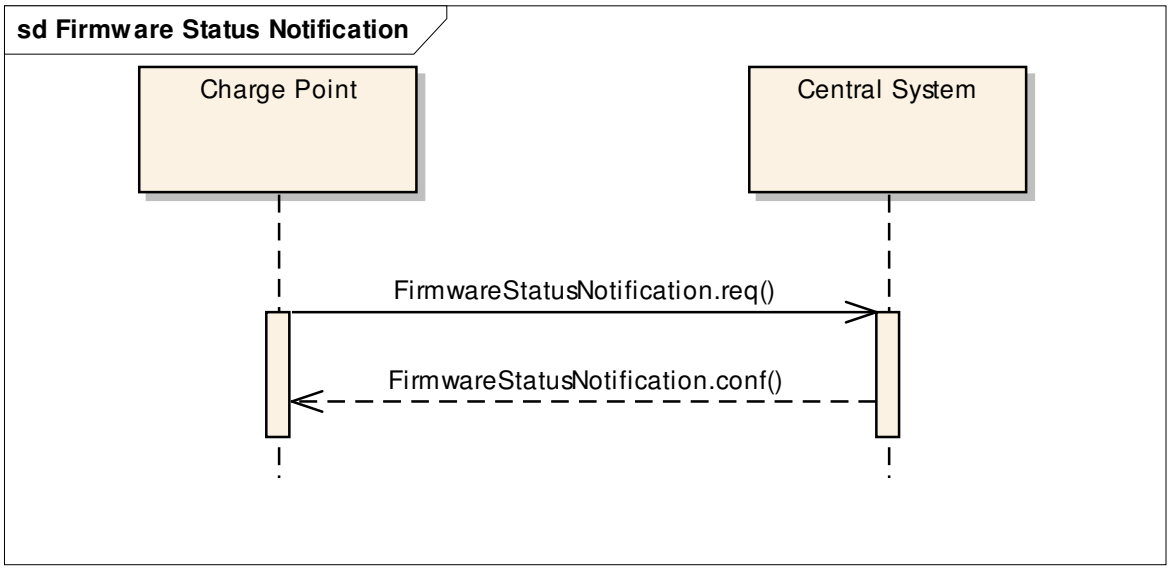
4.3 Diagnostics Status Notification



Charge Box sends a notification to inform the Central System when a diagnostics upload has finished.The Charge Box SHALL send a DiagnosticsStatusNotifcation.req PDU to inform the Central System that the upload of diagnostics has finished successfully or failed.

Upon receipt of a DiagnosticsStatusNotifcation.req PDU, the Central System SHALL respond with a DiagnosticsStatusNotifcation.conf.

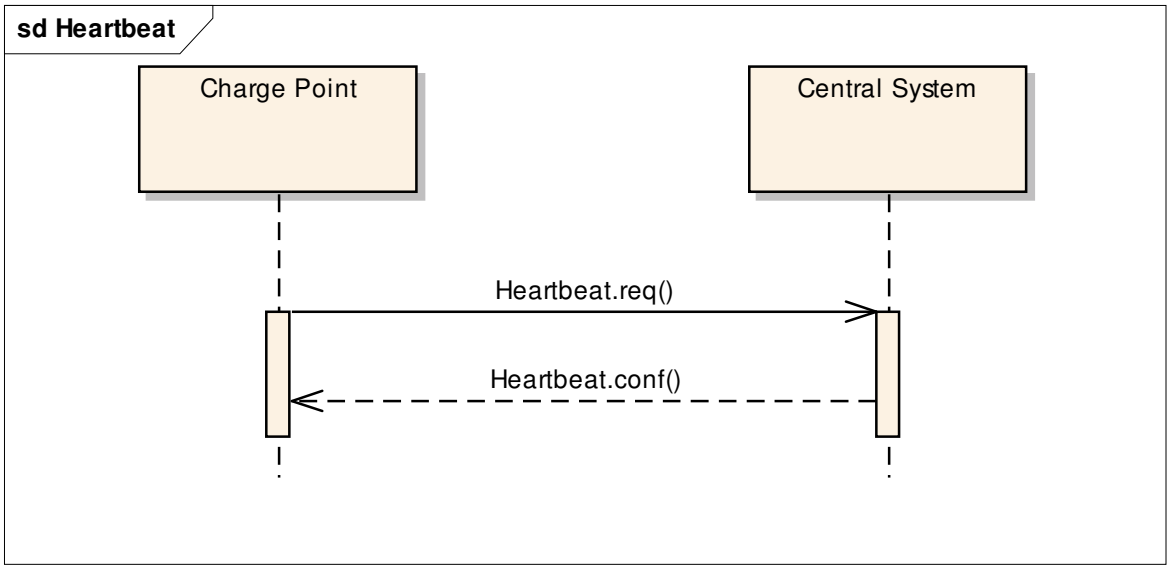
4.4 Firmware Status Notification



A Charge Box sends notifications to inform the Central System about the progress of the firmware update. The Charge Box SHALL send a FirmwareStatusNotifcation.req PDU for informing the Central System about the progress of the installation of a firmware update.

Upon receipt of a FirmwareStatusNotifcation.req PDU, the Central System SHALL respond with a FirmwareStatusNotifcation.conf.

4.5 Heartbeat



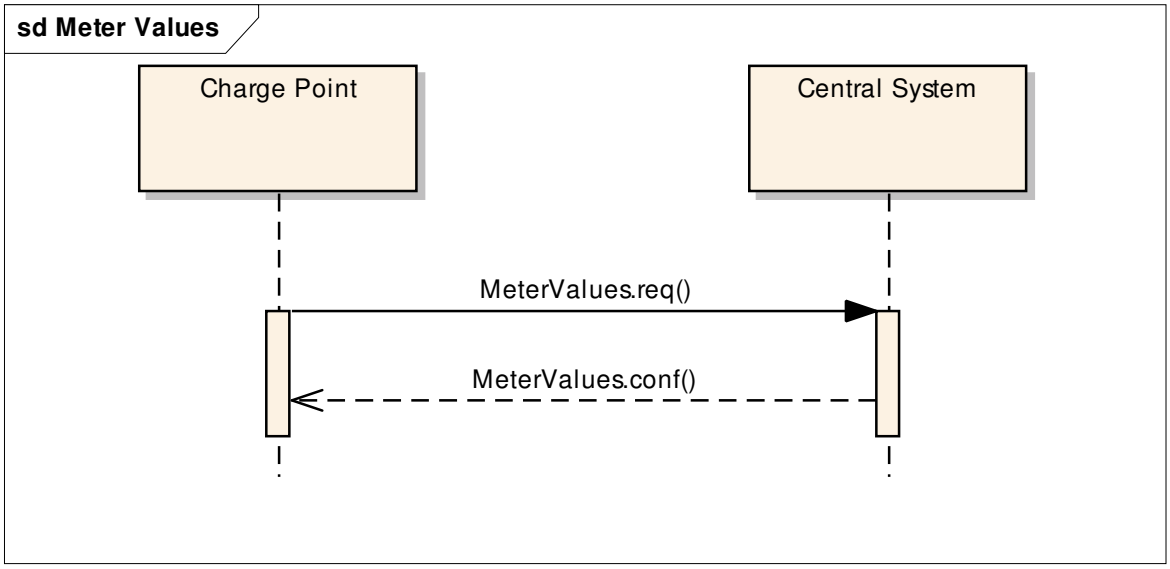
To let the Central System know that a Charge Point is still connected, a Charge Box sends a heartbeat after a configurable time interval.

The Charge Box SHALL send a Heartbeat.req PDU for ensuring that the Central System knows that a Charge Box is still alive.

Upon receipt of a Heartbeat.req PDU, the Central System SHALL respond with a Heartbeat.conf. The response PDU SHALL contain the current time of the Central System, which could be used by the Charge Box to synchronize its internal clock.

The Charge Box MAY skip sending a Heartbeat.req PDU when another PDU has been sent to the Central System within the configured heartbeat interval. This implies that a Central System SHOULD assume availability of a Charge Box whenever a PDU has been received, the same way as it would have, when it received a Heartbeat.req PDU.

4.6 Meter Values



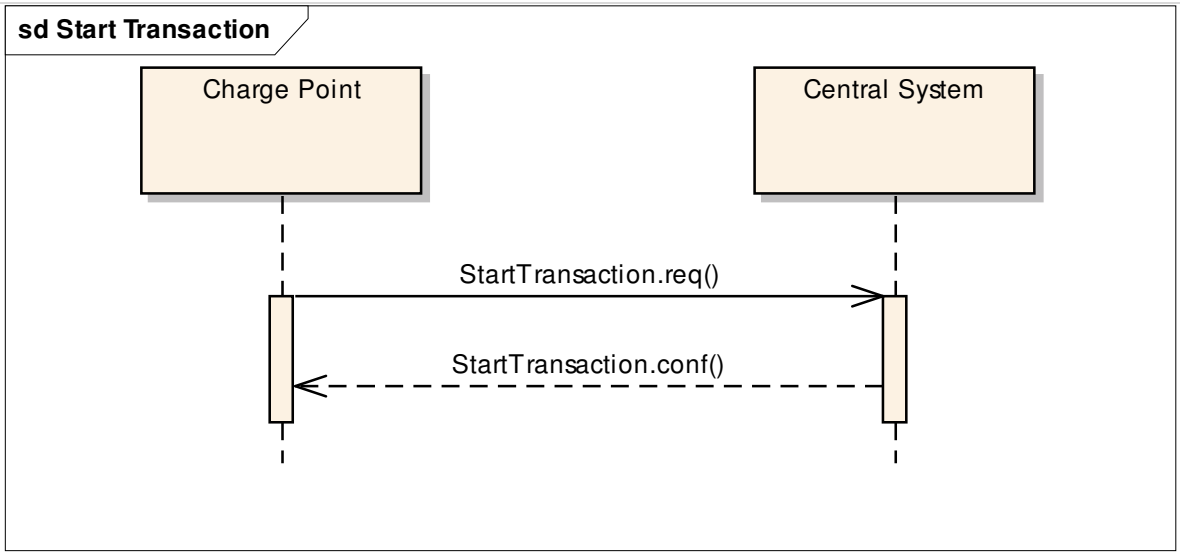
A Charge Box may sample the electricity meter to provide extra information about its meter values. It is up to the Charge Box to decide when it will send meter values.

The Charge Box SHALL send a MeterValues.req PDU for offloading meter values. The request PDU SHALL contain for each sample:

- The id of the connector from which samples were taken. A value of zero is used when samples were taken from the main power meter.
- A timestamp, containing the date and time on which the meter value has been sampled.
- A value, containing the sampled meter value in Wh (Watt-hours)

Upon receipt of a MeterValues.req PDU, the Central System SHALL respond with a MeterValues.conf.

4.7 Start Transaction



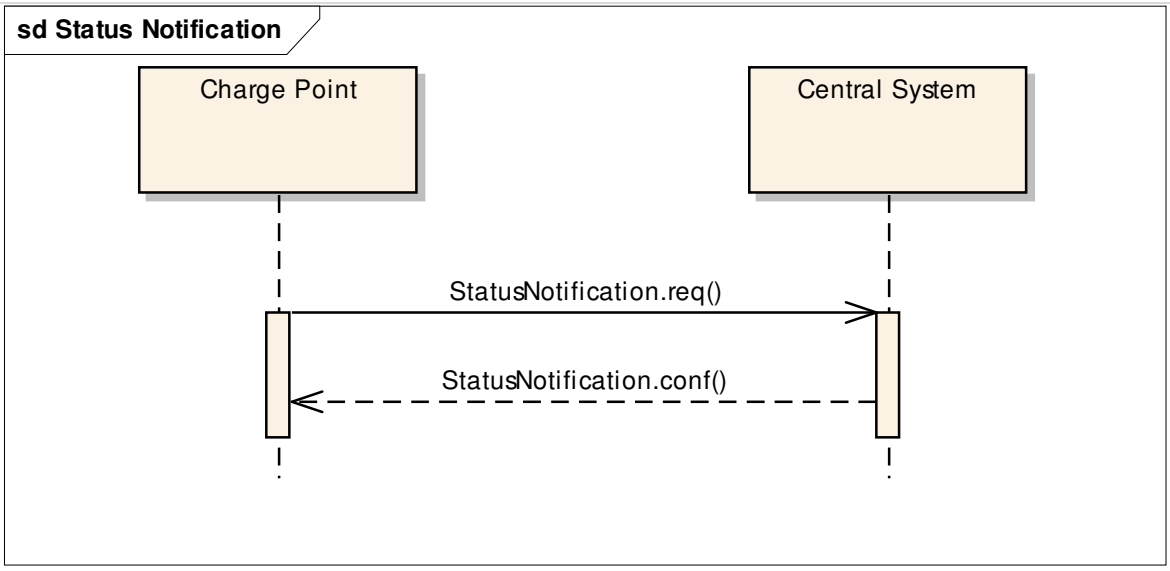
When an electric vehicle is allowed to start charging, the Charge Box needs to inform the Central System about this.

The Charge Box SHALL send a StartTransaction.req PDU to the Central System to inform it about the start of a charging transaction. Still, the Central System MUST verify validity of the transaction, because the identifier might have been authorized locally by the Charge Box using an out-of-date white list. The identifier, for instance, may have been blocked since it was added to the local white list.

Upon receipt of a StartTransaction.req PDU, the Central System SHALL respond with a StartTransaction.conf PDU. This response PDU MUST include a transaction id and an authorization status value.

Upon receipt of a StartTransaction.conf PDU with an authorization status value of 'Accepted' Charge Box SHALL update the white list entry with the values from the response. Upon receipt of a StartTransaction.conf PDU with an authorization status value other than 'Accepted' and if the transaction matching transaction-id is still in progress, Charge Box SHOULD stop the transaction and send the appropriate request as specified in Stop Transaction. For any authorizaton status value other than 'Accepted' or 'ConcurrentTx', Charge Box SHALL remove the id-tag from its local white list, so that next time it cannot be authorized locally by Charge Box, but will have to be explicitly authorized by the Central System.

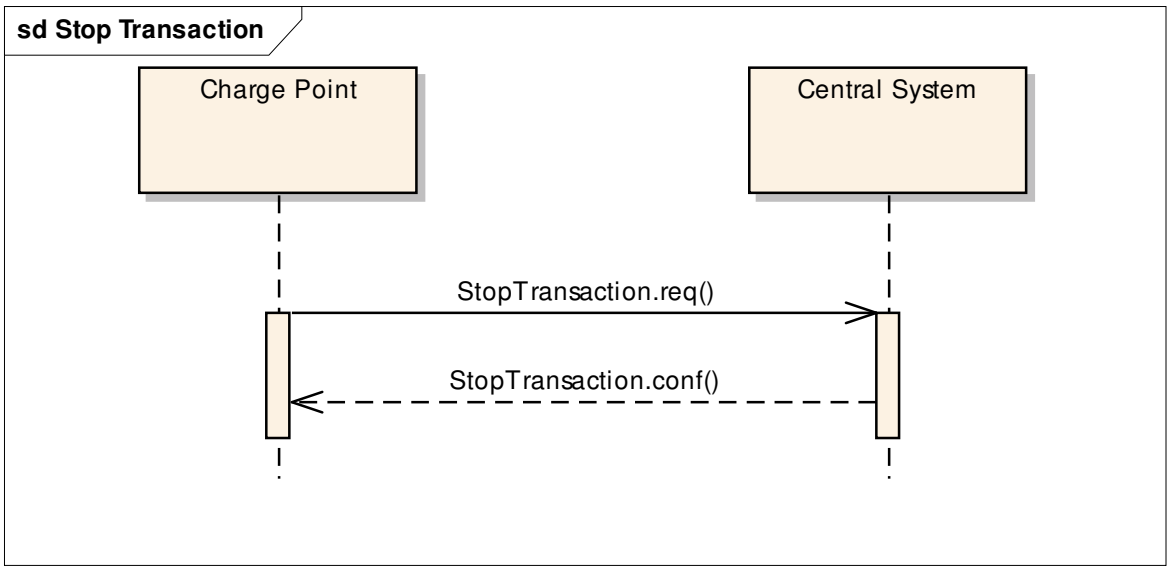
4.8 Status Notification



A Charge Box sends a notification to the Central System to inform the Central System about a status or error condition within the Charge Box.The Charge Box SHALL send an StatusNotification.req PDU when it becomes unavailable as result of an error condition or a scheduled Change Availability command. The Charge Box MAY send an StatusNotification.req PDU to inform the Central System of other error conditions.

Upon receipt of an StatusNotification.req PDU, the Central System SHALL respond with an StatusNotification.conf.

4.9 Stop Transaction



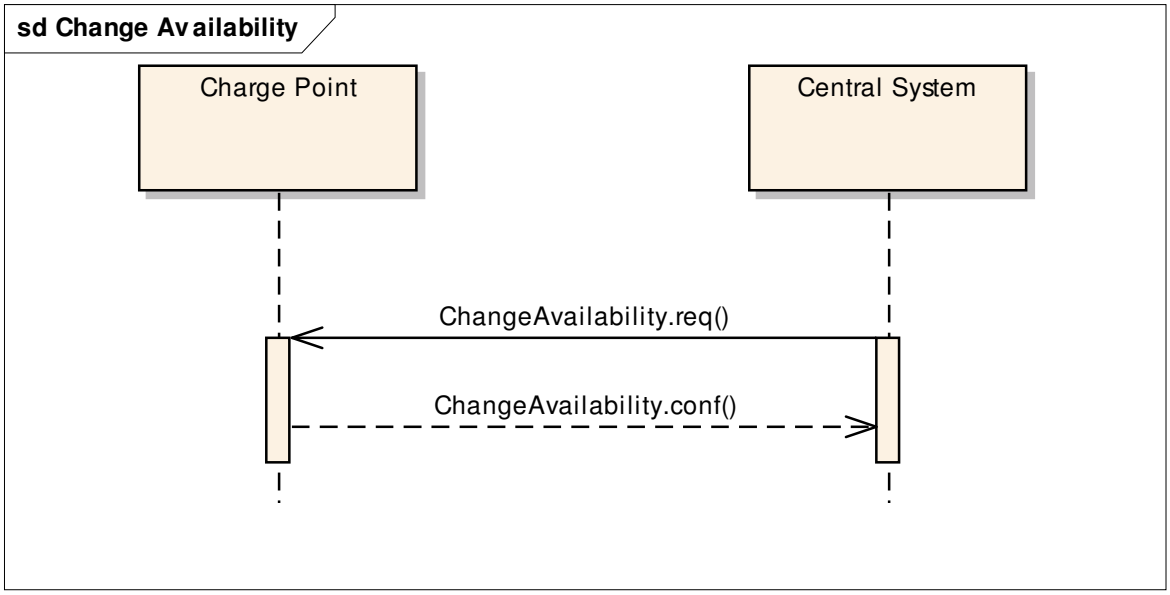
When an electric vehicle is allowed to stop charging; the Charge Box needs to be able to inform the Central System about this. The Charge Box SHALL send a StopTransaction.req PDU when a transaction needs to be stopped. A transaction MUST be stopped explicitly by the Charge Box.

Upon receipt of a StopTransaction.req PDU, the Central System SHALL respond with a StopTransaction.conf PDU. If authorization status value is 'Accepted' then Charge Box SHALL update the white list entry with the values from the response. For any authorization status value other than 'Accepted' Charge Box SHALL remove the id-tag from its local white list. The Central System SHALL always stop the transaction.

The id-tag in the request PDU MAY be omitted when the Charge Box itself needs to stop the transaction. For instance, when the Charge Box is requested to reset.

5 Operations Initiated by Central System

5.1 Change Availability

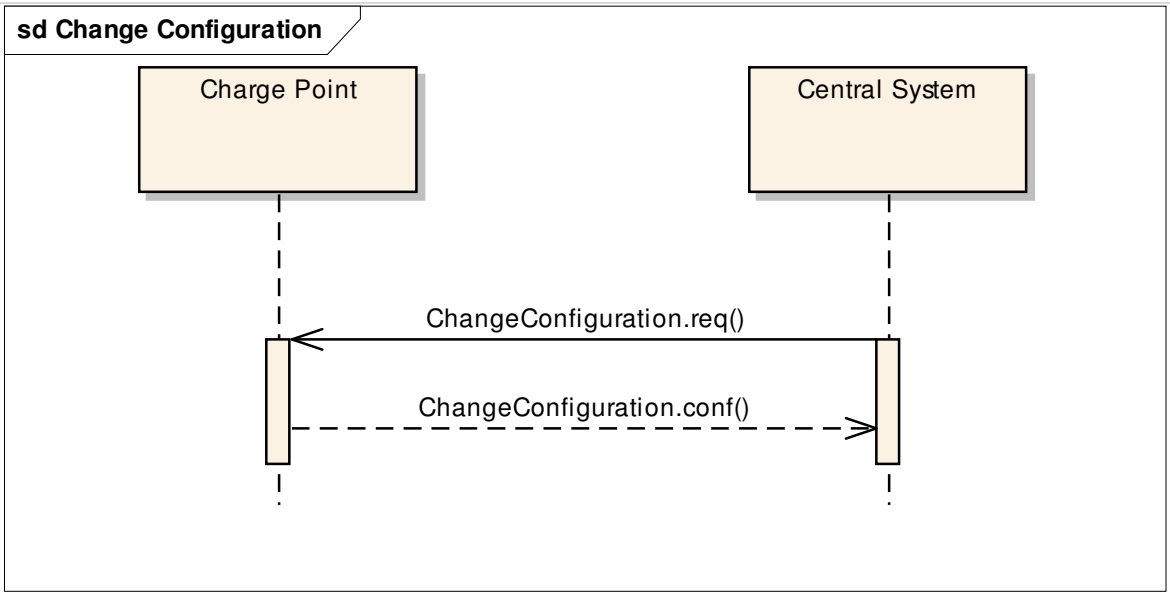


Central System can request a Charge Box to change its availability. A Charge Box is considered available (“operative”) when it is charging or ready for charging. A Charge Box is considered unavailable when it does not allow any charging. The Central System SHALL send a ChangeAvailability.req PDU for requesting a Charge Box to change its availability. The Central System can change the availability to available or unavailable.

Upon receipt of a ChangeAvailability.req PDU, the Charge Box SHALL respond with a ChangeAvailability.conf PDU. The response PDU SHALL indicate whether the Charge Box is able to change to the requested availability or not. When a transaction is in progress Charge Box SHALL respond with availability status 'Scheduled' to indicate that it is scheduled to occur after the transaction has finished. When that happens Charge Box SHALL inform Central System of its new availability status with a Status Notification.

In the event that Central System requests Charge Box to change to a status it is already in, Charge Box SHALL respond with availability status ‘Accepted’.

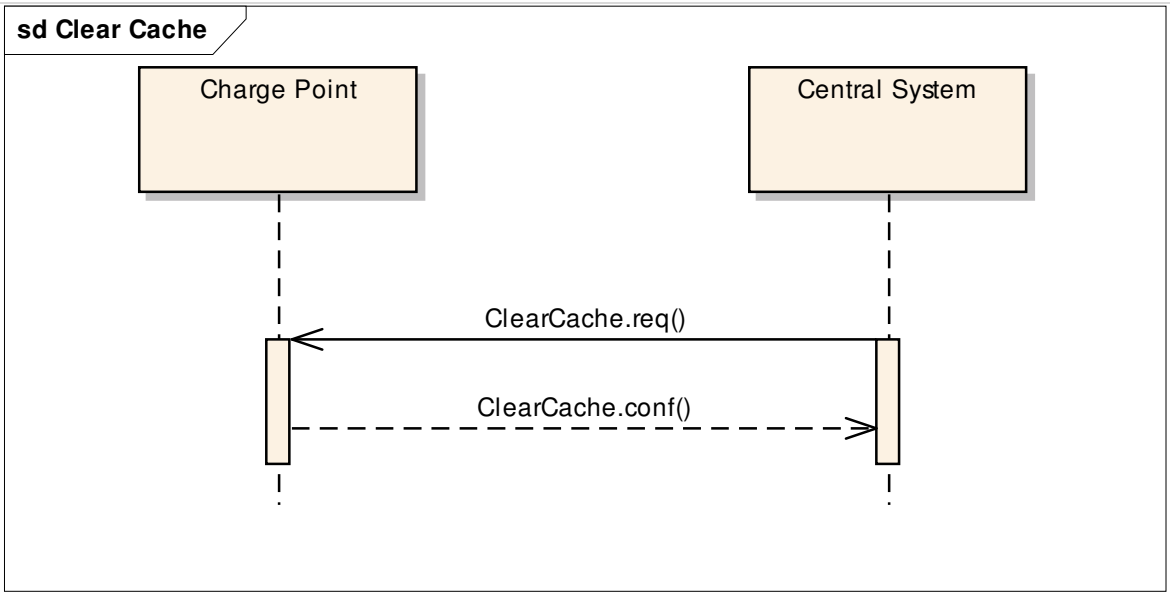
5.2 Change Configuration



Central System can request a Charge Box to change configuration parameters. To achieve this, Central System SHALL send a ChangeConfiguration.req. This request contains a key-value pair, where "key" is the name of the configuration setting to change and "value" contains the new setting for the configuration setting.

Upon receipt of a ChangeConfiguration.req Charge Box SHALL reply with a ChangeConfiguration.conf indicating whether it was able to executed the change. Content of "key" and "value" is not prescribed. If "key" does not correspond to a configuration setting supported by Charge Box, it SHALL reply with a status NotSupported. Otherwise it SHALL indicate success or failure with status Accepted or Rejected respectively.

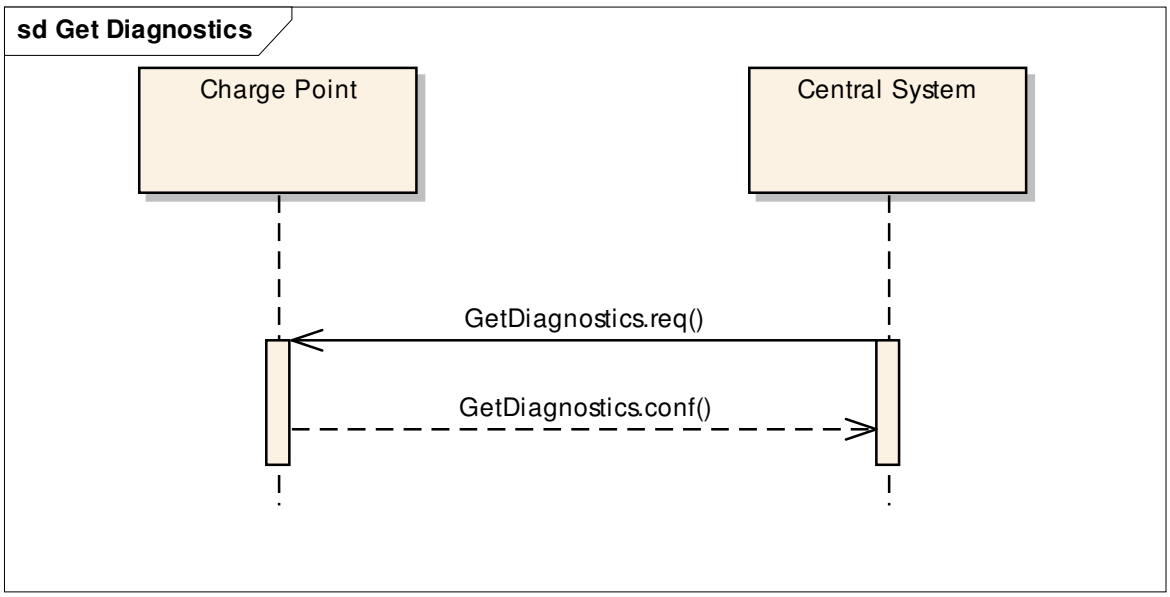
5.3 Clear Cache



Central System can request a Charge Box to clear its cache. For instance a Charge Box may cache previously authorized cards in a white list.The Central System SHALL send a ClearCache.req PDU for clearing the Charge Box's cache.

Upon receipt of a ClearCache.req PDU, the Charge Box SHALL respond with a ClearCache.conf PDU. The response PDU SHALL indicate whether the Charge Box was able to clear its cache.

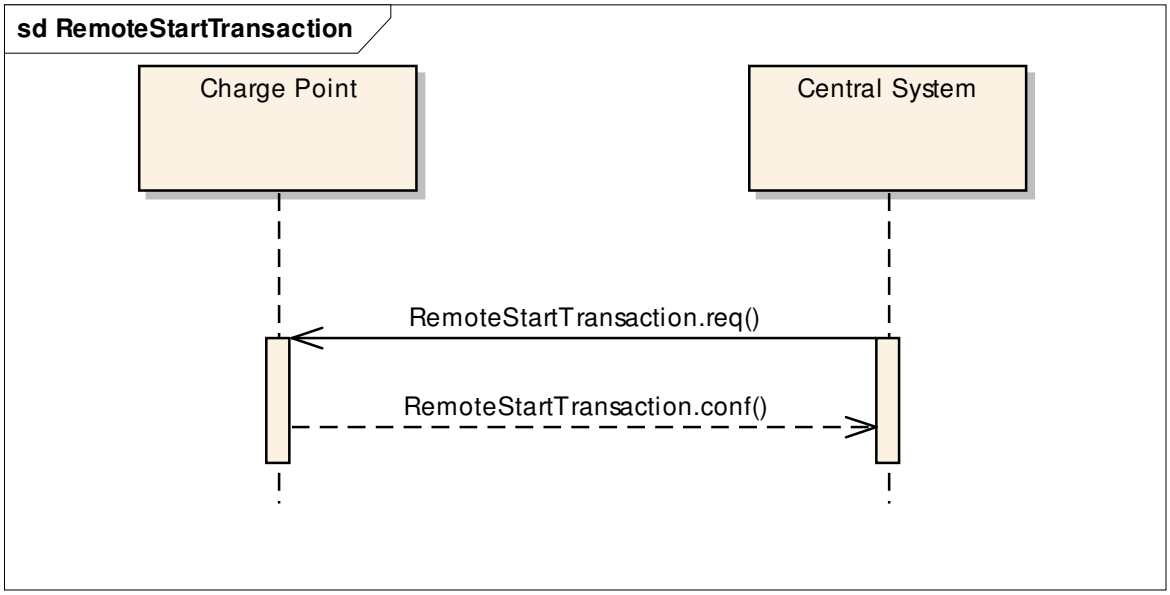
5.4 Get Diagnostics



Central System can request a Charge Box for diagnostic information.The Central System SHALL send a GetDiagnostics.req PDU for getting diagnostic information of a Charge Box with a location where the Charge Box MUST upload its diagnostic data to and optionally a begin and end time for the requested diagnostic information.

Upon receipt of a GetDiagnostics.req PDU, and if diagnostics information is available than Charge Box SHALL respond with a GetDiagnostics.conf PDU stating the name of the file containing the diagnostic information that will be uploaded. Charge Box SHALL upload a single compressed file. Format of the diagnostics file is not prescribed. If no diagnostics file is available, then GetDiagnostics.conf SHALL not contain a file name.

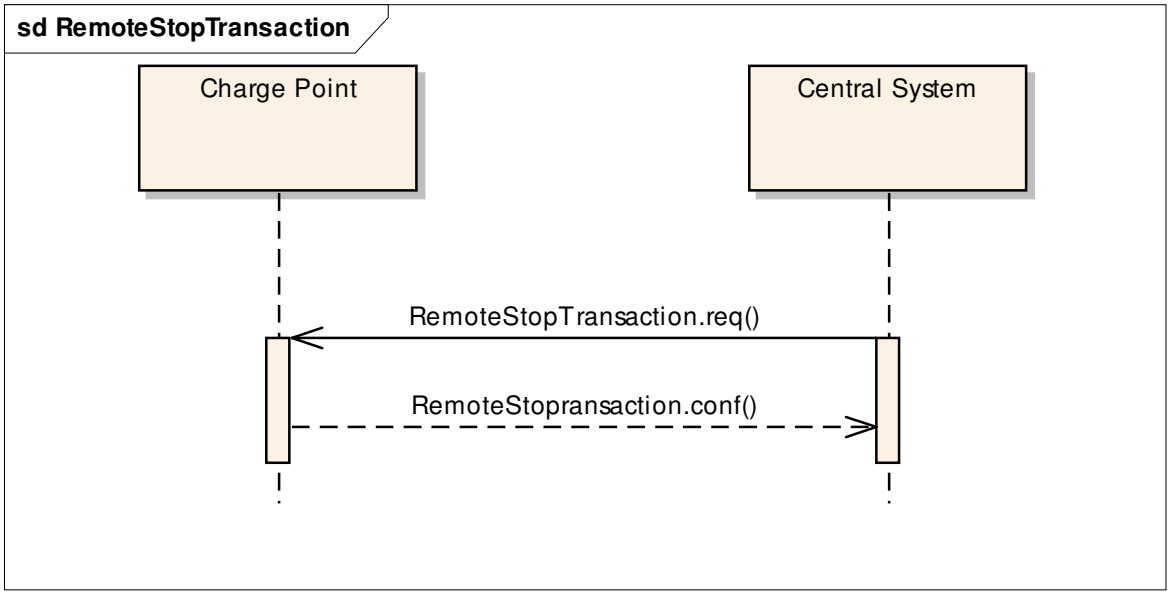
5.5 RemoteStartTransaction



Central System can request a Charge Box to start a transaction by sending a RemoteStartTransaction.req. Upon receipt Charge Box SHALL reply with RemoteStartTransaction.conf and a status indicating whether it is able to start a transaction or not.

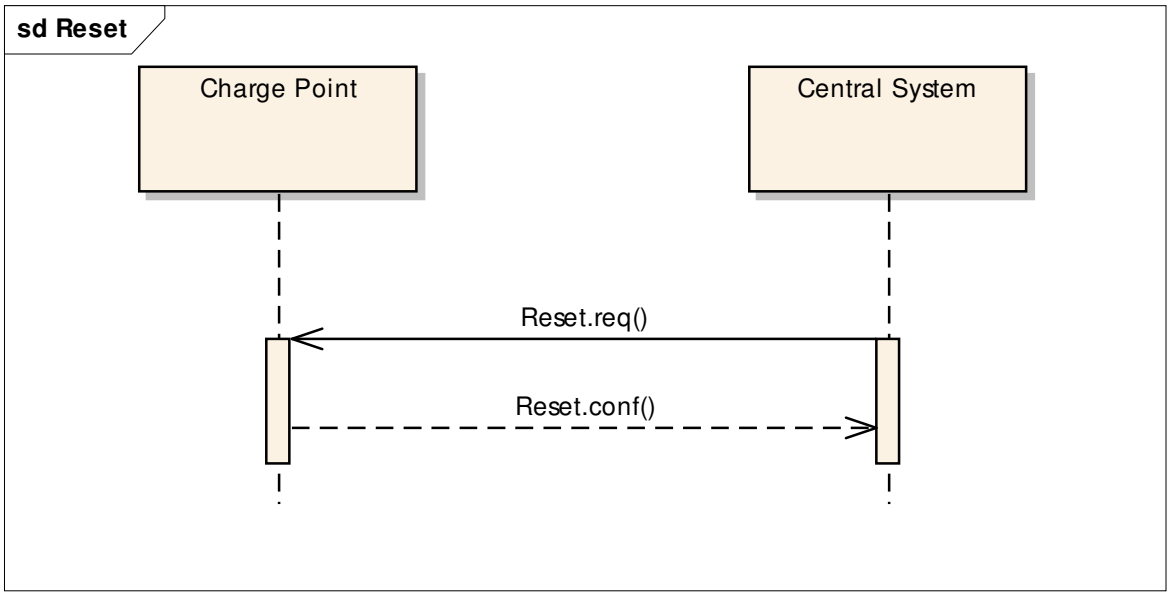
The RemoteStartTransaction.req SHALL contain an identifier (IdTag), which Charge Box SHALL use, if it is able to start a transaction, to send a StartTransaction.req to Central System. The transaction is started in the same way as described in Start Transaction. The RemoteStartTransaction.req MAY contain a connector id if the transaction is to be started on a specific connector.

5.6 RemoteStopTransaction



Central System can request a Charge Box to stop a transaction by sending a RemoteStopTransaction.req to Charge Box with the identifier of the transaction. Charge Box SHALL reply with RemoteStopTransaction.conf to indicate whether it is indeed able to the transaction.

5.7 Reset

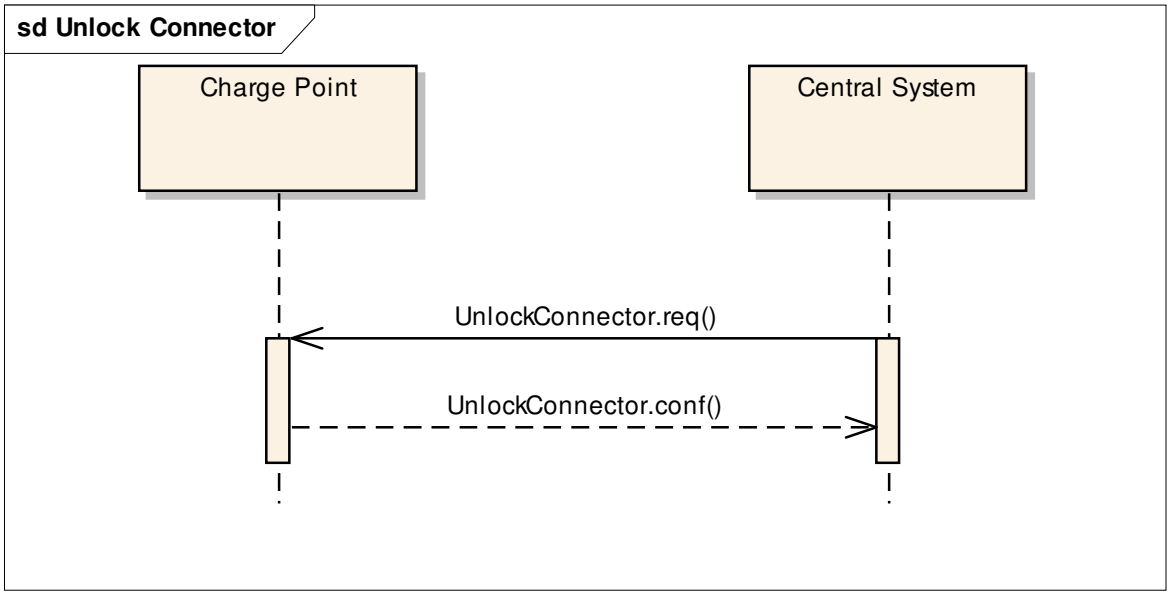


The Central System SHALL send a Reset.req PDU for requesting a Charge Box to reset itself. The Central System can request a hard or a soft reset. Upon receipt of a Reset.req PDU, the Charge Box SHALL respond with a Reset.conf PDU. The response PDU SHALL indicate whether the Charge Box is willing to reset itself.

At receipt of a soft reset, Charge Box SHALL return to its basic idle state. If any transaction is in progress it SHALL be terminated normally, as in Stop Transaction.

At receipt of a hard reset Charge Box SHALL attempt to terminate any transaction in progress normally as in Stop Transaction and then perform a reboot.

5.8 Unlock Connector

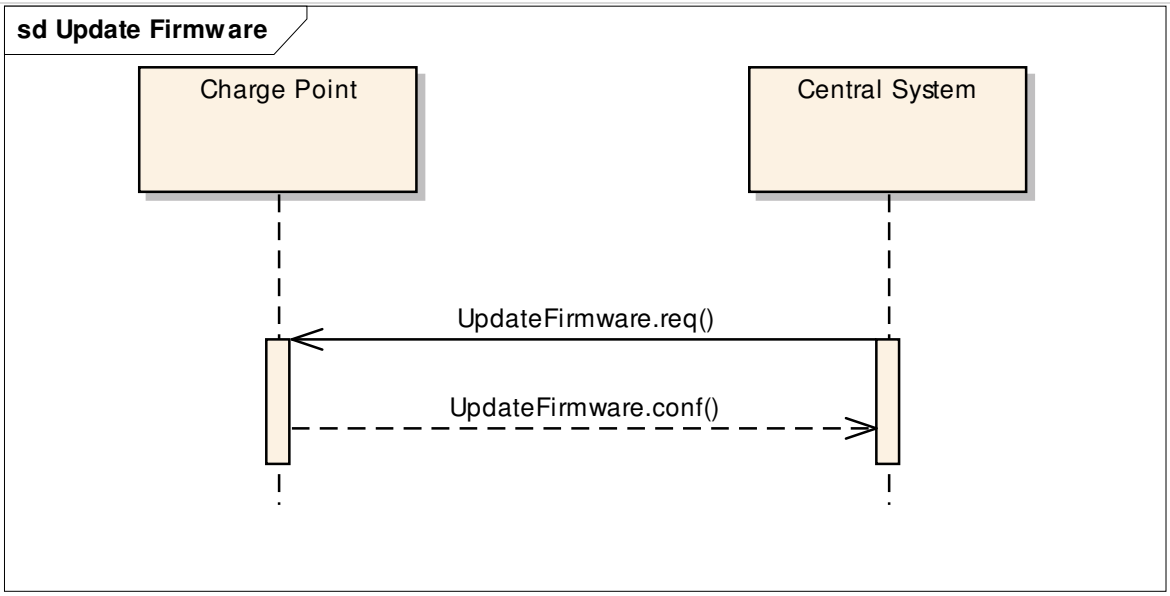


Central System can unlock a connector of a Charge Box. The Central System SHALL send an UnlockConnector.req PDU for requesting a Charge Box to unlock one of its connector.

Upon receipt of an UnlockConnector.req PDU, the Charge Box SHALL respond with a UnlockConnector.conf PDU. The response PDU SHALL indicate whether the Charge Box was able to unlock its connector.

If there was a transaction in progress on the specific connector, then Charge Box SHALL finish the transaction first as described in Stop Transaction.

5.9 Update Firmware



Central System can notify a Charge Box that it needs to update its firmware.The Central System SHALL send an UpdateFirmware.req PDU to inform the Charge Box about the availability of new firmware. The PDU SHALL contain a date and time after which the Charge Box is allowed to retrieve the new firmware and the location from which the firmware can be downloaded.

Upon receipt of an UpdateFirmware.req PDU, the Charge Box SHALL respond with a UpdateFirmware.conf PDU. The Charge Box SHOULD start retrieving the firmware as soon as possible after retrieve-date.

6 Messages

6.1 Authorize.conf

This contains the field definition of the Authorize.conf PDU sent by the Central System to the Charge Box as response to a Authorize.req PDU.

Field Name	Field Type	Card.	Description
idTagInfo	IdTagInfo	1..1	Mandatory.This contains information about authorization status, expiry and parent id.

6.2 Authorize.req

This contains the field definition of the Authorize.req PDU sent by the Charge Box to the Central System.

Field Name	Field Type	Card.	Description
idTag	string[15]	1..1	Mandatory This contains the identifier that needs to be authorized.

6.3 BootNotification.conf

This contains the field definition of the BootNotification.conf PDU sent by the Central System to the Charge Box as response to a BootNotification.req PDU.

Field Name	Field Type	Card.	Description
currentTime	dateTime	0..1	Optional. This contains the Central System's current time.

heartbeatInterval	integer	0..1	Optional. This contains the interval in seconds of the heartbeats.
status	RegistrationStatus	1..1	Mandatory.This contains whether the Charge-Box has been registered within the System Central.

6.4 BootNotification.req

This contains the field definition of the BootNotification.req PDU sent by the Charge Box to the Central System.

Field Name	Field Type	Card.	Description
chargeBoxSerialNumber	string[25]	0..1	Optional.This contains a value that identifies the serial number of the Charge Box inside the Charge Point..
chargePointModel	string[20]	1..1	Mandatory. This contains a value that identifies the model of the ChargePoint.
chargePointSerialNumber	string[25]	0..1	Optional.This contains a value that identifies the serial number of the Charge Point.
chargePointVendor	string[20]	1..1	Mandatory. This contains a value that identifies the vendor of the ChargePoint.
firmwareVersion	string[20]	0..1	Optional. This contains the firmware version of the Charge Box.
iccid	string[20]	0..1	Optional. This contains the ICCID of the modem's SIM card.

imsi	string[20]	0..1	Optional. This contains the IMSI of the modem's SIM card.
meterSerialNumber	string[25]	0..1	Optional. This contains the serial number of the main power meter of the Charge Point.
meterType	string[25]	0..1	Optional. This contains the type of the main power meter of the Charge Point.

6.5 ChangeAvailability.conf

This contains the field definition of the ChangeAvailability.conf PDU return by Charge Box to Central System.

Field Name	Field Type	Card.	Description
status	AvailabilityStatus	1..1	Mandatory.This indicates whether the Charge Box can perform the availability change.

6.6 ChangeAvailability.req

This contains the field definition of the ChangeAvailability.req PDU sent by the Central System to the Charge Box.

Field Name	Field Type	Card.	Description
connectorId	integer connectorId >= 0	1..1	Mandatory. The id of the connector for which availability needs to change. Id '0' (zero) is used if the availability of the charge point as a whole needs to change.

type	AvailabilityType	1..1	Mandatory.This contains the type of availability change that the Charge Box should perform.

6.7 ChangeConfiguration.conf

This contains the field definition of the ChangeConfiguratoin.conf PDU returned from Charge Box to Central System.

Field Name	Field Type	Card.	Description
status	ConfigurationStatus	1..1	Mandatory. Returns whether configuration change has been accepted.

6.8 ChangeConfiguration.req

This contains the field definition of the ChangeConfiguration.req PDU sent by Central System to Charge Box. It is RECOMMENDED that the content and meaning of the 'key' and 'value' attributes is agreed upon between Charge Box and Central System.

Field Name	Field Type	Card.	Description
key	string[50]	1..1	Mandatory. The name of the configuration setting to change. Predefined keys are: **HeartBeatInterval** (in seconds) **ConnectionTimeOut** (in seconds) **ProximityContactRetries**

			(in times) **ProximityLockRetries** (in times) **ResetRetries** (in times) **BlinkRepeat** (in times) **LightIntensity** (in %) **ChargePointId** (string) **MeterValueSampleInterval** (in seconds)
value	string[500]	1..1	Mandatory. The new value as string for the setting.

6.9 ClearCache.conf

This contains the field definition of the ClearCache.conf PDU sent by the Charge Box to the Charge Box as response to a ClearCache.req PDU.

Field Name	Field Type	Card.	Description
status	ClearCacheStatus	1..1	

6.10 ClearCache.req

This contains the field definition of the ClearCache.req PDU sent by the Central System to the Charge Box.

No fields are defined.

6.11 DiagnosticsStatusNotification.conf

This contains the field definition of the DiagnosticsStatusNotification.conf PDU sent by the Central System to the Charge Box as response to a DiagnosticsStatusNotification.req PDU.

No fields are defined.

6.12 DiagnosticsStatusNotification.req

This contains the field definition of the DiagnosticsStatusNotification.req PDU sent by the Charge Box to the Central System.

Field Name	Field Type	Card.	Description
status	DiagnosticsStatus	1..1	Mandatory.This contains the status of the diagnostics upload.

6.13 FirmwareStatusNotification.conf

This contains the field definition of the FirmwareStatus.conf PDU sent by the Central System to the Charge Box as response to a FirmwareStatus.req PDU.

No fields are defined.

6.14 FirmwareStatusNotification.req

This contains the field definition of the FirmwareStatus.req PDU sent by the Charge Box to the Central System.

Field Name	Field Type	Card.	Description
status	FirmwareStatus	1..1	Mandatory.This contains the progress status of the

			firmware installation.

6.15 GetDiagnostics.conf

This contains the field definition of the GetDiagnostics.conf PDU sent by the Charge Box to the Central System as response to a GetDiagnostics.req PDU.

Field Name	Field Type	Card.	Description
fileName	string[255]	0..1	Optional.This contains the name of the file with diagnostic information that will be uploaded. This attribute is not present when no diagnostic information is available.

6.16 GetDiagnostics.req

This contains the field definition of the GetDiagnostics.req PDU sent by the Central System to the Charge Box.

Field Name	Field Type	Card.	Description
location	anyURI	1..1	Mandatory. This contains the location (directory) where the diagnostics file shall be uploaded to.
retries	integer	0..1	Optional.This specifies how many times Charge Box must try to upload the diagnostics before giving up. If this field is not present, it is left to Charge Box to decide how many times it wants to retry.
retryInterval	integer	0..1	Optional. The interval in seconds after which a retry

			may be attempted. If this field is not present, it is left to Charge Box to decide how long to wait between attempts.
startTime	dateTime	0..1	Optional. This contains the date and time of the oldest logging information to include in the diagnostics.
stopTime	dateTime	0..1	Optional. This contains the date and time of the latest logging information to include in the diagnostics.

6.17 Heartbeat.conf

This contains the field definition of the Heartbeat.conf PDU sent by the Central System to the Charge Box as response to a Heartbeat.req PDU.

Field Name	Field Type	Card.	Description
currentTime	dateTime	1..1	Mandatory.This contains the current time of the Central System.

6.18 Heartbeat.req

This contains the field definition of the Hearbeat.req PDU sent by the Charge Box to the Central System.

No fields are defined.

6.19 MeterValues.conf

This contains the field definition of the MeterValues.conf PDU sent by the Central System to the Charge Box as response to a MeterValues.req PDU.

No fields are defined.

6.20 MeterValues.req

This contains the field definition of the MeterValues.req PDU sent by the Charge Box to the Central System.

Field Name	Field Type	Card.	Description
connectorId	integer connectorId >= 0	1..1	Mandatory. This contains a number (>0) designating a connector of the Charge Point.‘0’ (zero) is used to designate the main power meter.
values	MeterValue	0..*	Optional.The sampled meter values with timestamps.

6.21 RemoteStartTransaction.conf

This contains the field definitions of the RemoteStartTransaction.conf PDU sent from Charge Box to Central System.

Field Name	Field Type	Card.	Description
status	RemoteStartStopStatus	1..1	Mandatory. Status indicating whether Charge Box accepts the request to start a transaction.

6.22 RemoteStartTransaction.req

This contains the field definitions of the RemoteStartTransaction.req PDU sent to Charge Box by Central System.

Field Name	Field Type	Card.	Description
connectorId	int connectorId > 0	0..1	Optional. Number of the connector on which to start the transaction.
idTag	string[15]	1..1	Mandatory. The identifier that Charge Box must use to start a transaction.

6.23 RemoteStopTransaction.conf

This contains the field definitions of the RemoteStopTransaction.conf PDU sent from Charge Box to Central System.

Field Name	Field Type	Card.	Description
status	RemoteStartStopStatus	1..1	Mandatory. Status indicating whether Charge Box accepts the request to stop a transaction.

6.24 RemoteStopTransaction.req

This contains the field definitions of the RemoteStopTransaction.req PDU sent to Charge Box by Central System.

Field Name	Field Type	Card.	Description
transactionId	integer	1..1	Mandatory. The identifier of the transaction which Charge Box is requested to stop.

6.25 Reset.conf

This contains the field definition of the Reset.conf PDU sent by the Charge Box to the Central System as response to a Reset.req PDU.

Field Name	Field Type	Card.	Description
status	ResetStatus	1..1	Mandatory.This indicates whether the Charge Box can perform the reset.

6.26 Reset.req

This contains the field definition of the Reset.req PDU sent by the Central System to the Charge Box.

Field Name	Field Type	Card.	Description
type	ResetType	1..1	Mandatory. This contains the type of reset that the Charge Box should perform.

6.27 StartTransaction.conf

This contains the field definition of the StartTransaction.conf PDU sent by the Central System to the Charge Box as response to a StartTransaction.req PDU.

Field Name	Field Type	Card.	Description
idTagInfo	IdTagInfo	1..1	Mandatory.This contains information about authorization status, expiry and parent id.
transactionId	integer	1..1	Mandatory.This contains the transaction id supplied by the Central System.

6.28 StartTransaction.req

This section contains the field definition of the StartTransaction.req PDU sent by the Charge Box to the Central System.

Field Name	Field Type	Card.	Description
connectorId	integer connectorId > 0	1..1	Mandatory.This identifies which connector of the Charge Box is used.
idTag	string[15]	1..1	Mandatory.This contains the identifier for which a transaction has to be started.
meterStart	integer	1..1	Mandatory.This contains the meter value in Wh for the connector at start of the transaction.
timestamp	dateTime	1..1	Mandatory.This contains the date and time on which the transaction is started.

6.29 StatusNotification.conf

This contains the field definition of the StatusNotifcation.conf PDU sent by the Central System to the Charge Box as response to an StatusNotification.req PDU.

No fields are defined.

6.30 StatusNotification.req

This contains the field definition of the StatusNotification.req PDU sent by the Charge Box to the Central System.

Field Name	Field Type	Card.	Description
connectorId	integer connectorId >= 0	1..1	Mandatory. The id of the connector for which the status is reported. Id '0' (zero) is used if the status is for the charge point as a whole.
errorCode	ChargePointErrorCode	1..1	Mandatory.This contains the error code reported by the Charge Point.
status	ChargePointStatus	1..1	Mandatory.This contains the current status of the Charge Point.

6.31 StopTransaction.conf

This contains the field definition of the StopTransaction.conf PDU sent by the Central System to the Charge Box as response to a StopTransaction.req PDU.

Field Name	Field Type	Card.	Description
idTagInfo	IdTagInfo	0..1	Optional.This contains information about authorization status, expiry and parent id. It is optional, because a transaction may have been stopped without an identifier.

6.32 StopTransaction.req

This contains the field definition of the StopTransaction.req PDU sent by the Charge Box to the Central System.

Field Name	Field Type	Card.	Description

idTag	string	0..1	Optional.This contains the identifier which requested to stop the charging. It is optional because a charge point may terminate charging without the presence of an id-tag, e.g. in case of a reset.
meterStop	integer	1..1	Mandatory.This contains the meter value in Wh for the connector at end of the transaction.
timestamp	dateTime	1..1	Mandatory.This contains the date and time on which the transaction is stopped.
transactionId	integer	1..1	Mandatory.This contains the transaction-id as received by the StartTransaction.conf.

6.33 UnlockConnector.conf

This contains the field definition of the UnlockConnector.conf PDU sent by the Charge Box to the Central System as response to an UnlockConnector.req PDU.

Field Name	Field Type	Card.	Description
status	UnlockStatus	1..1	Mandatory. This indicates whether the Charge Box has unlocked the connector.

6.34 UnlockConnector.req

This contains the field definition of the UnlockConnector.req PDU sent by the Central System to the Charge Box.

Field Name	Field Type	Card.	Description

connectorId	integer connectorId > 0	1..1	Mandatory.This contains the identifier of the connector to be unlocked.

6.35 UpdateFirmware.conf

This contains the field definition of the UpdateFirmware.conf PDU sent by the Charge Box to the Central System as response to a UpdateFirmware.req PDU.

No fields are defined.

6.36 UpdateFirmware.req

This contains the field definition of the UpdateFirmware.req PDU sent by the Central System to the Charge Box.

Field Name	Field Type	Card.	Description
location	anyURI	1..1	Mandatory.This contains a string containing a URI pointing to a location from which to retrieve the firmware.
retries	integer	0..1	Optional.This specifies how many times Charge Box must try to download the firmware before giving up. If this field is not present, it is left to Charge Box to decide how many times it wants to retry.
retrieveDate	dateTime	1..1	Mandatory.This contains the date and time after which the Charge Box must retrieve the (new) firmware.

retryInterval	integer	0..1	Optional. The interval in seconds after which a retry may be attempted. If this field is not present, it is left to Charge Box to decide how long to wait between attempts.

7 Types

7.1 AuthorizationStatus

Enumeration

Status as response to an Authorize.req.

Field Name	Field Type	Description
Accepted		Identifier is allowed for charging.
Blocked		Identifier has been blocked. Not allowed for charging.
Expired		Identifier has expired. Not allowed for charging.
Invalid		Identifier is unknown. Not allowed for charging.
ConcurrentTx		Identifier is already involved in another transaction and multiple transactions are not allowed. (Only relevant for a StartTransaction.req.)

7.2 AvailabilityStatus

Enumeration

Status returned in response to ChangeAvailability.req.

Field Name	Field Type	Description
Accepted		Request has been accepted and will be executed.
Rejected		Request has not been accepted and will not be executed.

Scheduled		Request has been accepted and will be executed when transaction(s) in progress have finished.

7.3 AvailabilityType

Enumeration

Requested availability change in ChangeAvailability.req.

Field Name	Field Type	Description
Inoperative		Charge point is not available for charging.
Operative		Charge point is available for charging.

7.4 ChargePointErrorCode

Enumeration

Charge Point status reported in StatusNotification.req.

Field Name	Field Type	Description
ConnectorLockFailure		Failure to lock or unlock connector.
HighTemperature		Temperature inside charge point is too high.
Mode3Error		Problem with Mode 3 connection to vehicle.
NoError		No error to report.
PowerMeterFailure		Failure to read power meter.
PowerSwitchFailure		Failure to control power switch.
ReaderFailure		Failure with ID tag reader.

ResetFailure		Unable to perform a reset.

7.5 ChargePointStatus

Enumeration

Status reported in StatusNotification.req.

States considered Operative are: *Available, Occupied.*

States considered Inoperative are: *Unavailable, Faulted.*

Field Name	Field Type	Description
Available		At least one connector is available for charging. (Operative).
Occupied		All connectors of charge point are occupied. (Operative).
Unavailable		Charge point is not available for charging. It has been explicitly set to unavailable. (Inoperative).
Faulted		Charge point has reported an error and is no longer available. (Inoperative).

7.6 ClearCacheStatus

Enumeration

Response to ClearCache.req.

Field Name	Field Type	Description
Accepted		Command has been executed.
Rejected		Command has not been executed.

7.7 ConfigurationStatus

Enumeration

Status in ChangeConfiguration.conf.

Field Name	Field Type	Description
Accepted		Configuration key supported and setting has been changed.
Rejected		Configuration key supported, but setting could not be changed.
NotSupported		Configuration key is not supported.

7.8 DiagnosticsStatus

Enumeration

Status in DiagnosticsStatusNotification.req.

Field Name	Field Type	Description
Uploaded		Diagnostics information has been uploaded.
UploadFailed		Uploading of diagnostics failed.

7.9 FirmwareStatus

Enumeration

Status of a firmware download as reported in FirmwareStatusNotification.req.

Field Name	Field Type	Description
Downloaded		New firmware has been downloaded by charge point.
DownloadFailed		Charge point failed to download firmware.
InstallationFailed		Installation of new firmware has failed.

Installed		New firmware has successfully been installed in charge point.

7.10 IdTagInfo

Class

Contains status information about an identifier. It is returned in Authorization, Start Transaction and Stop Transaction responses.

Field Name	Field Type	Description
expiryDate	dateTime	Optional.This contains the date at which id-tag should be removed from the White List..
parentIdTag	string[15]	Optional.This contains the parent-identifier.
status	AuthorizationStatus	Mandatory.This contains whether the id-tag has been accepted or not by the Central System.

7.11 MeterValue

Class

Collection of meter values in MeterValues.req.

Field Name	Field Type	Description
timestamp	dateTime	Time of meter value.
value	integer	Value in Wh.

7.12 RegistrationStatus

Enumeration

Result of registration in response to BootNotification.req.

Field Name	Field Type	Description
Accepted		Charge point is accepted by Central System.
Rejected		Charge point is not accepted by Central System. This happens when the charge point id is not known by Central System.

7.13 RemoteStartStopStatus

Enumeration

The result of a RemoteStartTransaction.req or RemoteStopTransaction.req request.

Field Name	Field Type	Description
Accepted		Command will be executed.
Rejected		Command will not be executed.

7.14 ResetStatus

Enumeration

Result of Reset.req.

Field Name	Field Type	Description
Accepted		Command will be executed.
Rejected		Command will not be executed.

7.15 ResetType

Enumeration

Type of reset requested by Reset.req.

Field Name	Field Type	Description

Hard		Full reboot of charge box software.
Soft		Return to initial status, gracefully terminating any transactions in progress.

7.16 UnlockStatus

Enumeration

Status in response to UnlockConnector.req.

Field Name	Field Type	Description
Accepted		Command will be executed.
Rejected		Command will not be executed.

8 Binding to Transport Protocol

This section describes how the OCPP PDUs can be conveyed over SOAP.

The rationale behind using SOAP as a transport is that SOAP already provides the infrastructure of sending messages. SOAP has a good support in the industry, which results in tools that improve the ease of implementing the protocol.

The used version of SOAP MUST be 1.2. See [SOAP].

8.1 Charge Box Identity

To be able for the Central System to identify uniquely a Charge Box, a Charge Box MUST send in each request PDU, its identifier in the SOAP header. The header name is "chargeBoxIdentity" of XSD type "string". For example:

```
<!-- Header containing the identifier of the sending Charge Box -->
<ns:chargeBoxIdentity>CB1234</ns:chargeBoxIdentity>
```

When a Central System needs to send requests to a Charge Box, the Central System MUST send in each request the "chargeBoxIdentity" for which Charge Box the request is intended. If the receiving Charge Box is not the intended one, then the Charge Box MUST send a SOAP Fault Response message, indicating that the identity is wrong (e.g. sub-code is "IdentityMismatch").

8.2 Fault Response

In cases where the receiving party (e.g. Charge Box or Central System) cannot process the request and the corresponding confirmation PDU doesn't have to ability to report the error, then the SOAP Fault Response Message SHOULD be used. This can be used, for instance, when the operation is not implemented or when an internal error has occurred.

The following fault codes can be used by the service:

Value belongs to the namespace "http://www.w3.org/2003/05/soap-envelope".

SubCode belongs to the namespace "urn://Ocpp/2010/08/".

- *Value* = Sender*, SubCode* = SecurityError; Sender failed authentication or is not authorized to use the requested operation.
- *Value* = Sender*, SubCode* = IdentityMismatch; Sender sent the wrong identity value.
- *Value* = Sender, *SubCode* = ProtocolError; Sender's message does not comply with protocol specification.
- *Value* = Receiver*, SubCode* = InternalError; An internal error occurred and the receiver is not able to complete the operation.
- *Value* = Receiver, *SubCode* = NotSupported; The receiver does not support the requested operation.

8.3 Mobile Networks

In cases where Charge Boxes uses an IP based mobile data network (GPRS, UMTS, HSPDA, etc.) to communicate with the Central System, it's possible that the Central System cannot initiate a network connection to a Charge Box directly. For example, a Network Operator may use dynamic IP addresses.

To overcome this problem, a Central System SHOULD send a SMS to a Charge Box. The SMS SHOULD contain the Charge Box identifier, but the SMS MAY also be empty. How an SMS is sent to an SMSC and delivered to the Charge Box is outside the scope of this specification. Upon receipt of a SMS, the Charge Box MUST send a Heartbeat.req PDU to the Central System with WS-Addressing header "From", with the URL of the Charge Box. See [WS-ADDR] for a description of the Web Service Addressing standard. The Central System SHOULD use this URL for sending a pending command to.

When Charge Boxes are deployed in a mobile network, the Charge Box SHOULD set its URL in the WS-Addressing "From" field when sending a request. This way a Central System can first try sending commands to the supplied URL, before waking up the Charge Box via SMS.

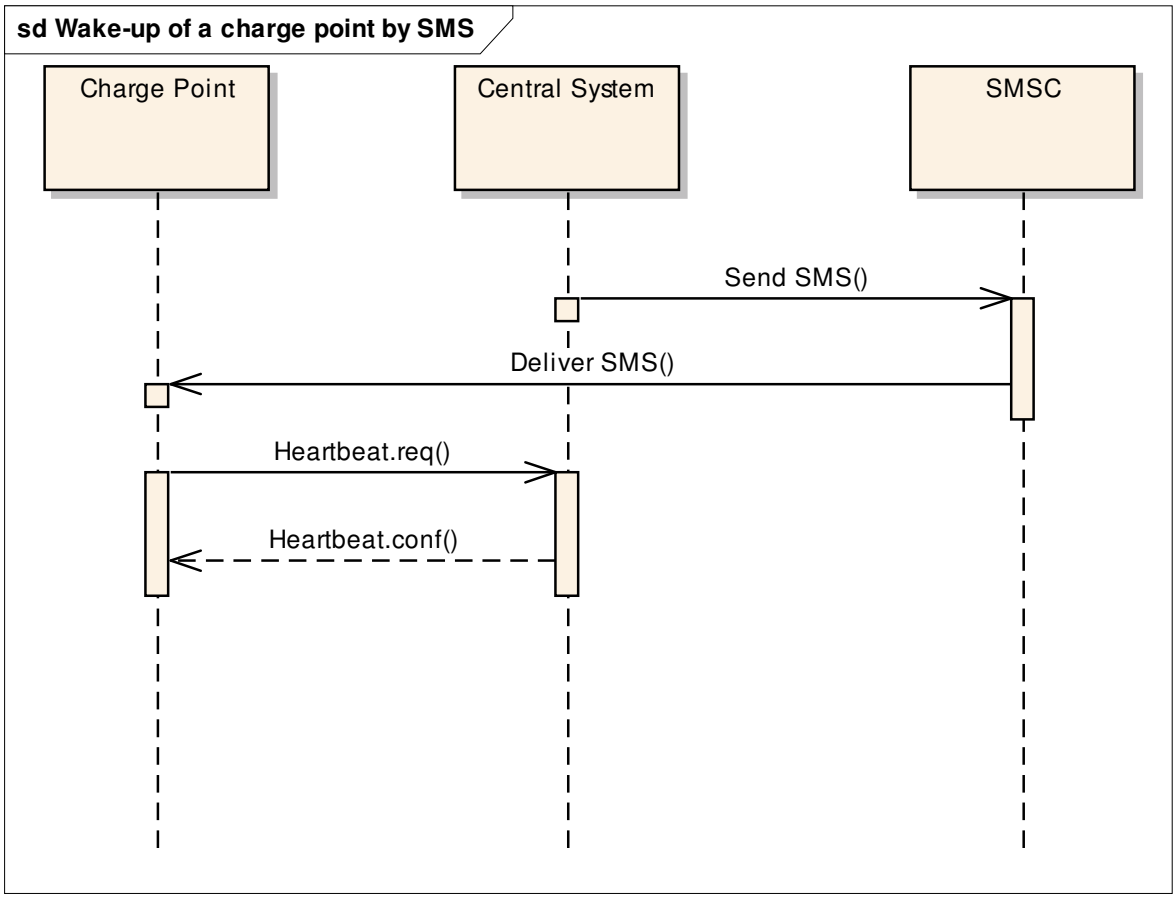


Figure: Wake-up of a charge point by SMS

8.4 Download Firmware

When a Charge Box is notified about new firmware, it needs to be able to download this firmware. The Central System supplies in the request an URL where the Charge Box can be downloaded. The URL also contains the protocol which must be used to download the firmware.

The firmware MUST be downloaded via FTP or FTPS. FTP(S) is a better optimized for large binary data then HTTP. Also FTP(S) has the ability to resume downloads. In case a download is interrupted, the Charge Box can resume downloading after the part it already has downloaded. The FTP URL is of format: *ftp://user*:*password*@*host*:*port*/*path* in which the parts *user*:*password*@, :*password* or :*port* may be excluded.

To ensure that the correct firmware is downloaded, it is RECOMMENDED that the firmware is also digitally signed.

8.5 Upload Diagnostics

When a Charge Box is requested to upload a diagnostics file, the Central System supplies in the request an URL where the Charge Box should upload the file. The URL also contains the protocol which must be used to upload the file.

The diagnostics file MUST be downloaded via FTP or FTPS. FTP is a better optimized for large binary data then HTTP. Also FTP has the ability to resume downloads. In case an upload is interrupted, the Charge Box can resume uploading after the part it already has uploaded. The FTP URL is of format: *ftp://user*:*password*@*host*:*port*/*path* in which the parts *user*:*password*@, :*password* or :*port* may be excluded.

8.6 Compression

In cases where bandwidth needs to be reduced, a communicating party can use of the HTTP capability of compressing data. HTTP defines a 'Content-Encoding' header, which contains the compression method.

The Charge Box and Central System MAY use HTTP compression. Compression can be performed on a HTTP request and/or response.

The Charge Box and Central System MUST support the 'gzip' and 'deflate' compression methods. These are the most common compression methods.

When using compression, one should take great care if indeed the message size decreases. If the message is small, then it's possible that the compression will increase the size.

8.7 Security

To avoid exposure of private sensitive data, the transport of SOAP messages SHOULD be secured with SSL/TLS (e.g. HTTPS).

For a receiving party to trust a received message, the sending party SHOULD use a client certificate.

8.8 WSDL

The versions of the WSDL descriptions corresponding to this OCPP description are:

- Central System service: urn://Ocpp/Cs/2010/10/
- Charge Point service: urn://Ocpp/Cp/2010/10/