

Table of Contents

1. OCPI.....	3
1.1. OCPI 2.2.....	3

4.1.7.1. Platforms	22
4.1.7.2. Message Routing Headers	22
4.1.7.3. Broadcast Push	23
4.1.7.4. Open Routing Request	24
4.1.7.5. GET All via Hubs	25

4.1.7.6. Overview of required/optional routing headers for different scenarios	26
4.1.7.7. GET All via Hubs	29
4.1.7.8. Timestamps and Objects send via Hubs	29
4.2. Unique message IDs	30
4.3. Interface endpoints	30

8.2.1. Sender Interface	46
8.2.1.1. GET Method	46
8.2.2. Receiver Interface	48
8.2.2.1. GET Method	48
8.2.2.2. PUT Method	49

8.2.2.3. PATCH Method	50
8.3. Object description	51
8.3.1. <i>Location</i> Object	52
8.3.1.1. Example public charging location	53
8.3.1.2. Example destination charging location	55

9.2.1.3. Receiver Interface	75
9.2.1.4. GET Method	76
9.2.1.5. PUT Method	76
9.2.1.6. PATCH Method	77
9.3. Object description	78

9.3.1. <i>Session</i> Object	78
9.3.1.1. Examples.	79
9.3.2. <i>ChargingPreferences</i> Object	80
9.4. Data types	81
9.4.1. <i>ChargingPreferencesResponse</i> <i>enum</i>	81

11.4.5. <i>TariffDimensionType</i> <i>enum</i>	116
11.4.6. <i>TariffRestrictions</i> <i>class</i>	117
11.4.6.1. Example: <i>Tariff</i> with <i>max_power</i> <i>Tariff Restrictions</i>	118
11.4.6.2. Example: <i>Tariff</i> with <i>max_duration</i> <i>Tariff Restrictions</i>	119
11.4.7. <i>TariffType</i> <i>enum</i>	120

12. <i>Tokens</i> module	121
12.1. Flow and Lifecycle	121
12.1.1. Push model	121
12.1.2. Pull model.	121
12.1.3. Real-time authorization	121

14.3. Flow.	142
14.3.1. Example of setting/updating a ChargingProfile by the Sender (typically the SCSP or eMSP).	143
14.3.2. Example of a setting/updating a ChargingProfile by the SCSP via the eMSP	143
14.3.3. Example of a removing/clearing ChargingProfile sent by the Sender (typically the eMSP or SCSP)	144
14.3.4. Example of a removing/clearing ChargingProfile send by the SCSP via the eMSP	145

14.3.5. Example of a GET ActiveChargingProfile send by the Sender (typically the eMSP or SCSP)	145
14.3.6. Example of a GET ActiveChargingProfile send by the SCSP via eMSP	146
14.3.7. Example of the Receiver (typically the CPO) sending an updated ActiveChargingProfile	146
14.3.8. Example of the Receiver (typically the CPO) sending an updated ActiveChargingProfile to the SCSP via the eMSP	147

15.4.1. <i>ClientInfo</i> Object	159
15.5. Data types	159
15.5.1. <i>ConnectionStatus</i> <i>enum</i>	159
16. Types	160
16.1. <i>CiString</i> <i>type</i>	160

16.2. DateTime <i>type</i>	160
16.3. DisplayText <i>class</i>	160
16.4. number <i>type</i>	160
16.5. Price <i>class</i>	160
16.5.1. Role <i>enum</i>	161

Copyright © 2014 – 2019 NKL. All rights reserved.

This document is made available under the *Creative Commons Attribution- NoDerivatives 4.0 International Public License*

(<https://creativecommons.org/licenses/by-nd/4.0/legalcode>).

Version History

Version	Date	Author	Description
2.2	30-09-06-2019	Robert de Leeuw <i>ihomer</i>	Added support for Roaming Hubs Adds support for Platforms with multiple/different roles, additional
			changelog
			changelog
			changelog
			command module real-time authorization changelog

1. OCPI

1.1. OCPI 2.2

[Message routing headers](#)

[Hub Client Info](#)

[Support Platforms with multiple/different roles, additional roles](#)

[Charging Profiles](#)

[Preference based Smart Charging](#)

[CDRs](#)

[Sessions](#)

[Tariffs](#)

[Locations](#)

[Tokens](#)

[Commands](#)

[changelog](#)

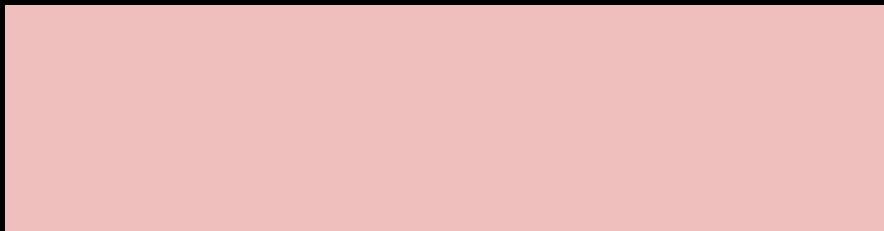
Starting in 2009, e-laad foundation and the predecessor of the eViolin association specified 2 standards in order to retrieve charge point details and active state. These are called the VAS interface and the Amsterdam interface. In this same period, a CDR format for the exchange of charge sessions between eViolin members was defined. This format is currently in use by the majority of the eViolin members. (eViolin is the branch organization for EV operators and service providers in NL and responsible for national

roaming and issuing of ID's). This resulted in 2014 in the development of OCPI.

An international group of companies already supports OCPI. Initiators are EV Box, The New Motion, ElaadNL, BeCharged, GreenFlux and Last Mile Solutions. Other participants include Next Charge, Freshmile, Plugsurfing, Charge-partner, Hubject, e-clearing.net, ihomer and Siemens. Several other major organizations and roaming platforms are interested in participating. The Netherlands Knowledge Platform for Charging Infrastructure (NKI) facilitates and coordinates this project to guarantee progress

<https://www.evroaming4.eu>

<https://www.nklnederland.nl/eciss>



<https://github.com/ocpi/ocpi>

2. Terminology and Definitions

2.1. Requirement Keywords

<https://www.ietf.org/rfc/rfc2119.txt>

not running their own platform. A lot of OI OS are also emulor. With OCPI 2.1.1 and earlier that meant having to set up an OCPI connection per role.

OCPI 2.2 introduces more roles and abstracts the role from the OCPI connection itself. OCPI 2.2 talks about Platforms connecting to Platforms, or Platforms connecting via Hubs to other Platforms. The Platform itself is not a role. The Platform provides services

for 1 or more roles.

Examples of platforms:

- A pure CPO: Not providing services to other CPOs. Not being an eMSP. Running its own software that connects via OCPI. Is defined in OCPI as a Platform has 1 CPO role, the CPO role of that company.

	CPO	eMSP	Hub	NSP	NAP	SCSP
CDRs						
Charging Profiles						
Commands						
Credentials						
Hub Client Info						
Locations						
Sessions						
Tariffs						
Tokens						
Versions						

	Broadcast push
	For example Locations, the CPO owns a Location (Charge Point), when a new Charge Point is added, the CPO calls PUT on the eMSP systems to inform them about new locations. See: Client Owned Objects

Term	Description
Configuration Module	OCPI Module needed to setup en maintain OCPI connections, but does not provide information for the EV driver: Credentials , Versions and Hub Client Info . Configuration Modules use message routing .
	Locations CDRs Tokens
	Open Routing Request

[eViolin](#)

[eViolin/Leden](#)

[bdew-codes.de](#)

[Provider ID List](#) <https://bdew-codes.de/Codenumbers/EMobilityId/ProviderIdList>

[EVSE Operator ID List](#) <https://bdew-codes.de/Codenumbers/EMobilityId/OperatorIdList>

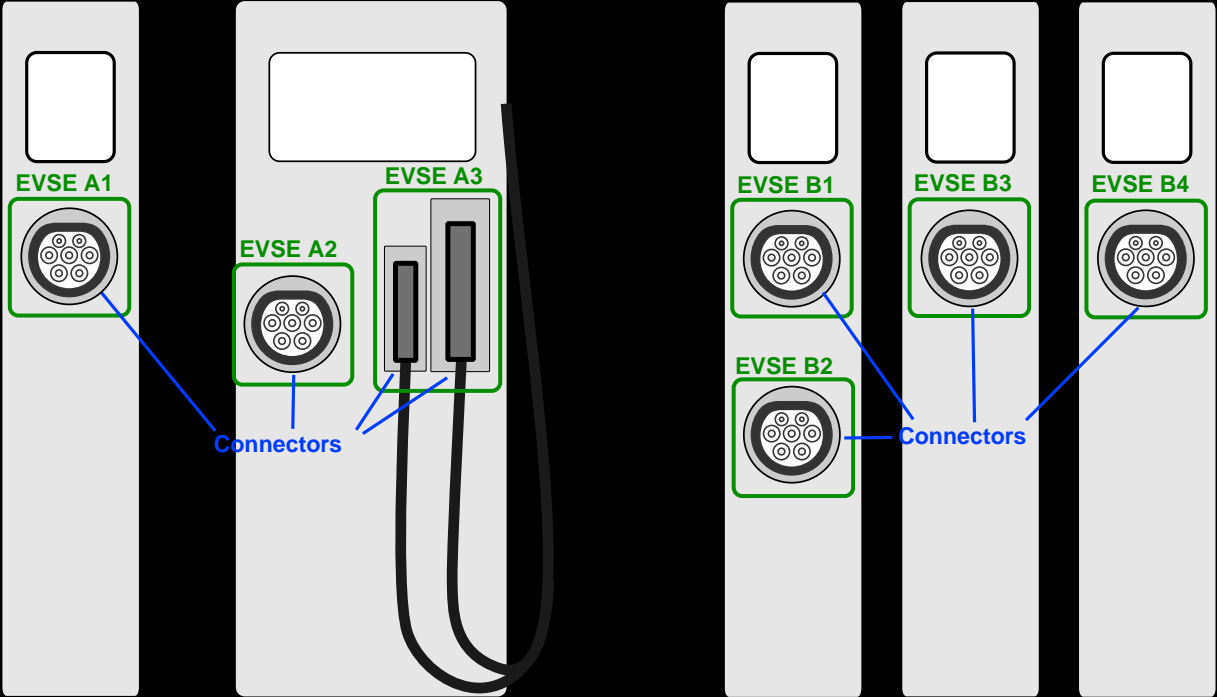
[austrian-](#)

[mobile-power.at](#)

The AFIREV organization will keep/keeps the registry for France. It provides operation id for CPO and emSP in compliance with eMI3 id structure. The prefix of these Ids is the "fr" country code. AFIREV will also be in charge of the definition of EVSE-Id structure, Charging-Pool-Id structure (location), and Contract-Id structure for France. AFIREV bases its requirements and recommendations on eMI3 definitions.

2.6. Charging topology

The charging topology, as relevant to the eMSP, consists of three entities:



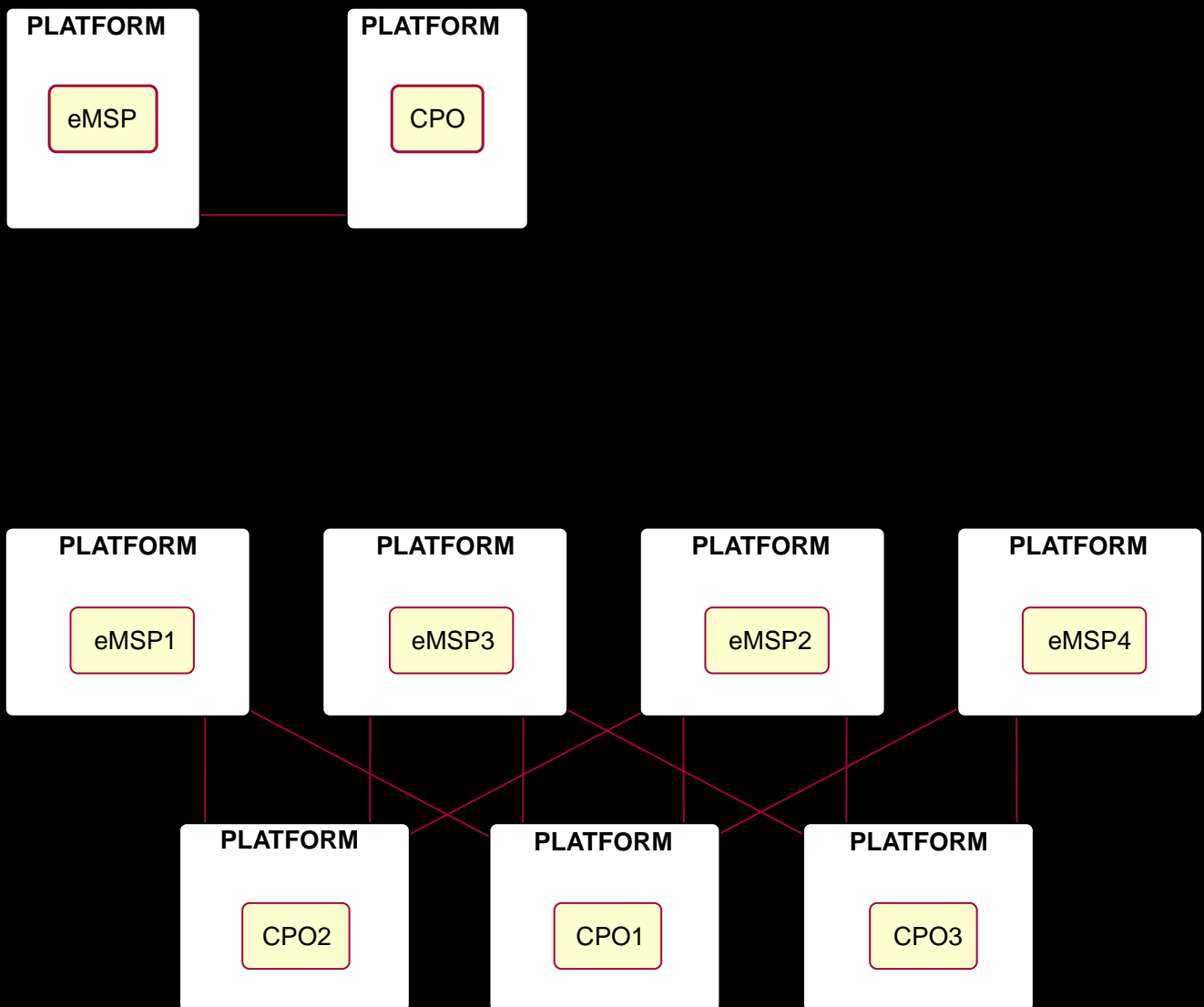
Symbol	Description	Type
?	An optional object. If not set, it might be <code>null</code> , or the field might be omitted. When the field is omitted and it has a default value, the value is the default value.	Object

Symbol	Description	Type
1	Required object.	Object
*	A list of zero or more objects. If empty, it might be <code>null</code> , <code>[]</code> or the field might be omitted.	[Object]
+	A list of at least one object.	[Object]

3. Supported Topologies

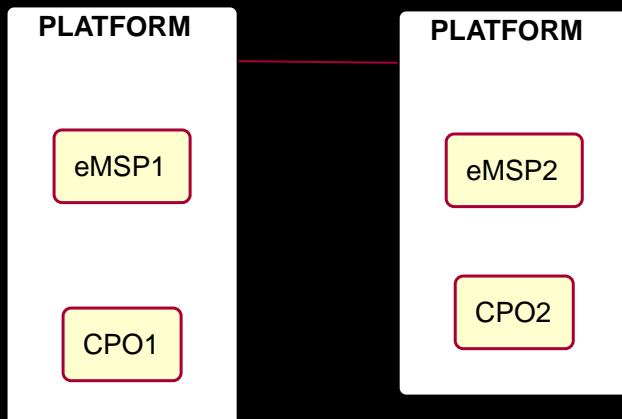
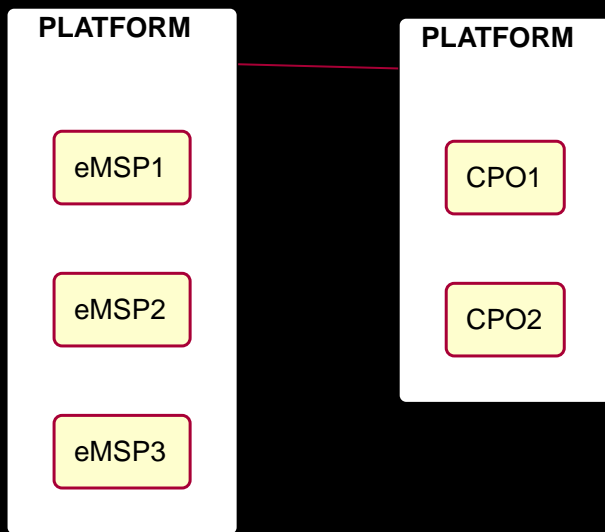
OCPI started as a bilateral protocol, for peer-to-peer communication. Soon parties started to use OCPI via Hubs, but OCPI 2.1.1 and earlier were not designed for that. OCPI 2.2 introduces a solution for this: [message routing](#).

EV Charging Market Roles



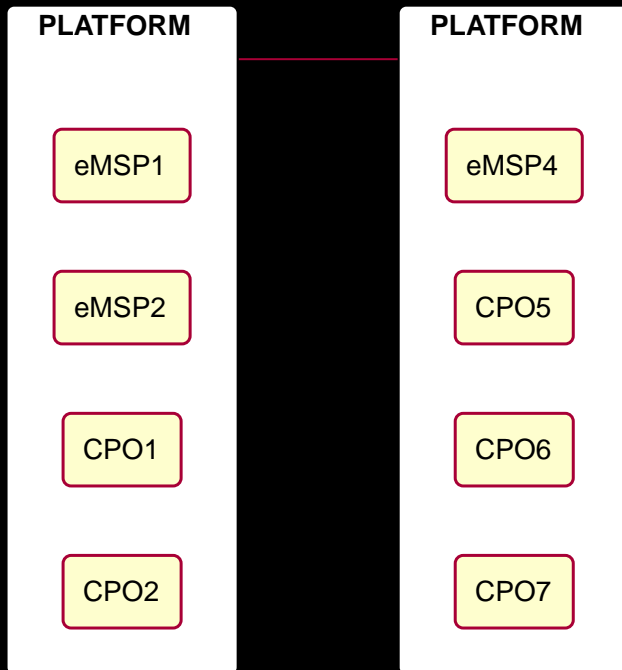
3.3. Peer-to-peer multiple the same roles

Some parties provide for example CPO or eMSP services for other companies. So the platform hosts multiple parties with the same role. This topology is a bilateral connection: peer-to-peer between two platforms, and both platforms can have multiple roles.



3.5. Peer-to-peer mixed roles

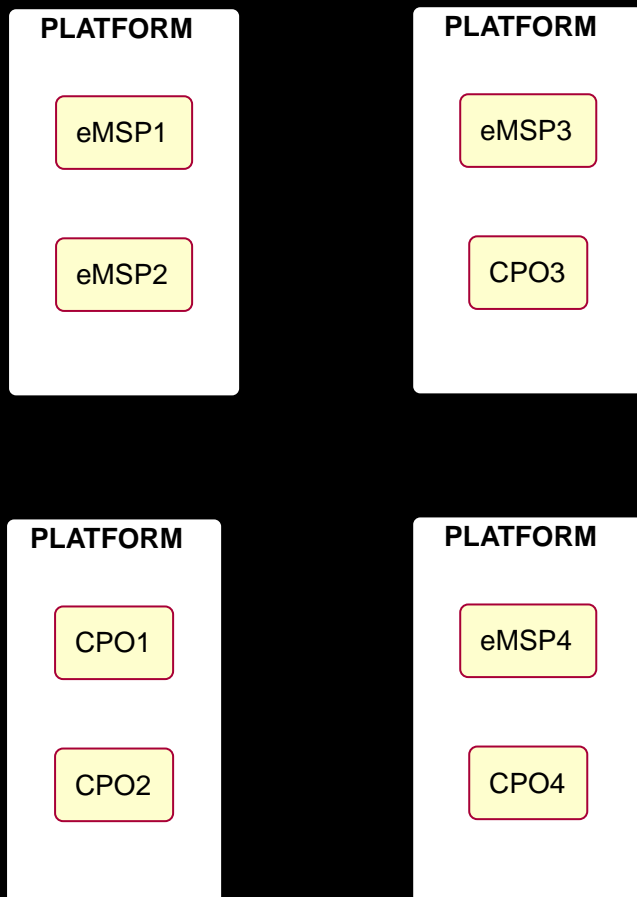
Some parties have dual roles, or provide them to other parties and then connect to other companies that do the same. This topology is a bilateral connection: peer-to-peer between two platforms, and both platforms have multiple different and also the same roles.



3.6. Multiple peer-to-peer

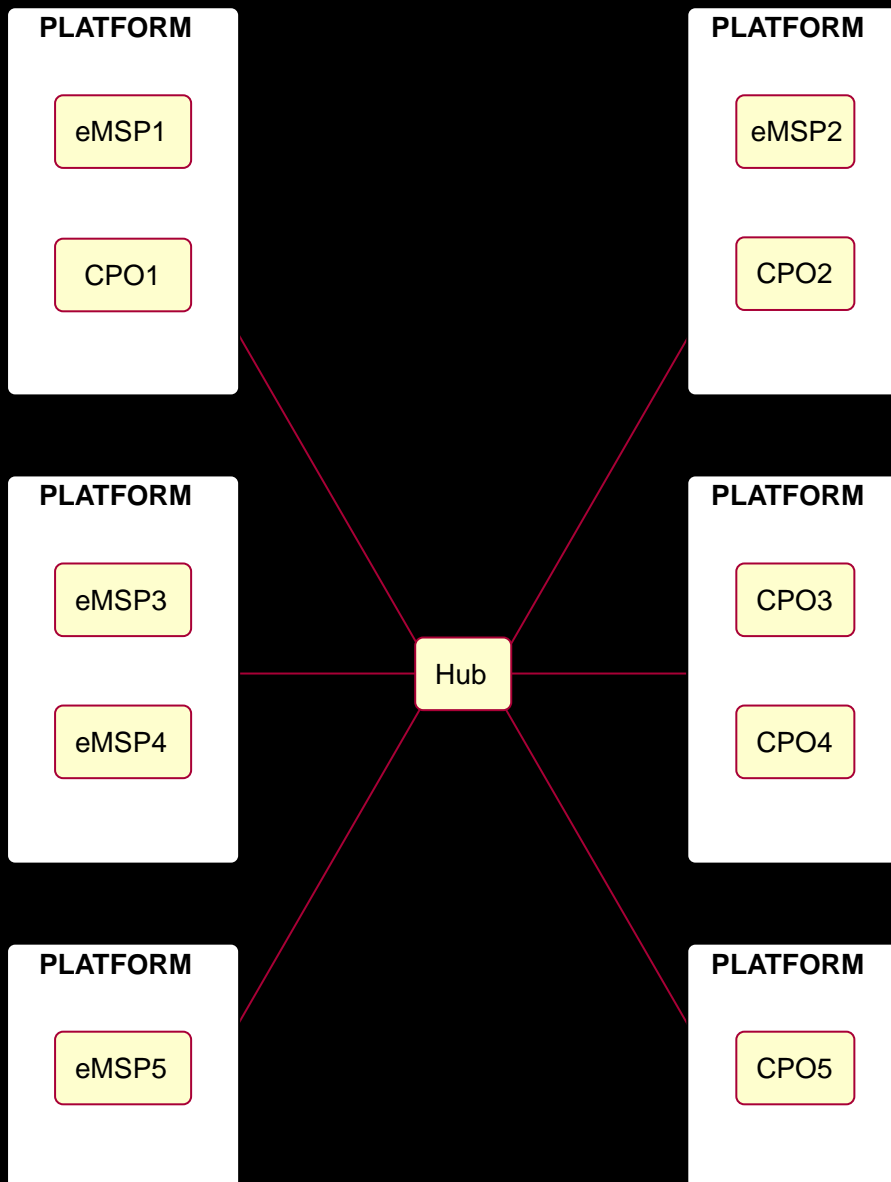
More a real-world topology when OCPI is used between market parties without a hub, all parties are platforms with multiple roles.

Disadvantage of this: requires a lot of connections between platforms to be setup, tested and maintained.



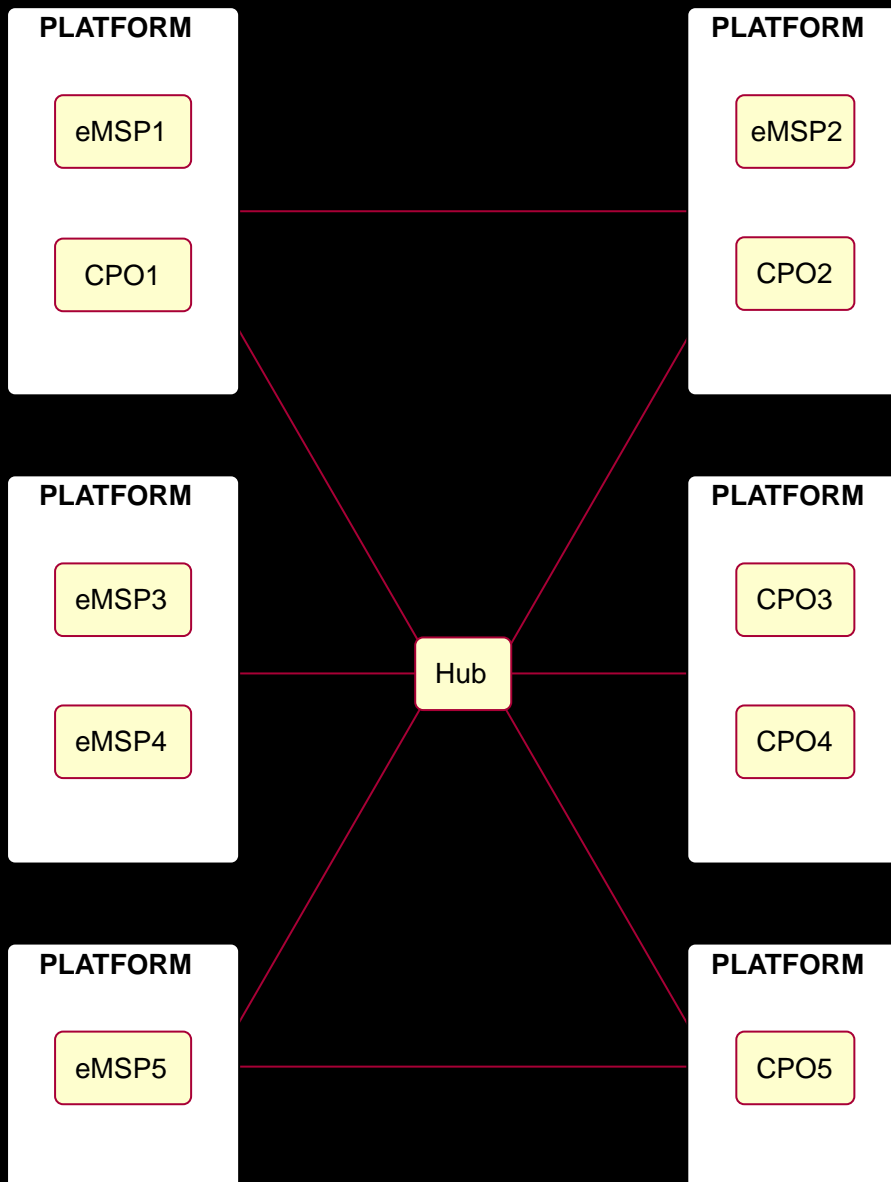
3.7. Platforms via Hub

This topology has all Platforms only connect via a Hub, all communication goes via the Hub.



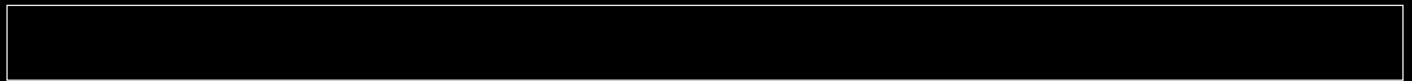
3.8. Platforms via Hub and direct

Not all Platforms will only communicate via a Hub. There might be different reasons for Platforms to still have peer-to-peer connections. The Hub might not yet support new functionality. The Platforms use a custom module for some new project, which is not supported by the Hub, etc.



4. Transport and format

4.1. JSON / HTTP implementation guide



'Credentials tokens'
Token Module

credentials module

Tokens

CREDENTIALS_TOKEN_A

credentials versions

4.1.4. Request format

The request method can be any of [GET](#), POST, [PUT](#), [PATCH](#) or DELETE. The OCPI protocol uses them in a way similar to REST APIs.

Method	Description
GET	Fetches objects or information.
POST	Creates new objects or information.
PUT	Updates existing objects or information.
PATCH	

	DateTime	
	DateTime	

Paginated Responses

For pagination to work correctly, it is important that multiple calls to the same URL (including query parameters): result in the same objects being returned by the server. For this to be the case, the sequence of objects mustn't change, or as little as possible. It is best practice to return the oldest (by creation date, not the `last_updated` field) first. While a client crawls over the pages (multiple GET requests every time to the 'next' page Link), a new object might be created on the server. The client detects this: the x-

Total-Count will be higher on the next call. But the client doesn't have to correct for this. Only the last page will be different (or an additional page). So the client will not be required to crawl all pages all over again when the client has reached to last page it has retrieved all relevant pages and is up to date.

Some query parameters can cause concurrency problems. For example the `date_to` query parameter. When


```
https://www.server.com/ocpi/cpo/2.2/cdrs/?limit=2000
```

The server should return (when the server has enough objects and the limit is the amount of objects the server wants to send is

100.) The `X-Limit` HTTP header should be set to 100 as well.

```
Link: <https://www.server.com/ocpi/cpo/2.2/cdrs/?offset=100&limit=100>; rel="next"
```



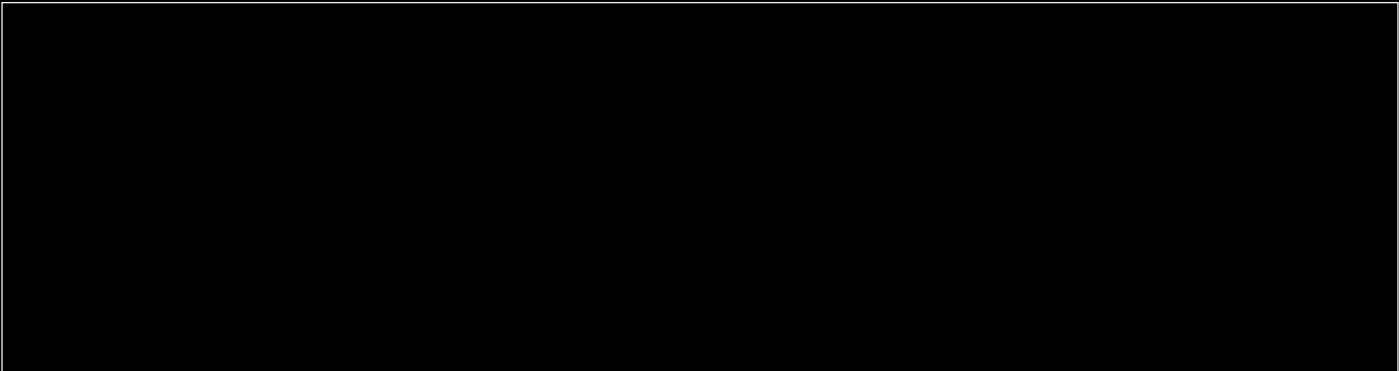
When a client pushes a Client Owned Object, but the {object-id} in the URL is different from the id in the object being pushed, server implementations are advised to return an [OCPI status code: 2001](#).

4.1.6. Response format

The content that is sent with all the response messages is an 'application/json' type and contains a JSON object with the following properties:

			Status Codes
	string		
	DateTime		

w3.org



4.1.6.3. Example: Tokens GET Response with one Token object. (CPO end-point) (one object)

```
{
  "data": {
    "uid": "012345678",
    "type": "RFID",
    "contract_id": "FA54320",
    "visual number": "DF000-2001-8999",
```

4.1.7. Message Routing

When the development of OCPI was started, it was designed for peer-to-peer communication between CPO and eMSP. This has advantages, but also disadvantages. Having to set up and maintain OCPI connections to a lot of parties requires more effort than

doing it for only a couple of connections. By communication via one or more Hubs, the amount of OCPI connections is reduced, while still being able to offer roaming to a lot of different parties and customers.

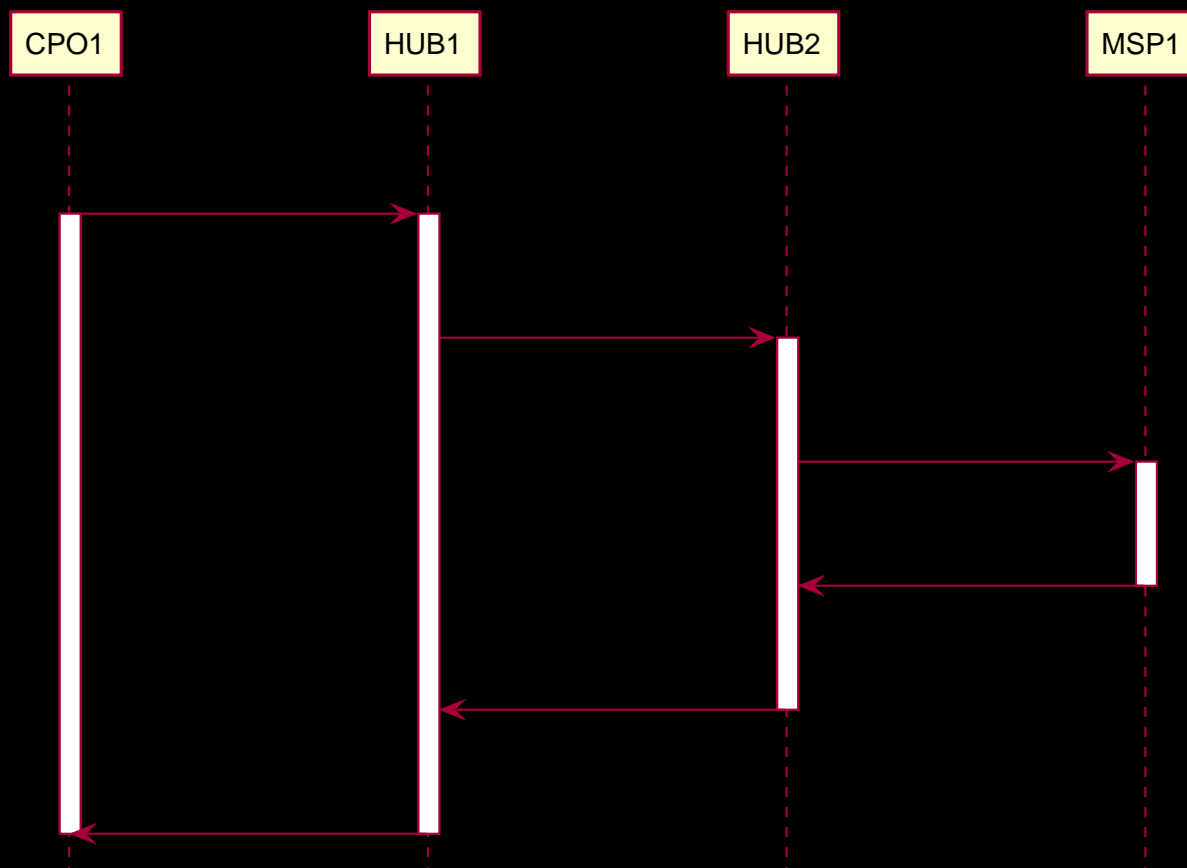
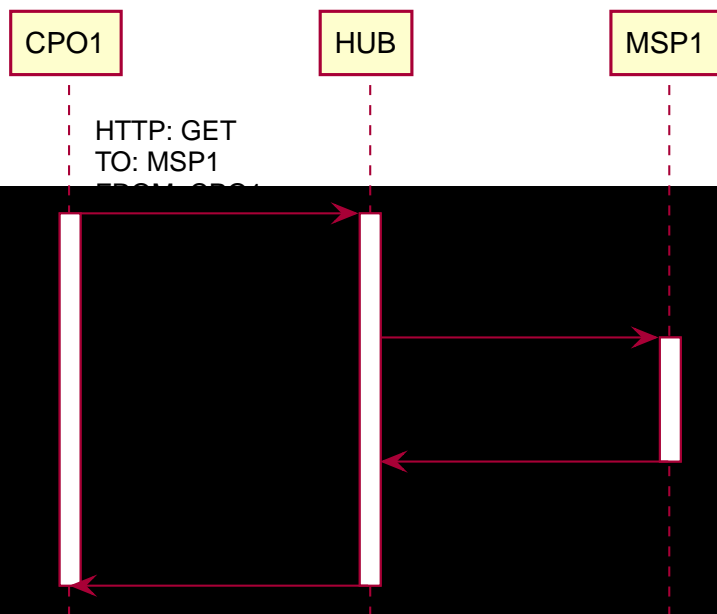
With the introduction of Message Routing, OCPI is now better usable for communication via Hubs.

All examples/sequence diagrams in this section use the roles CPO and eMSP as examples, they could be switched, it could be

Tokens Locations CDRs

Credentials Versions Hub Client Info

	CiString	
	CiString	
	CiString	
	CiString	

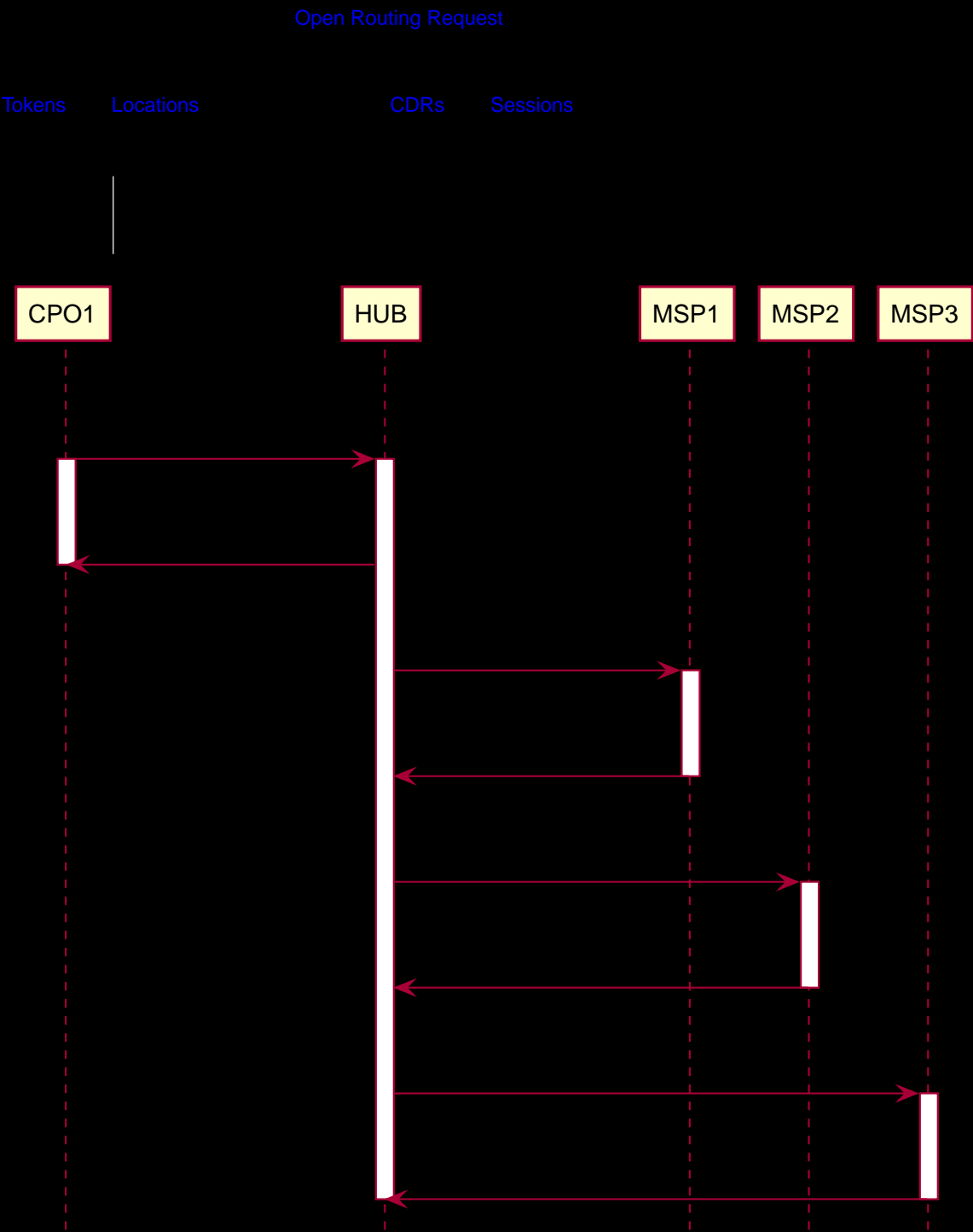


eMSPs though.)

When using Broadcast Push, the Hub broadcasts received information to all connected clients. To send data through a Hub might be very useful to share information like Locations or Tokens with all parties connected to the Hub that have implemented the corresponding module. This means only one request to the Hub will be necessary, as all connected clients will be served by the

Hub.

To send a Broadcast Push, the client uses the party-id and country-code of the Hub in the 'OCPI-to-' headers. The Hub parses the request and sends a response to the client, which optionally contains its own party-id and country-code in the 'OCPI-from-' headers. The Hub then sends the pushed data to any client implementing the corresponding applicable module, using its own party-id and country-code in the 'OCPI-from-' headers. The client receiving a Push from a Hub (with the Hub's information in the

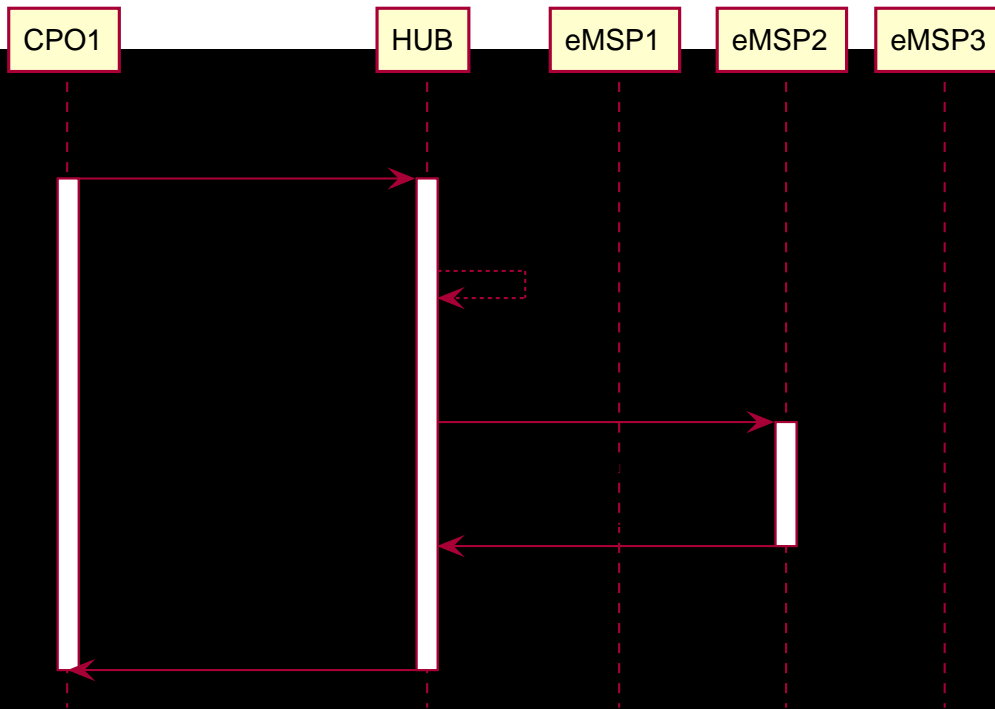


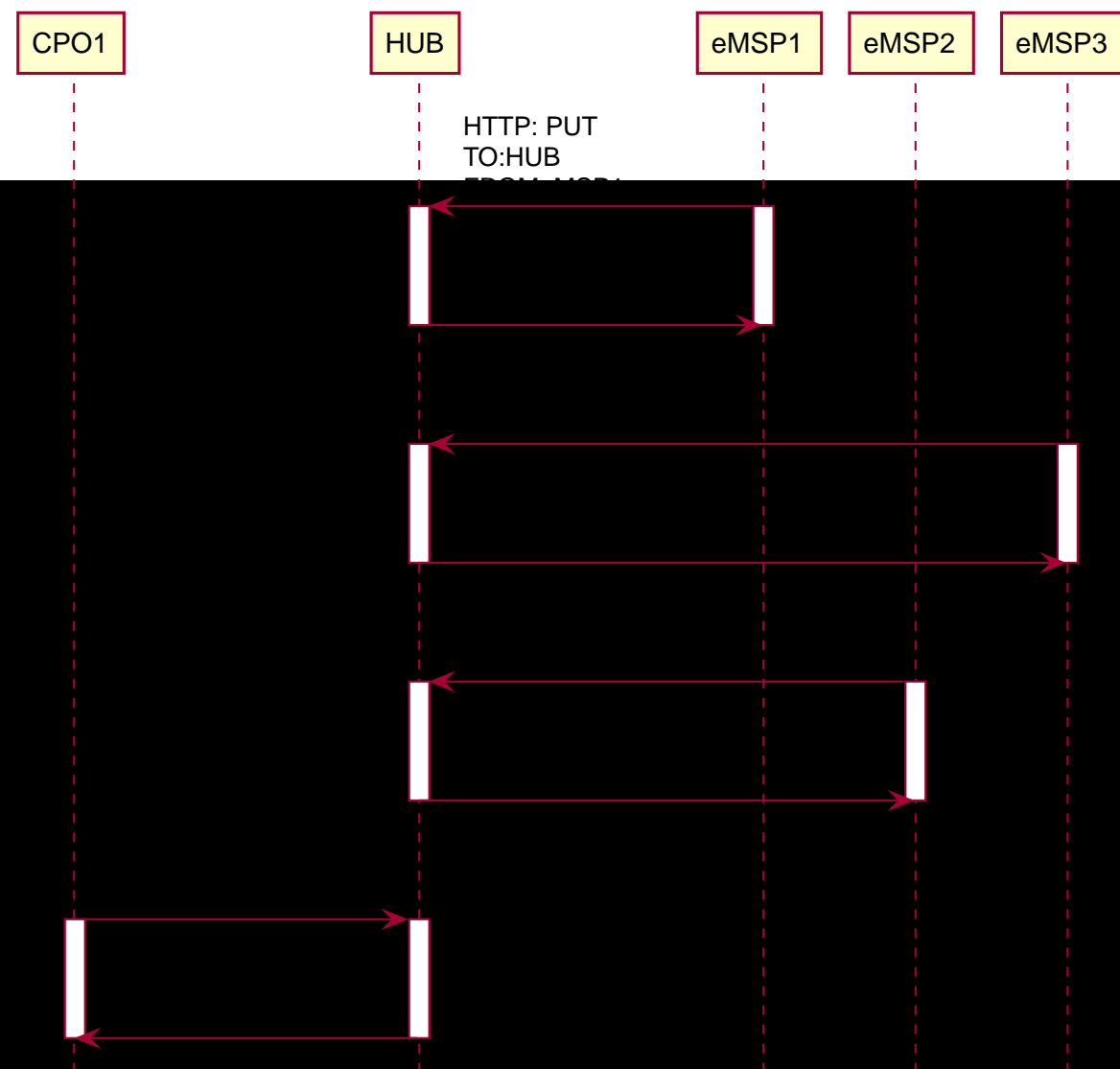
4.1.7.4. Open Routing Request

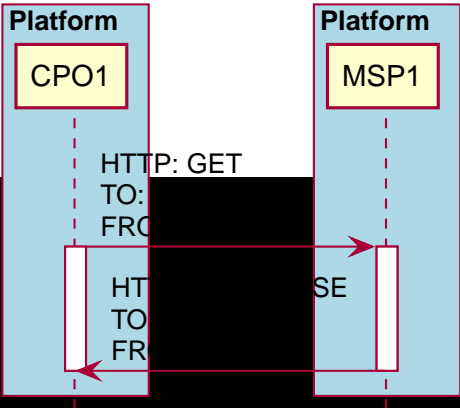
When a Hub has the intelligence to route messages based on the content of the request, or the requesting party does not know the destination of a request, the 'OCPI-to-' headers can be omitted in the request towards the Hub. The Hub can then decide to which party a request needs to be routed, or that it needs to be broadcasted if the destination cannot be determined.

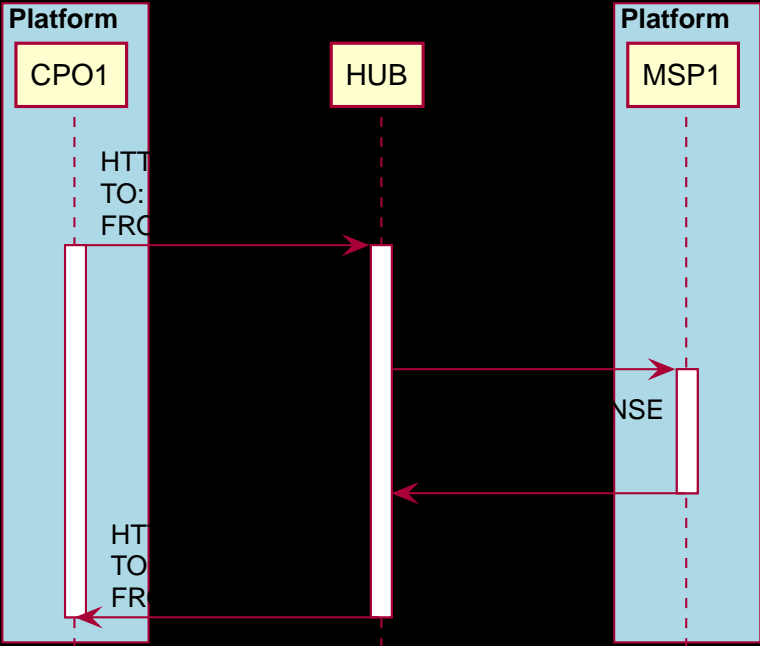
This has nothing to do with [Broadcast Push](#) though, as [Broadcast Push](#) only works for the [Push model](#), not for [GET](#) requests.

Open Routing Requests are possible for GET ([Not GET ALL](#)), POST, PUT, PATCH and DELETE.





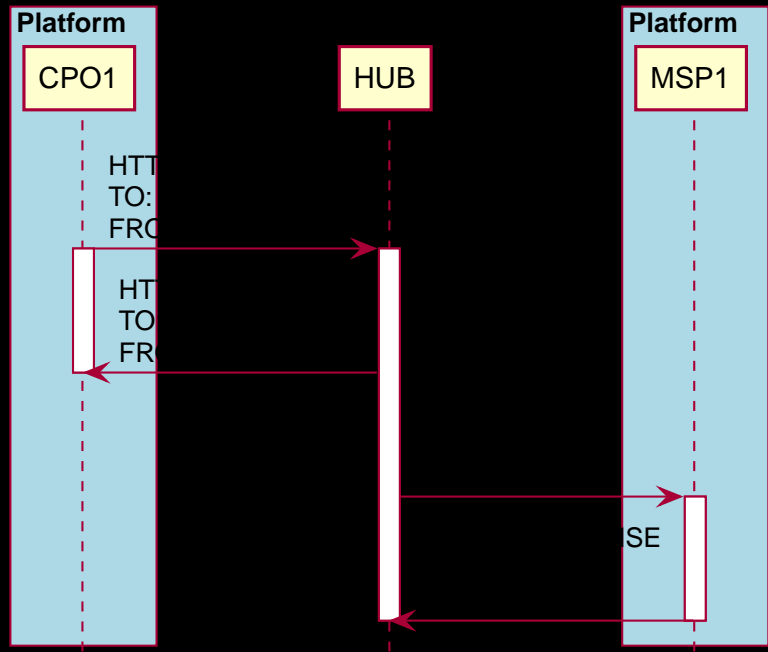




Push to the Hub. Broadcast

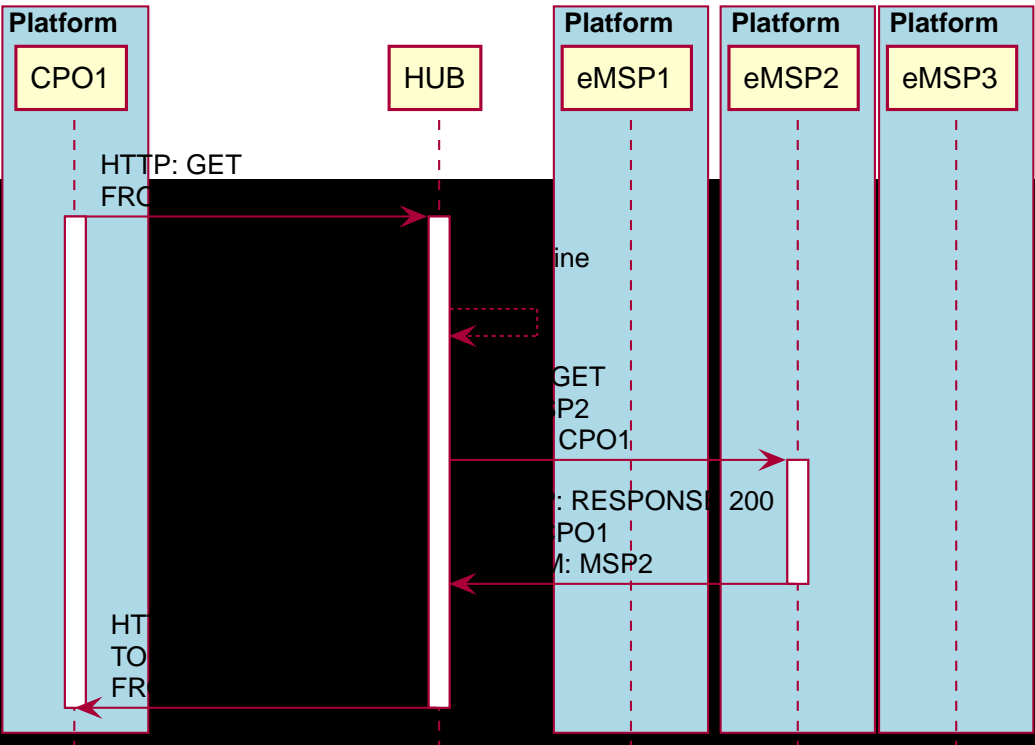
Name	Route	TO Headers	FROM Headers
Broadcast request	Requesting platform to Hub	Hub	Requesting-party

Name	Route	TO Headers	FROM Headers
Broadcast response	Hub to requesting platform	Requesting-party	Hub
Broadcast request	Hub to receiving platform	Receiving-party	Hub
Broadcast response	Receiving platform to Hub	Hub	Receiving-party

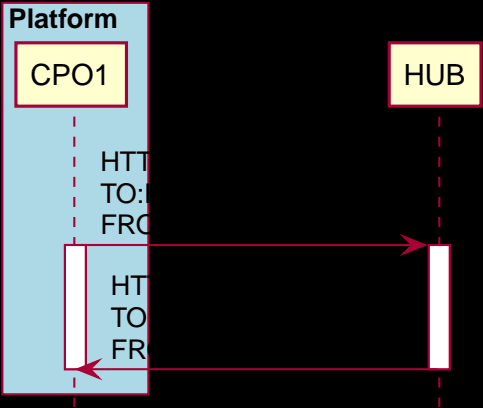


the routing of a request

needs to be determined by the Hub itself



GET All via a Hub

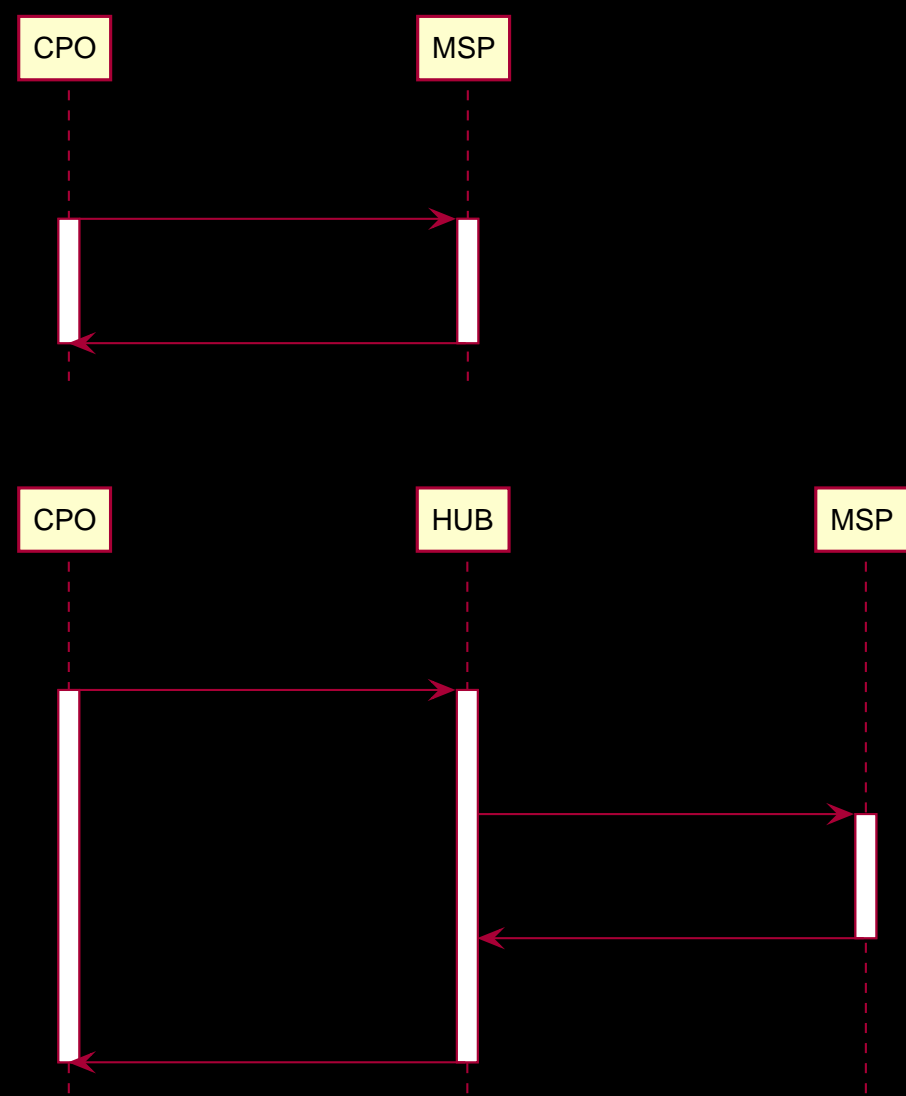


When GET Objects are sent via Hubs, the Table_updated field SHALL NOT be updated by the HUB.

4.2. Unique message IDs

For debugging issues, OCPI implementations are required to include unique IDs via HTTP headers in every request/response.

HTTP Header	Description



4.3. Interface endpoints

As OCPI contains multiple interfaces. Different endpoints are available for messaging. The protocol is designed such that the exact URLs of the endpoints can be defined by each party. It also supports an interface per version.

The locations of all the version-specific endpoints can be retrieved by fetching the API information from the versions endpoint. Each version-specific endpoint will then list the available endpoints for that version. It is strongly recommended to insert the protocol version into the URL.

For example: `/ocpi/cpo/2.2/locations` and `/ocpi/emsp/2.2/locations`.

5. Status codes

There are two types of status codes:

• Transport-related (HTTP)

2001	Invalid or missing parameters
2002	Not enough information, for example: Authorization request with too little information.
2003	Unknown Location, for example: Command: START_SESSION with unknown location.

Code	Description
2004	Unknown Token, for example: 'real-time' authorization of an unknown Token.
29xx	Reserved range for custom client error status codes (2900-2999).

6. Versions module

Type: Configuration Module

This is the required base module of OCPI. This module is the starting point for any OCPI connection. Via this module, clients can [which versions](#) [which modules](#)

Version		

	VersionNumber		
	URL		

```
[
  {
    "version": "2.1.1",
    "url": "https://www.server.com/ocpi/2.1.1/"
  },
  {

```


	VersionNumber		
	Endpoint		

Property	Type	Card.	Description
identifier	ModuleID	1	Endpoint identifier.
role	InterfaceRole	1	Interface role this endpoint implements.

Property	Type	Card.	Description
url	URL	1	URL to the endpoint.

NOTE

for the `credentials` module, the role is not relevant as this module is the same for all roles.

Credentials & Registration

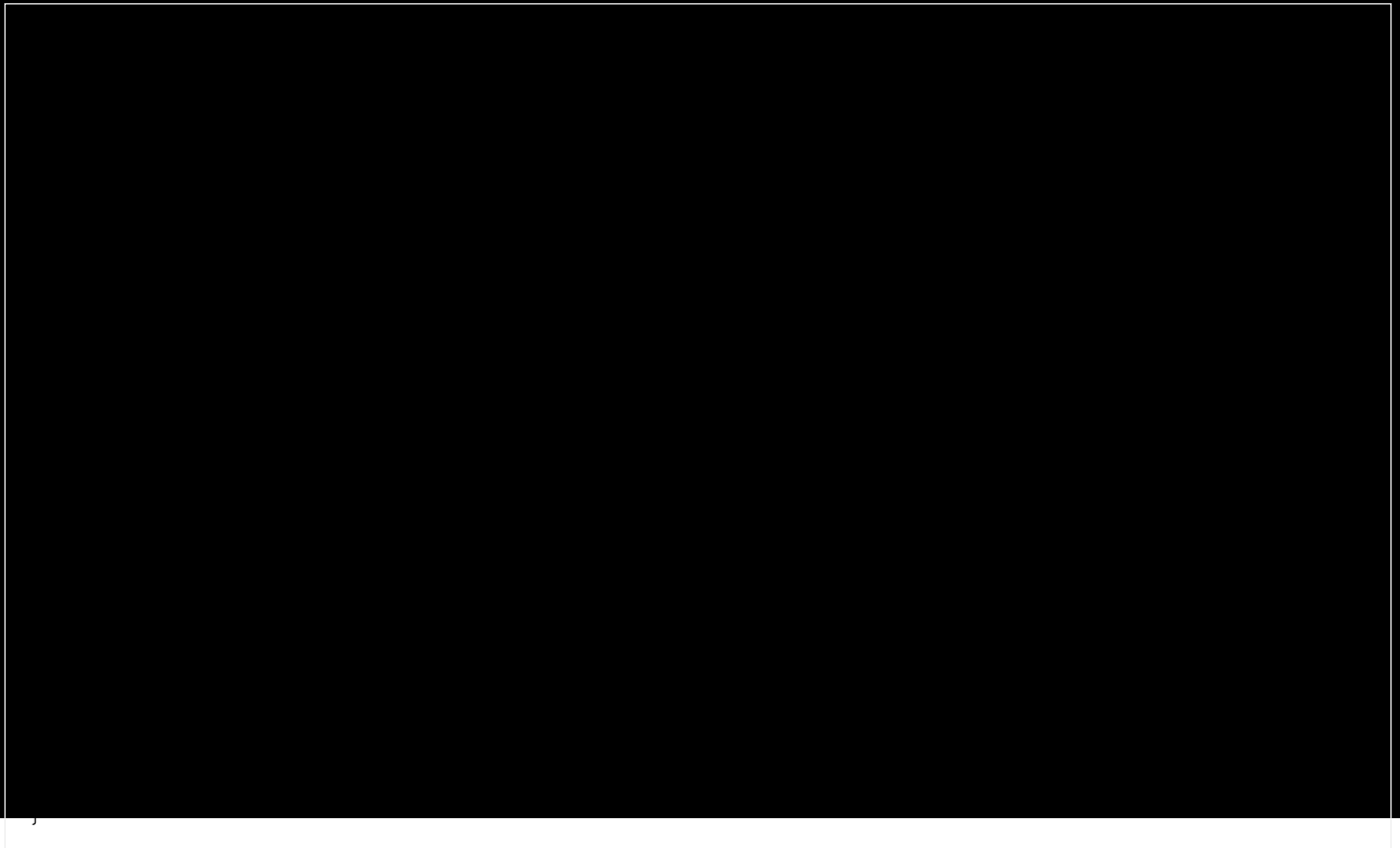
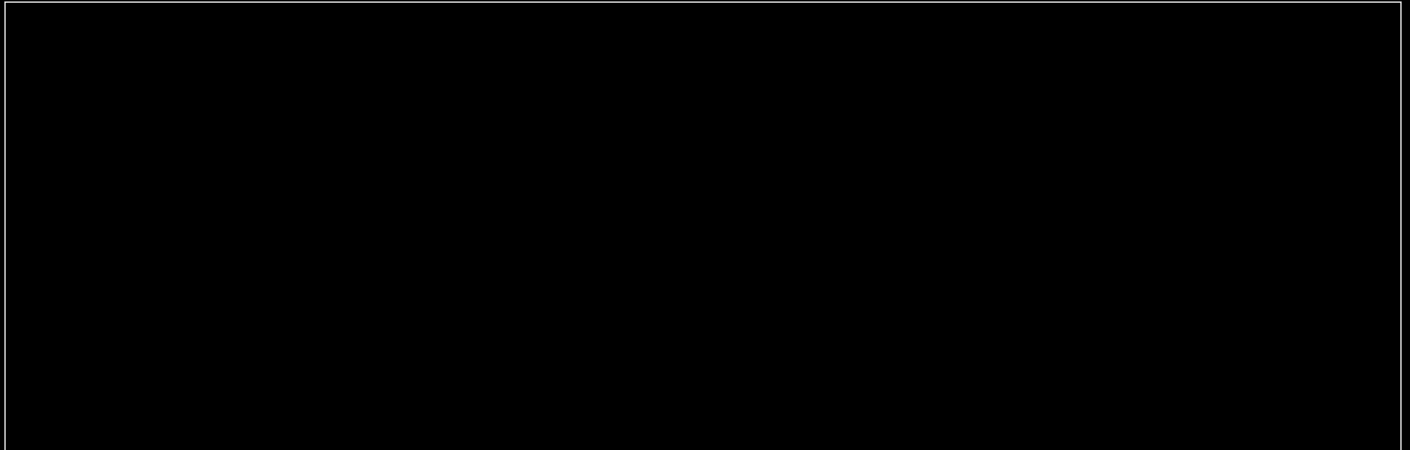
CDRs		
Charging Profiles		
Commands		
Credentials & Registration		
Hub Client Info		
Locations		
Sessions		
Tariffs		
Tokens		

Parties are allowed to create custom modules or customized versions of the existing modules. To do so, the `ModuleID` enum can be extended with additional custom moduleIDs. These custom moduleIDs MAY only be sent to parties with which there is an agreement to use a custom module. Do NOT send custom moduleIDs to parties you are not 100% sure will understand the custom moduleIDs. It is advised to use a prefix (e.g. country-code + party-id) for any custom moduleID, this ensures that the moduleID will

not be used for any future module of OCPI.

For example: `nltm-tokens`

6.2.6. GET



7. *Credentials* module

Module Identifier: `credentials`

Type: Configuration Module

[HTTP Authorization header](#)

OCPI Registration process

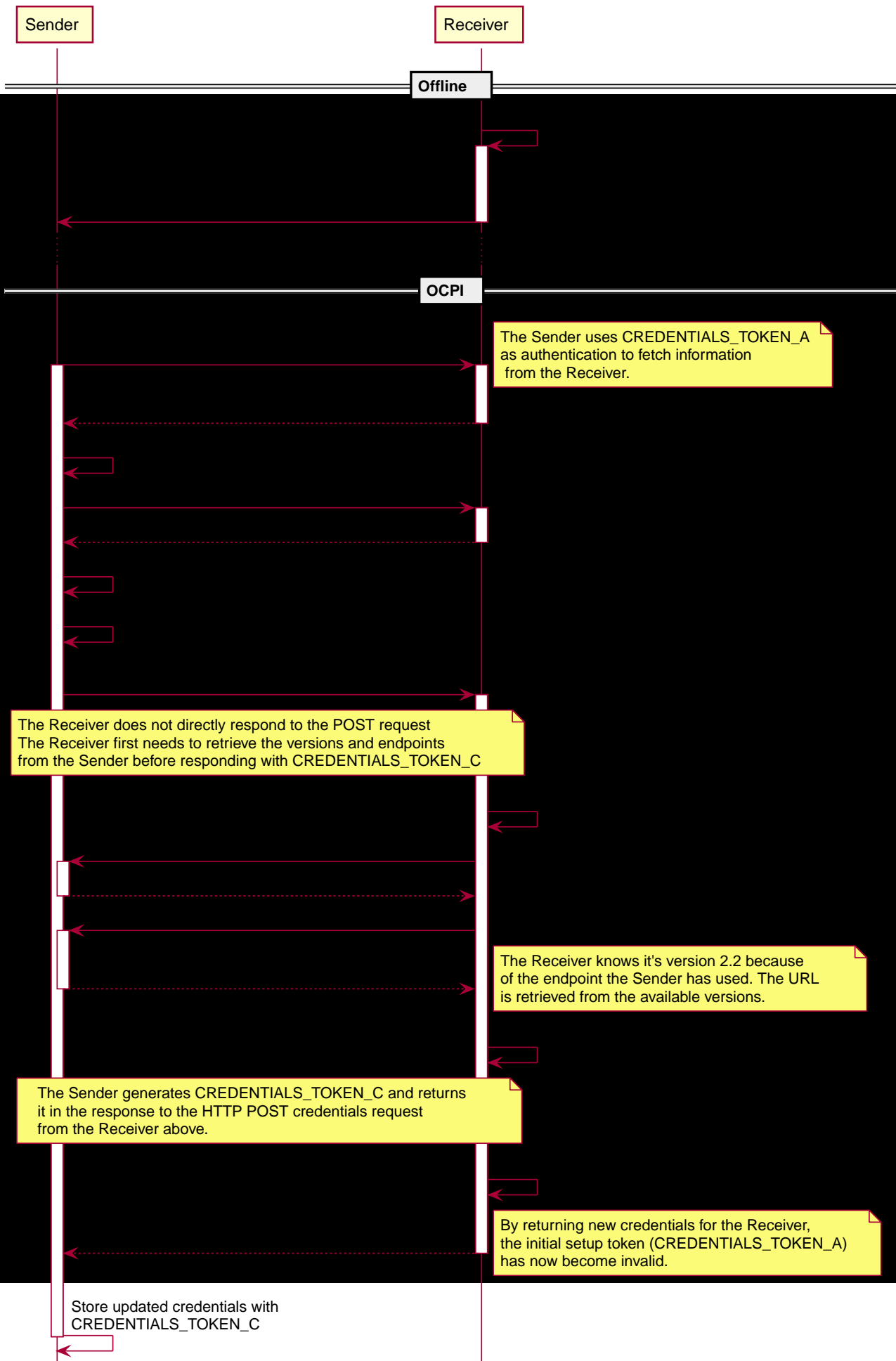


Figure 22. The OCPI registration process

Due to its symmetric nature of the credentials module, any platform can be Sender and or the Receiver for this module.

7.1.2. Updating to a newer version

At some point, both platforms will have implemented a newer OCPI version. To start using the newer version, one platform has to



The credentials (or parts thereof, such as the credentials token) can be updated by sending the new credentials via a PUT request to the credentials endpoint of the current version, similar to the update procedure described above.

Security advices: When one of the connecting platforms suspects that a credentials token is compromised, that platform SHALL

initiate a credentials token update as soon as possible. It is advisable to renew the credentials tokens at least once a month, in case it was not detected that the credentials where compromised.

7.1.5. Errors during registration

3001

3003

GET	
POST	
PUT	
DELETE	

A `POST` initiates the registration process for this endpoint's version. The server must also fetch the client's endpoints for this version.

If successful, the server must generate a new credentials token and respond with the client's new credentials to access the server's system. The credentials object in the response also contains extra information about the server such as its business details.

This method MUST return a HTTP status code 405: method not allowed if the client has already been registered before.

7.2.3. PUT Method

Provides the server with updated credentials to access the client's system. This credentials object also contains extra information

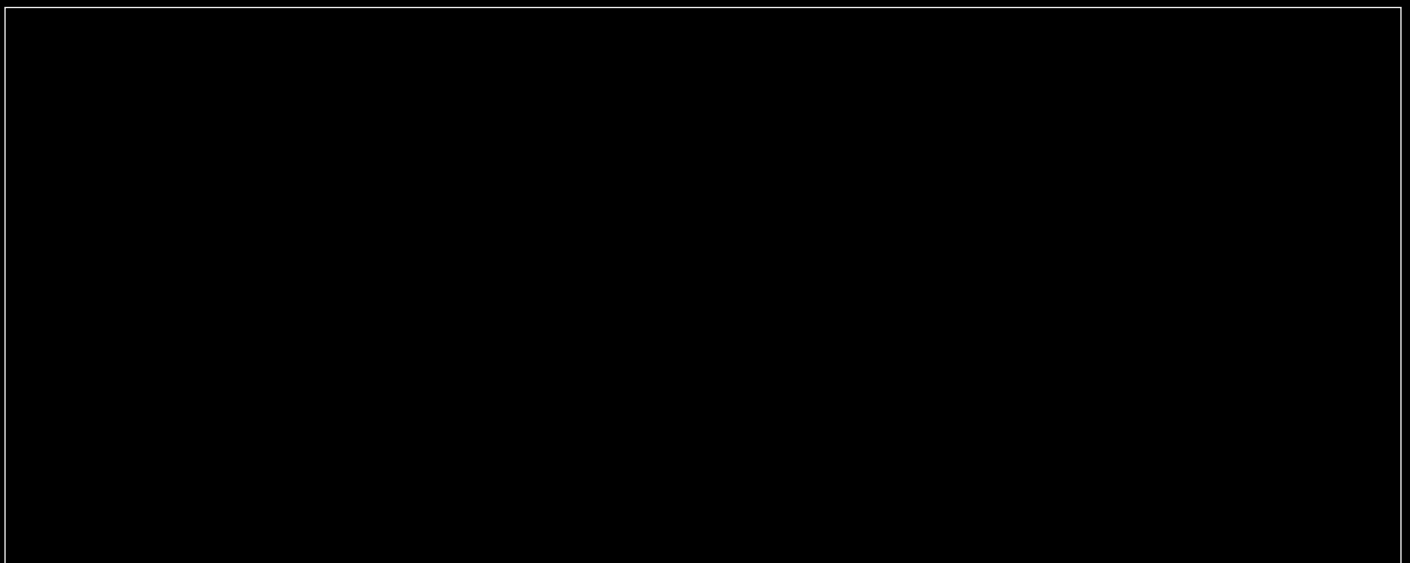
	string		
	URL		
	CredentialsRole		



OCPI party_id and/or the country_code.

7.3.2. Examples

Example of a minimal CPO credentials object:



Example of a CPO credentials object for a platform that provides services for 3 CPOs:

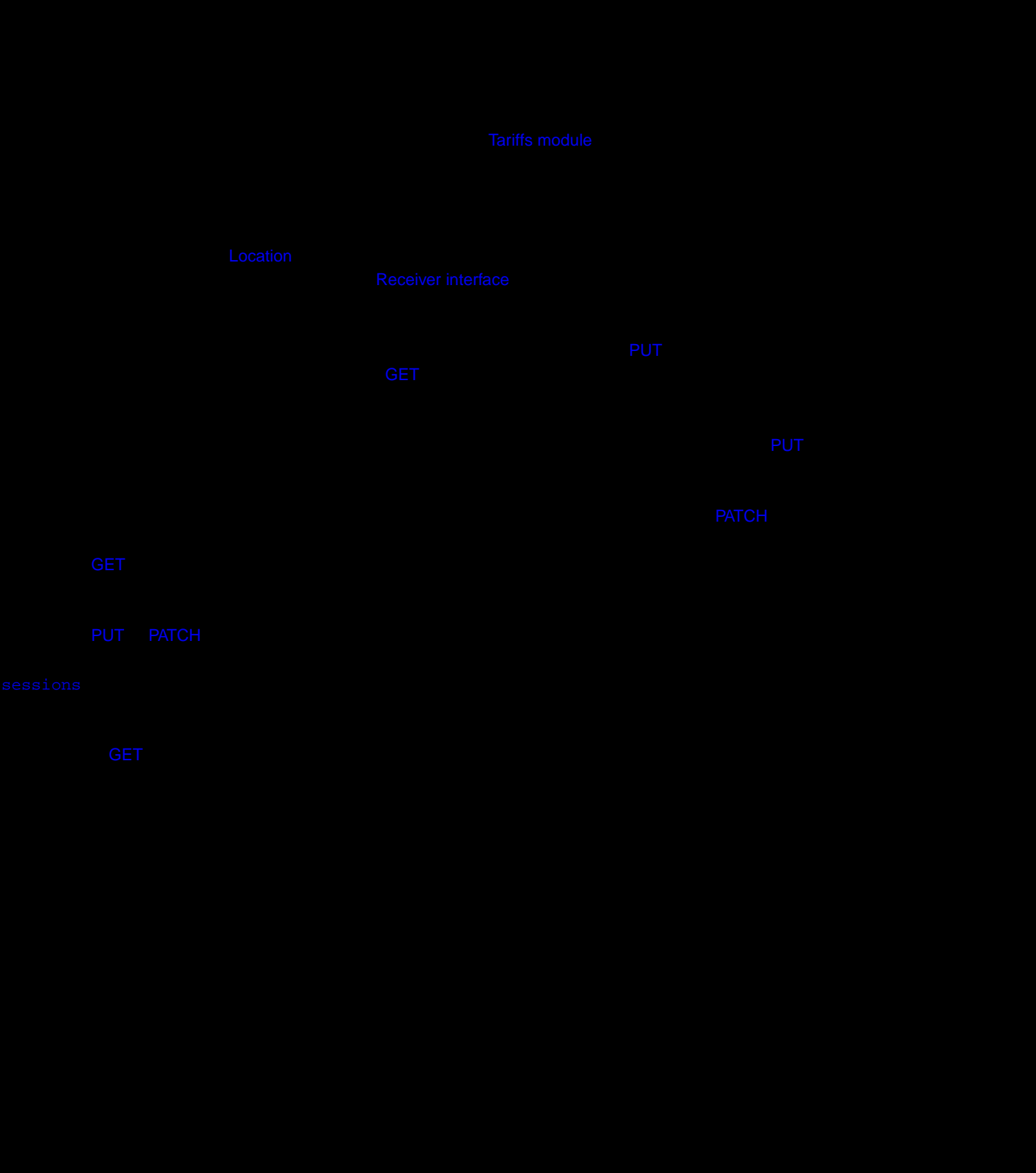
```
{
  "token": "9e80aca8-28be-11e9-b210-d663bd873d93",
  "url": "https://ocpi.example.com/versions/",
  "roles": [{
    "role": "CPO",
    "party id": "EXO",
```

	Role		
	BusinessDetails		
	CiString		
	CiString		

8. Locations module

Module Identifier: locations

Data owner: CPO



OCPI does not have this model:

- OCPI has Location at the highest level.

- Each location can have multiple EVSE
- Every EVSE can have one or more Connectors.

When mapping OCPP Charge Points to OCPI, there are 2 options:

|

GET Object interface

GET	paginated

Depending on the URL Segments provided, the GET request can either be used to retrieve information about a list of available Locations (with EVSEs and Connectors) at a CPO ([GET List](#)) or it can be used to retrieve information about one specific Location, EVSE or Connector ([GET Object](#)).

GET List: Request Parameters

Endpoint structure definition:

{locations_endpoint_url}?[date_from={date_from}][&[date_to={date_to}][&[offset={offset}][&[limit={1

paginated		pagination	
	DateTime		
	DateTime		

pagination

Location		

https://www.server.com/ocpi/cpo/2.2/locations/LOC1

https://www.server.com/ocpi/cpo/2.2/locations/LOC1/3256

https://www.server.com/ocpi/cpo/2.2/locations/LOC1/3256/1

The following parameters can be provided as URL segments in the same order.

Parameter	Datatype	Required	Description
location_id	CiString(36)	yes	Location.id of the Location object to retrieve.
	CiString		
	CiString		

Location		
EVSE		
Connector		

Client Owned Objects

party_idcountry_code

GET	
	PUT
PUT	
PATCH	
	PATCHFlow and Lifecycle

PATCH call.

Request Parameters

The following parameters can be provided as URL segments.

Parameter	Datatype	Requi	Description
	CiString		
	CiString		
	CiString		
	CiString		
	CiString		

Location		
EVSE		
Connector		

	Connector	EVSE
Location		
	EVSE	Location

	party_id	country_code
Receiver Interface		

	CiString		
	CiString		
	CiString		
evse_uid	CiString(36)	no	Evse.uid, required when an EVSE or Connector object is pushed.
connector_id	CiString(36)	no	Connector.id, required when a Connector object is pushed.

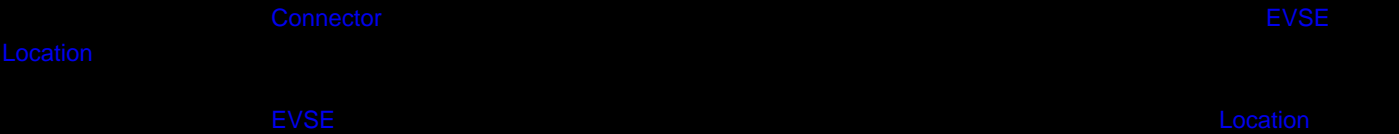
Request Body

The request body contains the new/updated object.

When the PUT contains a [Connector](#) Object, the Receiver SHALL also set the new `last_updated` value on the parent [EVSE](#) and [Location](#) Objects.

EVSE		Location
Location		
EVSE		
Connector		

PUT



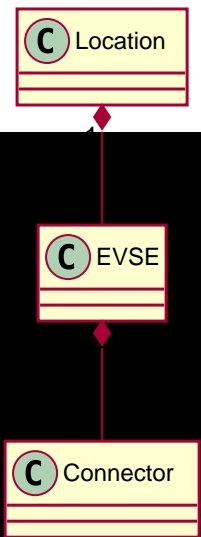
Example: a simple status update

This is the most common type of update message. It is used to notify eMSPs that the status of an EVSE changed. In this case it is the EVSE with uid 3255 of the Location with id 1012.

PATCH To URL: <https://www.server.com/ocpi/emsp/2.2/locations/NL/TNM/1012/3255>

```
{  
  "status": "CHARGING",  
  "last_updated": "2019-06-24T12:39:09Z"  
}
```

Locations class diagram



Token

Token

PublishToken

	CiString		
	CiString		
	CiString		
	PublishTokenType		

name	string(255)	?	Display name of the location.
address	string(45)	1	Street/block name and house number if available.
city	string(45)	1	City or town.

Property	Type	Card.	Description
postal_code	string(10)	?	Postal code of the location, may only be omitted when the location has no postal code: in some countries charging locations at highways don't have postal codes.
	string		
	string		
	GeoLocation		
	AdditionalGeoLocation		
	ParkingType		
	EVSE		
	DisplayText		
	BusinessDetails		Credentials
	BusinessDetails		
	BusinessDetails		
	Facility		
	string		http://www.iana.org/time-zones
	Hours		
	Image		
	EnergyMix		
	DateTime		

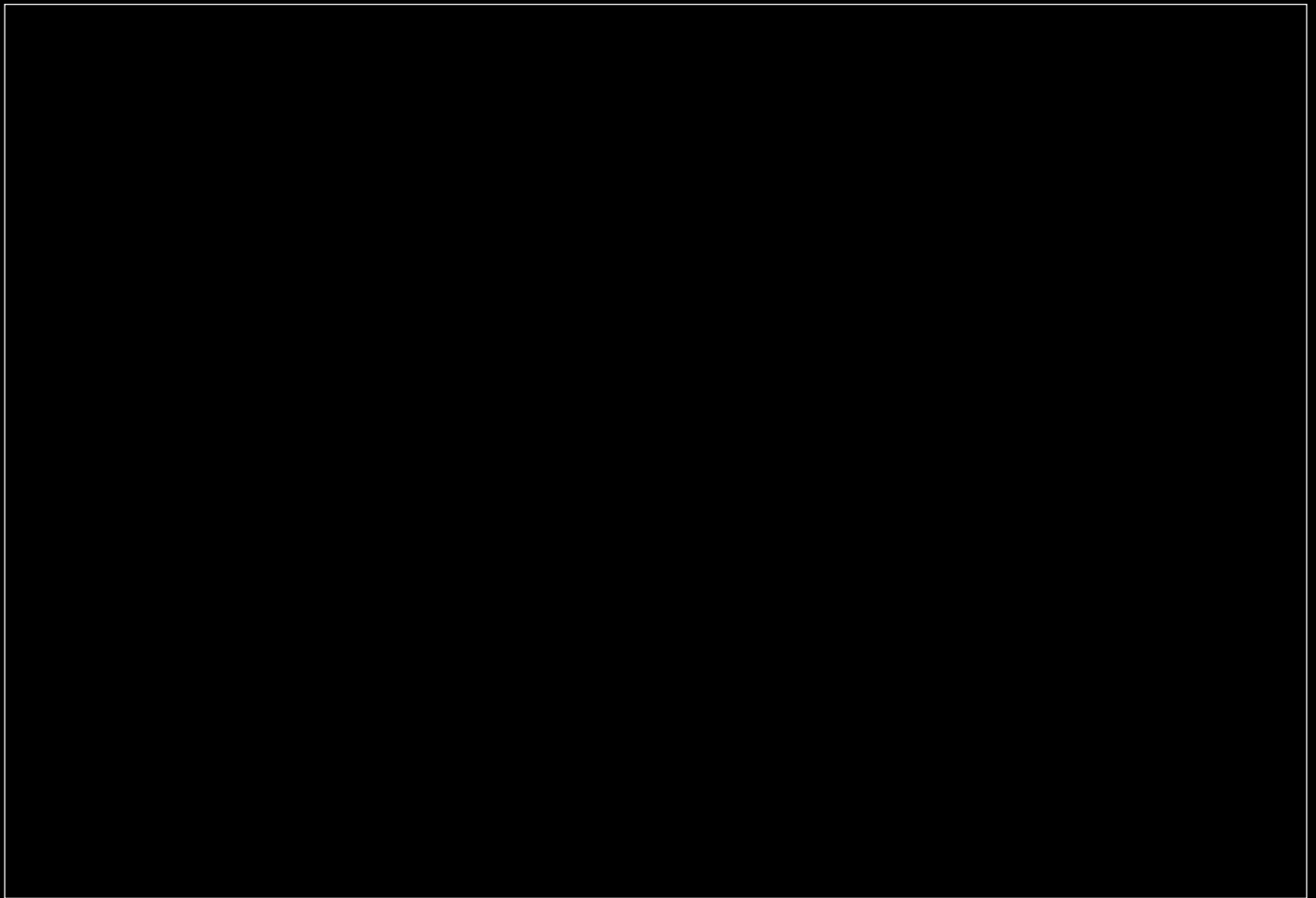
CDRs

```
{  
  "country_code": "BE",  
  "party_id": "BEC",  
  "id": "LOC1",  
  "publish": true,  
  "name": "Gent Zuid",
```

```
  "time_zone": "Europe/Brussels",  
  "last_updated": "2015-06-29T20:39:09Z"  
}
```

8.3.1.2. Example destination charging location

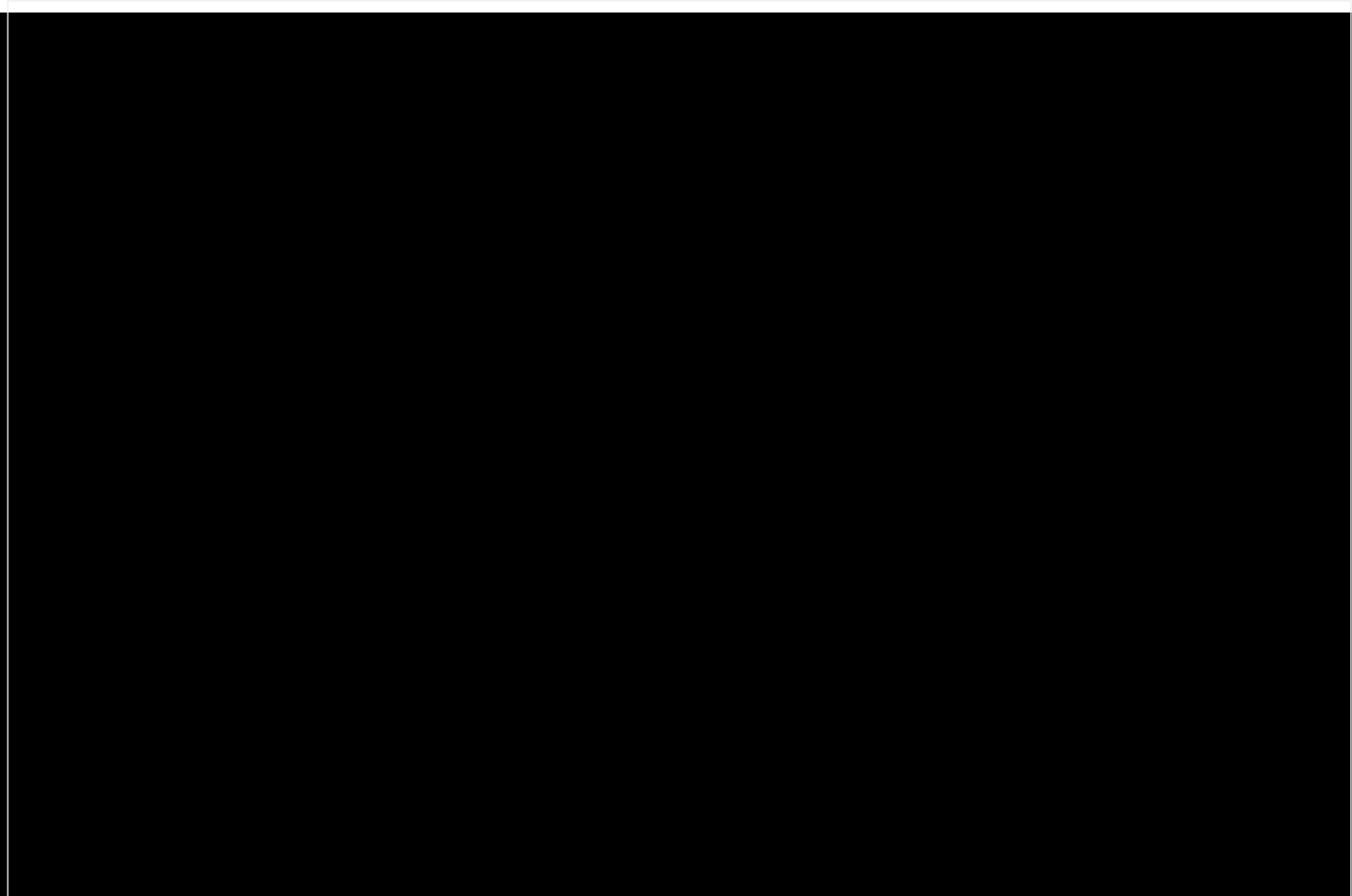
This is an example of a destination charging location. This is a Location where only guests, employees or customers can charge. For an EV driver, it can be useful to know if he/she can charge at his destination.



In case the EV driver is not billed for charging, there is, in such a case, no reason to publish the location via OCPI.

- `publish = false`

- `publish_allowed_to` not used
- `parking_type` = not used`
- `EVSE.parking_restrictions = CUSTOMERS` May still be useful so a support desk can also tell this to a customer.



Tokens

Tokens


```
{  
  "country_code": "NL",  
  "party_id": "ALL",  
  "id": "f76c2e0c-a6ef-4f67-bf23-6a187e5ca0e0",  
  "publish": false,  
  "publish allowed to": [{
```

Token

Tokens

```
{  
  "country_code": "DE",  
  "party_id": "ALL",  
  "id": "a5295927-09b9-4a71-b4b9-a5ffdfa0b77",  
  "publish": false,  
  "publish allowed to": [{
```

```
{
  "country_code": "SE",
  "party_id": "EVC",
  "id": "cbb0df21-d17d-40ba-a4aa-dc588c8f98cb",
  "publish": true,
  "name": "P-Huset Leonard",
```

8.3.2. EVSE Object

The *EVSE* object describes the part that controls the power supply to a single EV in a single session. It always belongs to a [Location](#) object. The object only contains directions to get from the location itself to the EVSE (i.e. *floor*, *physical_reference* or *directions*).

When the directional properties of an EVSE are insufficient to reach the EVSE from the *Location* point, then it typically indicates that the EVSE should be put in a different *Location* object (sometimes with the same address but with different coordinates/directions).

An *EVSE* object has a list of Connectors which can not be used simultaneously: only one connector per EVSE can be used at the

	CiString		
	CiString		http://emi3group.com/documents-links/
	Status		
	StatusSchedule		
	Capability		
	Connector		
	string		
	GeoLocation		
	string		
	DisplayText		
	ParkingRestriction		
	Image		
	DateTime		

EVSE

id	CiString(36)	1	Identifier of the Connector within the EVSE. Two Connectors may have the same id as long as they do not belong to the same <i>EVSE</i> object.
standard	ConnectorType	1	The standard of the installed connector.

Property	Type	Card.	Description
format	ConnectorFormat	1	The format (socket/cable) of the installed connector.
power_type	PowerType	1	
max_voltage	int	1	Maximum voltage of the connector (line to neutral for AC_3_PHASE), in
	CiString		<div>Tariff.type</div> <div>start_date_time end_date_time</div> <div>ProfileType</div>
	URL		
	DateTime		

	string		
	string		
	DisplayText		

	string		
website	URL	?	Link to the operator's website.
logo	Image	?	Image link to the operator's logo.

8.4.3. Capability *enum*

The capabilities of an EVSE.

Value	Description
	charging preferences
	started stopped
	reserved
	unlocked

DOMESTIC_D	Standard/Domestic household, type "D", 3 pin
DOMESTIC_E	Standard/Domestic household, type "E", CEE 7/5 3 pins
DOMESTIC_F	Standard/Domestic household, type "F", CEE 7/4, Schuko, 3 pins

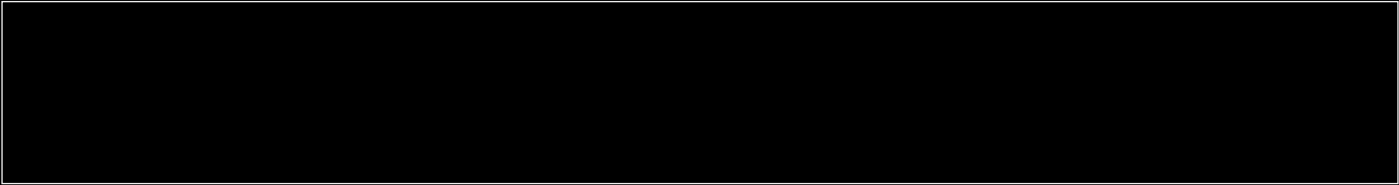
Value	Description
DOMESTIC_G	Standard/Domestic household, type "G", BS 1363, Commonwealth, 3 pins
DOMESTIC_H	Standard/Domestic household, type "H", SI-32, 3 pins
DOMESTIC_I	Standard/Domestic household, type "I", AS 3112, 3 pins

	EnergySource		
	EnvironmentallImpact		
	string		
	string		

8.4.6.1. Examples

Simple:

```
"energy_mix": {  
  "is_green_energy": true  
}
```



	EnergySourceCategory		
	number		

GENERAL_GREEN	All kinds of regenerative power sources.
SOLAR	Regenerative power from PV.
WIND	Regenerative power from wind turbines.
WATER	Regenerative power from water turbines.

8.4.9. EnvironmentalImpact class

Amount of waste produced/emitted per kWh.

Property	Type	Card.	Description
	EnvironmentalImpactCategory		
	number		

	DateTime		
	DateTime		

METRO_STATION	A metro station.
TRAIN_STATION	A train station.
AIRPORT	An airport.

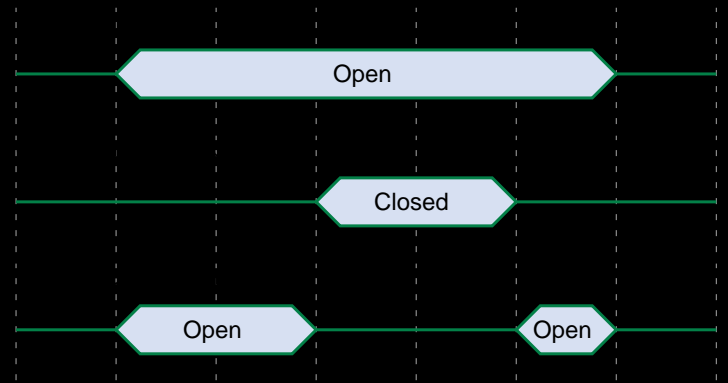
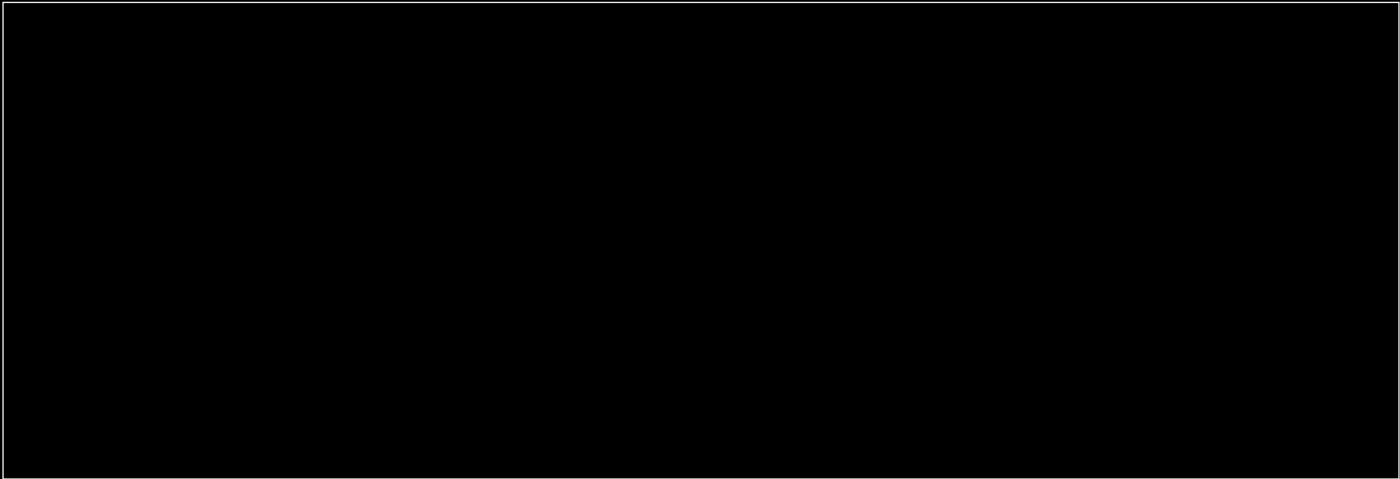
Value	Description
PARKING_LOT	A parking lot.
CARPOOL_PARKING	A carpool parking.
FUEL_STATION	A Fuel station.

	string		
	string		

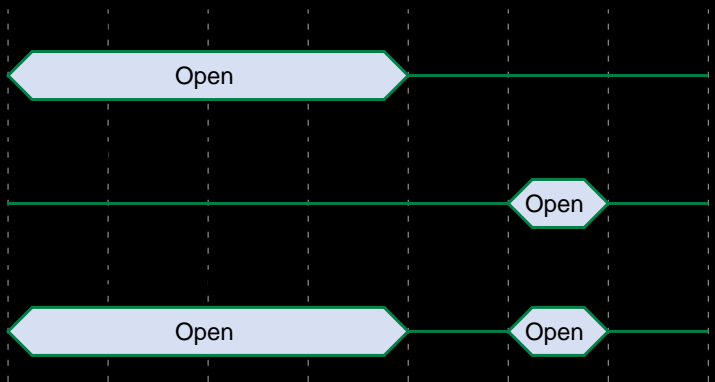
|

	RegularHours		RegularHours
	ExceptionalPeriod		
	ExceptionalPeriod		

}



```
{
  "twentyfourseven": false,
  "regular_hours": [{
    "weekday": 1,
    "period_begin": "00:00",
    "period_end": "04:00"
  }]
```



	URL		
	URL		
	ImageCategory		
	CiString		
width	int(5)	?	Width of the full scale image
height	int(5)	?	Height of the full scale image

8.4.16. ImageCategory enum

The category of an image to obtain the correct usage in a user presentation. The category has to be set accordingly to the image content in order to guarantee the right usage.

AC_1_PHASE	AC single phase.
AC_3_PHASE	AC three phase.
DC	Direct Current.

8.4.20. PublishTokenType *class*

Defines set of values that identify a token to which a Location might be published

At least on of the following fields SHALL be set: `uid`, `visual_number` or `group_id`

	CiString		
	TokenType		
	string		
	string		
	CiString		

	string		
	string		

```
"opening_times": {
  "regular_hours": [
    {
      "weekday": 1,
      "period_begin": "08:00",
      "period_end": "20:00"
```


BLOCKED	The EVSE/Connector is not accessible because of a physical barrier, i.e. a car.
CHARGING	The EVSE/Connector is in use.
INOPERATIVE	The EVSE/Connector is not yet active or it is no longer available (deleted).

Value	Description
OUTOFORDER	The EVSE/Connector is currently out of order.
PLANNED	The EVSE/Connector is planned, will be operating soon.
REMOVED	The EVSE/Connector was discontinued/removed.

	DateTime		
	DateTime		
	Status		

|

9. Sessions module

Module Identifier: sessions

Data owner: CDR



When a reservation results in a charging session for the same token, the Session object status is: ACTIVE

When a reservation does not result in a charging session, the Session object status SHALL be set to: COMPLETED.

A CDR might be created even if no energy was transferred to the EV, just for the costs of the reservation.

9.2. Interfaces and Endpoints

9.2.1. Sender Interface

GET	paginated
PUT	

paginated

pagination

	DateTime		
	DateTime		

Response Data

The response contains a list of Session objects that match the given parameters in the request, the header will contain the [pagination](#) related headers.

Any older information that is not specified in the response is considered no longer valid. Each object must contain all required fields. Fields that are not specified may be considered as null values.

Datatype	Card.	Description
Session	*	List of Session objects that match the

|

	CiString		

ChargingPreferences

ChargingPreferences		

ChargingPreferencesResponse

ChargingPreferencesResponse		

Client Owned Objectsparty_idcountry_code

Endpoint structure definition:

{sessions_endpoint_url}/{country_code}/{party_id}/{session_id}

Example:

Method	Description
GET	Retrieve a Session object from the eMSP's system with Session.id equal to {session_id}.
POST	
PUT	
PATCH	

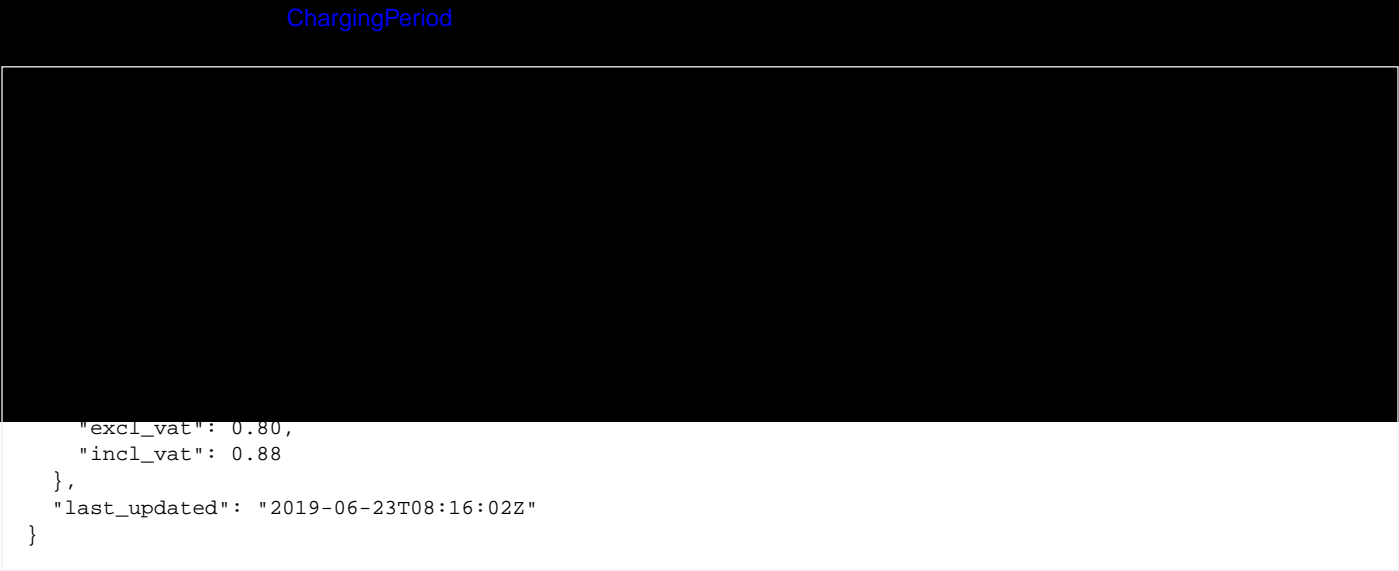
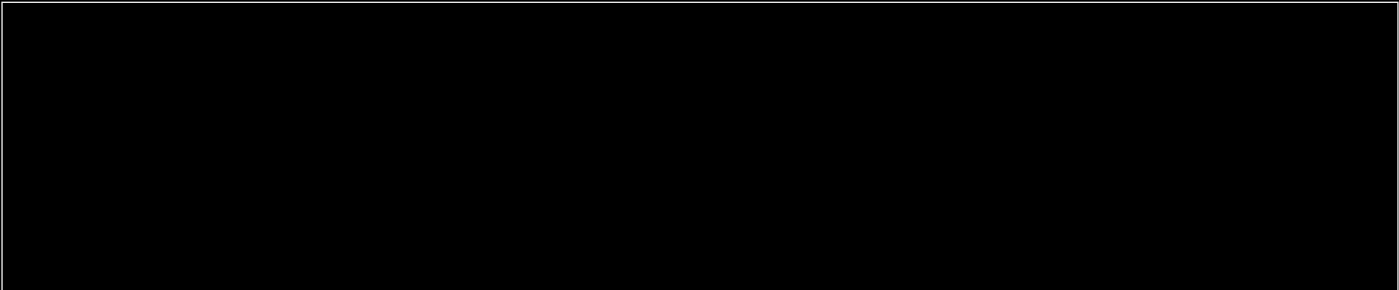
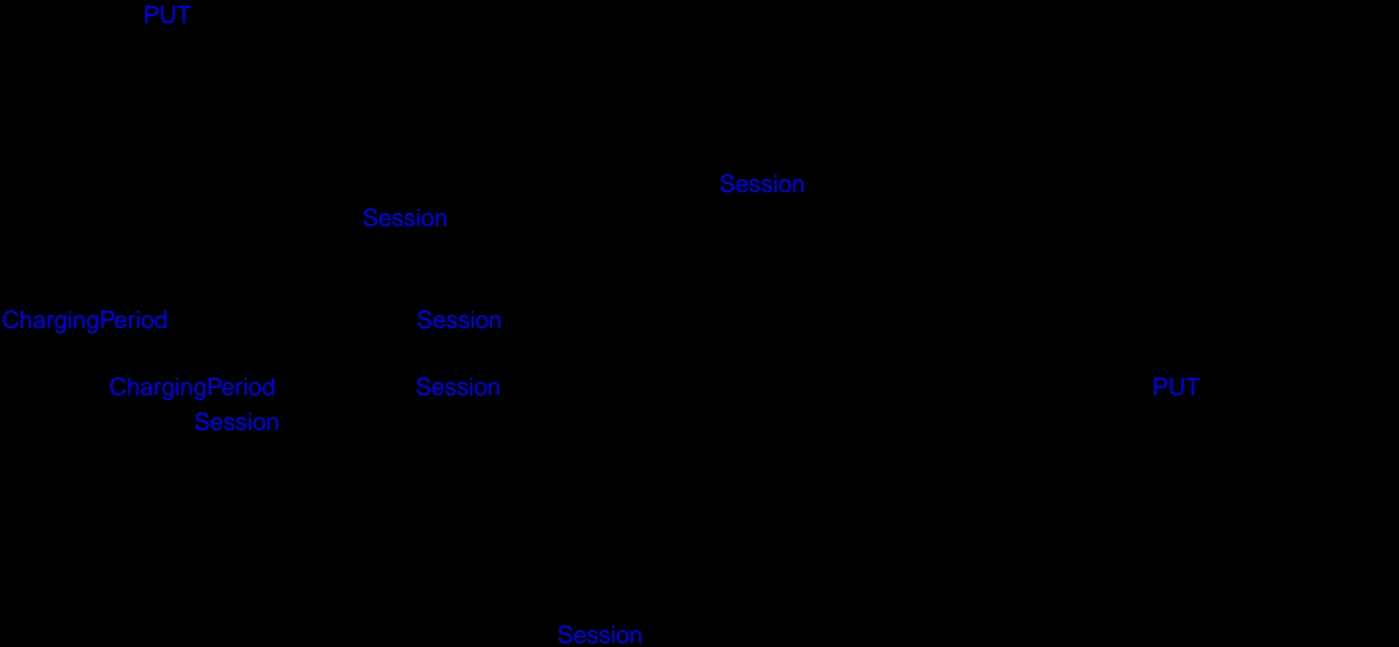
	CiString		
	CiString		
	CiString		

Session		

Session		

		red	
country_code	CiString(2)	yes	Country code of the CPO performing this PUT on the eMSP's system. This SHALL be the same value as the country_code in the Session object being pushed.

Parameter	Datatype	Required	Description
party_id	CiString(3)	yes	Party ID (Provider ID) of the CPO performing this PUT on the eMSP's system. This SHALL be the same value as the party_id in the Session object being pushed.
	CiString		



9.3. Object description

9.3.1. Session Object

CDR description			
	CiString		
	CiString		
	CiString		
	DateTime		ACTIVE PENDING PENDING ACTIVE ACTIVE
	DateTime		
	number		
	CdrToken		
	AuthMethod		
	CiString		authorization StartSession real-time
			been given by the eMSP that are relevant to this Session, the last given value SHALL be used here.
location_id	CiString(36)	1	Location.id of the Location object of this CPO, on which the charging session is/was happening.

Property	Type	Card.	Description
evse_uid	CiString(36)	1	EVSE.uid of the EVSE of this Location on which the charging session is/was happening.
connector_id	CiString(36)	1	Connector.id of the Connector of this Location the charging session is/was happening.
	string		
	string		
	ChargingPeriod		
	Price		
	SessionStatus		
	DateTime		

ReserveNow
real-time authorization



```
{
  "country_code": "BE",
  "party_id": "BEC",
  "id": "101",
  "start_date_time": "2015-06-29T22:39:09Z",
  "end date time": "2015-06-29T23:50:16Z",
```

	ProfileType		ConnectorProfileType ProfileType Tariff
	DateTime		
	number		
			or energy charged.

Property	Type	Card.	Description
discharge_allowed	boolean	?	The driver allows their EV to be discharged when needed, as long as the other preferences are met: EV is charged with the preferred energy (<i>energy_need</i>) until the preferred departure moment (<i>departure_time</i>). Default if omitted: false

	object using this state.
INVALID	The Session object using this state is declared invalid and will not be billed.
PENDING	The session is pending, it has not yet started. Not all pre-conditions are met. This is the initial state. The session might never become an <i>active</i> session.

Value	Description
RESERVATION	The session is started due to a reservation, charging has not yet started. The session might never become an <i>active</i> session.

10. CDRs module

Module Identifier: cdrs

Data owner: CPO



10.1.3. Pull model

eMSPs who do not support the Push model need to call [GET](#) on the CPO's CDRs endpoint to receive a list of CDRs.

This [GET](#) can also be used in combination with the Push model to retrieve CDRs after the system (re-)connects to a CPO, to get a list of CDRs *missed* during a downtime of the eMSP's system.

A CPO is not required to return all known CDRs, the CPO is allowed to return only the CDRs that are relevant for the requesting eMSP.

GET	paginated

[paginated](#), [pagination](#)

Parameter	Datatype	Required	Description
date_from	DateTime	no	Only return CDRs that have last_updated after or equal to this Date/Time (inclusive).
date_to	DateTime	no	Only return CDRs that have last_updated up to this Date/Time, but not

[pagination](#)

CDR		

GET	
POST	

To retrieve an existing URL from the eMSP's system, the URL, returned in the response to a POST of a new CDR, has to be used.

Response Data

The endpoint returns the requested CDR, if it exists.

Datatype	Card.	Description
CDR		

CDR		

	URL		

[Session](#) [Session](#)
session. The information is meant to be viewed by the driver while the charging session is ongoing.

The CDR on the other hand can be thought of as *sealed*, preserving the information valid at the moment in time the underlying session was started. This is a requirement of the main use case for CDRs, namely invoicing. If e.g. a street is renamed the day after a session took place, the driver should be presented with the name valid at the time the session was started. This guarantees that

the CDR will be recognized as correct by the driver and is not going to be contested.

The *CDR* object shall always contain information like Location, EVSE, Tariffs and Token as they were **at the start** of the charging session.

A CPO SHALL at least start (and add) a ChargingPeriod every moment/event that has relevance for the total costs of a CDR.

	CiString		
	CiString		
	CiString		something appended
	DateTime		
	DateTime		
	CiString		
	CdrToken		
	AuthMethod		
	CiString		authorization StartSession real-time
	CdrLocation		EVSE Connector
	string		
	string		
tariffs	Tariff	*	List of relevant Tariff Elements, see: Tariff . When relevant, a <i>Free of Charge</i> tariff should also be in this list, and point to a defined <i>Free of Charge</i> Tariff.

Property	Type	Card.	Description
charging_periods	ChargingPeriod	+	List of Charging Periods that make up this charging session. A session consists of 1 or more periods, where each period has a different relevant Tariff.
	SignedData		
	Price		
	Price		
	number		
	Price		
	number		
	Price		
	number		
	Price		
	Price		
	string		
	CiString		
	CiString		
	DateTime		

ReserveNow
real-time authorization

10.3.1.1. Example of a CDR


```
{  
  "country_code": "BE",  
  "party_id": "BEC",  
  "id": "12345",  
  "start_date_time": "2015-06-29T21:39:09Z",  
  "end date time": "2015-06-29T23:37:32Z",  
}
```

10.4.1. AuthMethod *enum*

Value	Description
AUTH_REQUEST	Authentication request has been sent to the eMSP.
COMMAND	Command like StartSession or ReserveNow used to start the Session, the Token provided in the Command was used as authorization.

	CdrDimension Type		
	number		

		ChargingPeriod
		ChargingPeriod
		ChargingPeriod
		ChargingPeriod
		ChargingPeriod
		ChargingPeriod
		ChargingPeriod
		ChargingPeriod
		ChargingPeriod

NOTE

OCPI makes it possible to provide SoC in the Session object. This information can be useful to show the current State of Charge to an EV driver during charging. Implementers should be aware that SoC is only available at some DC Chargers. Which is currently a small amount of the total amount of Charge Points. Of these DC Chargers, only a small percentage currently provides SoC via OCPP to the CPO. Then there is also the question if SoC is allowed to be provided to third-parties as it can be seen as privacy-sensitive information. So if an

Location

	CiString		
	string		
	string		
	string		
	string		
	string		
	GeoLocation		
	CiString		
	CiString		http://emi3group.com/documents-links/
	CiString		
	ConnectorType		
	ConnectorFormat		
	PowerType		

	CiString		
type	TokenType	1	Type of the token

Property	Type	Card.	Description
contract_id	CiString(36)	1	Uniquely identifies the EV driver contract token within the eMSP's platform (and suboperator platforms). Recommended to follow the specification for eMA ID from "eMI3 standard version V1.0" (http://emi3group.com/documents-links/) "Part 2: business objects."

	DateTime		
	CdrDimension		
	CiString		

	CiString		
	CiString		
	SignedValue		
	CiString		

|

		https://has-to-be.com
		https://alfen.com/de/downloads
		https://www.ebee.berlin

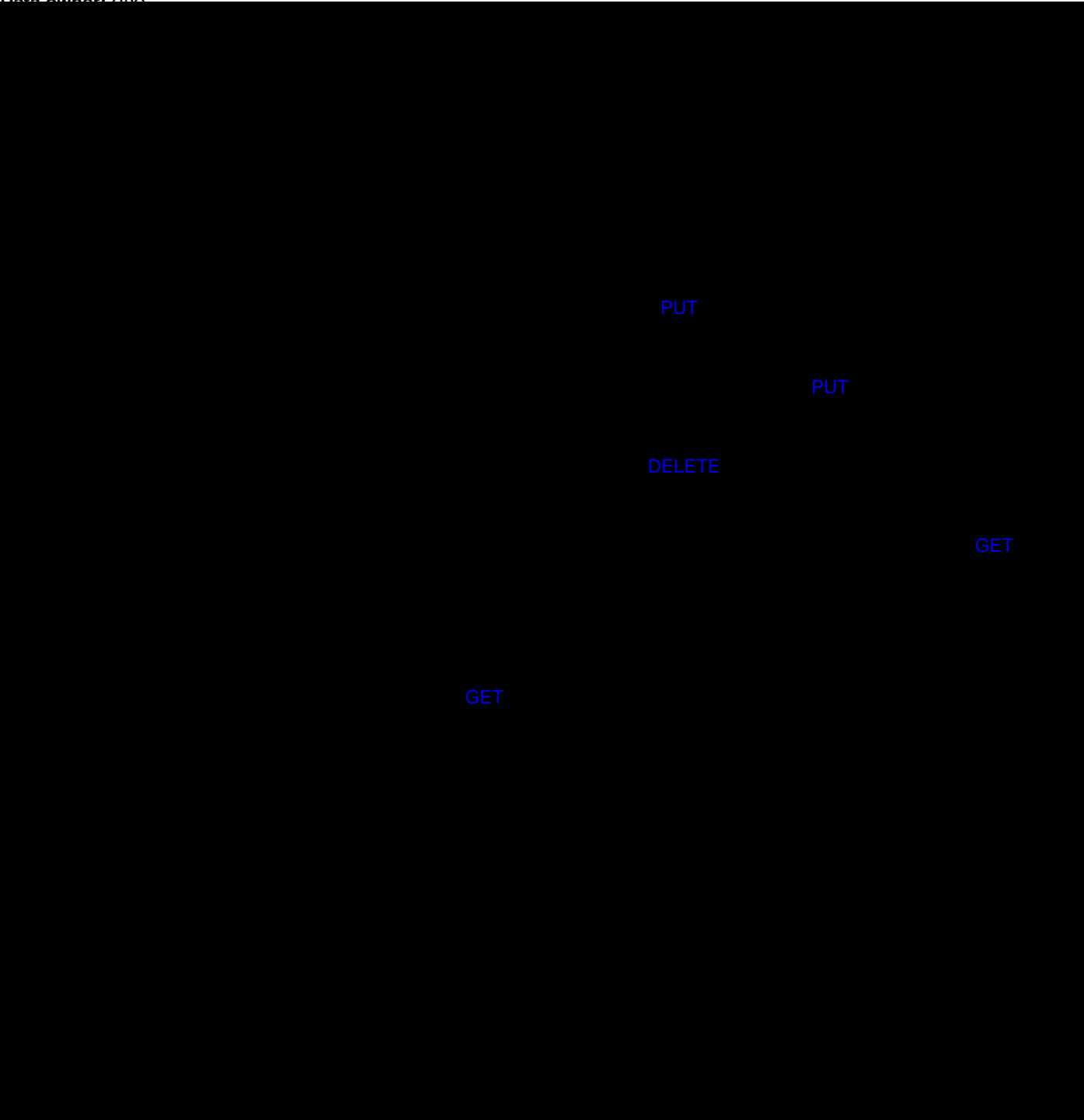
This class contains the signed and the plain/unsigned data. By decoding the data, the receiver can check if the content has not been altered.

Property	Type	Card.	Description
nature	CiString(32)	1	Nature of the value, in other words, the event this value belongs to. Possible values at moment of writing: - Start (value at the start of the Session) - End (signed value at the end of the Session)
	CiString		
	CiString		

11. Tariffs module

Module Identifier: tariffs

Data owner: GPO



GET	paginated
DELETE	n/a

11.2.1.1. GET Method

Fetch information about all Tariffs.

Endpoint structure definition:

paginated

pagination

	DateTime		
	DateTime		

pagination

Tariff		

Client Owned Objects

party_idcountry_code

Endpoint structure definition:

{tariffs_endpoint_url}/{country_code}/{party_id}/{tariff_id}

Example:

Method	Description
GET	Retrieve a Tariff as it is stored in the eMSP's system.
POST	
PUT	
DELETE	

	CiString		
	CiString		
	CiString		

Tariff		

Tariff		

Parameter	Datatype	Required	Description
country_code	CiString(2)	yes	Country code of the CPO performing the PUT request on the eMSP's system. This SHALL be the same value as the country_code in the Tariff object being pushed.
	CiString		
	CiString		

--	--	--	--

Connector object

	CiString		
	CiString		
	CiString		

A Tariff object consists of a list of one or more Tariff Elements, which can be used to create complex Tariff structures.

When the list of Tariff Elements contains more than one Element with the same Tariff Dimension (ENERGY/FLAT/TIME etc.), than the first Tariff Element with that Dimension in the list with matching Tariff Restrictions will be used. Only one Tariff per Element type

can be active at any point in time, but multiple Tariff Types can be active at once. IE you can have an ENERGY element and TIME element active at the same time, but only the first valid element of each.

When no Tariff Element with a specific Dimension is found for which the Restrictions match, and there is no Tariff Element in the list with the given Dimension without Restrictions, there will be no costs for that Tariff Dimension.

Tariff example

Free of Charge

	CiString		
	CiString		
	CiString		
	string		
	TariffType		Charging
		Preferences	
	DisplayText		
	URL		
	Price		
	Price		
	TariffElement		
	DateTime		Location
	DateTime		Location
	EnergyMix		
	DateTime		

NOTE

max_price: As the VAT might be built up of different parts, there might be situations where maximum cost including VAT is reached earlier or later than the maximum cost excluding VAT. So as a rule, they both apply: - The total cost of a Charging Session excluding VAT can never be higher than the max_price excluding VAT. - The total cost of a Charging Session including VAT can never be higher than the max_price including VAT.

NOTE

`start_date_time` and `end_date_time`: When the Tariff of a Charge Point (Location) is changed during an ongoing charging session, it is common to not switch the Tariff until the ongoing session is finished. But this is NOT a rule of OCPI. When charging at a Charge Point, a driver accepts the tariff which is valid when they start their charging session. If the Tariff of the Charge Point would change during the charging session, the driver might get billed something they didn't agree to when starting the session.

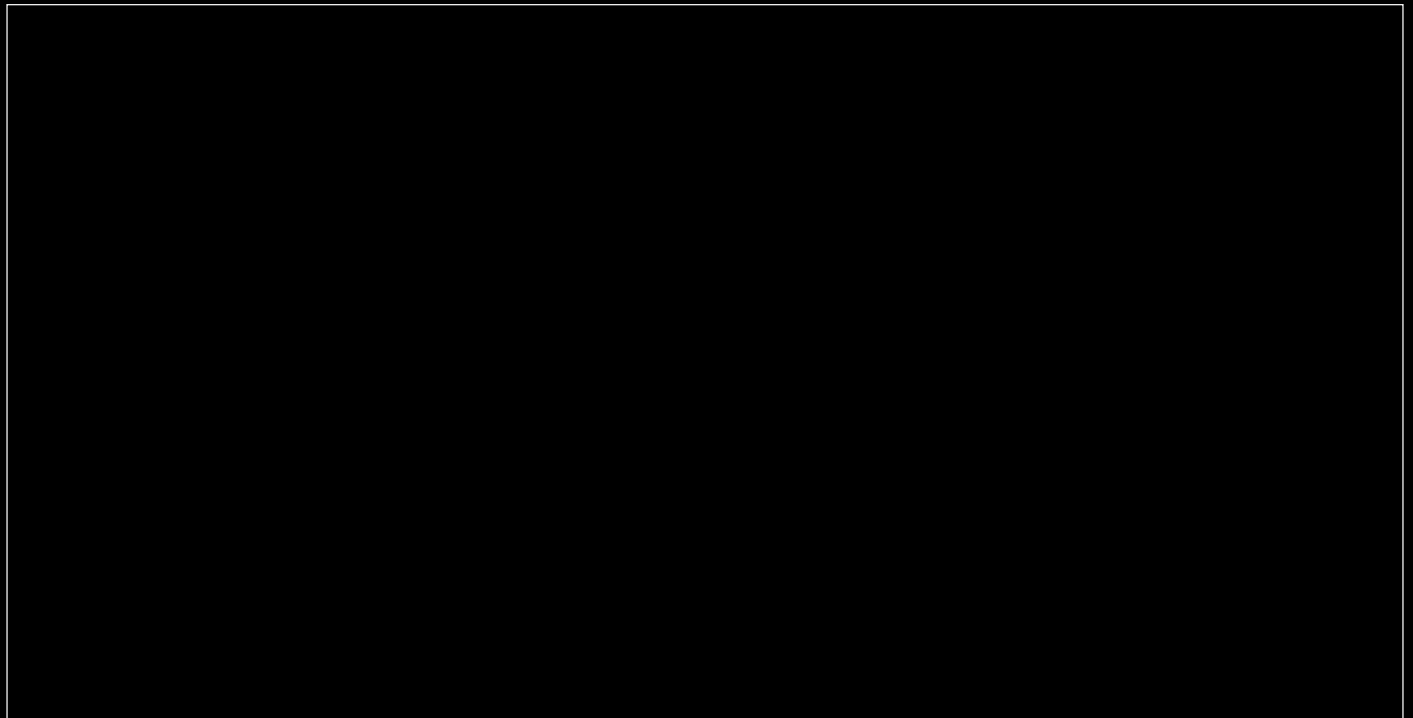
- Billed per 1 Wh

This tariff will result in total cost of 5.50 euro (excl. VAT) or 6.10 euro (incl. VAT) when 20 kWh are charged.

```
{
  "country_code": "DE",
  "party_id": "ALL",
  "id": "17",
  "currency": "EUR",
  "elements": [{
```

- Start or transaction fee
 - 0.50 euro (excl. VAT)
 - 20% VAT

- Energy
 - 0.25 euro per kWh (excl. VAT)
 - 10% VAT
 - Billed per 1 Wh



This tariff has an end date: 30 June 2019, which is typically used when a tariff is going to be replaced by a new tariff. A [Connector](#) of a [Location](#) can have multiple Tariffs (IDs) assigned. By assigning both, the old and the new tariff ID, they will automatically be replaced. It is not required to update all Locations at the same time, the old tariff can also be removed later.

For a charging session where 50 kWh are charged, this tariff will result in costs of 10.00 euro (excl. VAT) or 11.00 euro (incl. VAT) due to the price limit. If only 30 kWh were charged, the costs would be 8.00 euro (excl. VAT) and 8.85 euro (incl. VAT), as the start fee combined with the energy costs would be lower than the defined max price.

{

Simple Tariff example 3 euro per hour, 5 euro per hour parking

Example of a tariff where the driver pays for the time of using the Charge Point, but pays more when the car is no longer charging, to discourage the EV driver of leaving his EV connected when it is already full.

- Charging Time
 - 3.00 euro per hour (excl. VAT)
 - 10% VAT
 - Billed per 1 min (60 seconds)



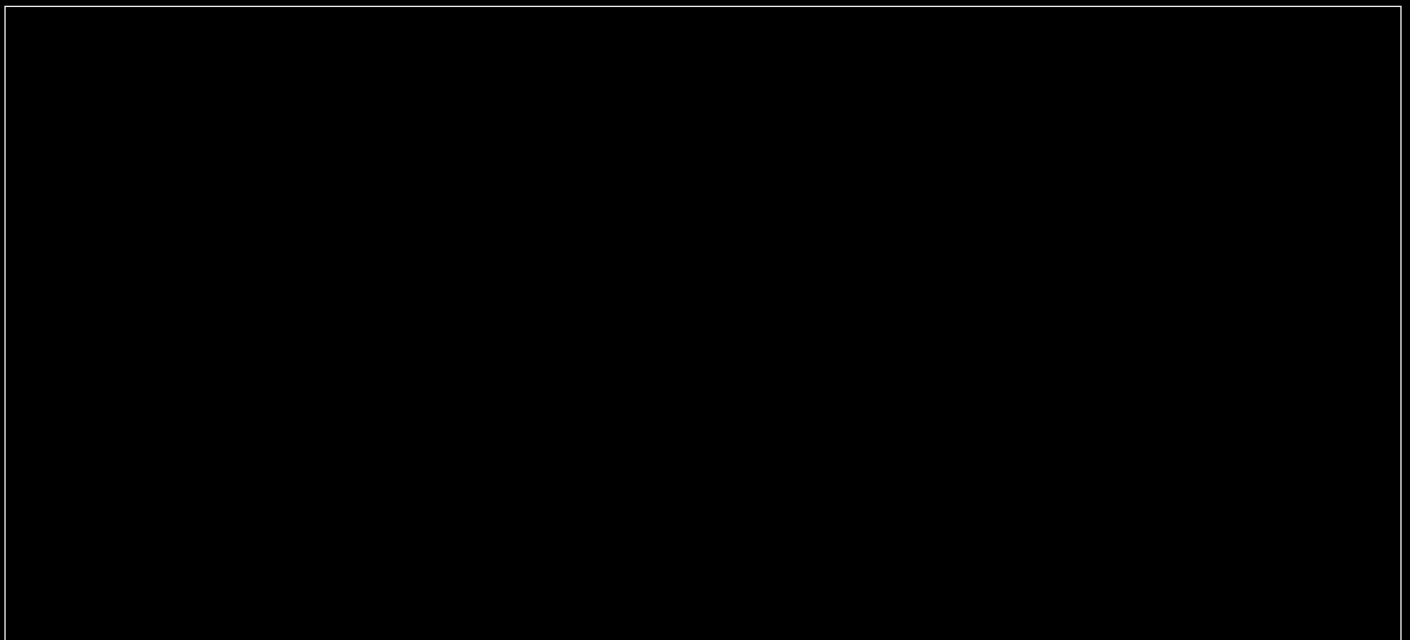
```
{
  "country_code": "DE",
  "party_id": "ALL",
  "id": "12",
  "currency": "EUR",
  "type": "AD HOC PAYMENT",
```

This examples shows the use of `tariff_alt_url`.

This examples shows a `PROFILE_CHEAP` tariff, which is a smart charging tariff. Drivers are able to select this tariff by setting the `profile_type` in their [Charging Preferences](#) to `CHEAP`. In such case, the price might not be fixed, but depend on the real-time

energy prices. To explain this to the driver, a short text inside `tariff_alt_text` might not be the best solution. Showing a graph could be better. Therefore it is also possible to provide an URL in `tariff_alt_url` to a site that explains the tariff better and in more detail.

- Energy



- 2.00 euro per hour (excl. VAT)
- 20% VAT
- Billed per 10 min (600 seconds)

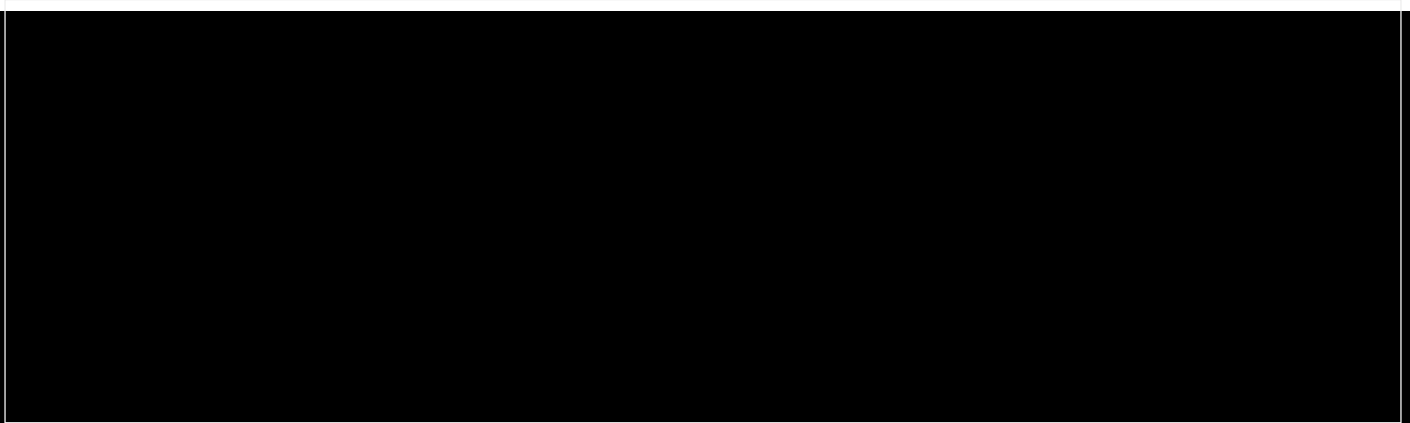
- When charging with more than 32A on weekends
 - 1.25 euro per hour (excl. VAT)
 - 20% VAT
 - Billed per 10 min (600 seconds)

```
{  
  "country_code": "DE",  
  "party_id": "ALL",  
  "id": "14",  
  "currency": "EUR",  
  "type": "REGULAR",
```

```
}  
},  
"last_updated": "2015-06-29T20:39:09Z"  
}
```

Free of Charge Tariff example

In this example no VAT is given because it is not necessary (as the `price` is 0.00). This might not always be the case though and it is of course permitted to add a VAT, even if the `price` is set to zero.



```
{
  "country_code": "DE",
  "party_id": "ALL",
  "id" : "52",
  "currency" : "EUR",
  "elements" : [ {
```

- 0.50 euro (excl. VAT)
- 20% VAT
- Energy

- 0.25 euro per kWh (excl. VAT)
- 10% VAT
- Billed per 1 Wh



For a charging session that was started 13 minutes after the reservation time, where the driver charges 20 kWh, this tariff will result in costs of 8.75 euro (excl. VAT) or 10.00 euro (incl. VAT). Because the reservation fee is billed per 5 minutes, the driver has to pay for 15 minutes of reservation even though they started the charging session 13 minutes after the reservation time.

```
{
  "country_code": "DE",
  "party_id": "ALL",
  "id": "20",
  "currency": "EUR",
  "elements": [{
```

Tariff example with reservation price

For a charging session that was started 22 minutes after the reservation time, where the driver charges 20 kWh, this tariff will result in costs of 6.50 euro (excl. VAT) or 7.30 euro (incl. VAT). Because the reservation fee is billed per 10 minutes, the driver has to pay for 30 minutes of reservation even though they started the charging session 22 minutes after the reservation time.

If the driver did not start a charging session and the reservation expired after the reserved time of 1 hour, the tariff would have

resulted in costs of 6.00 euro (excl. VAT) or 7.20 euro (incl. VAT). In case a reservation is not used, the driver has to pay the full amount of reserved time as well as an additional expiration fee as compensation for not charging at all.

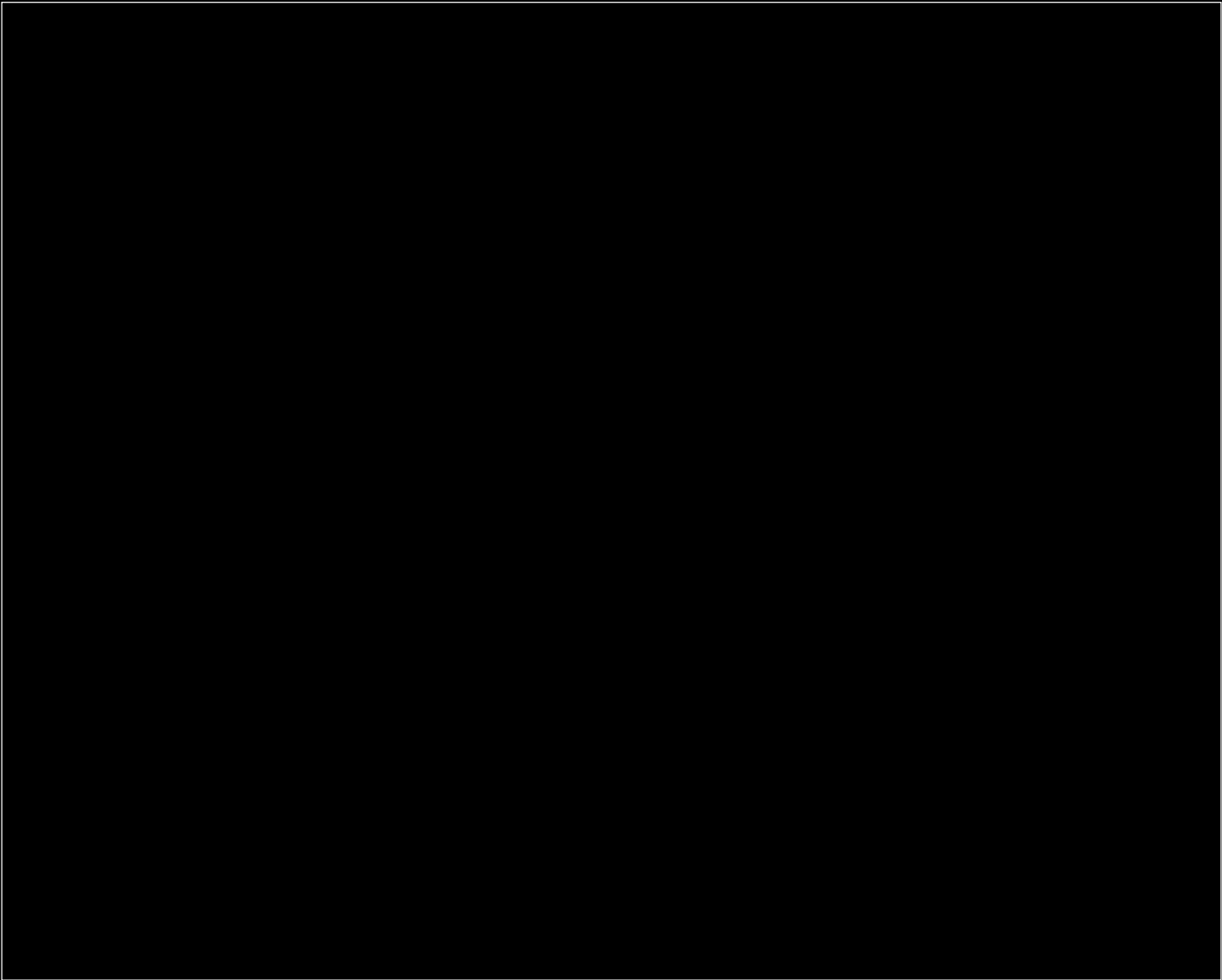
```
{  
  "country_code": "DE",  
  "reservation_price": 6.00  
}
```

- Billed per 1 Wh

This example is very similar to [Tariff example with reservation price](#) with the difference that expired reservations cost something and that reservation time is billed per 10 minutes. Also, the price for reservation is different.

For a charging session that was started 22 minutes after the reservation time, where the driver charges 20 kWh, this tariff will result in costs of 7.00 euro (excl. VAT) or 7.90 euro (incl. VAT). Because the reservation fee is billed per 10 minutes, the driver has to pay for 30 minutes of reservation even though they started the charging session 22 minutes after the reservation time.

If the driver did not start a charging session and the reservation expired after the reserved time of 1.5 hours, the tariff would have resulted in costs of 9.00 euro (excl. VAT) or 10.80 euro (incl. VAT). In case a reservation is not used, the driver has to pay the



FRIDAY	Friday
SATURDAY	Saturday
SUNDAY	Sunday

11.4.2. PriceComponent *class*

Property	Type	Card	Description
type	TariffDimensionType	1	Type of tariff dimension
	number		
	number		

```
{
  "country_code": "DE",
  "party_id": "ALL",
  "id": "22",
  "currency": "EUR",
  "elements": [
```

- 10 billable minutes of charging time after 17:00, generating costs of 0.40 euro. As the Price Component of the last Tariff Element being used has a `step_size` of 15 minutes, we bill a total charging duration of 15 minutes. When considering the already billed 5 minutes of charging time before 17:00, we are left with 10 minutes to bill after 17:00.
- Parking time will no be billed, the 2 minutes parking time is within the extra 5 minutes of charging billed because of the `step_size`.

Value	Description
FLAT	Flat fee without unit for <code>step_size</code>
PARKING_TIME	Time not charging: defined in hours, <code>step_size</code> multiplier: 1 second
TIME	Time charging: defined in hours, <code>step_size</code> multiplier: 1 second
	RESERVATION

	string		Location
	string		Location
	string		Location
	string		Location
	number		
	number		
	number		
	number		

Property	Type	Card.	Description
min_power	number	?	Minimum power in kW, for example 5. When the EV is charging with more than the defined amount of power, this TariffElement is/becomes active. If the charging power is or becomes lower, this TariffElement is not or no longer valid and becomes inactive. This describes NOT the minimum power over the entire
	number		
	DayOfWeek		
	ReservationRestrictionType		FLAT TIME

```
{  
  "country_code": "DE",  
  "party_id": "ALL",  
  "id": "1",  
  "currency": "EUR",  
  "type": "REGULAR",  
}
```

```
{  
  "country_code": "DE",  
  "party_id": "ALL",  
  "id": "2",  
  "currency": "EUR",  
  "type": "REGULAR",  
}
```

	Charging Preference CHEAP
	Charging Preference FAST
	Charging Preference GREEN
	Charging Preference REGULAR

12. Tokens module

Module Identifier: tokens

Data owner: xcp



NOTE

Real-time authorization might be asked for a charging location that is not published via the [Location](#) module, typically a private charger. In most cases this is expected to result in: `ALLOWED`.

NOTE

If real-time authorization is asked for a location, the eMSP SHALL NOT validate that charging is possible based on information like opening hours or EVSE status etc, this information might not be update to date.

12.2. Interfaces and endpoints

party_id country_code Client Owned Object

GET	
PUT	
PATCH	
	PUT

	CiString		
	CiString		
	CiString		
	TokenType		RFID

Response Data

The response contains the requested object.

Type	Card.	Description
Token	1	The requested Token object.

Example: invalidate a Token

```
PATCH To URL: https://www.server.com/ocpi/cpo/2.2/tokens/NL/TNM/012345678

{
  "status": "invalid"
}
```

GET	paginated
POST	

paginated

pagination

paginated

pagination

Parameter	Datatype	Required	Description
date_from	DateTime	no	Only return Tokens that have last_updated after or equal to this Date/Time (inclusive).

Parameter	Datatype	Required	Description
date_to	DateTime	no	Only return Tokens that have last_updated up to this Date/Time, but not including (exclusive).
effect	int	no	The effect of the first object returned. Default is 0.

pagination

Token		

2002

	CiString		
	TokenType		RFID

In the body an optional [LocationReferences](#) object can be given. The eMSP SHALL then validate if the Token is allowed to be used at this Location, and if applicable: which of the Locations EVSEs. The object with valid Location and EVSEs will be returned in the response.

Type	Card.	Description
LocationReferences	?	Location and EVSEs for which the Token is requested to be authorized.

Response Data

AuthorizationInfo

AuthorizationInfo		

	AllowedType		
	Token		
	LocationReferences		
	CiString		Session CDR
	DisplayText		

	CiString		
	CiString		
	CiString		
	TokenType		
	CiString		http://emi3group.com/documents-links/
			"Part 2: business objects."
visual_number	string (64)	?	Visual readable number/identification as printed on the Token (RFID card), might be equal to the contract_id.

Property	Type	Card.	Description
issuer	string(64)	1	Issuing company, most of the times the name of the company printed on the token (RFID card), not necessarily the eMSP.
group_id	CiString(36)	?	This ID groups a couple of tokens. This can be used to make two or more
	WhitelistType		
	string		
	ProfileType		Charging Preference ProfileType Set Charging Preferences ProfileType
	EnergyContract		
	DateTime		

|



```
{
  "country_code": "DE",
  "party_id": "TNM",
  "uid": "12345678905880",
  "type": "RFID",
  "contract_id": "DE8ACC12E46L89",

```


	string		
	string		

	CiString		
	CiString		

12.4.4. TokenType *enum*

Value	Description
AD_HOC_USER	One time use Token ID generated by a server (or App.) The eMSP uses this to bind a Session to a customer, probably an app user.
APP_USER	Token ID generated by a server (or App.) to identify a user of an App. The same user uses the

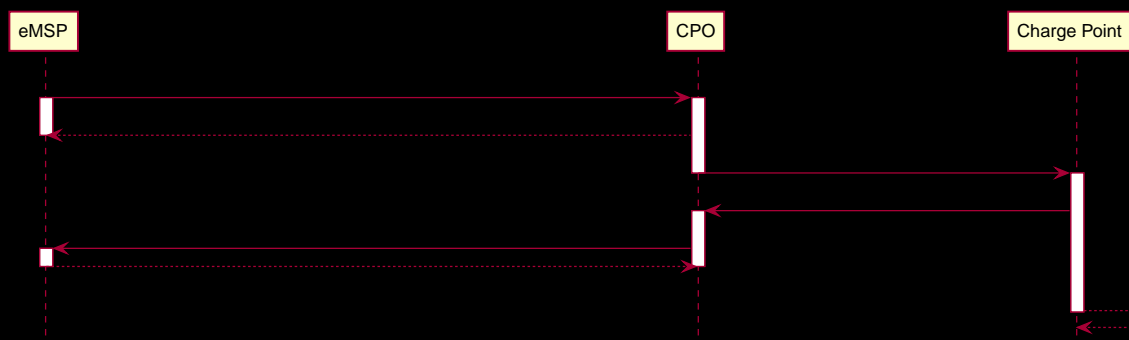
real-time authorization

	realtime authorization
	realtime authorization
	realtime authorization
	realtime authorizationrealtime
	authorization

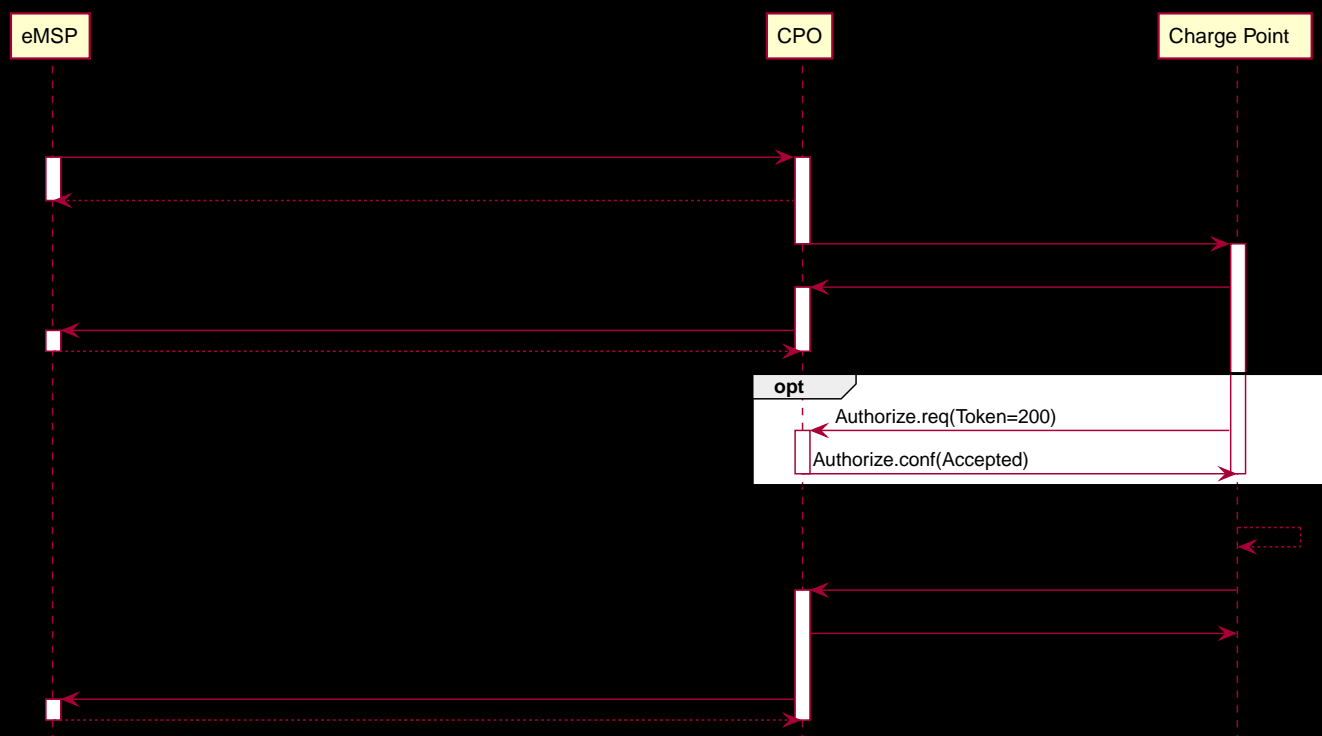
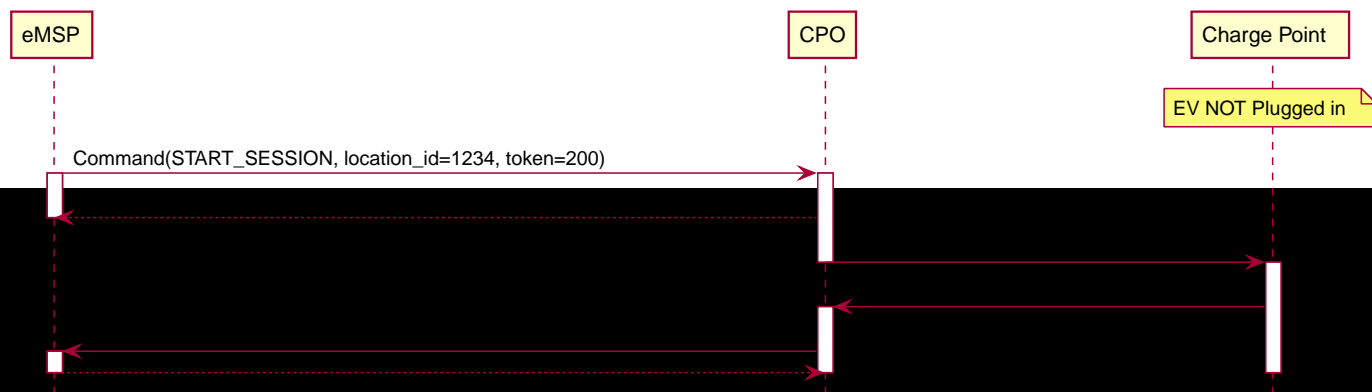
13. *Commands* module

Module Identifier: commands

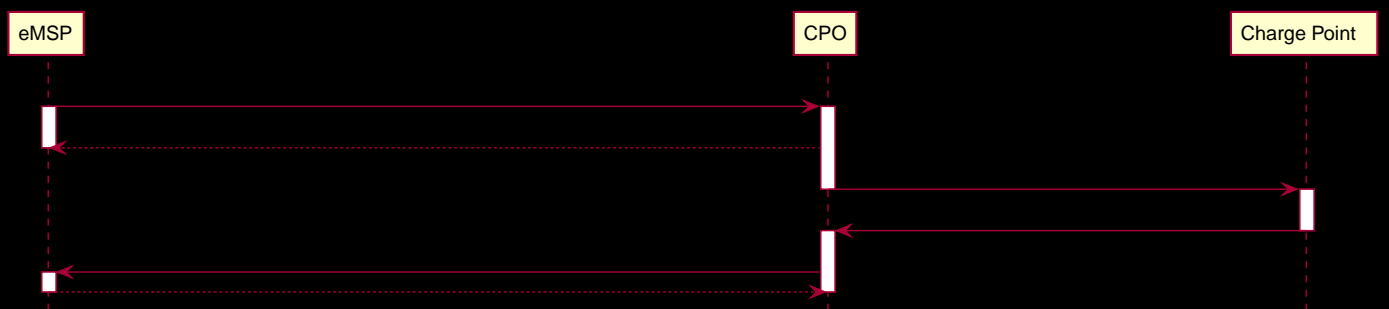
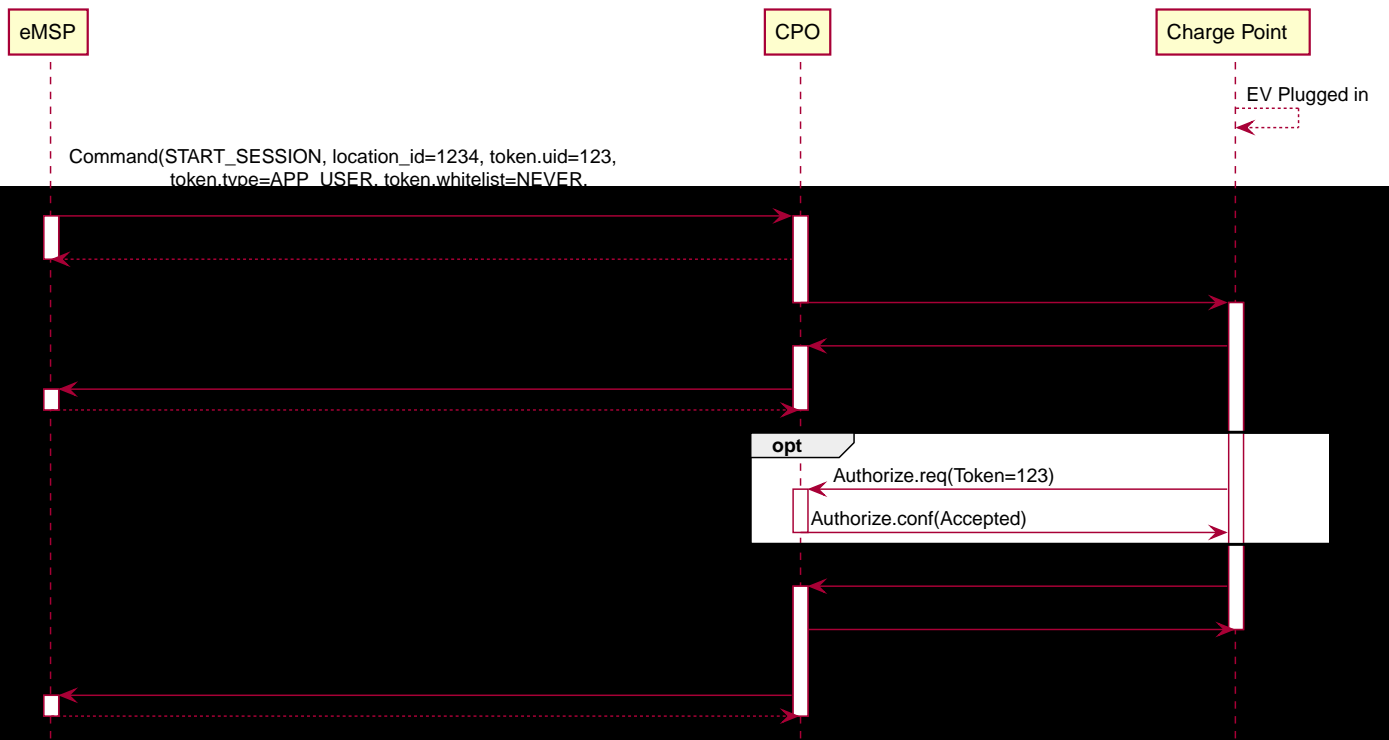
Type: Functional Module

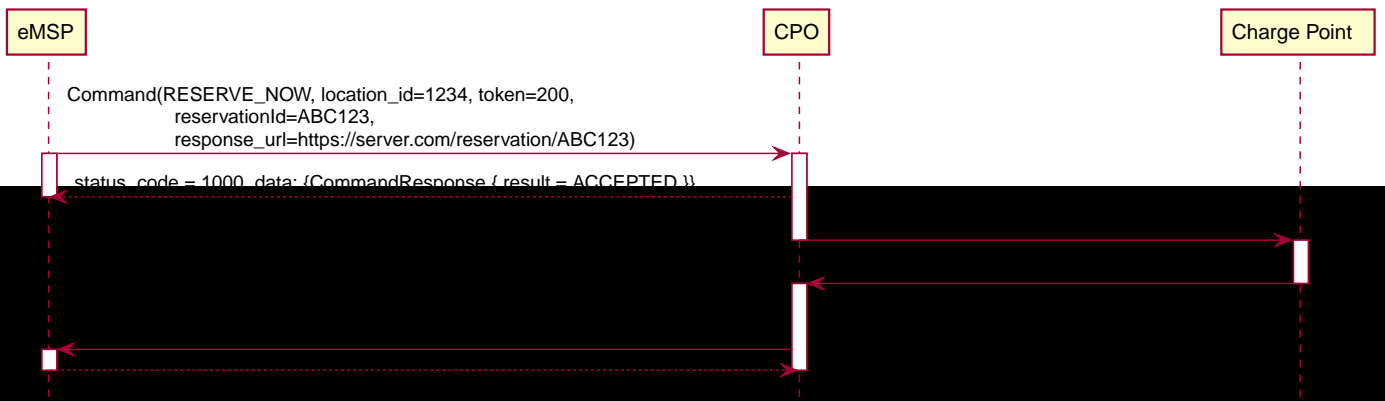


Example of a `START_SESSION` that is accepted, but no new Session is started because the EV is not plugged in, and this Charge Point does not allow a remote start without a cable already being plugged in.



realtime authorization





Endpoint structure definition:

```
{commands_endpoint_url}{command}
```

Examples:

https://www.server.com/ocpi/cpo/2.2/commands/START_SESSION

https://ocpi.server.com/commands/STOP_SESSION

https://server.com/ocpi/cpo/2.2/commands/RESERVE_NOW

POST	

	CommandType		

CancelReservation		
ReserveNow		
StartSession		
StopSession		
UnlockConnector		

Datatype	Card.	Description
CommandResponse	1	Result of the command request, by the CPO (not the Charge Point). So this indicates if the CPO understood the command request and was able to send it to the Charge Point. This is not the response by the Charge Point

POST	

CommandResult		

13.3. Object description

13.3.1. CancelReservation Object

With CancelReservation the Sender can request the Cancel of an existing Reservation. The CancelReservation needs to contain the `reservation_id` that was given by the Sender to the `ReserveNow`.

	URL		
	CiString		

	CommandResponseType		
	DisplayText		

	CommandResultType		
	DisplayText		

The eMSP provides a Token that has to be used by the Charge Point. The Token provided by the eMSP for the `ReserveNow` SHALL be authorized by the eMSP before sending it to the CPO. Therefor the CPO SHALL NOT check the validity of the Token provided before sending the request to the Charge Point.

If this is an OCPP Charge Point, the Charge Point decides if it needs to validate the given Token, in such case:

- If this Token is of type: AD_HOC_USER or APP_USER the CPO SHALL NOT do a [realtime authorization](#) at the eMSP for this .
- If this Token is of type: RFID, the CPO SHALL NOT do a [realtime authorization](#) at the eMSP for this Token at the given EVSE/Charge Point within 15 minutes after having received this ReserveNow.

The eMSP MAY use Tokens that have not been pushed via the [Token](#) module, especially AD_HOC_USER or APP_USER Tokens are

[Token](#)

	URL		
	Token		
	DateTime		
	CiString		
	CiString		
	CiString		
	CiString		Session CDR

[realtime authorization](#)

[realtime authorization](#)

[authorized](#)

The eMSP MAY use Tokens that have not been pushed via the [Token](#) module, especially AD_HOC_USER or APP_USER Tokens are only used by commands send by an eMSP. As these are never used locally at the Charge Point like RFID.

Unknown Tokens received by the CPO in the `StartSession` Object don't need to be stored in the `Token` module. In other words, when a Token has been received via `StartSession`, the same Token does not have to be returned in a Token GET request from the eMSP. However, the information of the Token SHALL be put in the `Session` and `CDR`.

An eMSP sending a `StartSession` SHALL only use Token that are owned by this eMSP in `StartSession`, using Tokens of other eMSPs is not allowed.

	URL		
	Token		
	CiString		
	CiString		
	CiString		Session CDR

	URL		
	CiString		

	URL		
	CiString		
	CiString		
	CiString		

13.4.1. CommandResponseType enum

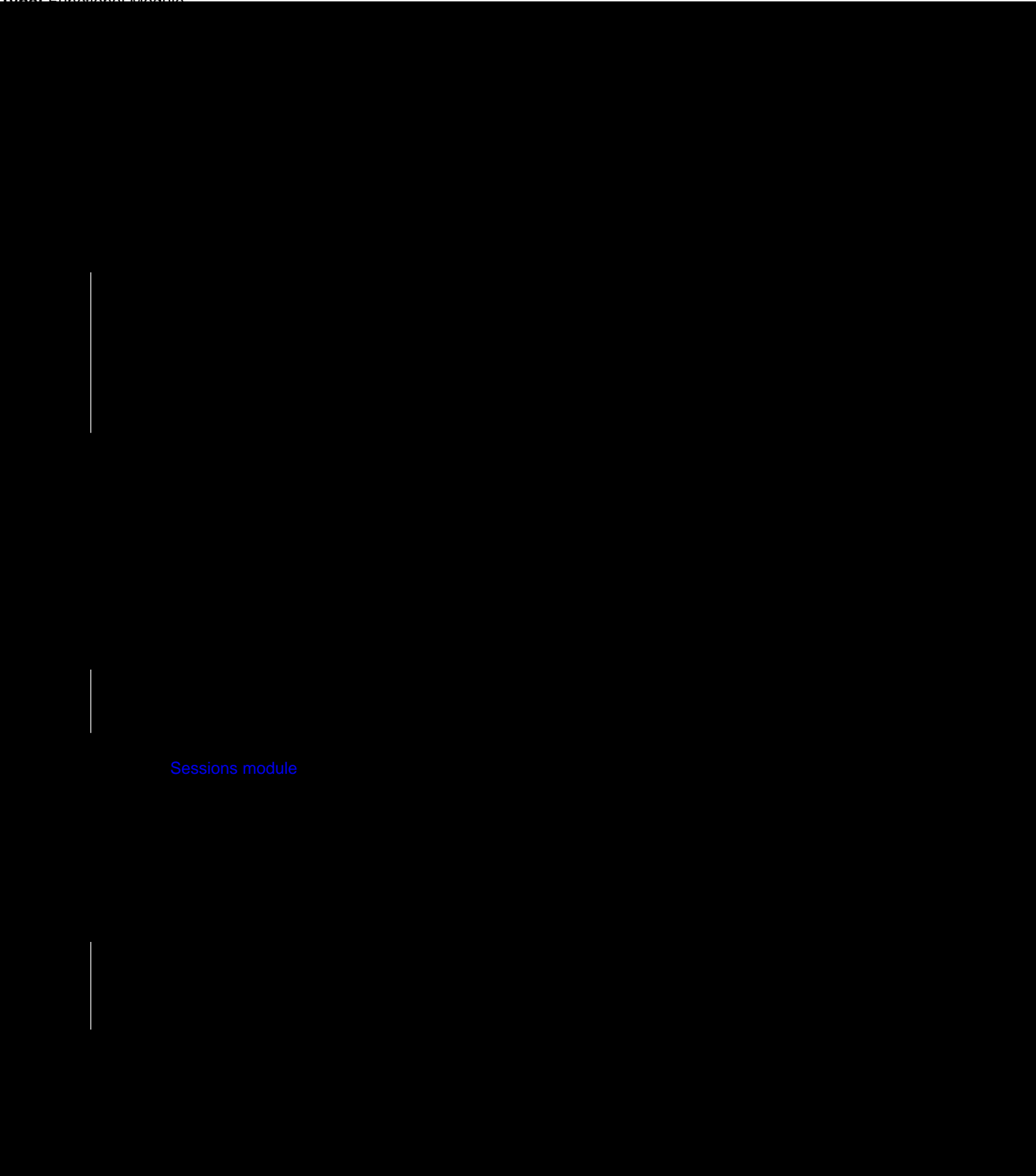
Response to the command request from the eMSP to the CPO.

Value	Description

14. ChargingProfiles module

Module Identifier: chargingprofiles

Type: Functional Module



free to do this.



Figure 36. Smart Charging Topology: The eMSP generates ChargingProfiles.

Session

Session

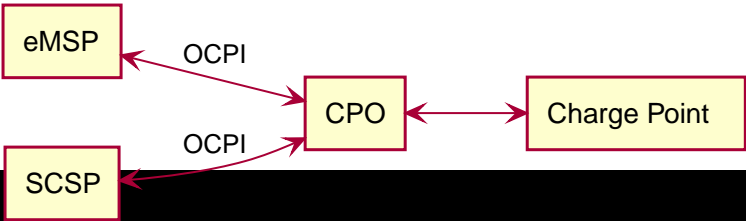


Session

Session

eMSPs customers.

In this topology, the eMSP is not aware that the CPO is using OCPI to receive Charging Profiles from the SCSP.



Commands

CPO PUT method

CPO GET method

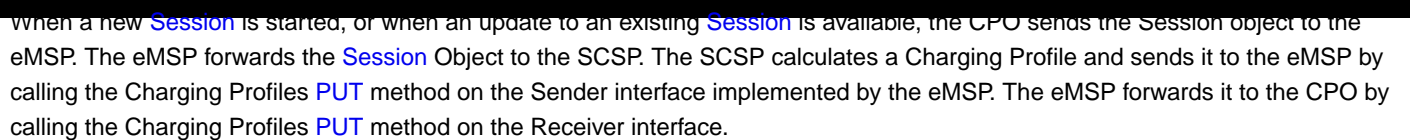
CPO DELETE method

MSP PUT method

MSP PUT method

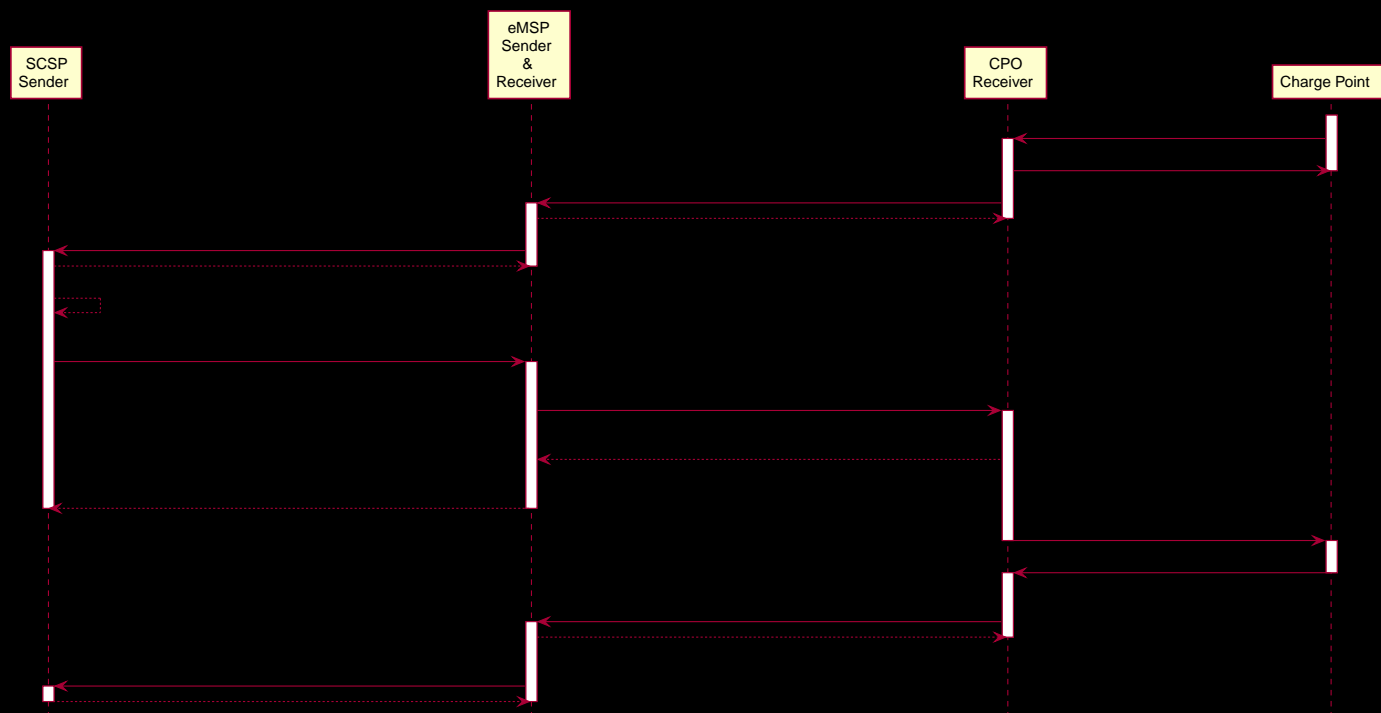
For calculating optimum ChargingProfiles it might be useful for the eMSP or SCSP to know the ChargingProfile that the Charge Point has planned for the Session: ActiveChargingProfile. The ActiveChargingProfile might differ from ChargingProfile requested via OCPI. There might be other limiting factors being taken into account by the CPO and or Charge Point, that limit the ChargingProfile. The ActiveChargingProfile profile can be requested by the Sender by calling the [CPO GET method](#) on the Charging Profile

The CPO can limit the amount of request that can be done on the Charging Profiles interface, this too prevent creating a too high load or data usages. To do this the CPO can reject a request on the Charging Profile Receiver interface be responding with: **TOO OFTEN**



The CPO responds to the eMSP, the response body will contain the response to the request, acknowledging the request was understood and can be forwarded to the Charge Point. The eMSP forwards this response to the SCSP.

The CPO sends the requests to the Charge Point. When the CPO receives a response from the Charge Point, that result is send to the eMSP by call the **POST** method, on the URL provided by the eMSP in the **PUT** request of the eMSP. The eMSP forwards this result to the the URL provided by the SCSP in the **PUT** request of the SCSP, this call will contain a **ChargingProfileResult** Object



DELETE

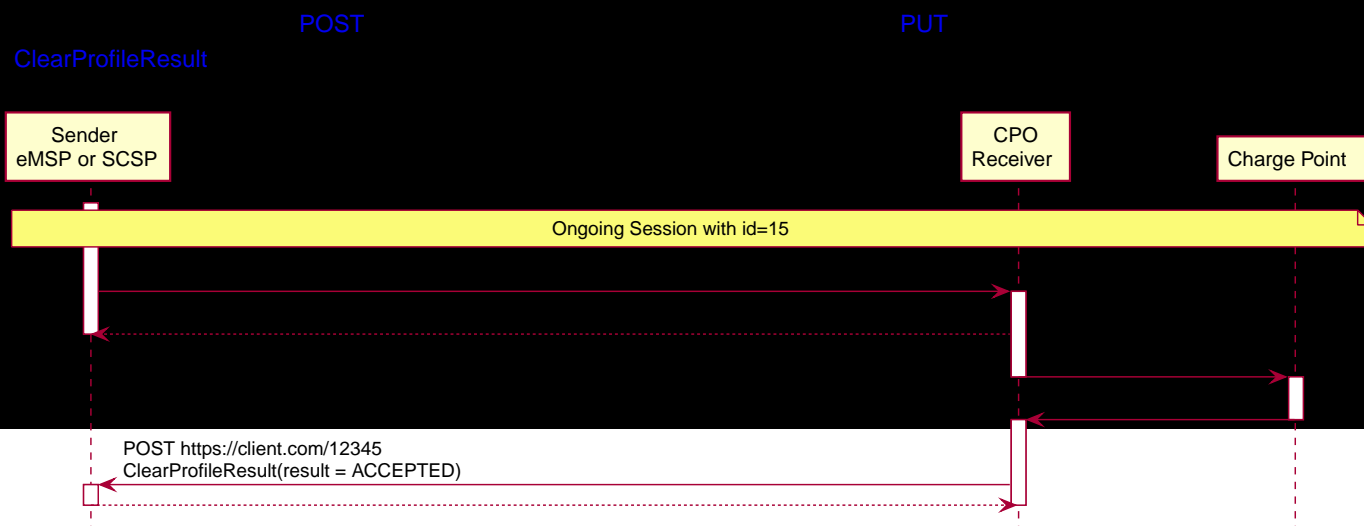
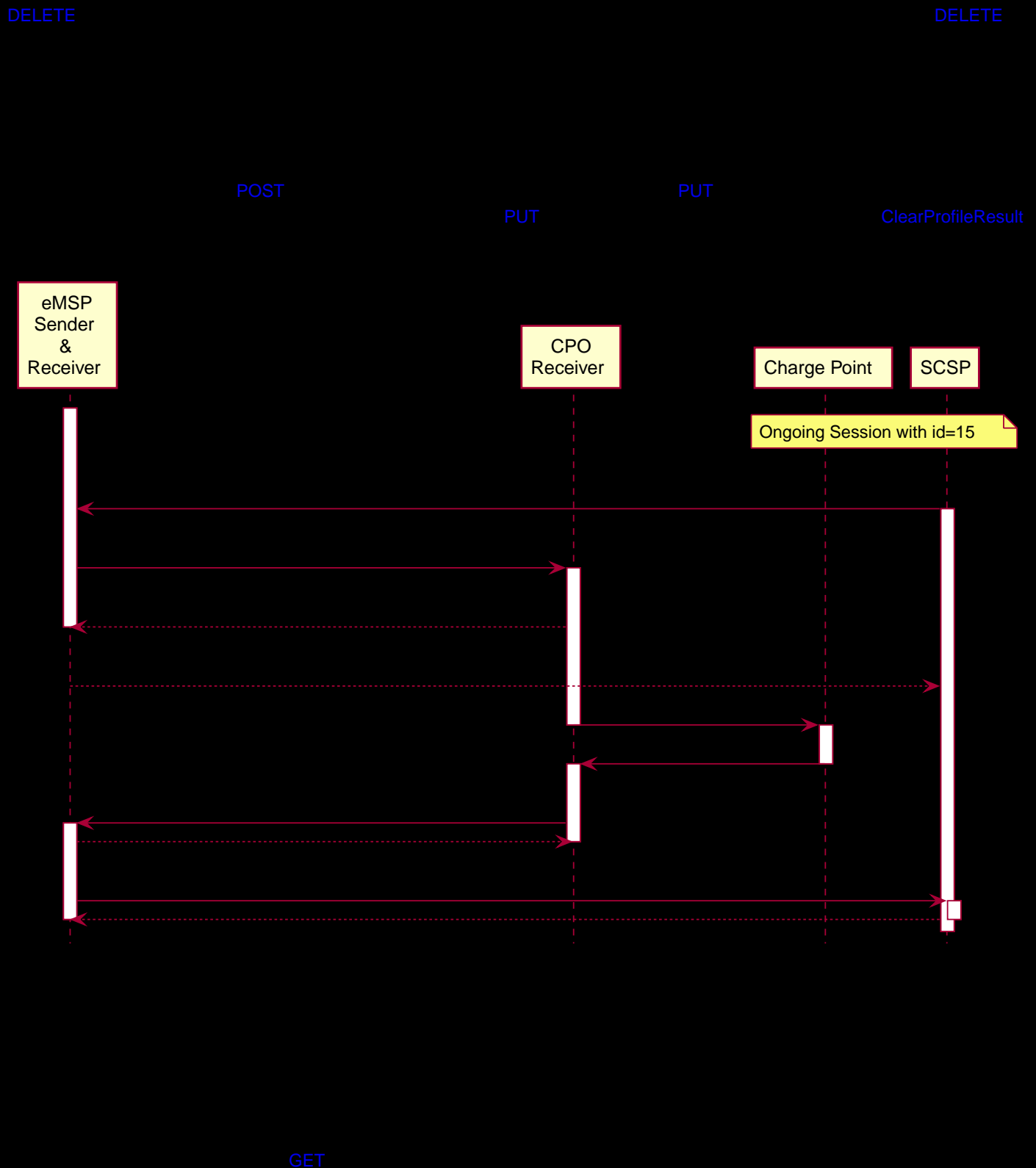


Figure 41. Example of a ClearChargingProfile.

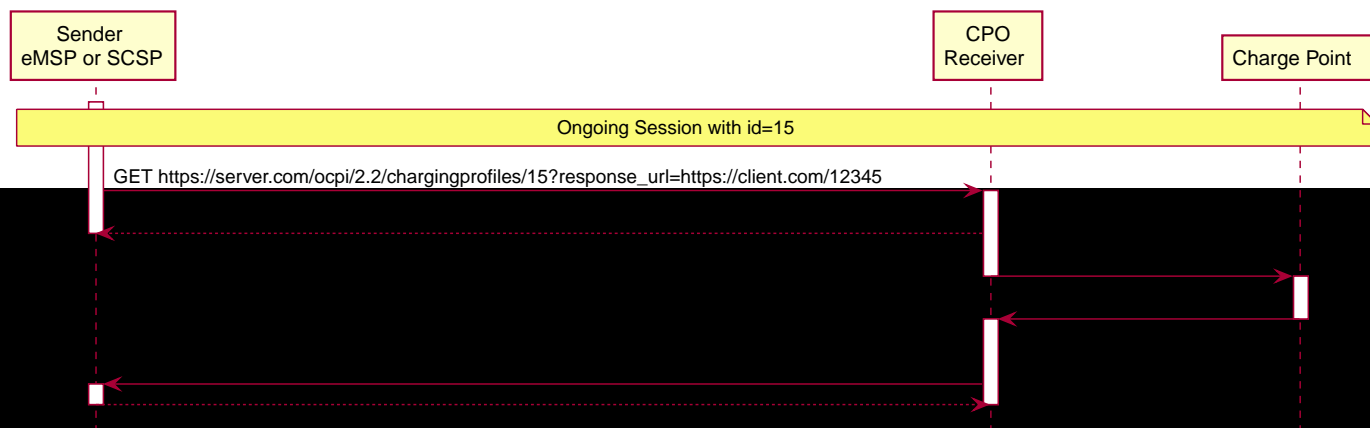
14.3.4. Example of a removing/clearing ChargingProfile send by the SCSP via the eMSP

The SCSP might want to remove the charging profile, for example the EV driver has selected to switch to charging with the highest cost available. The SCSP sends the MSP to the CPO to remove the active charging profile. This can be handled like the



was accepted and can be forwarded to the Charge Point.

The CPO sends a message to the Charge Point to retrieve the current active charging profile. When the CPO receives a response from the Charge Point, that **ActiveChargingProfile** is send to the eMSP by call the **POST** method, on the URL provided by the eMSP in the **PUT** request of the eMSP, this call will contain a **ActiveChargingProfileResult** Object.



GET
GET

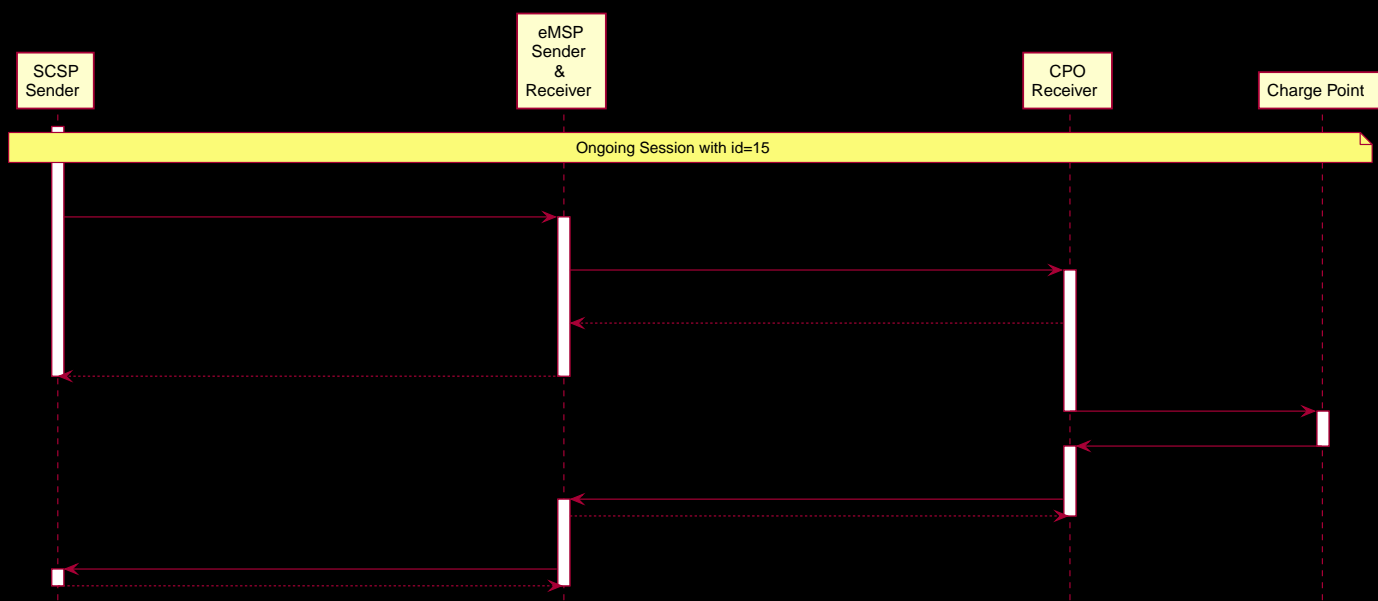
POST

GET

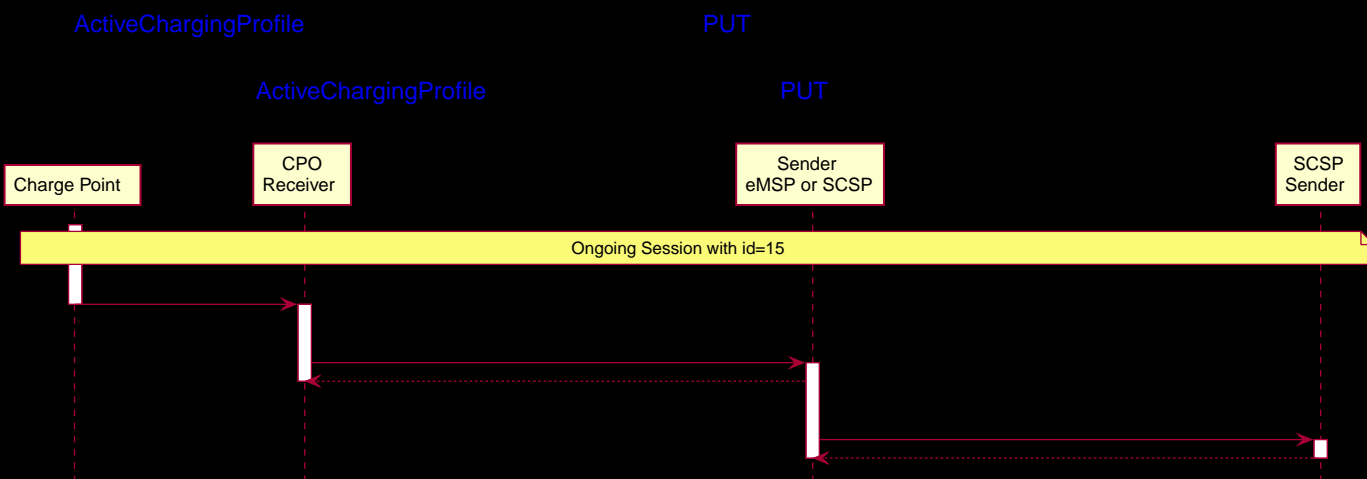
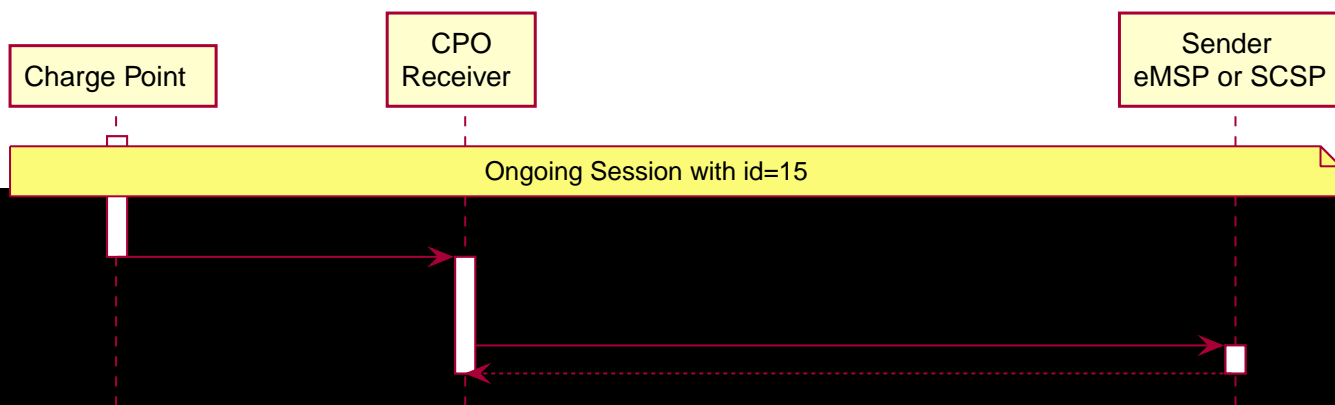
GET

ActiveChargingProfileResult

ActiveChargingProfileResult



When the CPO knows the [ActiveChargingProfile](#) of a Charge Point has changed, the Receiver (typically the CPO) sends this update [ActiveChargingProfile](#) to the Sender (typically the eMSP or SCSP), by calling the [PUT](#) method on the Sender interface.



GET	
PUT	
PATCH	n/a
DELETE	Cancels an existing ChargingProfile for a specific charging session.

14.4.1.1. GET Method

Retrieves the ActiveChargingProfile as it is currently planned for the the given session.

Endpoint structure definition:

--	--	--	--

	CiString		
	URL		ActiveChargingProfileResult

--	--	--	--

ChargingProfileResponse		

https://www.cpo.com/ocpi/2.2/chargingprofiles/1234

Request Parameters

The following parameters can be provided as URL segments.

Parameter	Datatype	Required	Description
	CiString		

SetChargingProfile			

ChargingProfileResponse			

|

	CiString		
	URL		ClearProfileResult

Response Data

The response contains the direct response from the Receiver (typically CPO), not the response from the EVSE itself, that will be

sent via an asynchronous POST on the Sender interface if this response is ACCEPTED.

Datatype	Card.	Description
ChargingProfileResponse	1	Result of the ChargingProfile DELETE request, by the CPO (not the location/EVSE). So this indicates if the CPO understood the ChargingProfile DELETE request and was able to

POST	
PUT	

<https://www.server.com/ocpi/2.2/12345678>

The content of the request body depends on the original request by the eMSP to which this POST is send as a result.

14.4.2.2. Request Body

Datatype	Card.	Description
Choice: one of three		
ActiveChargingProfileResult		
ChargingProfileResult		
ClearProfileResult		

Response Format

	CiString		

ActiveChargingProfile		
		EVSE.

14.4.2.6. Response Body

The response to the PUT on the eMSP interface SHALL contain the [Response Format](#) with the data field omitted.

14.5. Object description

	ChargingProfileResponseType		

	ChargingProfileResultType		
	ActiveChargingProfile		

	ChargingProfileResultType		

	ChargingProfileResultType		

14.5.5. SetChargingProfile Object

Object set to a CPO to set a Charging Profile.

Property	Type	Card.	Description
	ChargingProfile		
	URL		

	DateTime		
	ChargingProfile		

	DateTime		
			indefinitely or until end of the transaction in case startProfile is absent.
charging_rate_unit	ChargingRateUnit	1	The unit of measure.

Property	Type	Card.	Description
min_charging_rate	number	?	Minimum charging rate supported by the EV. The unit of measure is defined by the chargingRateUnit. This parameter is intended to be used by a local smart charging algorithm to optimize the power allocation for
	ChargingProfilePeriod		

ChargingProfile

	number		

15. *HubClientInfo* module

Module Identifier: hubclientinfo

Data owner: Hub

PUT

PUT

When the Hub invalidates a ClientInfo object (deleting is not possible), the Hub will send the updated ClientInfo object (with the field: status set to SUSPENDED, by calling the [PUT](#) method on the connected party ClientInfo endpoint with the updated ClientInfo object.

When the connected party is not sure about the state or existence of a ClientInfo object in the Hub system, the connected party can call the **GET** to request to ClientInfo object from the Hub system.

15.2.2. Pull model

GET

GET	
PUT	
	PUT

		red	
country_code	CiString(2)	yes	Country code of the requested ClientInfo object.
party_id	CiString(3)	yes	Party ID (Provider ID) of the requested ClientInfo object.

Response Data

The response contains the requested object.

Type	Card.	Description
ClientInfo		

ClientInfo		

	CiString		
	CiString		

--

GET	paginated
POST	n/a
PUT	n/a
PATCH	n/a

Method	Description
DELETE	n/a

15.3.2.1. GET Method

paginated	pagination		
	DateTime		
	DateTime		

pagination

ClientInfo		

15.4.1. ClientInfo Object

Property	Type	Card.	Description
party_id	CiString(3)	1	CPO or eMSP ID of this party. (following the 15118 ISO standard), as
	CiString		
	Role		
	ConnectionStatus		
	DateTime		

16. Types

16.1. CiString *type*

|

	string		
	string		

excl_vat	number	?	Price/Cost excluding VAT.
incl_vat	number	?	Price/Cost including VAT.

16.5.1. Role *enum*

Value	Description
CPO	Charge Point Operator Role

[w3.org spec](#)

17. Changelog

17.1. Changes between OCPI 2.1.1 and 2.2

			Token
CDRs / ChargingPeriod class	Medium / Medium	Minimal / Minimal	Added <code>tariff_id</code> field to ChargingPeriod, when the session switches from one tariff to another, this needs to be known, can be relevant with Preference based Smart Charging.

[illegible]

164

Context (Module / Object)	Expected Impact: eMSP / CPO	Expected Effort: eMSP / CPO	Description
Sessions /	Minor /	Minimal /	- Added <code>county_code</code> and <code>party_id</code> fields, making it easier to
			Token
Receiver GET & PUT methods	Minor	Minimal	<p>CiString, making them case-insensitive, which had always been the idea.</p> <p>Added <code>token_type</code> field, making it possible to make a distinction between different Token types with the same <code>uid</code>.</p>

Context (Module / Object)	Expected Impact: eMSP / CPO	Expected Effort: eMSP / CPO	Description
Tokens /	Minor /	Minimal /	Changed country_code, party_id and tariff_id from string to
		Minimum	character sets then only ASCII.