DirectPayment solution - OCPI 2.2.1

EVRoamingFoundation

info@evroaming.org

This is a separate module which can be used by CPOs and is an addition on the OCPI 2.2.1 protocol.
It supports direct payment with different Payment Terminal Providers.

[1]

March 2024

Working Document - changes can happen

[1] https://evroaming.org/

Revision History

Revision	Date	Author(s)	Description
1.0	29.08.23	PF	document created
1.1	03.10.23	PF	added examples to terminal and financial advice objects
1.2	09.10.23	PF	add reservation to transaction flow diagram, changed property names inside financial advice to capture instead of payment, removed start and end datetime from tariffs module, add token_id to StartSession, remove eMSP and replace with PTP
1.3	12.10.2023	PF	refactored financial-advice urls to use - instead of _, Added role information to all GET subsections titles, minor transaction flow adaptions
1.4	19.10.2023	PF	added business and functional useCase, added countryCode/partyId as alternative to customer_reference in a terminal object, clearer describe systems in transaction flow diagram
1.5	24.10.2023	PF	Added id and last_updated to tariff"light" module, added token.uid remark on startSession
1.6	08.12.2023	PF	Changed maxPrice to Price type. Added Terminals PUT endpoint. Added geolocation and address information to Terminal object
1.7	13.12.2023	PF	Added currency to FinancialAdviceConfirmation. Made invoice fields optional on terminal object.

1.8	21.02.2024	PF	Added last_updated fields to Terminal and financial advice confirmation objects. Changed type of Terminal.invoice_base_url to URL.
1.9	01.03.2024	PF	Updated useCases according to to the EVRF direct payment workgroup. Fixed minor typos.

Contents

Acronyms

CPO Charging Point Operator. 9–19, 21, 26, 28, 30, 32, 36, 38

CSMS Charger Station Management System. 14, 30

EFT Electronic Funds Transfer. 19, 21, 39

EVSE Electric Vehicle Supply Equipment. 18, 41, 42

Merchant In the context of this document, the merchant is (in most cases) the CPO. The merchant receives funds from the acquiring party in form of settlements directly into their bank account, for the charging services performed.. 7

Payment Terminal A payment terminal allows a merchant to capture card information and to transmit this data to the acquiring party for authorization and finally to transfer funds to the merchant. In order to provide acquiring services on a payment terminal strict protocols and certifications apply.. 10, 12–16, 21

PSP Payment Service Provider - In the context of this document this refers to the party providing acquiring services, which is typically a bank or institution that accepts and processes electronic payments. Merchant can enter into a direct contract with an acquirer or use the services of an intermediary (a payment service provider).. 10, 11, 16, 19, 39, 40, 42

PTP Payment Terminal Provider - In the context of this document, this refers to the party operating the payment terminal management system. This can be the terminal vendor (re/seller) and manufacturer.. 9–15, 17–19, 21, 22, 26, 28, 30, 32, 33, 36, 38, 39, 42–44

Chapter 1

Introduction

Starting from the beginning of 2024, the European Union will require direct payment at charging stations to streamline and enhance the electric vehicle charging experience. This move aims to create a more convenient and accessible system for EV users across EU member states, eliminating the need for multiple payment methods and subscriptions. By implementing direct payment, the EU aims to encourage wider EV adoption by reducing barriers and ensuring a standardized, user-friendly charging infrastructure throughout the region. To support the implementation of this regulation this solution was made.

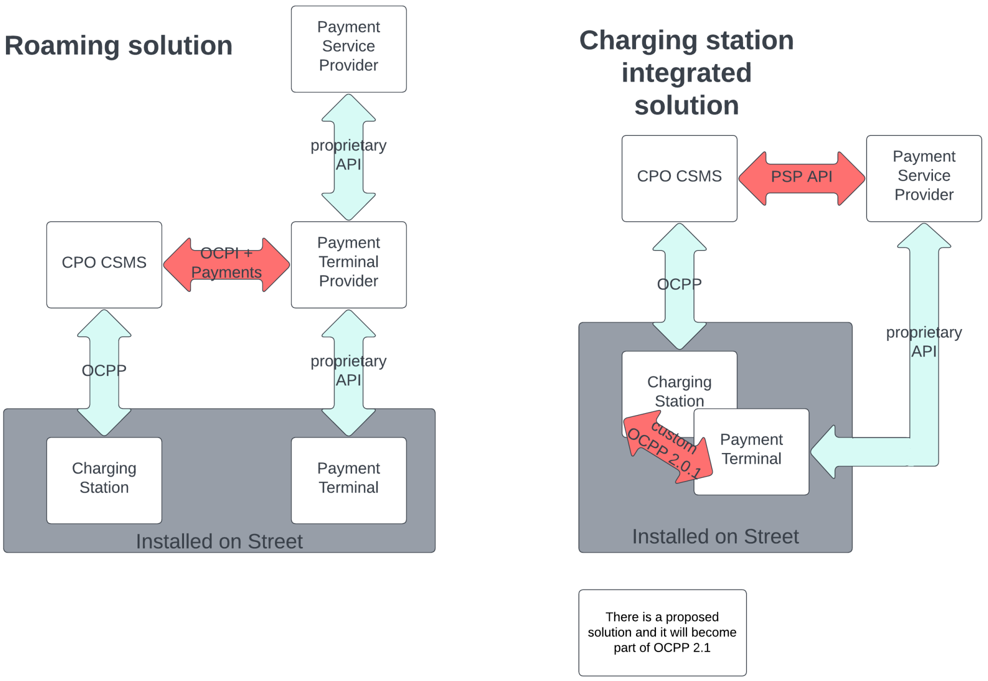

Figure 1.1: Difference between roaming and an integrated charging station solution

This should show the difference between roaming and an integrated charging station solution for direct payment support. There will be valid useCases for both approaches. The red arrows show "new"/"custom" implementations whereas the light blue ones show already existing ones.

The new payment module and all other changes are based on OCPI 2.2.1(https://github.com/ocpi/ocpi/releases/tag/2.2.1) and should be seen as an extension to the protocol. With this it should be possible to connect a CPO to a PTP. The new module provides the possibility to exchange information about the terminal and also about the financial advice at the end of a transaction. Therefore it consists of two new objects with their needed endpoints.

Chapter 2

Business UseCase

2.1 As a CPO I want to enable direct payment via payment terminals

Objective(s)	1. The PTP/PSP can process direct payment for EV charging on payment terminals. 2. As a CPO I want to link a charging station/group of charging station to specific Payment Terminals.
Description	When an EV driver uses a charging station, they may prefer the convenience of direct payment through a Payment Terminal rather than using an app or other methods or it can be required by regulations(e.g. AFIR, NEVI). This allows for faster transaction times and can be more familiar and intuitive for some users. By supporting this option, the CPO ensures greater accessibility for a wide range of EV drivers.
Actors	PTP, PSP, CPO
Preconditions	1. The Location is equipped with a Payment Terminal and is assigned to such a terminal by the CPO. 2. The EV driver has a valid ad-hoc/contactless payment method for payment. 3. The EV driver has viewed the price for charging and wishes to proceed. 4. The CPO has an agreement with either the PTP/PSP or both.

Postconditions	1. EV-drivers can pay for the charging session directly through the payment terminal. 2. PTP receives the payment details and can process the transaction accordingly with the PSP. 3. Charging session information, including payment details, is updated in real-time for the PTP and CPO. 4. The CPO will receive the payment. 5. The EV Driver receives an invoice from the CPO.

Chapter 3

Functional UseCases

3.1 As a CPO I want to display charging relevant information (e.g. connectors, tariffs etc.) on the payment terminal, so the EV driver is adequately informed.

Objective(s)	Ensure the terminal has and displays the latest data about the connectors and tariffs.
Description	The CPO system exchanges location (connector) + tariff information with the PTP via OCPI. This helps the EV-driver choose a suitable connector for charging.
Actors	PTP, CPO
Preconditions	1. The Payment Terminal is operational and online. 2. The CPO system is available for data requests.
Postconditions	1. The user can see the up-to-date charging relevant information on the Payment Terminal.

3.2 As a PTP, I want to remotely initiate a charging session upon the user's request.

Objective(s)	Allow the PTP system to request a charging session with the CPO.
Description	After the EV-driver selects a connector and reviews the price for charging the PTP authorises with the given payment method. Then the PTP system can request the CPO to start charging. The CPO responds with whether the request for start was accepted or not. Note: Maximum price in the OCPI tariffs module is used as the reservation amount in the pre-authorisation. Remark: In case there is only one connector this one can be autoselected.
Actors	PTP, CPO, EV-Driver
Preconditions	1. EVDriver has viewed the price for charging and wishes to proceed. 2. The Payment Terminal is operational and online. 3. The CPO system is available for data requests. 4. The payment method has been authorised.
Postconditions	1. Charging session is initiated.

3.3 As a CPO, I want to be able to stop a charging session.

Objective(s)	To enable the CPO to terminate a charging session, covering scenarios such as reaching the preauthorization limit, equipment-related issues, or receiving a manual stop request from the EV driver.
Description	The CPO is capable of ending a charging session for various reasons, including when the preauthorization limit set by the EV driver is reached, in case of a malfunction or safety concern at the charging station, or upon a manual stop request from the EV driver. It is crucial for the CPO to manage these terminations efficiently, ensuring communication with the PTP for payment processing and notifying the EV driver about the session's end and its reasons.
Actors	PTP, CPO, EV-Driver

Preconditions	1. A charging session was started at the Payment Terminal and is in progress.
Postconditions	1. Charging session is terminated gracefully, and relevant parties (evDriver/Payment Terminal/CSMS) are notified. 2. The PTP is notified for payment processing. 3. The EV driver is informed.

3.4 As a PTP, I want to request the CPO to stop a charging session.

Objective(s)	To oversee and facilitate the termination of charging sessions from a payment perspective, handling scenarios such as reaching payment limits or manual termination requests, and ensuring seamless communication with the CPO for accurate billing.
Description	The PTP plays a crucial role in managing the financial transactions associated with EV charging sessions. This includes terminating sessions when a preauthorized payment limit is reached or upon receiving a stop request. Stopping requests caused by the PTP is an edge case. Usually the CPO has the information about the current costs earlier and therefore will trigger the stopping.
Actors	PTP, CPO, EV-Driver
Preconditions	1. A charging session was started at the Payment Terminal and is in progress.
Postconditions	1. Charging session is terminated gracefully, and relevant parties (evDriver/Payment Terminal/CSMS) are notified. 2. The CPO has received the request to stop the charging session. 3. The EV driver is informed.

3.5 As a CPO, I want to tell the PTP which charging stations are linked to a specific terminal.

Objective(s)	Assigning locations to a Payment Terminal of a PTP so that it can be used for them.
Description	To make the payment process smoother and more organized, the CPO wishes to designate certain Payment Terminals to specific charging locations and/or EVSEs. This means that each Payment Terminal is only responsible for transactions at its assigned location(s)/EVSEs. By doing so, the CPO can monitor, manage, and maintain Payment Terminals and locations/EVSEs more efficiently.
Actors	PTP, CPO
Preconditions	1. Payment Terminals are set up and ready for assignments. 2. Locations are existing within the CPO's system.
Postconditions	1. The given location is now linked to a Payment Terminal. 2. EV drivers can select these locations using the assigned Payment Terminal.

3.6 As a CPO, I want to ensure that post-charging receipt are generated and communicated via the PTP to the user.

Objective(s)	If the CPO is the one who generates the receipt then an accurate receipt post-charging that reflects the associated costs and needed payment details needs to be created and communicated via the PTP to the user.
Description	After a charging session is completed, a receipt needs to be generated that clearly outlines the total cost, and other relevant transaction details. The CPO is responsible for ensuring this receipt is accurately created and then populated to the PTP in a timely manner.
Actors	PTP, CPO

Preconditions	1. The charging session has been successfully completed. 2. Payment was processed by the PSP. 3. The CPO has has received needed transaction information
Postconditions	1. The invoice has been accurately generated. 2. The invoice has been made available to the EV driver via the Payment Terminal UI.

Chapter 4

Usage flows

4.1 Terminal assignment

This flow shows the exchange and the assignment of the terminal object. This object will be owned and created by the PTP. After the object was pushed to/pulled by the CPO there will be the possibility to assign specific locations to this terminal. This assignment then will be pushed by the CPO to the PTP.

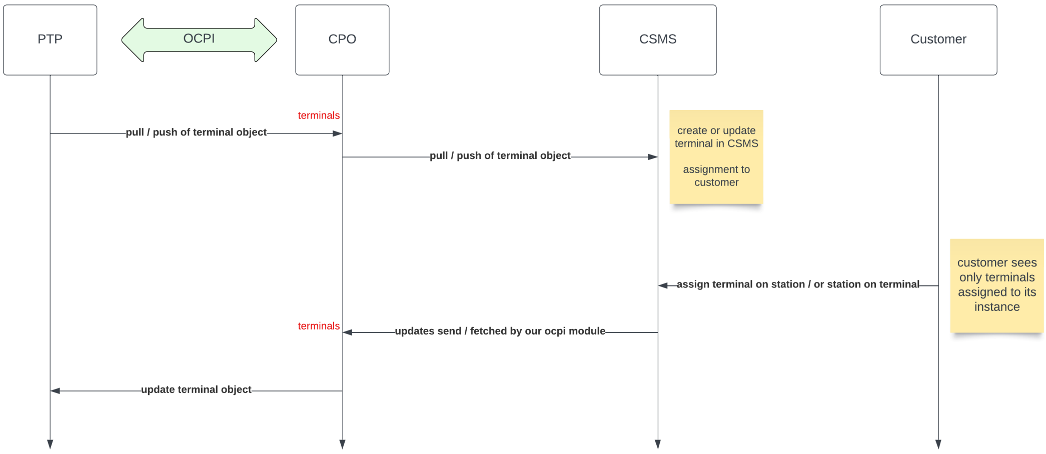
PTP
OCPI
CPO
CSMS
Customer
terminals
pull / push of terminal object
pull / push of terminal object
create or update terminal in CSMS
assignment to customer
customer sees only terminals assigned to its instance
assign terminal on station / or station on terminal
terminals
updates send / fetched by our ocpi module
update terminal object

Figure 4.1: Terminal assignment flow

4.2 Locations exchange

This flow shows the exchange of locations and their corresponding EVSE status updates. The PTP has to pull all the locations(that are assigned to one of his terminals) from the CPO so that he can map the location data to the assigned IDs on the terminal object. This part is similar to the OCPI standard flow for locations.

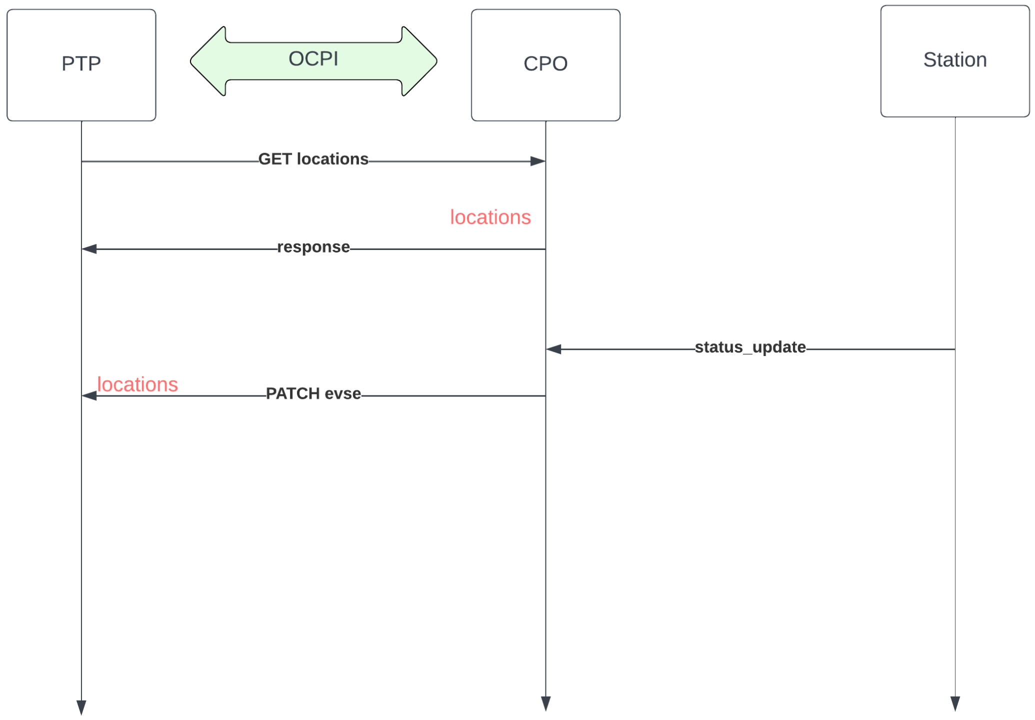

Figure 4.2: Location exchange flow

4.3 Transaction

This flow shows a single transaction in total. After choosing a specific connector on the terminal the PTP has to pull the current connector information to receive the applicable tariff_id. With this id the tariff should be pulled so that the alt_text can be shown on the terminal. Also the tariff is needed to reserve the needed preauth limit (stated in the max_price) at the PSP.

When the user accepts the tariff the PTP will reserve the max_price amount at the PTP and a start_session command will be sent to the CPO. The PTP will pass an authorization_reference with this request which will be used as the postfix of the invoice URL.

During the session there will be session updates pushed by the CPO if existing. For stopping the session there are 3 ways:

1. can be stopped by the car or by the station
2. a stop_session can be sent by the PTP when the preauth limit is reached
3. can also be stopped by the CPO backend if the preauth limit is reached

Now the PTP will receive a CDR with an invoice_reference_id. If this id is set then the invoice was created by the CPO, if not then the invoice will be created by the PTP.

If the CPO creates the invoice, the PTP has to push a financial-advice-confirmation object after he has done the capture at the PSP. When the CPO has received this object the previously created invoice has to be enriched with the required EFT data.

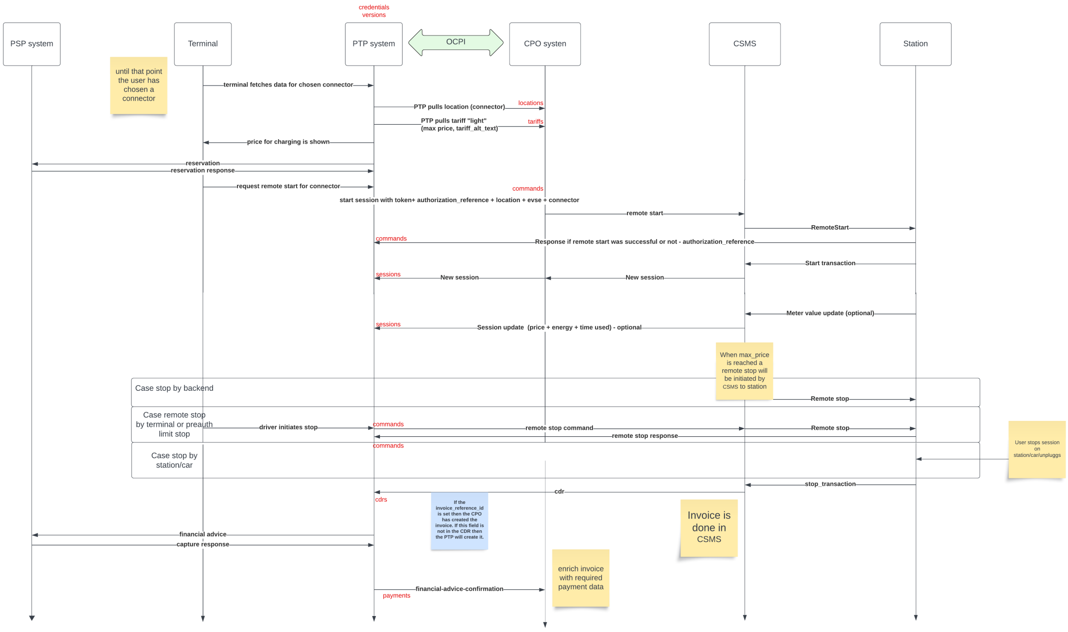
credentials
versions
PSP system
Terminal
PTP system
OCPI
CPO systen
CSMS
Station
until that point the user has chosen a connector
terminal fetches data for chosen connector
PTP pulls location (connector)
locations
PTP pulls tariff "light"
(max price, tariff_alt_text)
tariffs
price for charging is shown
reservation
reservation response
request remote start for connector
commands
start session with token+ authorization_reference + location + evse + connector
remote start
RemoteStart
commands
Response if remote start was successful or not - authorization_reference
Start transaction
sessions
New session
New session
Meter value update (optional)
sessions
Session update (price + energy + time used) - optional
When max_price is reached a remote stop will be initiated by CSMS to station
Case stop by backend
Remote stop
Case remote stop by terminal or preauth limit stop
driver initiates stop
commands
remote stop command
Remote stop
remote stop response
commands
Case stop by station/car
User stops session on station/car/unpluggs
stop_transaction
cdr
cdrs
If the invoice_reference_id is set then the CPO has created the invoice. If this field is not in the CDR then the PTP will create it.
Invoice is done in CSMS
financial advice
capture response
enrich invoice with required payment data
financial-advice-confirmation
payments

Figure 4.3: Transaction flow

Chapter 5

Payment module

Module Identifier: payments
Data owner: PTP
Type: Functional Module

This module should support the Payment Terminal use case for direct payment.
A terminal can handle multiple locations and/or EVSEs. It should be able for a CPO to assign them to a terminal object. At the end of a charging session there should be a CDR sent. There should be also send a financial-advice-confirmation from the PTP. This should contain the actual cost and EFT data. This object is only needed if the CPO creates the invoice.

5.1 Terminal

5.1.1 Endpoints

Method	Description
GET	pull all terminals as CPO
GET(byId)	
PATCH	terminal is updated by CPO(location assignment)
PUT	terminal is updated by CPO

Table 5.1: PTP side

5.1.1.1 GET from PTP

Depending on the URL Segments provided, the GET request can either be used to retrieve information about a list of available Terminals at a PTP (GET List) or it can be used to retrieve information about one specific Terminal (GET Object).

5.1.1.1.1 GET List: Request Parameters

Endpoint structure definition:

```
{terminals_endpoint_url}?
  [date_from={date_from}]&
  [date_to={date_to}]&
  [offset={offset}]&
  [limit={limit}]
```

Examples:
https://www.server.com/ocpi/ptp/2.2.1/payments/terminals/?date_from=2019-01-28T12:00:00&date_to=2019-01- 29T12:00:00

https://ocpi.server.com/2.2.1/payments/terminals/?offset=50

https://www.server.com/ocpi/2.2.1/payments/terminals/?date_from=2019-01-29T12:00:00&limit=100

https://www.server.com/ocpi/ptp/2.2.1/payments/terminals/?offset=50&limit=100

If the optional parameters date_from and/or date_to are provided, only Terminals with (last_updated) between the given date_from (including) and date_to (excluding) will be returned.
This request is paginated, it supports the pagination related URL parameters:

Property	Type	Cardinality	Description
date_from	DateTime	?	Only return Terminals that have last_updated after or equal to this Date/-Time (inclusive).
date_to	DateTime	?	Only return Terminals that have last_updated up to this Date/Time, but not including (exclusive).
offset	int	?	The offset of the first object returned. Default is 0.
limit	int	?	Maximum number of objects to GET

Table 5.2: Terminal GET request parameters

5.1.1.1.2 GET List: Response Data

This endpoint returns a list of Terminal objects. The header will contain the pagination related headers. Each object must contain all required fields. Fields that are not specified may be considered as null values. Any old information that is not specified in the response is considered no longer valid. For requests that use pagination, the response data provided by all the pages together is the new truth. Any old information not contained in any of the pages needs to be considered no longer valid.

Type	Cardinality	Description
Terminal	*	List of all Terminals.

Table 5.3: Terminal GET list response data

5.1.1.1.3 GET Object: Request Parameters

Endpoint structure definition for retrieving a Terminal:

```
{terminals_endpoint_url}/{terminal_id}
```

Examples:
https://www.server.com/ocpi/ptp/2.2.1/payments/terminals/55719888-ed09-4cca-82cc-803bdb77bf26

Property	Type	Cardinality	Description
terminal_id	CiAsciiString[1..36]	1	Terminal.terminal_id of the Terminal object to retrieve.

Table 5.4: Terminal GETById request parameters

5.1.1.1.4 GET Object: Response Data

The response contains the requested object:

Type	Cardinality	Description
Terminal	1	Requested Termninal.

Table 5.5: Terminal GETById response data

5.1.1.2 PATCH

This PATCH should be used by the CPO to assign location_ids to a terminal.

Example:
PATCH To URL: https://www.server.com/ocpi/ptp/2.2.1/payments/terminals/55719888-ed09-4cca-82cc-803bdb77bf26

```
{
  "location_ids": [
    "df37373d-1669-4127-a6ac-d86750095119",
    "a06dc823-3e5a-40c8-89cf-1b5b9e941412",
    "55719888-ed09-4cca-82cc-803bdb77bf26"
  ]
}
```

5.1.1.3 PUT

This PUT should be used by the CPO to update location data of a terminal. Example: PUT To URL: https://www.server.com/ocpi/ptp/2.2.1/payments/terminals/55719888-ed09-4cca-82cc-803bdb77bf26

```
{
  "customer_reference": "OMV",
  "invoice_base_url": "someNewURL",
  "location_ids": [
    "df37373d-1669-4127-a6ac-d86750095119",
    "a06dc823-3e5a-40c8-89cf-1b5b9e941412",
    "55719888-ed09-4cca-82cc-803bdb77bf26"
  ],
  "last_updated": "2019-12-10T17:16:15Z"
}
```

Method	Description
GET	PTP pulls terminal
GET(byId)	
POST	Terminal is pushed by PTP

Table 5.6: CPO side

5.1.1.4 GET from CPO

Depending on the URL Segments provided, the GET request can either be used to retrieve information about a list of Terminals that the PTP has created at the CPO (GET List) or it can be used to retrieve information about one specific created Terminal (GET Object).

5.1.1.4.1 GET List: Request Parameters

Endpoint structure definition:

```
{terminals_endpoint_url}?
  [date_from={date_from}]&
  [date_to={date_to}]&
  [offset={offset}]&
  [limit={limit}]
```

Examples:
https://www.server.com/ocpi/cpo/2.2.1/payments/terminals/?date_from=2019-01-28T12:00:00&date_to=2019-01- 29T12:00:00

https://ocpi.server.com/2.2.1/payments/terminals/?offset=50

https://www.server.com/ocpi/2.2.1/payments/terminals/?date_from=2019-01-29T12:00:00&limit=100

https://www.server.com/ocpi/cpo/2.2.1/payments/terminals/?offset=50&limit=100

If the optional parameters date_from and/or date_to are provided, only Terminals with (last_updated) between the given date_from (including) and date_to (excluding) will be returned.

This request is paginated, it supports the pagination related URL parameters:

Property	Type	Cardinality	Description
date_from	DateTime	?	Only return Terminals that have last_updated after or equal to this Date/Time (inclusive).
date_to	DateTime	?	Only return Terminals that have last_updated up to this Date/Time, but not including (exclusive).
offset	int	?	The offset of the first object returned. Default is 0.
limit	int	?	Maximum number of objects to GET

Table 5.7: Terminal GET request parameters

5.1.1.4.2 GET List: Response Data

This endpoint returns a list of Terminal objects. The header will contain the pagination related headers. Each object must contain all required fields. Fields that are not specified may be considered as null values. Any old information that is not specified in the response is considered no longer valid. For requests that use pagination, the response data provided by all the pages together is the new truth. Any old information not contained in any of the pages needs to be considered no longer valid.

Type	Cardinality	Description
Terminal	*	List of all Terminals.

Table 5.8: Terminal GET list response data

5.1.1.4.3 GET Object: Request Parameters

Endpoint structure definition for retrieving a Terminal:

```
{terminals_endpoint_url}/{terminal_id}
```

Examples:
https://www.server.com/ocpi/cpo/2.2.1/payments/terminals/55719888-ed09-4cca-82cc-803bdb77bf26

Property	Type	Cardinality	Description
terminal_id	CiAsciiString[1..36]	1	Terminal.terminal_id of the Terminal object to retrieve.

Table 5.9: Terminal GETById request parameters

5.1.1.4.4 GET Object: Response Data

The response contains the requested object:

Type	Cardinality	Description
Terminal	1	Requested Termninal.

Table 5.10: Terminal GETById response data

5.1.1.5 POST

The POST should be used by the PTP to create a newly shipped terminal on the CPO's system. Here, no location_ids should be included as the assignment will be done by the CPO. The object sent here can be just the terminal_id or an object with additional data if known through the terminal order process.

Example:
POST only id To URL: https://www.server.com/ocpi/cpo/2.2.1/payments/terminals/

```json
{
  "terminal_id": "452cf8a1-79aa-4a0e-9aee-dc788586053c"
}
```

POST object with additional data to URL:
https://www.server.com/ocpi/cpo/2.2.1/payments/terminals/

```json
{
  "terminal_id": "452cf8a1-79aa-4a0e-9aee-dc788586053c",
  "address": "Street 1",
  "city": "Vienna",
  "country": "AUT",
  "coordinates": {
    "latitude": "51.047599",
    "longitude": "3.729944"
  },
  "customer_reference": "Chargepoint",
  "invoice_base_url": "https://somecompany.com/invoices",
  "invoice_creator": "CPO",
  "location_ids": [],
  "last_updated": "2018-12-10T17:16:15Z"
}
```

5.1.2 Object definition

Property	Type	Cardinality	Description
terminal_id	CiAsciiString[1..36]	1	Unique ID that identifies a terminal.
customer_reference	CiAsciiString[1..36]	?	This reference will be used to link the terminal to a CSMS. The reference might also be provided via the order process.
party_id	CiString[3]	?	This is an alternative to the customer_reference which can be used.
country_code	CiString[2]	?	This is an alternative to the customer_reference which can be used.
address	CiString[45]	?	Street/block name and house number if available.
city	CiString[45]	?	City or town.
postal_code	CiString[10]	?	Postal code of the terminal, may only be omitted when the terminal has no postal code.
state	CiString[20]	?	State or province of the location, only to be used when relevant.
country	CiString[3]	?	ISO 3166-1 alpha-3 code for the country of this location.
coordinates	GeoLocation	?	Coordinates of the terminal.
invoice_base_url	URL	?	BaseURL to the downloadable invoice
invoice_creator	Enum	?	Describes which party creates the invoice for the eDriver.
location_ids	CiAsciiString[1..36]	+	List of all locations assigned to that terminal.
evse_uids	CiAsciiString[1..36]	+	List of all EVSEs assigned to that terminal.
last_updated	DateTime	1	Timestamp when this Terminal was last updated (or created).

Table 5.11: Terminal object

The invoice_creator enum consists of the following values:

- CPO
- PTP

5.1.3 Examples

Example of a newly created terminal sent from the PTP to the CPO:

```json
{
  "terminal_id": "452cf8a1-79aa-4a0e-9aee-dc788586053c",
  "customer_reference": "Chargepoint",
  "address": "Street 1",
  "city": "Vienna",
  "country": "AUT",
  "coordinates": {
    "latitude": "51.047599",
    "longitude": "3.729944"
  },
  "invoice_base_url": "https://somecompany.com/invoices",
  "invoice_creator": "CPO",
  "location_ids": [],
  "last_updated": "2018-12-10T17:16:15Z"
}
```

Example of a terminal with assigned locations:

```json
{
  "terminal_id": "9e94f62c-661b-4afa-b6da-019b58fab9ac",
  "address": "Street 1",
  "city": "Vienna",
  "country": "AUT",
  "coordinates": {
    "latitude": "51.047599",
    "longitude": "3.729944"
  },
  "customer_reference": "BP",
  "invoice_base_url": "https://somecompany.com/invoices",
  "invoice_creator": "PTP",
  "location_ids": [
    "df37373d-1669-4127-a6ac-d86750095119",
    "a06dc823-3e5a-40c8-89cf-1b5b9e941412",
    "55719888-ed09-4cca-82cc-803bdb77bf26"
  ],
  "last_updated": "2018-12-10T17:16:15Z"
}
```

Example of a terminal with assigned locations and assigned EVSEs:

```json
{
  "terminal_id": "9e94f62c-661b-4afa-b6da-019b58fab9ac",
  "address": "Street 1",
  "city": "Vienna",
  "country": "AUT",
  "coordinates": {
    "latitude": "51.047599",
    "longitude": "3.729944"
  },
  "customer_reference": "BP",
  "invoice_base_url": "https://somecompany.com/invoices",
  "invoice_creator": "PTP",
  "location_ids": [
    "df37373d-1669-4127-a6ac-d86750095119",
    "a06dc823-3e5a-40c8-89cf-1b5b9e941412",
    "55719888-ed09-4cca-82cc-803bdb77bf26"
  ],
  "evse_uids": [
    "17d5f8ea-8832-454f-aff5-257bc6a25353"
  ],
  "last_updated": "2018-12-10T17:16:15Z"
}
```

5.2 Financial Advice Confirmation

This object will be used by the PTP to describe a finished financial transaction at the PSP. It contains all needed information so that the CPO can enrich his invoice with it.

5.2.1 Endpoints

Method	Description
GET GET(byId)	pull all financial-advice-confirmation as CPO.

Table 5.12: PTP side

5.2.1.1 GET from PTP

Depending on the URL Segments provided, the GET request can either be used to retrieve information about a list of all financial advice confirmations at a PTP (GET List) or it can be used to retrieve information about one specific financial advice confirmation (GET Object).

5.2.1.1.1 GET List: Request Parameters

Endpoint structure definition:

```
{financial_advice_confirmation_endpoint_url}?
  [date_from={date_from}]&
  [date_to={date_to}]&
  [offset={offset}]&
  [limit={limit}]
```

Examples:
https://www.server.com/ocpi/ptp/2.2.1/payments/financial-advice-confirmations/?date_from=2019-01-28T12:00:00&date_to=2019-01-29T12:00:00

https://ocpi.server.com/2.2.1/payments/financial-advice-confirmations/?offset=50

https://www.server.com/ocpi/2.2.1/payments/financial-advice-confirmations/?date_from=2019-01-9T12:00:00&limit=100

https://www.server.com/ocpi/ptp/2.2.1/payments/financial-advice-confirmations/?offset=50&limit=100

If the optional parameters date_from and/or date_to are provided, only financial advice confirmations with (last_updated) between the given date_from (including) and date_to (excluding) will be returned.
This request is paginated, it supports the pagination related URL parameters:

Property	Type	Cardinality	Description
date_from	DateTime	?	Only return financial advice confirmations that have last_updated after or equal to this Date/Time (inclusive).
date_to	DateTime	?	Only return financial advice confirmations that have last_updated up to this Date/Time, but not including (exclusive).
offset	int	?	The offset of the first object returned. Default is 0.
limit	int	?	Maximum number of objects to GET

Table 5.13: Financial Advice confirmations GET request parameters

5.2.1.1.2 GET List: Response Data

This endpoint returns a list of financial advice confirmation objects. The header will contain the pagination related headers. Each object must contain all required fields. Fields that are not specified may be considered as null values. Any old information that is not specified in the response is considered no longer valid. For requests that use pagination, the response data provided by all the pages together is the new truth. Any old information not contained in any of the pages needs to be considered no longer valid.

Type	Cardinality	Description
Financial Advice confirmation	*	List of all Financial Advice confirmations.

Table 5.14: Financial Advice confirmation GET list response data

5.2.1.1.3 GET Object: Request Parameters

Endpoint structure definition for retrieving a Terminal:

```
{financial_advice_confirmations_endpoint_url}/{ ↩
   financial_advice_confirmation_id}
```

Examples:
https://www.server.com/ocpi/ptp/2.2.1/payments/financial-advice-confirmations/55719888-ed09-4cca-82cc-803bdb77bf26

Property	Type	Cardinality	Description
financial_advice_confirmation_id	CiAsciiString[1..36]	1	Financial Advice confirmation.id of the financial_advice_confirmation object to retrieve.

Table 5.15: Financial Advice confirmation GETById request parameters

5.2.1.1.4 GET Object: Response Data

The response contains the requested object:

Type	Cardinality	Description
Financial Advice confirmation	1	Requested Financial Advice confirmation.

Table 5.16: Financial Advice confirmation GETById response data

Method	Description
GET	pull all financial-advice-confirmations
GET(byId)	
POST	financial-advice-confirmations is sent by the PTP

Table 5.17: CPO side

5.2.1.2 GET from CPO

Depending on the URL Segments provided, the GET request can either be used to retrieve information about a list of Financial Advice confirmations that the PTP has created at the CPO (GET List) or it can be used to retrieve information about one specific created Financial Advice confirmation (GET Object).

5.2.1.2.1 GET List: Request Parameters

Endpoint structure definition:

```
{financial_advice_confirmations_endpoint_url}?
  [date_from={date_from}]&
  [date_to={date_to}]&
  [offset={offset}]&
  [limit={limit}]
```

Examples:
https://www.server.com/ocpi/cpo/2.2.1/payments/financial-advice-confirmations/?date_from=2019-01-28T12:00:00&date_to=2019-01- 29T12:00:00

https://ocpi.server.com/2.2.1/payments/financial-advice-confirmations/?offset=50

https://www.server.com/ocpi/2.2.1/payments/financial-advice-confirmations/?date_from=2019-01-29T12:00:00&limit=100

https://www.server.com/ocpi/cpo/2.2.1/payments/financial-advice-confirmations/?offset=50&limit=100

If the optional parameters date_from and/or date_to are provided, only Terminals with (last_updated) between the given date_from (including) and date_to (excluding) will be returned.
This request is paginated, it supports the pagination related URL parameters:

Property	Type	Cardinality	Description
date_from	DateTime	?	Only return Financial Advice confirmations that have last_updated after or equal to this Date/Time (inclusive).
date_to	DateTime	?	Only return Financial Advice confirmations that have last_updated up to this Date/Time, but not including (exclusive).
offset	int	?	The offset of the first object returned. Default is 0.
limit	int	?	Maximum number of objects to GET

Table 5.18: Financial Advice confirmations GET request parameters

5.2.1.2.2 GET List: Response Data

This endpoint returns a list of Financial Advice confirmation objects. The header will contain the pagination related headers. Each object must contain all required fields. Fields that are not specified may be considered as null values. Any old information that is not specified in the response is considered no longer valid. For requests that use pagination, the response data provided by all the pages together is the new truth. Any old information not contained in any of the pages needs to be considered no longer valid.

Type	Cardinality	Description
Financial Advice confirmation	*	List of all Financial Advice confirmation.

Table 5.19: Financial Advice confirmation GET list response data

5.2.1.2.3 GET Object: Request Parameters

Endpoint structure definition for retrieving a Financial Advice confirmation:

```
{financial_advice_confirmation_endpoint_url}/{ ↩
   financial_advice_confirmation_id}
```

Examples:
https://www.server.com/ocpi/cpo/2.2.1/payments/financial-advice-confirmations/55719888-ed09-4cca-82cc-803bdb77bf26

Property	Type	Cardinality	Description
financial_advice_confirmation_id	CiAsciiString[1..36]	1	Financial Advice confirmation.id of the financial_advice_confirmation object to retrieve.

Table 5.20: Financial Advice confirmation GETById request parameters

5.2.1.2.4 GET Object: Response Data

The response contains the requested object:

Type	Cardinality	Description
Financial Advice confirmation	1	Requested Financial Advice confirmation.

Table 5.21: Financial Advice confirmation GETById response data

5.2.1.3 POST

The POST should be used by the PTP to create a Financial Advice confirmation on the CPO's system. This will be used to get the status of the capture and also the required eft data to put on the invoice.

Example:
POST To URL: https://www.server.com/ocpi/cpo/2.2.1/payments/financial-advice-confirmations/

```
{
  "id": "452cf8a1-79aa-4a0e-9aee-dc788586053c",
  "authorization_reference": "pp-100100-1948213567",
  "total_costs": {
    "excl_vat": 4.00,
    "incl_vat": 4.40
  },
  "currency": "EUR",
  "eft_data": [
    "Mastercard",
    "AID: 1234",
    "Crypto: 3456",
    "Nr: **** **** **** 1234",
    "SEQ: 00",
    "Amount: EUR 4.40"
  ],
  "capture_status_code": "SUCCESS",
  "capture_status_message": "Capture successfull at PSP",
  "last_updated": "2018-12-10T17:16:15Z"
}
```

5.2.2 Object definition

Property	Type	Cardinality	Description
id	CiAsciiString[1..36]	1	Uniquely identifies the financial−advice−confirmation.
authorization_reference	CiAsciiString[1..36]	1	Reference to the authorization given by the PTP.
total_costs	Price	1	Real amount that was captured at the PSP. This is a consumer price with VAT.
currency	CiAsciiString[3]	1	ISO-4217 code of the currency of this tariff.
eft_data	CiAsciiString[1..tbd]	+	Invoice relevant data from the direct payment. EFT
capture_status_code	CaptureStatusCode	1	Code that identifies the financial advice status.
capture_status_message	CiAsciiString[1..255]	?	Message about any error at the financial advice.
last_updated	DateTime	1	Timestamp when this financial advice confirmation was last updated (or created).

Table 5.22: Financial Advice Confirmation object

The capture_status_code enum consists of the following values:

- SUCCESS
- PARTIAL_SUCCESS
- FAILED

5.2.3 Examples

Example of a successful capture at the PSP:

```json
{
  "id": "452cf8a1-79aa-4a0e-9aee-dc788586053c",
  "authorization_reference": "pp-100100-1948213567",
  "total_costs": {
    "excl_vat": 4.00,
    "incl_vat": 4.40
  },
  "currency": "EUR",
  "eft_data": [
    "Mastercard",
    "AID: 1234",
    "Crypto: 3456",
    "Nr: **** **** **** 1234",
    "SEQ: 00",
    "Amount: EUR 4.40"
  ],
  "capture_status_code": "SUCCESS",
  "capture_status_message": "Capture successfull at PSP",
  "last_updated": "2018-12-10T17:16:15Z"
}
```

Example of a unsuccessful capture at the PSP:

```json
{
  "id": "452cf8a1-79aa-4a0e-9aee-dc788586053c",
  "authorization_reference": "pp-100100-1948213567",
  "total_costs": {
    "excl_vat": 0.00,
    "incl_vat": 0.00
  },
  "currency": "EUR",
  "capture_status_code": "FAILED",
  "capture_status_message": "Capture unsuccessful at PSP",
  "last_updated": "2018-12-10T17:16:15Z"
}
```

Chapter 6

OCPI adjustments

This chapter contains adjustments to the standard of OCPI 2.2.1. All of them are cardinality changes that are needed only to support the direct payment useCase.

6.1 Locations module

6.1.1 Connector

There must always be a tariff_id to be able to show something to the customer.

Property	Type	Cardinality	Description
tariff_ids	CiAsciiString[1..36]	1+	Identifiers of the currently valid charging tariffs.

Table 6.1: Connector Object

6.1.2 EVSE

The physical_reference will be used to identify the EVSE within a charging site/pool. The EVSE.uid must be present because it is needed within the start_session command.

Property	Type	Cardinality	Description
physical_reference	CiAsciiString[1..16]	1	A number/string printed on the outside of the EVSE for visual identification.
uid	CiAsciiString[1..36]	1	

Table 6.2: EVSE Object

6.2 Tariffs module

This will be the minimal requirements for a tariff module to support a fast MVP. This **should** be extended to the full tariff module so that the PTP can calculate the whole tariff and has not just a textual description of it.

6.2.1 Tariff

Max_price must be set because it is needed in the preauth capture.

Property	Type	Cardinality	Description
id	CiAsciiString[1..36]	1	Uniquely identifies the tariff within the CPO's platform
max_price	Price	1	Maximum amount for the authorization at the PSP.
currency	CiAsciiString[3]	1	ISO-4217 code of the currency of this tariff.
tariff_alt_text	DisplayText	1+	List of multi-language alternative tariff info texts. Will be used to display the tariff on the terminal device.
last_updated	DateTime	1	Timestamp when this Tariff was last updated (or created).

Table 6.3: Tariff Object

6.3 Commands module

6.3.1 StartSession

All levels from location-EVSE-connector must be present in a start_session to clearly identify on which connector the charging should start. Also it is important that the authorization_reference is passed by as it will be used as part of the invoice URL at the end.

Property	Type	Cardinality	Description
evse_uid	CiAsciiString[1..36]	1	
connector_id	CiAsciiString[1..36]	1	
authorization_reference	CiAsciiString[1..36]	1	Reference to the authorization given by the PTP, when given, this reference will be provided in the relevant Session and/or CDR. This reference can be for example provided as pp-cardterminalId-paymentTokenTraceNo. Must be unique.

Table 6.4: StartSession Object

Remark: The Token.uid should be unique and it should be the paymentTokenTraceNo

6.4 Sessions module

6.4.1 Session

Property	Type	Cardinality	Description
authorization_reference	CiAsciiString[1..36]	1	Reference to the authorization given by the PTP, when given, this reference will be provided in the relevant Session and/or CDR. This Reference can be provided as pp-cardterminalId-paymentTokenTraceNo. Must be unique. Must reference to the StartSession object.

Table 6.5: Session Object

6.5 CDRs module

6.5.1 CDR

Property	Type	Cardinality	Description
authorization_reference	CiAsciiString[1..36]	1	Reference to the authorization given by the PTP, when given, this reference will be provided in the relevant Session and/or CDR. This reference can be provided as pp-cardterminalId-paymentTokenTraceNo. Must be unique. Must reference to the StartSession object.

Table 6.6: CDR Object