OCPP 2.0

Part 4 - Errata, v1.0

Table of Contents

1. Disclaimer

Version History

Version	Date	Author	Description
1.0	2019-12-20	Milan Jansen, Franc Buve and Robert de Leeuw	Version 1.0
Release Candidate	2019-12-01	Milan Jansen, Franc Buve and Robert de Leeuw	Release Candidate.

2. Scope

This document contains erratas on the OCPP 2.0 specification.

2.1. Terminology and Conventions

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

3. Errata

3.1. Page 6, Section 3.3. WebSocket Compression, Description WebSocket Compression optional for Charging Stations not clear

Description about WebSocket Compression handshake in RFC 7692 is causing a bit of confusion. It is not clear if compression is required or not.

Old text	RFC 7692 allows the Charging Station and the CSMS to do a negotiation during the connection setup. When both parties support the Compression Extension they will then use DEFLATE compression ([RFC1951]) when sending data over the line. When one of the parties doesn't support it, the JSON will be send uncompressed (like in OCPP 1.6J).
New text	**OCPP Requires the CSMS (and Local Controller) to support RFC 7692, WebSocket compression is seen as a relative simple way to reduce mobile data usage. For a Charging Station this is not a hard requirement, as this might be more complex to implement on an embedded platform, but as this is seen as efficient solution to reduce mobile data usage, it is RECOMMENDED to be implemented on a Charging Station that uses a mobile data connection.** RFC 769 allows the Charging Station and the CSMS to do a negotiation during the connection setup. When both parties support the Compression Extension they will then use DEFLATE compression ([RFC1951]) when sending data over the line. When one of the parties doesn't support it, the JSON will be sent uncompressed (like in OCPP 1.6J). **When the Charging Station detects that compression is not used, it is RECOMMENDED not to close the connection, as turning of compression can be very useful during development, testing and debugging.**

3.2. Page 6, Section 4.1.1. Synchronicity, Unclear when the CSMS is allowed to send messages

It is unclear that the CSMS does not have to wait for a response from Charging Station 1 when wanting to send a request to Charging Station 2.

Old text	A Charging Station or CSMS SHALL NOT send a CALL message to the other party unless all the CALL messages it sent before have been responded to or have timed out. A CALL message has been responded to when a CALLERROR or CALLRESULT message has been received with the message ID of the CALL message.
New text	A Charging Station or CSMS SHALL NOT send a CALL message to the other party unless all the CALL messages it sent before have been responded to or have timed out. **This does not mean that the CSMS cannot send a message to another Charging Station, while waiting for a response of a first Charging Station, this rule is per OCPP-J connection.** A CALL message has been responded to when a CALLERROR or CALLRESULT message has been received with the message ID of the CALL message.

3.3. Page 10, Section 4.2.3, Unclear if a system is allowed to drop a message when it is not conform the JSON schema

There are error codes like; PropertyConstraintViolation, OccurrenceConstraintViolation and TypeConstraintViolation. But there is no remark or requirement that explicitly allows a system to delete a complete message when it is not conform the JSON schema.

```
When a message contains any invalid OCPP and/or it is not conform the JSON
schema, the system is allowed to drop the message.
```

3.4. Page 10, Section 4.2.3, Missing CALLERROR example

The following example plus description needs be added, below the CALLERROR field descriptions.

For example, a CALLERROR could look like this:

```
[4,
  "162376037",
  "NotSupported",
  "SetDisplayMessageRequest not implemented",
  {}
]
```

3.5. Page 15, Section 7.2. Handling Signed Messages, incomplete requirement

The requirement is also applicable for the CSMS, not only the Charging Station.

Old text	When a Charging Station receives a signed request, and it supports digital signing, it SHALL send a signed reply.
New text	When a Charging Station **or CSMS** receives a signed request, and it supports digital signing, it SHALL send a signed reply.

3.6. Page 17, Section 8.4. WebSocketPingInterval, A recommendation needs to be added about how to configure the WebSocketPingInterval

Changed description:

Old description	A value of 0 disables client side websocket Ping / Pong. In this case there is either no ping / pong or the server initiates the ping and client responds with Pong. Positive values are interpreted as number of seconds between pings. Negative values are not allowed, SetConfiguration is then expected to return a Rejected result.
New description	A value of 0 disables client side websocket Ping / Pong. In this case there is either no ping / pong or the server initiates the ping and client responds with Pong. Positive values are interpreted as number of seconds between pings. Negative values are not allowed, SetConfiguration is then expected to return a Rejected result. **It is recommended to configure WebSocketPingInterval smaller then: MessageAttemptsTransactionEvent * MessageAttemptIntervalTransactionEvent. This will limit the chance of the resend mechanism for transaction-related messages being triggered by connectivity issues.**