OASIS OCPP TC CR Mark off-line transactions
Title: Mark off-line transactions

Version: v0.3

Date: 2016-12-09

Initiator: Robert de Leeuw (IHomer)

Supporters: n/a

Goal: For the processing of transaction in the Central System it is very useful to know if a transaction was started and/or stopped while the Charge Point was off-line.

Implementation Impact:

<table>
<thead>
<tr>
<th>Actor</th>
<th>Impact</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charge Point</td>
<td>Low</td>
</tr>
<tr>
<td>Central System</td>
<td>Low</td>
</tr>
</tbody>
</table>

Security Impact: n/a

Compliance Impact: Tests for off-line behavior have to start checking for this new flag.

Summary: By adding an optional field (flag): "offline" of type boolean to both the StartTransaction.req and StopTransaction.req, the Charge Point can notify the Central System that a transaction is started and/or stopped while it was off-line (no connection with the Central System). The Charge Point is required to set the "offline" flag when sending a Start or StopTransaction.req that was queued, because it could not be send when the Charge-Point was offline.

All text bellow can be seen as a new or improved functional block for the OCPP specification. If this is an improvement, copy the old functional block here and update it. Mark all the changes made.

1. Scope

(Add scope of the new/improved functionality here)
2. Use Cases

2.1. B04 - Offline Start Transaction (Updated)

Table 1. B04 - Offline Start Transaction

<table>
<thead>
<tr>
<th>No.</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Use case element name</td>
<td>Offline Start Transaction</td>
</tr>
<tr>
<td>2</td>
<td>ID</td>
<td>B04</td>
</tr>
<tr>
<td></td>
<td>Functional block</td>
<td>B. Transactions</td>
</tr>
<tr>
<td>3</td>
<td>Objective(s)</td>
<td>To enable the EV Driver to start a charging session while the Charge Point is Offline.</td>
</tr>
<tr>
<td>4</td>
<td>Description</td>
<td>This use case covers how the Charge Point, while Offline, is able to start a transaction using the Local Authorization List or the Authorization Cache.</td>
</tr>
<tr>
<td></td>
<td>Actors</td>
<td>Charge Point, Central System, User</td>
</tr>
<tr>
<td></td>
<td>Scenario description</td>
<td>1. The EV Driver presents an IdToken at the Charge Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. The Charge Point authorizes the EV Driver by using the Local Authorization List.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. The transaction starts.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4. The StartTransaction.req is stored/queued by the Charge Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5. The connection between Charge Point and Central System is restored.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6. The Charge Point starts to send queued messages</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7. The stored StartTransaction.req is send, notifying the Central System about the transaction that was started.</td>
</tr>
<tr>
<td>5</td>
<td>Prerequisite(s)</td>
<td>The Charge Point is Offline.</td>
</tr>
<tr>
<td>6</td>
<td>Postcondition(s)</td>
<td>Successful postcondition:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The EV Driver is authorized to Offline start a transaction.</td>
</tr>
<tr>
<td></td>
<td>Failure postcondition:</td>
<td>The EV Driver is not authorized to Offline start a transaction.</td>
</tr>
</tbody>
</table>
Figure 1. Sequence Diagram: Offline Start Transaction
2.1.1. B04 - Requirements

*These requirements are normative.*

Table 2. B04 - Requirements

<table>
<thead>
<tr>
<th>CP/CS</th>
<th>ID</th>
<th>Precondition</th>
<th>Requirement definition</th>
<th>M/O/C</th>
<th>Rational</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP</td>
<td>B04.FR.0 1</td>
<td>When Offline.</td>
<td>The Charge Point MUST queue any StartTransaction PDUs.</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>CP</td>
<td>B04.FR.0 2</td>
<td>After the connection is restored.</td>
<td>The Charge Point MUST send queued StartTransaction PDUs.</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>CP</td>
<td>B04.FR.0 3</td>
<td>B04.FR.02</td>
<td>The flag: &quot;offline&quot; SHALL be set to TRUE for any StartTransaction.req that belongs to a transaction that was started while the Charge Point was off-line.</td>
<td>M</td>
<td></td>
</tr>
</tbody>
</table>

2.2. B08 - Offline Stop Transaction (Updated)

Table 3. B08 - Offline Stop Transaction

<table>
<thead>
<tr>
<th>No.</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Use case element name</td>
<td>Offline Stop Transaction</td>
</tr>
<tr>
<td>2</td>
<td>ID</td>
<td>B08</td>
</tr>
<tr>
<td></td>
<td><em>Functional block</em></td>
<td>B. Transactions</td>
</tr>
<tr>
<td></td>
<td><em>Parent use case</em></td>
<td>B06 - Local Stop Transaction</td>
</tr>
<tr>
<td>3</td>
<td>Objective(s)</td>
<td>To enable the User, while the Charge Point is Offline, to stop a transaction.</td>
</tr>
<tr>
<td>4</td>
<td>Description</td>
<td>This use case describes how a User can stop a transaction while the Charge Point is Offline. While a transaction is ongoing and the Charge Point is Offline, the User presents his IdToken, if the Charge Points knows locally (without asking the Central System) that this IdToken is allowed to stop the transaction, it will stop the ongoing transaction. When the Charge Point restores the connection with the Central System, it needs to send the information about this Offline stopt transaction to the Central System.</td>
</tr>
<tr>
<td></td>
<td><em>Actors</em></td>
<td>Charge Point, Central System, User</td>
</tr>
</tbody>
</table>
### Scenario Description
1. The EV Driver presents IdToken to stop the transaction.
2. The Charge Point stops the energy offer.
3. The StopTransaction.req is stored/queued by the Charge Point.
4. The connection between Charge Point and Central System is restored.
5. The Charge Point starts to send queued messages
6. The stored StopTransaction.req is send, notifying the Central System about the transaction that was stopped.

### Alternative Scenario(s)
If the cable not is permanently attached, **N01 - Remotely Unlock Connector** applies.

### Prerequisite(s)
Charging session ongoing and connection lost.

### Postcondition(s)
Charge Point is in *Idle* state.

![Sequence Diagram: Offline Stop Transaction](image)

**Figure 2. Sequence Diagram: Offline Stop Transaction**

### Error Handling
n/a

### Remark(s)
Parent id check must be done on local authorization list and / or Authorization Cache if available.

### Test Case(s)
Test Case Root Id: 038
2.2.1. B08 - Requirements

Table 4. B08 - Requirements

<table>
<thead>
<tr>
<th>CP/CS</th>
<th>ID</th>
<th>Precondition</th>
<th>Requirement definition</th>
<th>M/O/C</th>
<th>Rational</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP</td>
<td>B08.FR.01</td>
<td>When Offline.</td>
<td>The Charge Point MUST queue any StopTransaction PDUs.</td>
<td></td>
<td>M</td>
</tr>
<tr>
<td>CP</td>
<td>B08.FR.02</td>
<td>After the connection is restored.</td>
<td>The Charge Point MUST send queued StopTransaction PDUs.</td>
<td></td>
<td>M</td>
</tr>
<tr>
<td>CP</td>
<td>B04.FR.03</td>
<td>B08.FR.02</td>
<td>The flag: &quot;offline&quot; SHALL be set to TRUE for any StopTransaction.req that belongs to a transaction that was stopped while the Charge Point was off-line.</td>
<td></td>
<td>M</td>
</tr>
</tbody>
</table>

2.3. B11 - Inform Central System of an Offline Occurred Transaction (Updated)

Table 5. B11 - Inform Central System of an Offline Occurred Transaction

<table>
<thead>
<tr>
<th>No.</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Use case element name</td>
<td>Inform Central System of an Offline Occurred Transaction</td>
</tr>
<tr>
<td>2</td>
<td>ID</td>
<td>B11</td>
</tr>
<tr>
<td></td>
<td>Functional block</td>
<td>B. Transactions</td>
</tr>
<tr>
<td>3</td>
<td>Objective(s)</td>
<td>To enable the Charge Point to inform the Central System that a transaction occurred while the Charge Point was Offline.</td>
</tr>
<tr>
<td>4</td>
<td>Description</td>
<td>This use case covers how the Charge Point starts and stops a transaction since connection loss.</td>
</tr>
<tr>
<td></td>
<td>Actors</td>
<td>Charge Point, Central System</td>
</tr>
<tr>
<td></td>
<td>Scenario description</td>
<td>1. The connection with the Central System is restored.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. The Charge Point sends a Heartbeat PDU to the Central System.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. The Central System sends StartTransaction.req to the Central System.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4. The Central responds with StartTransaction.conf, accepting the transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5. The Charge Point sends StopTransaction.req</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6. The Central System responds with StopTransaction.conf</td>
</tr>
<tr>
<td>5</td>
<td>Prerequisite(s)</td>
<td>At least one Offline transaction has taken place.</td>
</tr>
</tbody>
</table>

Alternative scenario(s) n/a
Figure 3. Sequence Diagram: Inform Central System of an Offline Occurred Transaction

2.3.1. B11 - Requirements

n/a

3. Messages

(New/changed messages)

3.1. StartTransaction.req (Updated)

This section contains the field definition of the StartTransaction.req PDU sent by the Charge Point to the Central System.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Card.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectorId</td>
<td>integer connectorId &gt; 0</td>
<td>1..1</td>
<td>Required. This identifies which connector of the Charge Point is used.</td>
</tr>
<tr>
<td>IdToken</td>
<td>IdToken</td>
<td>1..1</td>
<td>Required. This contains the identifier for which a transaction has to be started.</td>
</tr>
<tr>
<td>meterStart</td>
<td>integer</td>
<td>1..1</td>
<td>Required. This contains the meter value in Wh for the connector at start of the transaction.</td>
</tr>
<tr>
<td>reservationId</td>
<td>integer</td>
<td>0..1</td>
<td>Optional. This contains the id of the reservation that terminates as a result of this transaction.</td>
</tr>
<tr>
<td>timestamp</td>
<td>dateTime</td>
<td>1..1</td>
<td>Required. This contains the date and time on which the transaction is started.</td>
</tr>
<tr>
<td>offline</td>
<td>boolean</td>
<td>0..1</td>
<td>Optional. Indication that the transaction was started while the Charge Point was offline.</td>
</tr>
</tbody>
</table>

### 3.2. StopTransaction.req (Updated)

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

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Card.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IdToken</td>
<td>IdToken</td>
<td>0..1</td>
<td>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 IdToken, e.g. in case of a reset. A Charge Point SHALL send the IdToken if known.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Card.</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
<td>-------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>meterStop</td>
<td>integer</td>
<td>1..1</td>
<td>Required. This contains the meter value in Wh for the connector at end of the transaction.</td>
</tr>
<tr>
<td>timestamp</td>
<td>dateTime</td>
<td>1..1</td>
<td>Required. This contains the date and time on which the transaction is stopped.</td>
</tr>
<tr>
<td>transactionId</td>
<td>integer</td>
<td>1..1</td>
<td>Required. This contains the transaction-id as received by StartTransaction.conf.</td>
</tr>
<tr>
<td>reason</td>
<td>Reason</td>
<td>0..1</td>
<td>Optional. This contains the reason why the transaction was stopped. MAY only be omitted when Reason is &quot;Local&quot;.</td>
</tr>
<tr>
<td>offline</td>
<td>boolean</td>
<td>0..1</td>
<td>Optional. Indication that the transaction was stopped while the Charge Point was offline.</td>
</tr>
<tr>
<td>transactionData</td>
<td>MeterValue</td>
<td>0..*</td>
<td>Optional. This contains transaction usage details relevant for billing purposes.</td>
</tr>
</tbody>
</table>

4. Data Type
n/a

5. Configuration Keys
n/a

6. Test Cases
TODO

6.1. TC.01 - (Test Case Name)
7. WSDL Schema
(Additions/changes to the existing WSDL schemas)

8. JSON Schema
(Additions/changes to the existing JSON schemas)

Appendix

1. Sequence Diagram Source

1.1. OfflineStartTransaction.plantuml

```
@startuml
skinparam dpi 300
hide footbox

actor "User" as EV
participant "Charge Point" as CP
participant "Central System" as CS

activate EV
EV -> CP: Present IdToken()
activate CP

opt if supported
CP ->o CP: check local authorization list()
end

opt if supported
CP ->o CP: Check Authorization Cache()
end

alt LocalAuthorizeOffline=true & (Id in cache or (Id\nin local list & Valid)) or
(AllowOfflineTxForUnknownId=true\& Id not Invalid in local list)
    CP --> EV: optional notification
    CP ->o CP: lock connector()
    CP ->o CP: start EnergyOffer()
    CP -> CP: Store StartTransaction.req(offline=TRUE)

note over EV, CS
Connection loss can be minutes, but can also be days.
```
CP ->o CP: connection restored()

CP -> CS: Heartbeat.req()
activate CP
activate CS
CS --> CP: Heartbeat.conf()
deactivate CP
deactivate CS

CP ->o CP: send queued message()

CP -> CS: StartTransaction.req(offline=TRUE)
activate CS
CS --> CP: StartTransaction.conf()
deactivate CS
alt if authorized

note right of CP
Continue Regular Charging Session Ref #1
end note

alt if not authorized

note right of CP
Continue Start Charging Session-id not Accepted
end note

|||
end

deactivate CP
deactivate EV
@enduml

### 1.2. OfflineStopTransaction.plantuml

@startuml
skinparam dpi 300
hide footbox

actor "User" as EV
participant "Charge Point" as CP
participant "Central System" as CS

note over EV, CS
Charge Point is Offline and a transaction is ongoing.
end note
EV -> CP: Present IdToken()
activate EV
activate CP

CP --&gt; EV: optional notification

note over CS
Parent id check must
be done on local
authorization list and
/ or authorization
cache if available.
end note

opt if (id) startId) or (ParentId = ParentId of startId)

CP -&gt;o CP: stop EnergyOffer()
opt if cable not permanently attached
   CP -&gt;o CP: unlock connector()
end

CP -&gt; CP: Store StopTransaction.req(offline=TRUE)

note over EV, CS
Connection loss can be minutes, but can also be days.
end note

CP -&gt;o CP: connection restored()

CP -&gt; CS: Heartbeat.req()
activate CP
activate CS
CS --&gt; CP: Heartbeat.conf()
deactivate CP
deactivate CS

CP -&gt;o CP: send queued message()

CP -&gt; CS: StopTransaction.req(offline=TRUE)
activate CS
CS --&gt; CP: StopTransaction.conf()
deactivate CS
deactivate EV
deactivate CP
end

@enduml
1.3. OfflineTransaction.plantuml

@startuml
skinparam dpi 300
hide footbox

participant "Charge Point" as CP
participant "Central System" as CS

note over CP, CS
Charge Point is Offline and a transaction has occurred.
end note

activate CP
CP ->o CP: Connection restored()
CP -> CS: Heartbeat.req()
activate CS
CS --> CP: Heartbeat.conf()
deactivate CS

CP ->o CP: send queued message()

loop for all transactions since connection lost

CP -> CS: StartTransaction.req(offline=TRUE)
activate CS
CS --> CP: StartTransaction.conf()
deactivate CS

opt
CP -> CS: MeterValues.req()
activate CS
CS --> CP: MeterValues.conf()
deactivate CS
|||
end

note right of CS
MeterValues can also
be put in the StopTransaction
transactionData field
end note

CP -> CS: StopTransaction.req(offline=TRUE)
activate CS
CS --> CP: StopTransaction.conf()
deactivate CS
|||
deactivate CP
end
@enduml