OPEN
CHARGE
ALLIANCE

OCPP 1.6 Errata sheet

v3.0 Release, 2017-09-08

Errata sheet: for OCPP version: 1.6 FINAL.

Version History

VERSION	DATE	AUTHOR	DESCRIPTION
v3.0 Release	2017-09-08	Robert de Leeuw (IHomer)	3th release of the errata sheet New errata are marked with v3.0.
v2.0 Release	2017-03-27	Robert de Leeuw (IHomer) Klaas van Zuuren (ELaadNL) Brendan McMahon (ESB)	2nd release of the errata sheet Erratas have been reordered to match the chronological order of the 1.6 specification.
v1.0 Release	2016-03-31	Robert de Leeuw (IHomer) Klaas van Zuuren (ELaadNL)	First release

1. Scope

This document contains erratas on the OCPP 1.6 FINAL specification.

1.1. Terminology and Conventions

Bold: when needed to clarify differences, bold text might be used.

Since document version v3.0 errata are marked with a version number, indicating when an errata was added.

2. Major erratas

Problems with the content/definition of the messages, class and enumerations of the protocol.

Non known

3. Minor erratas

Improvements to the descriptions on how the protocol (should) work.

3.1. Page 7, section: 2.2: More than IEC 15118 might be taken into account for Smart Charging.

The definition of "Composite Charging Schedule" states that "IEC 15118 limits might be taken into account", but also other limits might be taken into account.

Old text	Also IEC 15118 limits might be taken into account.
New text	Local Limits might be taken into account.

3.2. Page 10, section: 3.2: Feature Profiles should be normative

The Feature Profiles paragraph states that it is "informative", but it is "normative".

Old text	This section is **informative**.
New text	This section is **normative**.

3.3. Page 18, section 3.5: Better description how to determine start/end of Energy Transfer Period

NEW: errata sheet v3.0

Description how the Central System can determine start/end of Energy Transfer Period can be improved.

Old text	A Central System MAY deduce the start and end of an Energy Transfer Period from the MeterValues that are sent during the Transaction.

New text	A Central System MAY deduce the start and end of an Energy Transfer Period from: the MeterValues that are sent during the Transaction, the status notifications: Charging, SuspendedEV and/or SuspendedEVSE. etc. Central System implementations need to take into account factors such as: Some EVs don't go to state SuspendedEV: they might continue to trickle charge. Some Charge Point don't even have a electrical meter.

3.4. Page 18, section 3.12.2: Add description of stacking without duration

NEW: errata sheet v3.0

When using stacking for Smart Charging: a high stack level without a duration will cause lower profiles to be never executed.

Add note at end of section 3.12.2

New text	NOTE: If you use Stacking without a duration, on the highest stack level, the Charge Point will never fall back to a lower stack level profile."

3.5. Page 33, new Chapter about time notations

In addition to "3.13 Time Zones" to following should have been added about time notations:

New text	**3.14 Time notations** Implementations MUST use ISO 8601 date time notation. Message receivers must be able to handle fractional seconds and time zone offsets (another implementation might use them). Message senders MAY save data usage by omitting insignificant fractions of seconds.

3.6. Page 35, section: 4.2: Boot Notification: Note on behaviour while not accepted by Central system

Old text	While not yet accepted by the Central System, the Charge Point may allow locally authorized transactions if it is configured to do so, as described in Local Authorization & Offline Behaviour. Parties who want to implement this behaviour must realize that it is uncertain if those transactions can ever be delivered to the Central System.
New text	A Charge Point Operator MAY choose to configure a Charge Point to accept transactions before the Charge Point is accepted by a Central System. Parties who want choose to implement this such behavior should realize that it is uncertain if those transactions can ever be delivered to the Central System. After a restart (for instance due to a remote reset command, power outage, firmware update, software error etc.) the Charge Point MUST again contact the Central System and SHALL send a BootNotification request. If the Charge Point fails to receive a BootNotification.conf from the Central System, and has no in-built non-volatile real-time clock hardware that has been correctly preset, the Charge Point may not have a valid date / time setting, making it impossible to later determine the date / time of transactions. It might also be the case (e.g. due to configuration error) that the Central System indicates a status other than Accepted for an extended period of time, or indefinitely. It is usually advisable to deny all charging services at a Charge Point if the Charge Point has never before been Accepted by the Central System (using the current connection settings, URL, etc.) since users cannot be authenticated and running transactions could conflict with provisioning processes.

3.7. Page 37, section: 4.5: Relation between FirmwareStatusNotification.req and FirmwareUpdate.req is missing.

There is no description about the relation between FirmwareUpdate.req and FirmwareStatusNotification.req in the specification.

Additional text	The FirmwareStatusNotification.req PDUs SHALL be sent to keep the Central System updated with the status of the update process, started by the Central System with a FirmwareUpdate.req PDU.

3.8. Page 38, section: 4.7: Missing description of configuration keys for MeterValues

There are a couple of required configuration keys that have a huge influence of how MeterValues work. But there is no reference or description for them in 4.7 Meter Values:

- ClockAlignedDataInterval
- MeterValuesAlignedData
- MeterValuesAlignedDataMaxLength
- MeterValuesSampledData
- MeterValuesSampledDataMaxLength
- MeterValueSampleInterval
- StopTxnAlignedData
- StopTxnAlignedDataMaxLength
- StopTxnSampledData
- StopTxnSampledDataMaxLength

New Chapter to be added to the specification:

3.14: Metering Data

This section is normative.

Extensive metering data relating to charging sessions can be recorded and transmitted in different ways depending on its intended purpose. There are two obvious use cases (but the use of meter values is not limited to these two):

- Charging Session Meter Values
- Clock-Aligned Meter Values

Both types of meter readings MAY be reported in standalone MeterValues.req messages (during a transaction) and/or as part of the *transactionData* element of the StopTransaction.req PDU.

3.14.1 Charging Session Meter Values

Frequent (e.g. 1-5 minute interval) meter readings taken and transmitted (usually in "real time") to the Central System, to allow it to provide information updates to the EV user (who is usually not at the charge point), via web, app, SMS, etc., as to the progress of the charging session. In OCPP, this is called "sampled meter data", as the exact frequency and time of readings is not very significant, as long as it is "frequent enough". "Sampled meter data" can be configured with the following configuration keys:

- MeterValuesSampledData
- MeterValuesSampledDataMaxLength
- MeterValueSampleInterval
- StopTxnSampledData

- StopTxnSampledDataMaxLength

MeterValueSampleInterval is the time (in seconds) between sampling of metering (or other) data, intended to be transmitted by "MeterValues" PDUs. Samples are acquired and transmitted periodically at this interval from the start of the charging transaction.

A value of "0" (numeric zero), by convention, is to be interpreted to mean that no sampled data should be transmitted.

MeterValuesSampledData is a comma separated list that prescribes the set of measurands to be included in a MeterValues.req PDU, every MeterValueSampleInterval seconds. The maximum amount of elements in the MeterValuesSampledData list can be reported by the Charge Point via: **MeterValuesSampledDataMaxLength**

StopTxnSampledData is a comma separated list that prescribes the sampled measurands to be included in the TransactionData element of StopTransaction.req PDU, every MeterValueSampleInterval seconds from the start of the charging session. The maximum amount of elements in the StopTxnSampledData list can be reported by the Charge Point via: **StopTxnSampledDataMaxLength**

3.14.2 Clock-Aligned Meter Values

Grid Operator might require meter readings to be taken from fiscally certified energy meters, at specific Clock aligned times (usually every quarter hour, or half hour).

"Clock-Aligned Billing Data" can be configured with the following configuration keys:

- ClockAlignedDataInterval
- MeterValuesAlignedData
- MeterValuesAlignedDataMaxLength
- StopTxnAlignedData
- StopTxnAlignedDataMaxLength

ClockAlignedDataInterval is the size of the clock-aligned data interval (in seconds). This defines the set of evenly spaced meter data aggregation intervals per day, starting at 00:00:00 (midnight).

For example, a value of 900 (15 minutes) indicates that every day should be broken into 96 15-minute intervals.

A value of "0" (numeric zero), by convention, is to be interpreted to mean that no clock-aligned data should be transmitted.

MeterValuesAlignedData is a comma separated list that prescribes the set of measurands to be included in a MeterValues.req PDU, every ClockAlignedDataInterval seconds. The maximum amount of elements in the MeterValuesAlignedData list can be reported by the Charge Point via: **MeterValuesAlignedDataMaxLength**

StopTxnAlignedData is a comma separated list that prescribes the set of clock-aligned periodic measurands to be included in the TransactionData element of StopTransaction.req PDU for every ClockAlignedDataInterval of the charging session. The maximum amount of elements in the StopTxnAlignedData list can be reported by the Charge Point via: **StopTxnAlignedDataMaxLength**

3.14.3 Multiple Locations/Phases

When a Charge Point can measure the same measurand on multiple locations or phases, all possible locations and/or phases SHALL be reported when configured in one of the relevant configuration keys.

For example: A Charge Point capable of measuring *Current.Import* on *Inlet* (all 3 phases) (grid connection) and *Outlet* (3 phases per connector on both its connectors). *Current.Import* is set in MeterValuesSampledData. MeterValueSampleInterval is set to 300 (seconds). Then the Charge Point should send:

- a MeterValue.req with: connectorId = 0; with 3 SampledValue elements, one per phase with location = Inlet.
- a MeterValue.req with: connectorId = 1; with 3 SampledValue elements, one per phase with location = Outlet.
- a MeterValue.req with: connectorId = 2; with 3 SampledValue elements, one per phase with location = Outlet.

3.14.4 Unsupported measurands

When a Central System sends a ChangeConfiguration.req to a Charge Point with one of the following configuration keys:

- MeterValuesAlignedData
- MeterValuesSampledData
- StopTxnAlignedData
- StopTxnSampledData

If the comma separated list contains one or more measurands that are not supported by this Charge Point, the Charge Point SHALL respond with: ChangeConfiguration.conf with: status = Rejected. No changes SHALL be made to the currently configuration.

NEW: errata sheet v3.0

3.14.5 No metering data in a Stop Transaction

When the configuration keys: **StopTxnAlignedData** and **StopTxnSampledData** are set to an empty string, the Charge Point SHALL not put meter values in a StopTransaction.req PDU.

3.9. Page 41, section: 4.9: Status transition from Preparing to Finishing (B6) is possible

NEW: errata sheet v3.0

The original 1.6 does not allow a transition from Preparing to Finishing (B6). But there is a use case where this is possible:

A Charge Point with 2 connectors and 1 RFID reader. Driver 1 connects his charging cable. State goes to "Preparing" User does not authorize, times out. Then the Charge Point has to go to state: "Finishing". It should NOT go to "Available". Because if the charging cable remains plugged in and another EV driver swipes his RFID (before plugging in), that would start a transaction on the already plugged in cable. By going to "Finishing" this is prevented.

Table top page 41	Add **B6**
Table on page 42	B6: Timed out. Usage was initiated (e.g. insert plug, bay occupancy detection), but idTag not presented within timeout.

3.10. Page 42, section: 4.9 (B1): Missing reference to configuration key: "ConnectionTimeOut"

The configuration key: "ConnectionTimeOut" is never referenced in the specification, section: 4.9 needs a note explaining how the state: "Preparing" and the configuration key: "ConnectionTimeOut" work together.

Old text	Intended usage is ended (e.g. plug removed, bay no longer occupied, second presentation of idTag, time out on expected user action)
New text	Intended usage is ended (e.g. plug removed, bay no longer occupied, second presentation of idTag, time out **(configured by the configuration key: ConnectionTimeOut)** on expected user action)

3.11. Page 50, section: 5.7: More than IEC 15118 might be taken into account for Smart Charging.

It is stated that "IEC 15118 limits might be taken into account", but also other limits might be taken into account.

Old text	Also IEC 15118 limits might be taken into account.
New text	Local Limits might be taken into account.

3.12. Page 51, section: 5.7: Get Composite Schedule: First sentences is not clear

In the description of Get Composite Schedule, it is not clear, what are the start and end points in time of the schedule that is to be sent.

Old text	Upon receipt of a GetCompositeSchedule.req, the Charge Point SHALL calculate the **scheduled time intervals up to the Duration is met and send them** to the central system.
New text	Upon receipt of a GetCompositeSchedule.req, the Charge Point SHALL calculate the **Composite Charging Schedule intervals, from the moment the request PDU is received: Time X, up to X + Duration, and send them in the GetCompositeSchedule.conf PDU** to the central system.

3.13. Page 51, section: 5.7: Improve use of connectorId '0' in GetCompositeSchedule

In the description of the use of connectorId '0' in GetCompositeSchedule it is not clear that it can also mean the Charge Point reports current instead of power.

Old text	If the ConnectorId in the request is set to '0', the Charge Point SHALL report the total expected **energy flow of the Charge Point for** the requested time period.
New text	If the ConnectorId in the request is set to '0', the Charge Point SHALL report the total expected **power or current the Charge Point expects to consume from the grid during** the requested time period.

3.14. Page 51, section: 5.9: Relation between GetDiagnostics.req and DiagnosticsStatusNotification.req is missing.

There is no description about the relation between GetDiagnostics.req and DiagnosticsStatusNotification.req in the specification.

The following diagram should replace the diagram in 5.9.

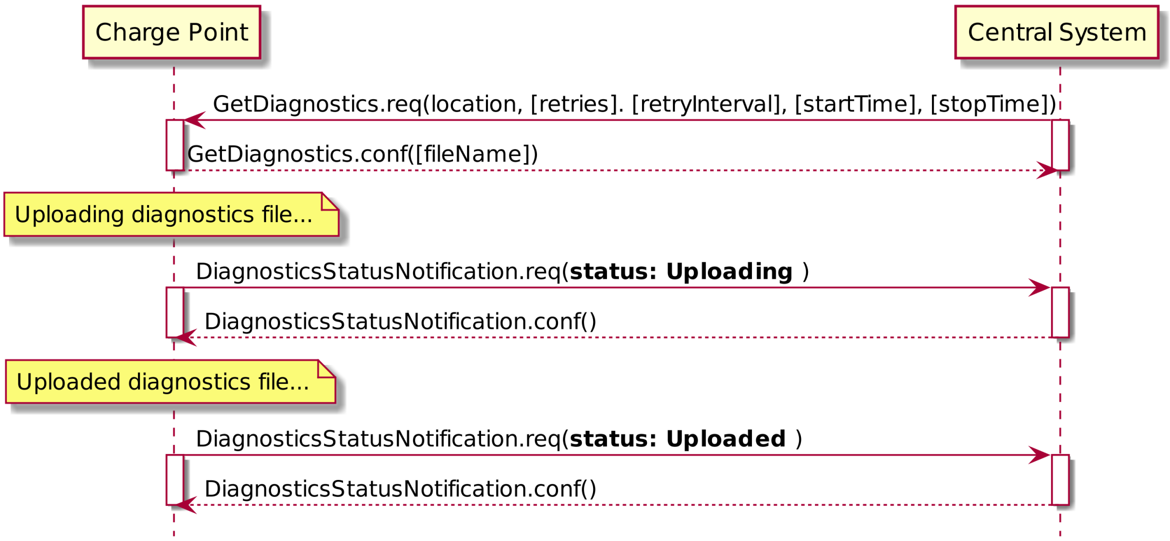

Charge Point
Central System
GetDiagnostics.req(location, [retries]. [retryInterval], [startTime], [stopTime])
GetDiagnostics.conf([fileName])
Uploading diagnostics file...
DiagnosticsStatusNotification.req(**status: Uploading**)
DiagnosticsStatusNotification.conf()
Uploaded diagnostics file...
DiagnosticsStatusNotification.req(**status: Uploaded**)
DiagnosticsStatusNotification.conf()

Figure 1. Sequence Diagram: get diagnostics

Additional text	During uploading of a diagnostics file, the Charge Point MUST send DiagnosticsStatusNotification.req PDUs to keep the Central System updated with the status of the upload process.

3.15. Page 52, section: 5.11. Remote Start Transaction: .conf status is accepted request

Old text	Central System can request a Charge Point to start a transaction by sending a RemoteStartTransaction.req. Upon receipt, the Charge Point SHALL reply with RemoteStartTransaction.conf and a status indicating whether it **is able** to start a transaction **or not**.
New text	Central System can request a Charge Point to start a transaction by sending a RemoteStartTransaction.req. Upon receipt, the Charge Point SHALL reply with RemoteStartTransaction.conf and a status indicating whether it **has accepted the request and will attempt** to start a transaction.

3.16. page 54, section: 5.12. Remote Stop Transaction: .conf status is accepted request

Old text	RemoteStopTransaction.req to Charge Point with the identifier of the transaction. Charge Point SHALL reply with RemoteStopTransaction.conf **to indicate whether it is indeed able to stop the transaction.**
New text	RemoteStopTransaction.req to Charge Point with the identifier of the transaction. Charge Point SHALL reply with RemoteStopTransaction.conf **and a status indicating whether it has accepted the request and a transaction with the given transactionId is ongoing and will be stopped.**

3.17. Page 56, section: 5.14: Improved description of Soft/Hard Reset

NEW: errata sheet v3.0

The descriptions of Soft/Hard reset can be improved, not very clear what is the difference between the 2.

Additional text, between par 1 & 2	After receipt of a Reset.req, The Charge Point SHALL send a StopTransaction.req for any ongoing transaction before performing the reset. If the Charge Point fails to receive a StopTransaction.conf form the Central System, it shall queue the StopTransaction.req.

Old text	At receipt of a soft reset, the Charge Point SHALL return to a state that behaves as just having been booted. If any transaction is in progress it SHALL be terminated normally, before the reset, as in Stop Transaction.
New text	At receipt of a soft reset, the Charge Point SHALL stop ongoing transactions gracefully and send StopTransaction.req for every ongoing transaction. It should then restart the application software (if possible, otherwise restart the processor/controller).

Old text	At receipt of a hard reset the Charge Point SHALL attempt to terminate any transaction in progress normally as in StopTransaction and then perform a reboot.
New text	At receipt of a hard reset the Charge Point SHALL restart (all) the hardware, it is not required to gracefully stop ongoing transaction. If possible the Charge Point sends a StopTransaction.req for previously ongoing transactions after having restarted and having been accepted by the Central System via a BootNotification.conf. This is a last resort solution for a not correctly functioning Charge Points, by sending a "hard" reset, (queued) information might get lost.

3.18. Page 58, section: 5.16.2: RemoteStart with ChargingProfile: TransactionId should not be set

It is not clear that in a remoteStartTransaction.req with ChargingProfile, the transactionId should not be set.

Old text	If the Central System includes a ChargingProfile, the ChargingProfilePurpose MUST be set to TxProfile.
New text	If the Central System includes a ChargingProfile, the ChargingProfilePurpose MUST be set to TxProfile **and the transactionId SHALL NOT be set**.

3.19. Page 58, section: 5.16.2: Meaning of note on RemoteStart with ChargingProfile is not clear

In the description of RemoteStartTransaction with a ChargeProfile there is a note, but the meaning of the note is not clear.

Old text	The Charge Point SHOULD add the TransactionId to the received profile once the transaction is reported to the central system.
New text	The Charge Point SHALL apply the given profile to the newly started transaction. This transaction will get a transactionId assigned by Central System via a startTransaction.conf. When the Charge Point receives a setChargingProfile.req, with the transactionId for this transaction, with the same StackLevel as the profile given in the remoteStartTransaction.req, the Charge Point SHALL replace the existing charging profile, otherwise it SHALL install/stack the profile next to the already existing profile(s).

3.20. Page 58, Section 5.16.4: Smart Charging fall back to default unclear

NEW: errata sheet v3.0

Old text	When recurrencyKind is used in combination with a chargingSchedule duration shorter than the recurrencyKind period, the Charge Point SHALL fall back to default behavior after the chargingSchedule duration ends.
New text	When recurrencyKind is used in combination with a chargingSchedule duration shorter than the recurrencyKind period, the Charge Point SHALL fall back to default behavior after the chargingSchedule duration ends. **This fall back means that the Charge Point SHALL use a ChargingProfile with a lower stackLevel if available. If no other ChargingProfile is available, the Charge Point SHALL allow charging as if no ChargingProfile is installed. If the chargingSchedulePeriod and/or duration is longer then the recurrence period duration, the remainder periods SHALL not be executed.**

3.21. Page 58, Section 5.16.4: Not defined what to do with a charging schedule period longer then recurrence.

NEW: errata sheet v3.0

It is not defined what to do with a chargingSchedulePeriod and/or duration that is longer then the recurrence period.

Add the following note after the NOTE about "chargingSchedulePeriod longer than duration"

New text	NOTE: When recurrencyKind is used in combination with a chargingSchedulePeriod and/or duration that is longer then the recurrence period duration, the remainder periods SHALL not be executed.

3.22. Page 58, Section 5.16.4: First ChargingSchedulePeriod should start with StartSchedule = 0

NEW: errata sheet v3.0

The obvious is not defined: The first ChargingSchedulePeriod StartSchedule in a ChargingSchedule should be 0.

Add the following note after the NOTE about "chargingSchedulePeriod longer than duration"

New text	NOTE: The StartSchedule of the first ChargingSchedulePeriod in a ChargingSchedule SHALL always be 0.

3.23. Page 60, section: 5.17: Description TriggerMessage for MeterValues not clear

The description of what a Charge Point should do when it receives a TriggerMessage.req PDU with requestedMessage: MeterValues, is not clear.

Old text	A MeterValues message triggered in this way for instance **SHOULD** return the most recent measurements for all measurands configured in configuration key MeterValuesSampledData.
New text	A MeterValues message triggered in this way for instance **SHALL** return the most recent measurements for all measurands configured in configuration key MeterValuesSampledData.

3.24. Page 61, section: 5.19: Relation between FirmwareUpdate.req and FirmwareStatusNotification.req is missing.

There is no description about the relation between FirmwareUpdate.req and FirmwareStatusNotification.req in the specification. In paragraph 3.3 on page 15, there is a more elaborate diagram of firmware update, including the relationship between UpdateFirmware.req and FirmwareStatusNotification.req, but that paragraph is informative.

The following diagram should replace the diagram in 5.19.

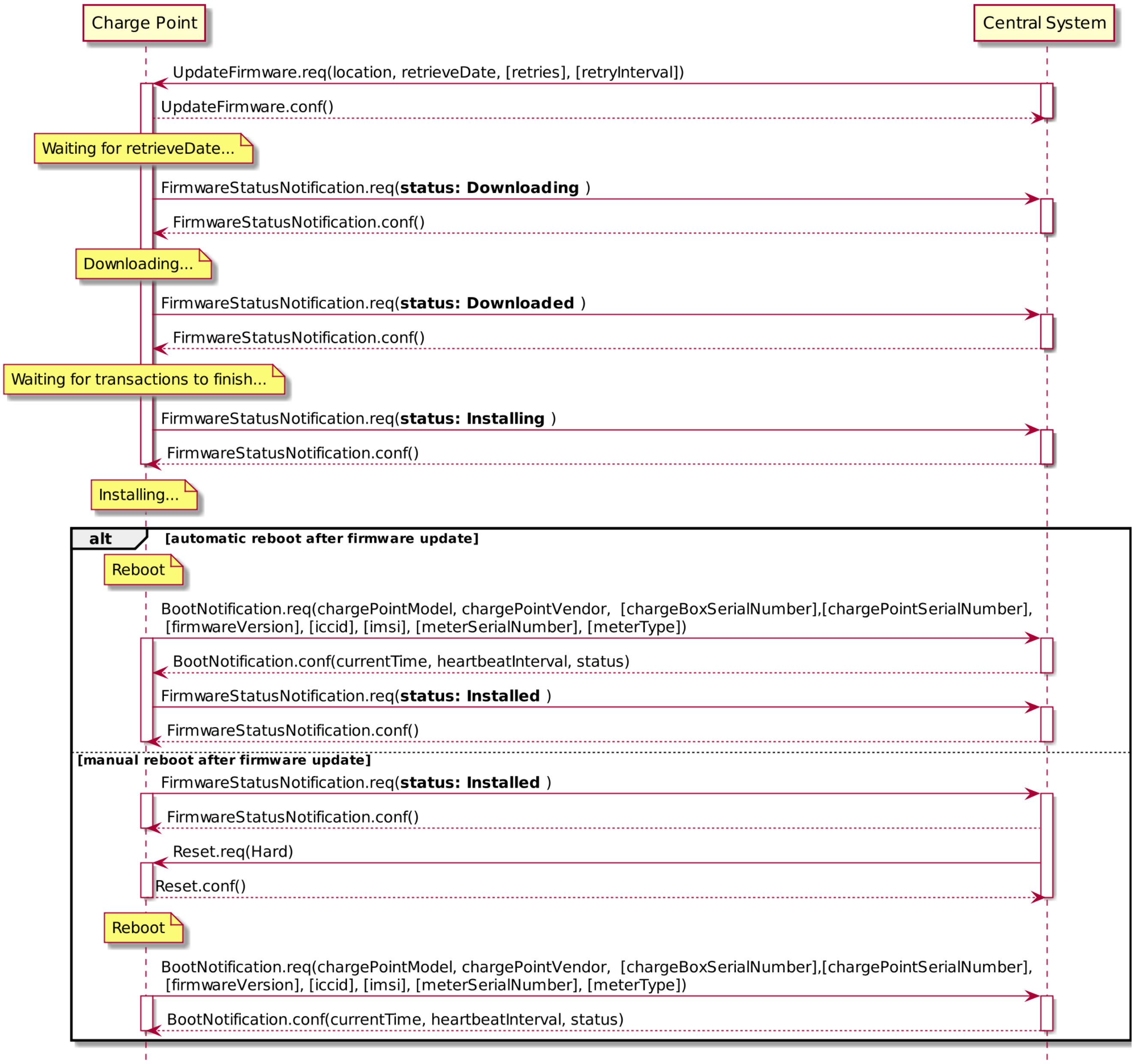

Charge Point
Central System
UpdateFirmware.req(location, retrieveDate, [retries], [retryInterval])
UpdateFirmware.conf()
Waiting for retrieveDate...
FirmwareStatusNotification.req(status: Downloading)
FirmwareStatusNotification.conf()
Downloading...
FirmwareStatusNotification.req(status: Downloaded)
FirmwareStatusNotification.conf()
Waiting for transactions to finish...
FirmwareStatusNotification.req(status: Installing)
FirmwareStatusNotification.conf()
Installing...
alt
[automatic reboot after firmware update]
Reboot
BootNotification.req(chargePointModel, chargePointVendor, [chargeBoxSerialNumber],[chargePointSerialNumber], [firmwareVersion], [iccid], [imsi], [meterSerialNumber], [meterType])
BootNotification.conf(currentTime, heartbeatInterval, status)
FirmwareStatusNotification.req(status: Installed)
FirmwareStatusNotification.conf()
[manual reboot after firmware update]
FirmwareStatusNotification.req(status: Installed)
FirmwareStatusNotification.conf()
Reset.req(Hard)
Reset.conf()
Reboot
BootNotification.req(chargePointModel, chargePointVendor, [chargeBoxSerialNumber],[chargePointSerialNumber], [firmwareVersion], [iccid], [imsi], [meterSerialNumber], [meterType])
BootNotification.conf(currentTime, heartbeatInterval, status)

Figure 2. Sequence Diagram: firmware update

Additional text	During downloading and installation of the firmware, the Charge Point MUST send FirmwareStatusNotification.req PDUs to keep the Central System updated with the status of the update process. The sequence diagram above is an example. It is good practice to first reboot the Charge Point to check the new firmware is booting and able to connect to the Central System, before sending the status: **Installed**. It is not a requirement.

3.25. Page 61, section: 5.19: No description new firmware should be installed.

There is no requirement on the installation of new firmware.

Additional text	The Charge Point SHALL, if the new firmware image is "valid", install the new firmware as soon as it is able to.

3.26. page 61, section: 5.19. Firmware installation during charging session

It is advised to not stop charging session to install new firmware, but wait until session has ended.

Additional text	If it is not possible to continue charging during installation of firmware, it is RECOMMENDED to wait until Charging Session has ended (Charge Point idle) before commencing installation. It is RECOMMENDED to set connectors that are not in use to UNAVAILABLE while the Charge Point waits for the Session to end.

This errata note is also applicable for OCPP versions 1.2 and 1.5.

3.27. Page 70, section: 6.22: Unclear when to use the fields: scheduleStart and chargingSchedule in GetCompositeSchedule.conf.

The message: GetCompositeSchedule.conf contains 3 optional fields, for two of these fields: "scheduleStart" and "chargingSchedule" it is not clear when they should or should not be used:

Add to the description the field: scheduleStart	If status is "Rejected", this field may be absent.
Add to the description the field: chargingSchedule	If status is "Rejected", this field may be absent.

3.28. Page 84, section: 6.55: Description for UpdateFirmware.req: retrieveDate is ambiguous

The description of the field: "retrieveDate" is ambiguous. It should not be "must", but: "is allowed to"

Old text	Required. This contains the date and time after which the Charge Point **must** retrieve the (new) firmware.
New text	Required. This contains the date and time after which the Charge Point **is allowed to** retrieve the (new) firmware.

3.29. Page 88, section: 7.7: Description SuspendedEVSE and SuspendedEV to strict, not all chargers have a contactor

Descriptions for SuspendedEVSE and SuspendedEV seems to imply that an EVSE has a contactor, but that is not always the case, for example: wireless charging.

SuspendedEVSE	Old text	When the contactor of a Connector opens upon request of the EVSE, e.g. due to a smart charging restriction or as the result of StartTransaction.conf indicating that charging is not allowed (Operative)
SuspendedEVSE	New text	When the EV is connected to the EVSE but the EVSE is not offering energy to the EV, e.g. due to a smart charging restriction, local supply power constraints, or as the result of StartTransaction.conf indicating that charging is not allowed etc. (Operative)
SuspendedEV	Old text	When the EVSE is ready to deliver energy but contactor is open, e.g. the EV is not ready.
SuspendedEV	New text	When the EV is connected to the EVSE and the EVSE is offering energy but the EV is not taking any energy. (Operative)

3.30. Page 90, section: 7.8: validFrom fields are allowed for TxProfiles.

In the class definition of ChargingProfile, the field: "validFrom", is defined as: "Not to be used when ChargingProfilePurpose is TxProfile." The specification denotes that the field ValidTo and ValidFrom are not to be used in combination with profiletype TxProfile. This note should have been deleted in the final version. With the decision to support stacking in combination with ProfileType TxProfile, the use of ValidFrom and ValidTo fields is unavoidable, since otherwise the profile with the highest StackLevel will be active until it is uninstalled.

Old text	Optional. Point in time at which the profile starts to be valid. If absent, the profile is valid as soon as it is received by the Charge Point. **Not to be used when ChargingProfilePurpose is TxProfile.**
New text	Optional. Point in time at which the profile starts to be valid. If absent, the profile is valid as soon as it is received by the Charge Point.

3.31. Page 91, section: 7.8: validTo fields are allowed for TxProfiles.

In the class definition of ChargingProfile, the field: "validTo", is defined as: "Not to be used when ChargingProfilePurpose is TxProfile." The specification denotes that the field ValidTo and ValidFrom are not to be used in combination with profiletype TxProfile. This note should have been deleted in the final version. With the decision to support stacking in combination with ProfileType TxProfile, the use of ValidFrom and ValidTo fields is unavoidable, since otherwise the profile with the highest profile will be active until it is uninstalled.

Old text	Optional. Point in time at which the profile stops to be valid. If absent, the profile is valid until it is replaced by another profile. **Not to be used when ChargingProfilePurpose is TxProfile.**
New text	Optional. Point in time at which the profile stops to be valid. If absent, the profile is valid until it is replaced by another profile.

3.32. Page 91, section: 7.10: Description of TxProfile/TxDefaultProfile in ChargingProfilePurposeType in relation with RemoteStartTransaction unclear

It is not completely clear what the correct ProfilePurpose should be in a remoteStartTransaction.req.

TxDefaultProfile	Old text	Default profile **to be used for new transactions**.
TxDefaultProfile	New text	Default profile **that can be configured in the Charge Point. When a new transaction is started, this profile SHALL be used, unless it was a transaction that was started by a remoteStartTransaction.req with a ChargeProfile that is accepted by the Charge Point**.
TxProfile	Old text	Profile with constraints to be imposed by the Charge Point on the current transaction. A profile with this purpose SHALL cease to be valid when the transaction terminates.
TxProfile	New text	Profile with constraints to be imposed by the Charge Point on the current transaction **, or on a new transaction when this is started via a RemoteStartTransaction.req with a ChargeProfile**. A profile with this purpose SHALL cease to be valid when the transaction terminates.

3.33. Page 92, section: 7.12: ChargingRateUnit value descriptions need more clarification

Using a ChargingRateUnit W for AC charging is potentially very complicated, and, if used, the calculation is tricky The description of the values of the ChargingRateUnit should be improved.

VALUE	OLD DESCRIPTION	NEW DESCRIPTION
W	Watts (power).	Watts (power). This is the TOTAL allowed charging power. If used for AC Charging, the phase current should be calculated via: Current per phase = Power / (Line Voltage * Number of Phases). The "Line Voltage" used in the calculation is not the measured voltage, but the set voltage for the area (hence, 230 of 110 volt). The "Number of Phases" is the numberPhases from the ChargingSchedulePeriod. It is usually more convenient to use this for DC charging. Note that if numberPhases in a ChargingSchedulePeriod is absent, 3 SHALL be assumed.
A	Amperes (current).	Amperes (current). The amount of Ampere per phase, not the sum of all phases. It is usually more convenient to use this for AC charging.

3.34. Page 93, section: 7.13: First ChargingSchedulePeriod should start with StartSchedule = 0

NEW: errata sheet v3.0

There is no requirement that explains the obvious: The first ChargingSchedulePeriod should start with StartSchedule = 0.

Updated description for the chargingSchedulePeriod field:

Old text	Required. List of ChargingSchedulePeriod elements defining maximum power or current usage over time.
New text	Required. List of ChargingSchedulePeriod elements defining maximum power or current usage over time. **The StartSchedule of the first ChargingSchedulePeriod SHALL always be 0.**

3.35. page 94, section: 7.14 ChargingSchedulePeriod Limit can also be in Watts

ChargingSchedulePeriod field: Limit can be in Watts or Ampere.

Old text	Required. Power limit during the schedule period, expressed in Amperes. Accepts at most one digit fraction (e.g. 8.1).
New text	Required. Charging rate limit during the schedule period, in the applicable chargingRateUnit, for example in Amperes or Watts. Accepts at most one digit fraction (e.g. 8.1).

3.36. page 94 - 95, section: 7.15 - 7.19 CiStringXXXType should be defined as type

The CiStringXXTypes are defined as Class, that contain a Field Name, but they should have been defined as Type (without Field Name)

Paragraphs effected:

- 7.15 CiString20Type
- 7.16 CiString25Type
- 7.17 CiString50Type
- 7.18 CiString255Type

- 7.19 CiString500Type

Old definition:

7.15 CiString20Type

Class

Generic used case insensitive string of 20 characters.

FIELD NAME	FIELD TYPE	DESCRIPTION
cistring20	CiString[20]	String is case insensitive.

New definition:

7.15 CiString20Type

Type

Generic used case insensitive string of 20 characters.

FIELD TYPE	DESCRIPTION
CiString[20]	String is case insensitive.

These Changes have no effect on the WSDL and JSON definitions, they are defined correct in WSDL and JSON Schemas.

3.37. Page 98, section: 7.27: Definition of IdTagInfo misses cardinality

Every class definition in OCPP 1.6 contains a column called: "Card." (cardinality), but this column is missing in the definition of IdTagInfo.

New definition:

7.27 IdTagInfo

Class

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

If expiryDate is not given, the status has no end date.

FIELD NAME	FIELD TYPE	CAR D.	DESCRIPTION
expiryDate	dateTime	0..1	Optional. This contains the date at which idTag should be removed from the Authorization Cache.
parentIdTag	IdToken	0..1	Optional. This contains the parent-identifier.
status	AuthorizationStatus	1..1	Required. This contains whether the idTag has been accepted or not by the Central System.

3.38. page 98, section: 7.28. idToken: idToken field is of wrong type

The field: idToken should have been of the type CiString20Type, it is case insensitive.

Old text	String[20]
New text	CiString20Type

To prevent interoperability issues: Do NOT update WSDL files!

Note: For future version: idToken was missed when added the CiStringXXTypes. Might be removed as type, and all field of the type idToken in other Classes will then be replaced by CiString20Type.

3.39. Page 98, section: 7.28: IdToken should be defined as type

The IdToken is defined as Class, that contain a Field Name, but they should have been defined as Type (without Field Name) The WSDL and JSON Schema are correct, was only wrong in the specification. (was also wrong in OCPP 1.5)

New definition:

7.28 IdToken

Type

Contains the identifier to use for authorization. It is a case insensitive string. In future releases this may become a complex type to support multiple forms of identifiers.

FIELD TYPE	DESCRIPTION
CiString20Type	IdToken is case insensitive.

3.40. Page 99, section: 7.31: Not definition of .register and .interval.

The are a couple of measurands that are defined as .register or .interval. But there is no definition of what that means.

Old:

VALUE	DESCRIPTION
Energy.Active.Export.Register	Energy exported by EV (Wh or kWh)
Energy.Active.Import.Register	Energy imported by EV (Wh or kWh)
Energy.Reactive.Export.Register	Reactive energy exported by EV (varh or kvarh)
Energy.Reactive.Import.Register	Reactive energy imported by EV (varh or kvarh)
Energy.Active.Export.Interval	Energy exported by EV (Wh or kWh)
Energy.Active.Import.Interval	Energy imported by EV (Wh or kWh)
Energy.Reactive.Export.Interval	Reactive energy exported by EV. (varh or kvarh)
Energy.Reactive.Import.Interval	Reactive energy imported by EV. (varh or kvarh)

New:

> Import is energy flow from the Grid to the Charge Point, EV or other load. Export is energy flow from the EV to the Charge Point and/or from the Charge Point to the Grid.

VALUE	DESCRIPTION
Energy.Active.Export.Register	Numerical value read from the "active electrical energy" (Wh or kWh) register of the (most authoritative) electrical meter measuring energy exported (to the grid).
Energy.Active.Import.Register	Numerical value read from the "active electrical energy" (Wh or kWh) register of the (most authoritative) electrical meter measuring energy imported (from the grid supply).
Energy.Reactive.Export.Register	Numerical value read from the "reactive electrical energy" (VARh or kVARh) register of the (most authoritative) electrical meter measuring energy exported (to the grid).
Energy.Reactive.Import.Register	Numerical value read from the "reactive electrical energy" (VARh or kVARh) register of the (most authoritative) electrical meter measuring energy imported (from the grid supply).
Energy.Active.Export.Interval	Absolute amount of "active electrical energy" (Wh or kWh) exported (to the grid) during an associated time "interval", specified by a Metervalues ReadingContext, and applicable interval duration configuration values (in seconds) for "ClockAlignedDataInterval" and "MeterValueSampleInterval".
Energy.Active.Import.Interval	Absolute amount of "active electrical energy" (Wh or kWh) imported (from the grid supply) during an associated time "interval", specified by a Metervalues ReadingContext, and applicable interval duration configuration values (in seconds) for "ClockAlignedDataInterval" and "MeterValueSampleInterval".
Energy.Reactive.Export.Interval	Absolute amount of "reactive electrical energy" (VARh or kVARh) exported (to the grid) during an associated time "interval", specified by a Metervalues ReadingContext, and applicable interval duration configuration values (in seconds) for "ClockAlignedDataInterval" and "MeterValueSampleInterval".

VALUE	DESCRIPTION
Energy.Reactive.Import.Interval	Absolute amount of "reactive electrical energy" (VARh or kVARh) imported (from the grid supply) during an associated time "interval", specified by a Metervalues ReadingContext, and applicable interval duration configuration values (in seconds) for "ClockAlignedDataInterval" and "MeterValueSampleInterval".

ℹ All "Register" values relating to a single charging transaction, or a non-transactional consumer (e.g. charge point internal power supply, overall supply) MUST be monotonically increasing in time.

The actual quantity of energy corresponding to a reported ".Register" value is computed as the register value in question minus the register value recorded/reported at the start of the transaction or other relevant starting reference point in time. For improved auditability, ".Register" values SHOULD reported exactly as they are directly read from a non-volatile register in the electrical metering hardware, and SHOULD NOT be re-based to zero at the start of transactions. This allows any "missing energy" between sequential transactions, due to hardware fault, mis-wiring, fraud, etc. to be identified, by allowing the Central System to confirm that the starting register value of any transaction is identical to the finishing register value of the preceding transaction on the same connector.

3.41. Page 103, section: 7.37: RecurrencyKindType definition is ambiguous.

NEW: errata sheet v3.0

The definition of RecurrencyKindType is ambiguous. It is not clear when a Charge Profile should recur.

Changes to the table

VALUE	OLD DESCRIPTION	NEW DESCRIPTION
Daily	The schedule restarts at the beginning of the next day.	The schedule restarts every 24 hours, at the same time as in the startSchedule.
Weekly	The schedule restarts at the beginning of the next week (defined as Monday morning).	The schedule restarts every 7 days, at the same time and day-of-the-week as in the startSchedule.

3.42. Page 105, section: 7.42: Improved description of Soft/Hard Reset

NEW: errata sheet v3.0

The descriptions of Soft/Hard reset can be improved, not very clear what is the difference between the 2.

Changes to the table

VALUE	OLD DESCRIPTION	NEW DESCRIPTION
Hard	Full reboot of Charge Point software.	Restart (all) the hardware, the Charge Point is not required to gracefully stop ongoing transaction. If possible the Charge Point sends a StopTransaction.req for previously ongoing transactions after having restarted and having been accepted by the Central System via a BootNotification.conf. This is a last resort solution for a not correctly functioning Charge Point, by sending a "hard" reset, (queued) information might get lost.
Soft	Return to initial status, gracefully terminating any transactions in progress.	Stop ongoing transactions gracefully and sending StopTransaction.req for every ongoing transaction. It should then restart the application software (if possible, otherwise restart the processor/controller).

3.43. Page 106, section: 7.45: No UnitOfMeasure for Measurand Frequency.

There is no UnitOfMeasure for Measurand: Frequency.

Add to the description for Frequency on page 100, section: 7.31:	OCPP 1.6 does not have a UnitOfMeasure for frequency, the UnitOfMeasure for any SampledValue with measurand: Frequency is Hertz.

3.44. Page 107, section: 7.42: UnlockConnector with unknown ConnectorId

NEW: errata sheet v3.0

It has not been specified how a Charge Point should respond when a Central System request an Unlock Connector for an unknown ConnectorId.

Preferably the response would have been: "Rejected", so that will be added to OCPP 2.0. For OCPP 1.6 we cannot add extra states, so we have to use "NotSupported". "UnlockFailed" should not be used for this, "UnlockFailed" is really for when the locking mechanism detects a failed unlock attempt.

Changes to the table

VALUE	OLD DESCRIPTION	NEW DESCRIPTION
UnlockFailed	Failed to unlock the connector.	Failed to unlock the connector: **The Charge Point has tried to unlock the connector and has detected that the connector is still locked or the unlock mechanism failed**.
NotSupported	Charge Point has no connector lock	Charge Point has no connector lock, **or ConnectorId is unknown**.

3.45. Page 112, section: 9.1.6: Improved description configuration key: "ConnectionTimeOut"

The description of the configuration key: "ConnectionTimeOut" can be improved

Old text	Interval **(from successful authorization)** until incipient charging session is automatically canceled due to failure of EV user to (correctly) insert the charging cable connector(s) into the appropriate connector(s).
New text	Interval **from beginning of status: 'Preparing'** until incipient session is automatically canceled, due to failure of EV driver to (correctly) insert the charging cable connector(s) into the appropriate socket(s). **The Charge Point SHALL go back to the original state, probably: 'Available'**

3.46. OCPP JSON Schema: StopTransaction reason is optional

The OCPP 1.6 JSON schema message definitions set, as originally published, contain an error that renders it inconsistent with both the written OCPP 1.6 specification and the SOAP/XML Schema/WSDLs schema definitions.

In the StopTransactionRequest message structure, the "reason" element is incorrectly marked as being "required", whereas the human-readable protocol specification document and the SOAP WSDL service definition file both explicitly define it to be "optional", with an explicitly defined implied default value ("Local"), as stated in Sections 4.10 (p46) and 6.49 (p82) of the OCPP 1.6 Specification.

The updated "OCPP_1.6_Schema.zip" file, as now published, contains a revised "StopTransaction.json" file, in which the reason element is now NOT specified as "required".

4. Typos

Typos, fixes to incorrect links/reference, improve terms used etc. that have no impact on the description of the way the protocol works.

4.1. Generic: Typo Field Type: DateTime should be dateTime

dateTime field type is misspelled a couple of times as: DateTime (with upper-case **D**)

PAGE	SECTION	MESSAGE/CLASS	FIELD NAME
24	3.12.2	NOTE add the bottom	validFrom
70	6.22	GetCompositeSchedule.conf	scheduleStart
90	7.8	ChargingProfile	validFrom
91	7.8	ChargingProfile	validTo
93	7.13	ChargingProfile	startSchedule

4.2. Generic: Use of Energy Meter vs Power Meter

The terms Energy Meter and Power Meter are use throughout the specification, but they are not used consistently and the term: Electrical Meter seems to fit most cases even better.

List of all textual improvements for this:

PAGE	SECTION	OLD TEXT	NEW TEXT
8	2.2	Defines the wiring order of the phases between the **energy** meter (or if absent, the grid connection), and the Charge Point connector.	Defines the wiring order of the phases between the **electrical** meter (or if absent, the grid connection), and the Charge Point connector.
38	4.7	A Charge Point MAY sample the **energy** meter or other sensor/transducer hardware to provide extra information about its meter values.	A Charge Point MAY sample the **electrical** meter or other sensor/transducer hardware to provide extra information about its meter values.

PAGE	SECTION	OLD TEXT	NEW TEXT
38	4.7	The Charging Point SHALL report all phase number dependent values from the **power** meter (or grid connection when absent) point of view.	The Charging Point SHALL report all phase number dependent values from the **electrical** meter (or grid connection when absent) point of view.
63	6.3	This contains the serial number of the main **power** meter of the Charge Point.	This contains the serial number of the main **electrical meter** of the Charge Point.
63	6.3	This contains the type of the main **power** meter of the Charge Point.	This contains the type of the main **electrical** meter of the Charge Point.
87	7.6	Failure to read power meter.	Failure to read **electrical/energy/** power meter.
116	9.1.21	The phase rotation per connector in respect to the connector’s **energy** meter (or if absent, the grid connection).	The phase rotation per connector in respect to the connector’s **electrical** meter (or if absent, the grid connection).

4.3. Page 13, section: 3.2: Typo in text about SupportedFeatureProfiles

Below the table with the mapping of messages to feature profiles, there is a typo: "charging profiles" instead of "feature profiles".

Old text	The support for the specific **charging** profiles is reported by the SupportedFeatureProfiles configuration key.
New text	The support for the specific **feature** profiles is reported by the SupportedFeatureProfiles configuration key.

4.4. Page 18, section: 3.4.4: Typo in Unknown Offline Authorization

There is a typo in the text about Unknown Offline Authorization.

Old text	When connection **the** the Central Server is restored
New text	When connection **to** the Central Server is restored

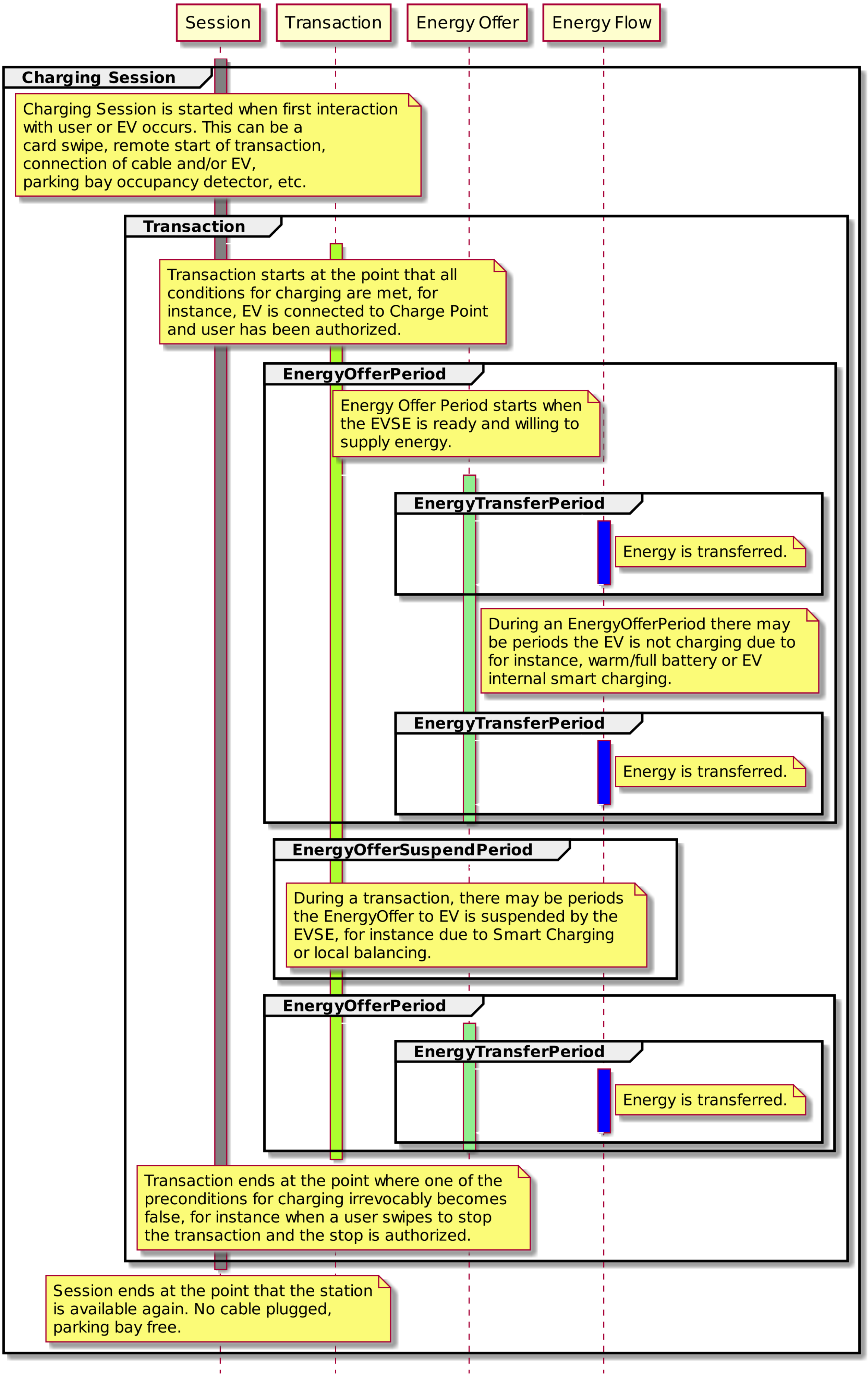
4.5. Page 19, section: 3.5: Definition of OCPP Transaction, Session, EnergyOfferPeriod etc missing.

In the table on page 19, there are some terms used that are never described in the specification.

Changes to the diagram on page 19:

OLD TEXT	NEW TEXT
Session	Charging Session
OCPP Transaction	Transaction

Update diagram for section: 3.5 on page 19:


Session
Transaction
Energy Offer
Energy Flow
Charging Session
Charging Session is started when first interaction with user or EV occurs. This can be a card swipe, remote start of transaction, connection of cable and/or EV, parking bay occupancy detector, etc.
Transaction
Transaction starts at the point that all conditions for charging are met, for instance, EV is connected to Charge Point and user has been authorized.
EnergyOfferPeriod
Energy Offer Period starts when the EVSE is ready and willing to supply energy.
EnergyTransferPeriod
Energy is transferred.
During an EnergyOfferPeriod there may be periods the EV is not charging due to for instance, warm/full battery or EV internal smart charging.
EnergyTransferPeriod
Energy is transferred.
EnergyOfferSuspendPeriod
During a transaction, there may be periods the EnergyOffer to EV is suspended by the EVSE, for instance due to Smart Charging or local balancing.
EnergyOfferPeriod
EnergyTransferPeriod
Energy is transferred.
Transaction ends at the point where one of the preconditions for charging irrevocably becomes false, for instance when a user swipes to stop the transaction and the stop is authorized.
Session ends at the point that the station is available again. No cable plugged, parking bay free.

Changes to the table of definitions in section:. 2.2 on page 7

TERM	OLD DESCRIPTION	NEW DESCRIPTION
Charging Session	Part of a transaction during which the EV is allowed to request energy	a Charging Session is started when first interaction with user or EV occurs. This can be a card swipe, remote start of transaction, connection of cable and/or EV, parking bay occupancy detector, etc.

Additions to the table of definitions in section:. 2.2 on page 7

TERM	DESCRIPTION
Energy Offer Period	Energy Offer Period starts when the EVSE is ready and willing to supply energy.
Energy Offer SuspendPeriod	During a transaction, there may be periods the EnergyOffer to EV is suspended by the EVSE, for instance due to Smart Charging or local balancing.

Changes throughout the entire specification, correcting incorrect term use.

PAGE	PAR.	OLD TEXT	NEW TEXT
42	4.9	C6: **Charging session** is stopped by user or a Remote Stop Transaction message and further user action is required (e.g. remove cable, leave parking bay)	C6: **Transaction** is stopped by user or a Remote Stop Transaction message and further user action is required (e.g. remove cable, leave parking bay)
42	4.9	D6: **Charging session** is stopped and further user action is required	D6: **Transaction** is stopped and further user action is required
43	4.9	E6: **Charging session** is stopped and further user action is required	E6: **Transaction** is stopped and further user action is required
43	4.9	F2: User restart charging session (e.g. reconnects cable, presents idTag again)	F2: User restart charging session (e.g. reconnects cable, presents idTag again)**, thereby creating a new Transaction**
88	7.7	Preparing: When a Connector becomes no longer available for a new user but **no charging session is active**. Typically a Connector is **occupied** when a user presents a tag, inserts a cable or a vehicle occupies the parking bay	Preparing: When a Connector becomes no longer available for a new user but **there is no ongoing Transaction (yet)**. Typically a Connector is **in preparing state** when a user presents a tag, inserts a cable or a vehicle occupies the parking bay
88	7.7	Finishing: When a **charging session** has stopped at a Connector, but the Connector is not yet available for a new user, e.g. the cable has not been removed or the vehicle has not left the parking bay	Finishing: When a **Transaction** has stopped at a Connector, but the Connector is not yet available for a new user, e.g. the cable has not been removed or the vehicle has not left the parking bay.
91	7.9	such as the start of a **session**	such as the start of a **Transaction**
111	9.1.5	or partial interval, at the beginning or end of a **charging session**	or partial interval, at the beginning or end of a **Transaction**
112	9.1.6	Interval (from successful authorization) until incipient **charging session** is automatically canceled due to failure of EV user to (correctly) insert the charging cable connector(s) into the appropriate connector(s).	Interval (from successful authorization) until incipient **Transaction** is automatically canceled due to failure of EV user to (correctly) insert the charging cable connector(s) into the appropriate connector(s).
117	9.1.25	Clock-aligned periodic measurand(s) to be included in the TransactionData element of StopTransaction.req MeterValues.req PDU for every ClockAlignedDataInterval of the **charging session**	Clock-aligned periodic measurand(s) to be included in the TransactionData element of StopTransaction.req MeterValues.req PDU for every ClockAlignedDataInterval of the **Transaction**

PAGE	PAR.	OLD TEXT	NEW TEXT
117	9.1.27	Sampled measurands to be included in the TransactionData element of StopTransaction.req PDU, every MeterValueSampleInterval seconds from the start of the **charging session**	Sampled measurands to be included in the TransactionData element of StopTransaction.req PDU, every MeterValueSampleInterval seconds from the start of the **Transaction**
121	9.4.4	If defined and true, this Charge Point support switching from 3 to 1 phase during a **charging session**.	If defined and true, this Charge Point support switching from 3 to 1 phase during a **Transaction**.

4.6. Page 34, section: 4.2: Boot Notification diagram: Interval

Figure 13: "Sequence Diagram: Boot Notification" contains a typo.

Old text	BootNotification.conf(currentTime, **heartbeatInterval**, status)
New text	BootNotification.conf(currentTime, **interval**, status)

4.7. Page 47, section: 4.10: Typo description in stop transaction

There is a typo in the description of StopTransactionOnEVSideDisconnect set to true.

Old text	Setting StopTransactionOnEVSideDisconnect to true will prevent sabotage acts **top** stop the energy flow by unplugging not locked cables on EV side.
New text	Setting StopTransactionOnEVSideDisconnect to true will prevent sabotage acts **to** stop the energy flow by unplugging not locked cables on EV side.

4.8. Page 50, section: 5.5: Clear Charging Profile sequence diagram: incorrect .conf message

The sequence diagram on page 50 for Clear Charging Profile contains a typo. It contains the incorrect response: "ClearCache.conf" instead of "ClearChargingProfile.conf"

Update diagram for section: 5.5 on page 50:

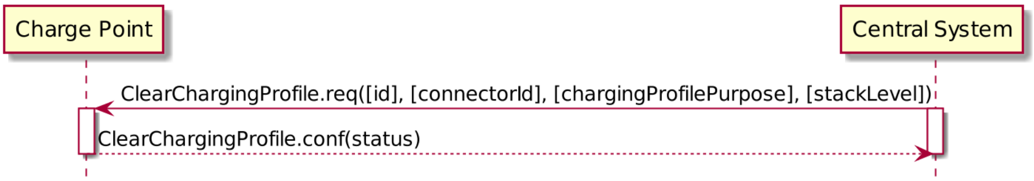

Charge Point
Central System
ClearChargingProfile.req([id], [connectorId], [chargingProfilePurpose], [stackLevel])
ClearChargingProfile.conf(status)

Figure 3. Updated Sequence Diagram: Clear Charging Profile

4.9. Page 55, section: 5.13: Missing MAY in description Reservation Parent idTag

The text MAY is missing in the description of getting the parent-id for a reservation via Authorize.req.

Old text	The Authorize.conf response **contains** the parent-id.
New text	The Authorize.conf response **MAY contain** the parent-id

4.10. Page 56, section: 5.14: Typo in reset description

NEW: errata sheet v3.0

There is a typo in the description of Reset response.

Old text	The response PDU SHALL include whether the Charge Point **is** will attempt to reset itself.
New text	The response PDU SHALL include whether the Charge Point will attempt to reset itself.

4.11. Page 60, section: 5.18: Central System sends Unlock Connector

In the paragraph about Unlock Connector, "Charge Point" and "Central Server" are mixed up.

Old text	To do so, The **Charge Point** SHALL send
New text	To do so, The **Central System** SHALL send

4.12. Page 71, section: 6.23: Typo in GetConfiguration.req

There is a typo in the text about GetConfiguration.req.

Old text	This contains the field definition of the GetConfiguration.req PDU sent by **the the** Central System to the Charge Point
New text	This contains the field definition of the GetConfiguration.req PDU sent by **the** Central System to the Charge Point

4.13. Page 91, section: 7.9: ChargingProfileKindType misses description and 'were used'

There is no 'were used' in the definition of ChargingProfileKindType

Additional text	Kind of charging profile, as used in: ChargingProfile

4.14. Page 91, section: 7.10: Description of ChargePointMaxProfile in ChargingProfilePurposeType contains unused words

The description ChargePointMaxProfile in ChargingProfilePurposeType contains words that should not have been here.

Old text	Configuration for the maximum power or current available for an entire Charge Point.**SetChargingProfile.req message.**
New text	Configuration for the maximum power or current available for an entire Charge Point.

4.15. Page 91, section: 7.10: ChargingProfilePurposeType misses description and 'were used'

There is no 'were used' in the definition of ChargingProfilePurposeType

Additional text	Purpose of the charging profile, as used in: ChargingProfile

4.16. Page 92, section: 7.13: ChargingSchedule misses description and 'were used'

There is no 'were used' in the definition of ChargingSchedule.

Additional text	Charging schedule structure defines a list of charging periods, as used in: GetCompositeSchedule.conf and ChargingProfile

4.17. Page 93, section: 7.14: ChargingSchedulePeriod misses description and 'were used'

There is no 'were used' in the definition of ChargingSchedulePeriod.

Additional text	Charging schedule period structure defines a time period in a charging schedule, as used in: ChargingSchedule

4.18. Page 96, section: 7.22. Typo in descriptions ConfigurationStatus

Word 'is' is missing in descriptions of ConfigurationStatus

OLD TEXT	NEW TEXT
Configuration key supported and setting has been changed.	Configuration key **is** supported and setting has been changed.
Configuration key supported, but setting could not be changed.	Configuration key **is** supported, but setting could not be changed.
Configuration key supported and setting has been changed,	Configuration key **is** supported and setting has been changed,

4.19. Page 101, section: 7.33: MeterValue misses used by StopTransaction.req

In the description of MeterValue there is a link to the usage of MeterValue: MeterValue.req, MeterValue it is also used in StopTransaction.req.

Old text	Collection of one or more sampled values in MeterValues.req.
New text	Collection of one or more sampled values in MeterValues.req **and StopTransaction.req**.

4.20. Page 102, section: 7.35: Descriptions Transaction.Begin, Transaction.End swapped

The descriptions of Transaction.Begin and Transaction.End are swapped.

Old:

VALUE	DESCRIPTION
Transaction.Begin	Value taken at **end** of transaction.
Transaction.End	Value taken at **start** of transaction.

New:

VALUE	DESCRIPTION
Transaction.Begin	Value taken at **start** of transaction.
Transaction.End	Value taken at **end** of transaction.

4.21. Page 106, section: 7.45: UnitOfMeasure links to incorrect usage

In the description of UnitOfMeasure there are links to the usage of UnitOfMeasure. There incorrectly say: MeterValues.req and StopTransaction.req, should be: SampledValue

Old text	Allowable values of the optional "unit" field of a Value element, as used in **MeterValues.req and StopTransaction.req messages**.
New text	Allowable values of the optional "unit" field of a Value element, as used in **SampledValue**.

4.22. Page 110, section: 9: Typo in Standard Configuration Key Names & Values

There is a typo in the text about Standard Configuration Key Names & Values.

Old text	In case **the the** accessibility is read-write, the Central System can also write the value for the key using ChangeConfiguration
New text	In case **the** accessibility is read-write, the Central System can also write the value for the key using ChangeConfiguration.

4.23. Page 121, section: 9.4.4: ConnectorSwitch3to1PhaseSupported: Type should be boolean

There is a typo in the type definition of the configuration key: ConnectorSwitch3to1PhaseSupported: Type should be boolean, is now bool.

Old type	bool
New type	boolean

4.24. Page 123, section: A.1: Power.Factor is missing in the list of new enum values

In the list of new values for the enum: Measurand, the value: Power.Factor is missing, should be added.

Old text	Added enum Current.Offered, Frequency, Power.Offered, RPM and SoC
New text	Added enum Current.Offered, Frequency, **Power.Factor,** Power.Offered, RPM and SoC

5. Known issues that will not be fixed

Non known