National Parking Platform

National Parking Platform

Developer Guide

Version 2.20

Publication 14th of April, 2026

Contents

Introduction

This document should be read with the NPP API Specification. Whilst the API Specification serves as an interface reference, the NPP Developer Guide provides guidance for developers based on concrete use cases. The use cases typically relate to specific roles:

- Parking Operator
- Parking Service Provider
- Connected (Enforcement System) Supplier

You should read this document with your own NPP role in mind.

> *This document is a working document. It will be updated when new use cases are introduced or necessary changes are recognised (e.g. additional details and explanations based on developer feedback/questions).*

To further support developers, a PostMan™ collection is provided that includes sample requests for the use cases.

Use Case Overview

This section provides an overview of the use cases covered in this version of the NPP Developer Guide. Use it as a springboard to the use cases relevant to you.

1. Service Provider Use Cases

1.1 Retrieve Inventory Information

The Inventory API provides operational data such as parking location details, applicable right specifications, and rate table information. It will be used in particular as a data source for the service provider's tariff engine.

Information can be called up (e.g. for initial load and cyclic sync purposes), and the NPP push notification service provides new data and updates to subscribers to avoid the need for continuous polling.

1.2 Create Session (Pay on Arrival / Check in, check out)

A service provider notifies the NPP about a newly-issued parking right and a parking session using this right.

1.3 Extend Session

An existing parking session is extended.

1.4 Adjacent Sessions

As a precaution, the service provider checks whether the customer has already purchased a session elsewhere.

1.5 Transaction

A service provider posts a transaction for consideration in the monthly reconciliation run.

1.6 Close-out Message

A service provider posts the close-out message for the current reconciliation period. It defines which previously submitted transactions (payments and refunds) shall be included in the consolidated reconciliation report.

2. Connected Supplier Use Cases

In this first version of the document, enforcement system providers are the only category of connected suppliers.

2.1 Check Parking Right (Fetch)

The enforcement system actively fetches parking right information.

2.2 Parking Right (Push Notification)

The enforcement system (subscribed to the NPP push service) receives a notification of a new or updated parking right.

2.3 Sessions (Fetch)

The enforcement system actively fetches session information.

2.4 Sessions (Push Notification)

The enforcement system (subscribed to the NPP push service) receives a notification of a new or updated session.

3. Operator Use Cases

3.1 Upload New Rate Table

A parking operator uploads a new tariff.

3.2 Upload New Location

A parking operator uploads a new location record to the NPP.

3.3 Update Location

A parking operator updates an existing location record.

3.4 Upload New Right Specification

A parking operator uploads a new right specification (e.g. charging hours).

3.5 Update Right Specification

A parking operator updates an existing right specification.

3.6 Reconciliation Report

An operator retrieves monthly reconciliation information.

4. Shared Use Cases

4.1 Read List of Organisations on NPP

An NPP user reads the list of organisations (contacts) known to the NPP.

Use Case Details

This section provides detailed examples for all current use cases.

Preliminary Remarks

The examples in this session presume the existence of

- one operator OPERATOR1 representing city council COUNCIL1
- two ficticious service providers *PROVIDER1* and *PROVIDER2*
- two off-street parking locations *CARPARK1* and *CARPARK2*
- two on-street parking locations *STREET1* and *STREET2*

For better readability, we often use telling ids instead of UUID-type ids.

The HTTP requests shown in this document are presumed to include all mandatory headers, in particular:

- Authorization: Bearer *{valid access token}*
- Accept: application/json
- Content-type: application/json
- X-Client-Version: *{your application version}*

1. Service Provider Use Cases

1.1 Retrieve Inventory Information

The Inventory API provides operational data such as parking location details, applicable right specifications, and rate table information. It will be used in particular as a data source for the service provider's tariff engine.

Information can be called up (e.g. for initial load and cyclic sync purposes), and the NPP push notification service provides new data and updates to subscribers to avoid the need for continuous polling.

1.1.1 Initial Load (Locations)

1.1.1.1 Read Locations

```
GET /v4/parking/places?expand=all
```

This call returns a paginated list of all known locations. A typical response will have the following format:

```
{
	"meta": {
		"referenceInstant": 1750166111,
		"offset": 0,
		"pageSize": 200,
		"total": 2
	},
	"data": [
		{
			"id": "CARPARK1",
			"version": 1,
			// …details of car park 1
		},
		{
			"id": "CARPARK2",
			"version": 1,
			// …details of car park 2
		}
	]
}
```

The *meta* data header provides pagination information (*offset*: index of the first item on this page, *pageSize*: maximum number of data items per page, *total*: total number of result records spread across *{total/pageSize}* pages, *referenceInstant*: timestamp when the information was retrieved from the NPP database).

If – in later calls – you only want to retrieve changes to this initial data set, you can use the *modified_since* query parameter with the value of *referenceInstant*:

```
GET /v4/parking/places?expand=all&modified_since=1750166111
```

Using this mechanism, you can maintain a copy of all relevant data in your own system.

1.1.1.2 Location Details

Now, let's have a look at the returned location details. For our ficticious car park 1, they look like this:


```
{
	"id": "CARPARK1",
	"version": 1,
	"layer": 1,
	"type": "place",
	"name": [{"language":"en","string":"Market Place"}],
	"description": [{"language":"en","string":"more details…"}],
	"parentId": {
		"id": "ROOTNODEID",
		"version": 1
	},
	"indicativePlacePointLocation": {
		"type": "Point",
		"coordinates": [-2.146692,54.298062]
	},
	"rightSpecifications": [
		{
			"id": "RIGHTSPEC1",
			"version": 1
		}
	],
	"paymentMethods": [
	    {
	        "paymentMode": ["payOnEntry"]
	    }
	],
	"placeStreetAddress": {
	    "postCode": "CD10 3AC",
	    "city": [{ "language": "en","string": "COUNCIL1 City"}],
	    "addressLines": [
	        {
	            "type": "street",
	            "order": 0,
	            "text": "92 Market Street"
	        }
	    ]
	},
	"contacts": [
	    {
	        "organisationName": [{"language":"en","string":"Council1"}],
	        "type": "operator",
	        "emailCommonData": [
	            {
	                "address": "enquiries@testcouncil.gov.uk",
	                "typeCode": "customerService"
	            }
	        ]
	    }
	],
	"operatorDefinedReference": {
		"id": "CARPARK1_COST_CODE",
		"version": 1,
		"className": "CostCode"
	},
```

```
        "responsibilityRoleAssignments": [
                {
                    "type": "operator",
                    "contactPoints": [
                        {
                            "id": "TESTCOUNCIL",
                            "version": 1,
                            "organisationName": [
                                {
                                    "language": "en",
                                    "string": "Test Council"
                                }
                            ]
                        }
                    ]
                }
        ]

}
```


Notes

Field(s)	Notes
id, version	unique identification of the location
name, description	parking location name and optional further descriptive text
type	type of hierarchy element; for now, this will always be "place" (with the exception of the hierarchy root element which is of type "campus")
layer	layer in the place hierarchy; for now the root element is on level 0, and all other locations are direct children of the root (and hence are on layer 1); might change in the future, e.g. when describing sub-areas offering electric vehicle charging
parentId	identification of the parent (root node in our case)
indicativePointLocation	coordinates of a point on the map to represent the location of the car park / parking location; please note that due to the underlying standard, coordinates are specified in the order LON, LAT
placeStreetAddress	postal address
paymentMethods	payment timing indicator; will be "payOnEntry" for the traditional pay-on-arrival use case
rightSpecifications	list of one or more right specifications detailing applicable tariffs and potential eligibility criteria
contacts	operator contact information
operatorDefinedReference	(optional) operator-defined cost code for this location
responsibilityRoleAssignments	Indicates relevant responsibilities at/for this location. At a minimum, it is specified who operates this location. See also /contacts endpoint under "Shared Use Cases"

1.1.2 Initial Load (Right Specifications)

Using the reference from the previous /places response ("rightSpecifications"), you can now retrieve further details of the rules that apply when parking at this location: applicable tariffs, potential eligibility constraints, valid periods, etc..

1.1.2.1 Read a specific Right Specification

```
GET /v4/parking/rights/specs/RIGHTSPEC1?expand=all
```


```
{
    "id": "RIGHTSPEC1",
    "version": 1,
    "description": [{"language":"en","string": "Market Place Hours"}],
    "hierarchyElements": [
        {
            "id": "CARPARK1",
            "version": 1
        }

    ],
    "type": "oneTimeUseParking",
    "rateEligibility": [
        {
            "id": "874b4c04-102d-4a8e-ba01-55ea304e3666",
            "version": 1,
            "rateTable": {
                "id": "TARIFF1",
                "version": 1
            }
        }
    ],
    "validity": {
        "validityStatus": "definedByValidityTimeSpec",
        "validityTimeSpecification": {
            "overallStartTime": "2025-07-03T00:00:00Z",
            "validPeriods": [
                {
                    "periodName": [{"language":"en","string":"Mon-Sun"}],
                    "recurringDayWeekMonthPeriod": [
                        {
                            "applicableDay": [
                                "monday",
                                "tuesday",
                                "wednesday",
                                "thursday",
                                "friday",
                                "saturday",
                                "sunday"
                            ]
                        }
                    ],
                }
            ]
        }
    }
}
```

Notes

Field(s)	Notes
id, version	unique identification of the right specification
hierarchyElements	list of locations to which this right specification applies (i.e. multiple locations can share the same right specification)
type	the type of right, for now typically "oneTimeUseParking"
rateEligibility	points to one or more applicable tariffs and potential eligibility criteria
validity. validityTimeSpecification. overallStartTime	defines the effective date of the right specification
validity. validityTimeSpecification. overallEndTime	defines the termination date of the right specification; can be null to indicate open-ended validity (end be set when a successor right specification is added)
validity. validityTimeSpecification. validPeriods	defines time periods within which a right specification is valid

1.1.3 Initial Load (Tariffs)

Using the reference from the previous /rights/specs response ("rateEligibility[].rateTable"), you can now retrieve further details of the applicable tariff.

1.1.3.1 Read a specific Tariff

```
GET /v4/parking/rates/TARIFF1?expand=all
```


```
{
    "id": "TARIFF1",
    "version": 1,
    "rateTableName": [{"language":"en","string": "Standard Tariff"}],
    "rateLineCollections": [
        {
            "applicableCurrency": "GBP",
            "collectionSequence": 0,
            "maxTime": "PT2H",
            "minTime": "PT30M",
            "rateLines": [
                {
                    "sequence": 0,
                    "description": [
                        {
                            "language": "en",
                            "string": "up to 30 minutes"
                        }
                    ],
                    "rateLineType": "incrementingRate",
                    "value": 0.5,
                    "incrementPeriod": "PT30M",
                    "usageCondition": "once"
                },
                {
                    "sequence": 1,
                    "description": [
                        {
                            "language": "en",
                            "string": "up to 1 hour"
                        }
                    ],
                    "rateLineType": "incrementingRate",
                    "value": 0.5,
                    "incrementPeriod": "PT30M",
                    "usageCondition": "once"
                },
                {
                    "sequence": 2,
                    "description": [
                        {
                            "language": "en",
                            "string": "up to 2 hours"
                        }
                    ],
                    "rateLineType": "incrementingRate",
                    "value": 1.0,
                    "incrementPeriod": "PT1H",
                    "usageCondition": "once"
```

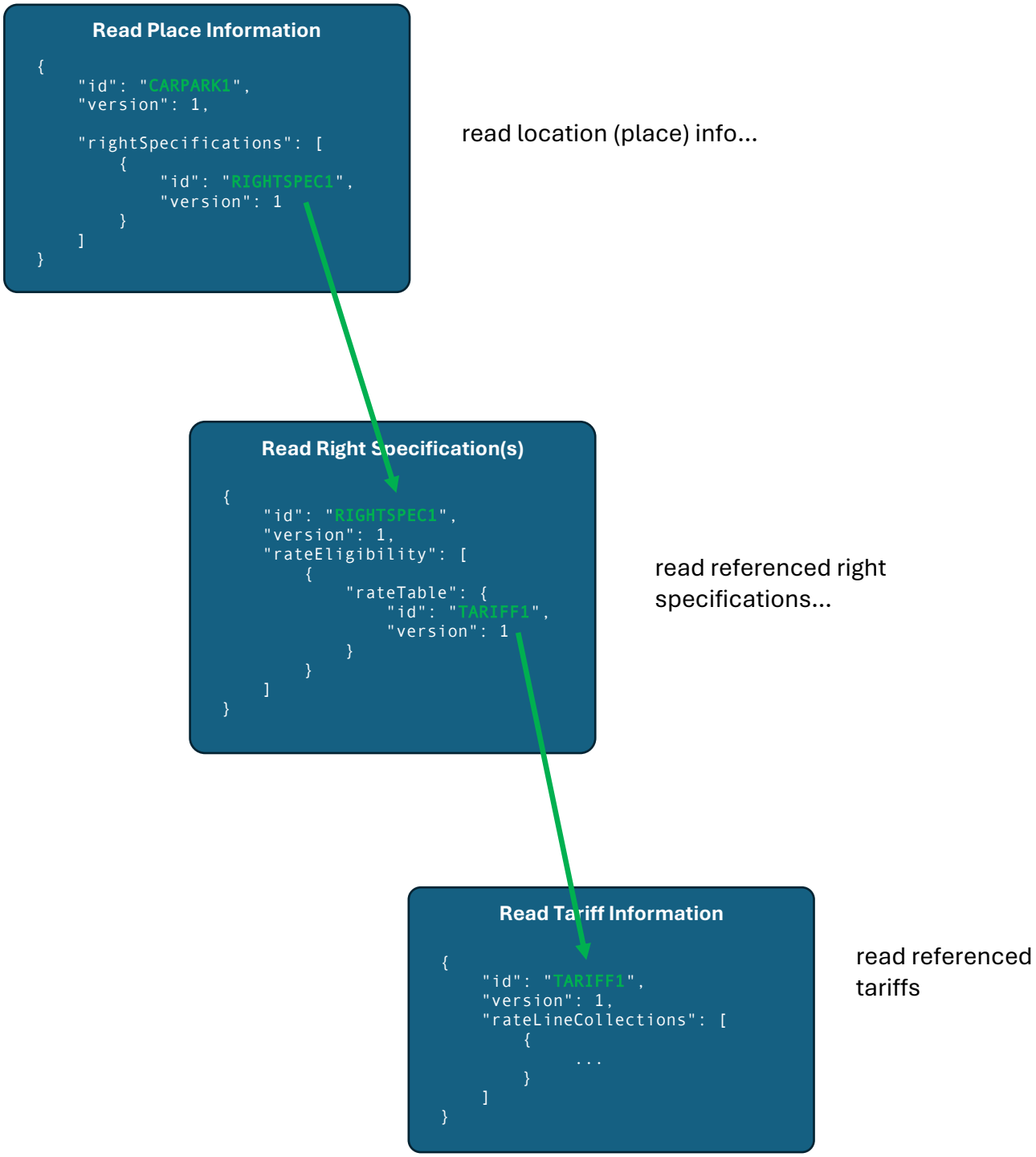
```
                }
            ],
            "relativeTimes": true
        }
    ],
    "availability": "public",
    "rateResponsibleParty": "COUNCIL1"
}
```

Notes

Field(s)	Notes
id, version	unique identification of the tariff (APDS: rate table)
Name	user-friendly name of the tariff
rateLineCollections[]	overall tariff settings, in the example: maximum duration of stay (2 hours) and minimum time charged (30 minutes)
rateLineCollections[] .rateLines	individual tariff steps, to be ordered by "sequence", individual step duration specified by "incrementPeriod", individual step value specified by "value" (i.e. the individual step values must be added together)

1.1.4 Inventory Information: Summary

As you have seen, inventory data collection is a three-step process.



Based on these interconnected datasets, your tariff engine must select and calculate the correct amount due.

1.1.5 APDS Refresher: Eligibility

There might be situations where certain constraints apply. These are expressed through eligibility criteria and associated (APDS) qualifications.

A defined set of eligibility criteria can then be used to make a specific tariff conditional

In the above diagram, the referenced Rate Table is only available to drivers/vehicle meeting the associated eligibility criteria. A typical example would be the requirement to be a member of a specific group of users to be eligible for a certain tariff (e.g. blue badge tariff).

1.1.5.1 Emission-based Charging

Another example are tariffs that are dependent on the emission of vehicles. The following example shows a right specification that is bound to emission-based eligibility criteria: the vehicle manufacture date must not be before the year 2000, and the vehicle must emit less than 100 gram of CO2 per kilometer. The referenced tariff is only applicable under these conditions.


```
{
    "id": "RIGHTWITHEMISSIONCRITERIA",
    "version": 1,
    "description": [
        {
            "language": "en",
            "string": "right specification with emission related criteria"
        }
    ],
    "hierarchyElements": [
        {
            "id": "CARPARK2",
            "version": 1
        }
    ],
    "type": "oneTimeUseParking",
    "rateEligibility": [
        {
            "id": "ELI0001",
            "version": 1,
            "rateTable": {
                "id": "REDUCEDTARIFF1",
                "version": 1
            },
            "eligibility": {
                "qualifications": [
                    {
                        "emissions": {
                            "vehicleCharacteristics": [
                                {
                                    "unit": "year",
                                    "boundaries": [
                                        {
                                            "comparisonOperator":
"greaterThanOrEqualTo",
                                            "value": 2000
                                        }
                                    ],
                                    "class": "manufactureDate"
                                },
                                {
                                    "unit": "gkm",
                                    "boundaries": [
                                        {
                                            "comparisonOperator": "lessThan",
                                            "value": 100
                                        }
                                    ],
                                    "class": "emissionValue"
                                }
                            ]
```

```
                        }
                    }
                ]
            }
        }
    ]
}
```

Unfortunately, there is no standard or best practices for the definition of vehicle emission criteria. Different Parking Operators chose different approaches and classifications. The APDS data model has been adapted to accommodate for this heterogeneous situation.

The APDS **Emissions** class makes it possible to specify one or more (emissions-related) vehicle characteristics in order to create a kind of profile for a specific vehicle class. The **vehicle characteristics** are expressed as a list of **value ranges**.

A **Value Range** consists of

- a **Class** indicating the type of value range
- a **Unit** of measure in which this class of range is expressed
- a lower and/or upper limit - **Boundary** - within which matching vehicles must lie

Currently, the following **Value Range Classes** are defined:

- batteryCapacity
- engineCapacityPetrol
- engineCapacityDiesel
- manufactureDate
- emissionValue

Defined **Units of Measure** are:

- cubicCentimetres
- cubicInches
- year
- gkm (gram per kilometer)
- gpm (gram per mile)
- kWh (kilowatt hours)
- eurolevel

With this definition in mind, go back and take another look at the example above.

1.1.5.2 Diesel/No Diesel Differentiation

Another vehicle classification is based on the type of fuel is uses. For this purpose, the APDS **Qualification** class offers the **propulsionEnergyType** attribute. The following example shows two **Eligibility** definitions, one for Diesel-powered vehicles, and one for non-Diesel vehicles:

Diesel

```
{
	"eligibilityName": [{ "language": "en", "string": "Diesel Vehicle"}],
	"qualifications": [
		{
			"propulsionEnergyType": ["diesel"]
		}
	]
}
```

Non-Diesel

```
{
	"eligibilityName": [{ "language": "en", "string": "Non-Diesel Vehicle"}],
	"qualifications": [
		{
		   "propulsionEnergyType": [
		      "battery","petrol","lpg","hydrogen"
		   ]
		}
	]
}
```

1.1.5.3 General Convention (Energy Source Types)

The APDS model defines energy source types that are not used within the NPP context. The NPP convention for energy source type references is this:

- For petrol-powered vehicles, the NPP only differentiates between ***petrol*** and ***petrolBatteryHybrid***. Specialisations concerning octane and lead are not used, they all fall under the ***petrol*** categorisation.
- The same applies to the Diesel energy source. The NPP only differentiates between ***diesel*** and ***dieselBatteryHybrid***. Hence, ***biodiesel*** counts as ***diesel***.
- The NPP only uses ***lpg***, and ***liquidGas*** counts as ***lpg***.
- Fully electric vehicles are categorised as ***battery***.
- The ***unknown*** enum is used in all situations where the energy source is either unknown/uncategorised or cannot be otherwise categorised (using the set of APDS-defined enumeration values).
- There is currently no NPP use case where ***all*** would be required.
- The ***other*** category is not used, because it creates contextual dependencies.

1.2 Create Session (Pay on Arrival / Check in, check out)

A service provider notifies the NPP about a newly-issued parking right and a parking session using this right. A customer has purchased 1 hour of parking at *CARPARK1*.

When the driver purchases the parking right (APDS term: assigned right), the Service Provider backend will have to classify the intended transaction:

- Location: what is the NPP place requested? → CARPARK1
- Which is/are the applicable right specification(s)? → CARPARK1-RIGHT1
- What is the tariff to apply? → TARIFF1

The determined right specification (CARPARK1-RIGHT1) must be referenced in the assigned right communicated to the NPP. The amount to be collected must be calculated based on the referenced tariff (TARIFF1).

Step 1: post new parking right

In a first step, the service provider posts the details of the newly-issued parking right to the NPP. This also includes information about collected payments.

```
POST /v4/parking/rights/assigned

{
	“id”: “NEW-PARKING-RIGHT-1”,
	“version”: 1,
	“rightHolder”: {
		“credentials”: [
			{
				“type”: “licensePlate”,
				“identifier”: {
					“id”: “TST001”,
					“className”: “UKNumberPlate”
				},
				“issuer”: [
					{
						“language”: “en”,
						“string”: DVLA”
					}
				]
			}
		]
	},
	“assignedRightIssuer”: {
		“id”: “PROVIDER1”,
		“version”: 1
	},
	“rightSpecification”: {
		“id”: “CARPARK1-RIGHT1”,
		“version”: 1
	},
	“issuanceTime”: “2025-05-20T10:02:00Z”,
	“expiry”: “2025-05-20T11:02:00Z”,
	“issueMethod”: “electronic”,
	“monetaryValue”: {
		“taxIncluded”: true,
		“value”: {
			“currencyType”: “GBP”,
			“currencyValue”: 2
		}
```

```
        },
        “payments”: [
                {
                        “id”: “23d45a6d-947b-4932-ac8b-0d97ec3a328d”,
                        “version”: 1,
                        “dateCollected”: “2025-05-20T10:01:00Z”,
                        “startPeriodCovered”: “2025-05-20T10:02:00Z”,
                        “endPeriodCovered”: “2025-05-20T11:02:00Z”,
                        “paymentLines”: [
                                {
                                        “id”: “403511a6-3e79-40ab-851b-da45b4ea8b34”,
                                        “version”: 1,
                                        “value”: {
                                                “currencyType”: “GBP”,
                                                “currencyValue”: 2
                                        },
                                        “idCode”: “PARKING-FEE-VAT20”,
                                        “identifierId”: “PROVIDER1-PAYMENT-REF-0001”,
                                        “paymentType”: “payment”
                                }
                        ]
                }
        ]
}
```

The NPP responds with:

```
HTTP/2 201

{
        “code”: 201,
        “status”: “CREATED”,
        “message”: “right with id NEW-PARKING-RIGHT-1 created”
}
```

Step 2: post new parking session

In a second step, the session record (making use of the previously-posted assigned right) is sent to the NPP.

```
POST /v4/parking/sessions

{
	“id”: “PROVIDER-GENERATED-SESSION-ID-1”,
	“version”: 1,
	“actualStart”: “2025-05-20T10:02:00Z”,
	“actualEnd”: “2025-05-20T11:02:00Z”,
	“initiator”: {
		“id”: “PROVIDER1”,
		“version”: 1
	},
	“identifiedCredentials”: [
		{
			“type”: “licensePlate”,
			“identifier”: {
				“id”: “TST001”,
				“className”: “UKNumberPlate”
			},
			“issuer”: [
				{
					“language”: “en”,
					“string”: DVLA”
				}
			]
		}
	],
	“hierarchyElement”: {
		“id”: “CARPARK1”,
		“version”: 1
	},
	“segments”: [
		{
			“id”: “PROVIDER-GENERATED-SESSION-ID-1-SEGMENT-1”,
			“version”: 1,
			“actualStart”: “2025-05-20T10:02:00Z”,
			“actualEnd”: “2025-05-20T11:02:00Z”,
			“assignedRight”: {
				“id”: “NEW-PARKING-RIGHT-1”,
				“version”: 1
			},
			“validationType”: [
				“licensePlate”
			]
		}
	],
	“identifiedVehicle”: {
		“make”: “AUDI”,
		“color”: “BLUE”,
		“country”: “GB”
	}
}

HTTP/2 201

{
	“code”: 201,
	“status”: “CREATED”,
	“message”: “session with id PROVIDER-GENERATED-SESSION-ID-1 created”
}
```

This new session currently has exactly one segment (see the APDS specification for more details on the concept of sessions and segments). In this case, the session's start and end timestamp correspond to the segment's boundaries. This will change in the next example ("Extend Session").

1.3 Extend Session

An existing parking session is extended, in our example by 1 hour. In APDS terms, this means that a second (or third ...) segment is added to the session representing the extension.

Step 1: post extension parking right

The new segment (standing for the extension) requires a new parking right to be issued by the service provider. This parking right is sent to the NPP.

```
POST /v4/parking/rights/assigned
```

```
{
	"id": "NEW-PARKING-RIGHT-2",
	"version": 1,
	"rightHolder": {
		"credentials": [
			{
				"type": "licensePlate",
				"identifier": {
					"id": "TST001",
					"className": "UKNumberPlate"
				},
				"issuer": [
					{
						"language": "en",
						"string": DVLA"
					}
				]
			}
		]
	},
	"assignedRightIssuer": {
		"id": "PROVIDER1",
		"version": 1
	},
	"rightSpecification": {
		"id": "CARPARK1-RIGHT1",
		"version": 1
	},
	"issuanceTime": "2025-05-20T11:02:00Z",
	"expiry": "2025-05-20T12:02:00Z",
	"issueMethod": "electronic",
	"monetaryValue": {
		"taxIncluded": true,
		"value": {
			"currencyType": "GBP",
			"currencyValue": 2
		}
	},
	"payments": [
		{
			"id": "bc092d9f-c50e-4f09-b306-29a7fcba361d",
			"version": 1,
			"dateCollected": "2025-05-20T11:01:00Z",
			"startPeriodCovered": "2025-05-20T11:02:00Z",
```

```
                    “endPeriodCovered”: “2025-05-20T12:02:00Z”,
                    “paymentLines”: [
                        {
                            “id”: “4318a313-151f-42b6-bf8d-e5578755c141”,
                            “version”: 1,
                            “value”: {
                                “currencyType”: “GBP”,
                                “currencyValue”: 2
                            },
                            “idCode”: “PARKING-FEE-VAT20”,
                            “identifierId”: “PROVIDER1-PAYMENT-REF-0002”,
                            “paymentType”: “payment”
                        }
                    ]
                }
        ]
}
```

The NPP responds with:

```
HTTP/2 201

{
        “code”: 201,
        “status”: “CREATED”,
        “message”: “right with id NEW-PARKING-RIGHT-2 created”
}
```

Step 2: send session update to reflect the extension

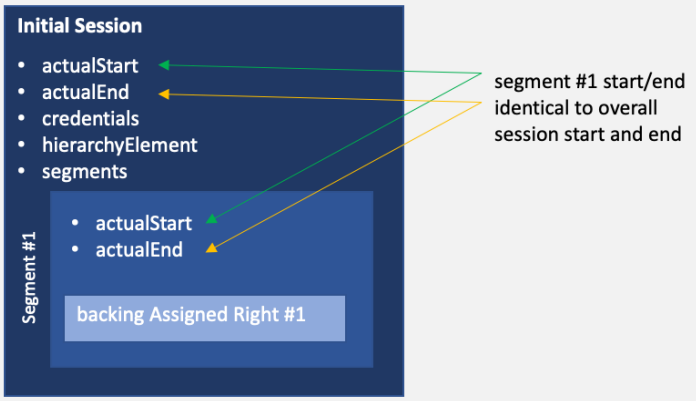
In a next step the updated session record (including the extension segment) is sent to the NPP.

```
PUT /v4/parking/sessions/ PROVIDER-GENERATED-SESSION-ID-1

{
        “id”: “PROVIDER-GENERATED-SESSION-ID-1”,
        “version”: 1,
        “actualStart”: “2025-05-20T10:02:00Z”,
        “actualEnd”: “2025-05-20T12:02:00Z”,
        “initiator”: {
                “id”: “PROVIDER1”,
                “version”: 1
        },
        “identifiedCredentials”: [
                {
                        “type”: “licensePlate”,
                        “identifier”: {
                                “id”: “TST001”,
                                “className”: “UKNumberPlate”
                        },
                        “issuer”: [
                                {
                                        “language”: “en”,
                                        “string”: DVLA”
                                }
                        ]
                }
        ],
        “hierarchyElement”: {
                “id”: “CARPARK1”,
                “version”: 1
        },
        “segments”: [
                {
                        “id”: “PROVIDER-GENERATED-SESSION-ID-1-SEGMENT-1”,
```

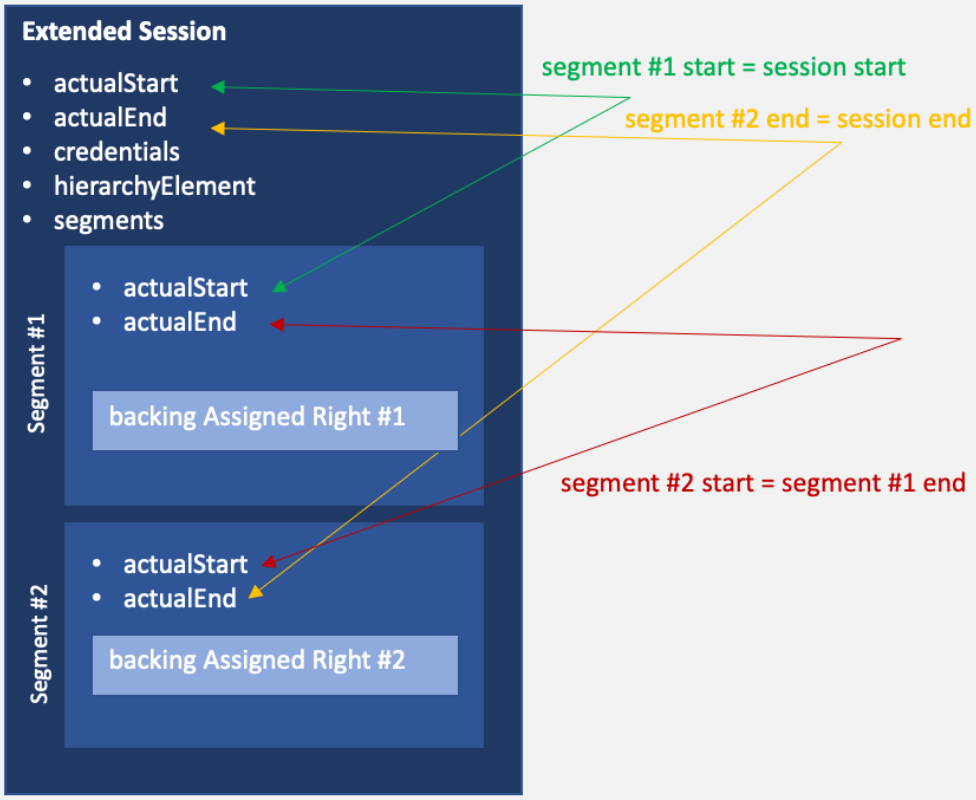
```
                        “version”: 1,
                        “actualStart”: “2025-05-20T10:02:00Z”,
                        “actualEnd”: “2025-05-20T11:02:00Z”,
                        “assignedRight”: {
                                “id”: “NEW-PARKING-RIGHT-1”,
                                “version”: 1
                        },
                        “validationType”: [
                                “licensePlate”
                        ]
                },
                {
                        “id”: “PROVIDER-GENERATED-SESSION-ID-1-SEGMENT-2”,
                        “version”: 1,
                        “actualStart”: “2025-05-20T11:02:00Z”,
                        “actualEnd”: “2025-05-20T12:02:00Z”,
                        “assignedRight”: {
                                “id”: “NEW-PARKING-RIGHT-2”,
                                “version”: 1
                        },
                        “validationType”: [
                                “licensePlate”
                        ]
                }

        ],
        “identifiedVehicle”: {
                “make”: “AUDI”,
                “color”: “BLUE”,
                “country”: “GB”
        }
}

HTTP/2 200

{
        “code”: 200,
        “status”: “OK”,
        “message”: “session with id PROVIDER-GENERATED-SESSION-ID-1 updated”
}
```


Please take note of the updated contents (marked in bold). The session has now two segments: one representing the first hour, one representing the extension. The session start now matches the first segment’s start, and the session end corresponds to the second segment’s end time. The two segments are adjacent to each other. Each segment is backed by a corresponding assigned right. A session’s timespan must always be completely covered by its segments. Gaps are not allowed.

APDS Refresher: Sessions and Segments



The **AssignedRight** stands for the (typically paid-for) right to park in one or more locations.
A **Session** is the act of actively making use of parking rights.
A **Session** comprises one (minimum) or more **Segments**.
Each **Segment** points to its backing **AssignedRight**.



1.4 Adjacent Sessions

As a precaution, the service provider checks whether the customer has already purchased a session elsewhere.

{ will be provided in the next revision of this document }

1.5 Transaction

Use case: a service provider posts a transaction for consideration in the monthly reconciliation run.

Whenever there is a transaction that is relevant for reconciliation, a service provider creates a corresponding transaction record (class name in API specification: *ReconciliationTransaction*) and sends it to the NPP in near realtime. The NPP collects those transactions in order to use them when compiling the consolidated reconciliation report after the current reconciliation period ended. A transaction can have one of three possible types:

- **payment** – a purchase was made, and a payment was collected
- **refund** – a previously made purchase has been refunded
- **cancellation** – use this type in case a previous transaction had to be reverted, but not in form of a regular refund (for the time being, it is unlikely that this transaction type will be needed)

The transaction must be sent shortly after the underlying assigned right has been sent to the NPP. The following table lists the data elements expected by the NPP to be found in a transaction:

Date Element Name	Description
Provider ID	service provider identifier
Operator ID	parking operator identifier
Transaction ID	unique identifier for the transaction
Transaction Type	type of transaction (one of: payment, refund, cancellation)
Transaction Time	timestamp (ISO 8601/RFC 3339)
Transaction Reference	in case of a refund/cancellation: id of the original transaction (if available)
Location ID	location for which a fee was charged/refunded
VAT Rate	applied VAT rate (percentage, currently 0 or 20)
Total Amount	total transaction amount (including VAT)
Net Amount	net transaction amount (excluding VAT)
VAT Amount	VAT amount (if any)
Commission Rate	applied commission rate (percentage, e.g. 2.5)
Commission VAT Rate	VAT rate applied to commission (percentage, currently 20)
Commission Total Amount	total amount charged to operator (including VAT)
Commission Net Amount	net amount of commission charged to operator (excluding VAT)
Commission VAT Amount	VAT amount included in commission
Remittance Total	amount paid to the operator

1.5.1 Example Payment Transaction

The following example demonstrates how to send a new (payment) transaction to the NPP:

```
POST /v4/parking/reconciliation/transactions

{
	"providerId": "PROVIDER1",
	"operatorId": "OPERATOR27",
	"transactionId": "UNIQUEPROVIDERGENERATEDTRANSACTIONID1",
	"transactionType": "payment",
	"transactionTime": "2025-07-17T17:23:02Z",
	"transactionReference": null,
	"locationId": "CARPARK1",
	"vatRate": 20,
	"totalAmount": 6,
	"netAmount": 5,
	"vatAmount": 1,
	"commissionRate": 2.5,
	"commissionVatRate": 20,
	"commissionTotalAmount": 0.18,
	"commissionNetAmount": 0.15,
	"commissionVatAmount": 0.03,
	"remittanceTotal": 5.82
}
```

1.5.2 Example Refund Transaction

The following example demonstrates how to send a new (refund) transaction to the NPP:

```
{
	"providerId": "PROVIDER1",
	"operatorId": "OPERATOR27",
	"transactionId": "UNIQUEPROVIDERGENERATEDTRANSACTIONID2",
	"transactionType": "refund",
	"transactionTime": "2025-07-18T09:21:12Z",
	"transactionReference": "UNIQUEPROVIDERGENERATEDTRANSACTIONID1",
	"locationId": "CARPARK1",
	"vatRate": 20,
	"totalAmount": -6,
	"netAmount": -5,
	"vatAmount": -1,
	"commissionRate": 0,
	"commissionVatRate": 20,
	"commissionTotalAmount": 0,
	"commissionNetAmount": 0,
	"commissionVatAmount": 0,
	"remittanceTotal": -6
}
```

> Note: for now, it is presumed that payment transactions are eligible for commission, even if they are refunded later on. The further assumption is that refund transactions are not eligible for commission. This is a policy decision yet to be confirmed/adjusted by the NPP board, so the convention might be subject to change in the near future.
>
> You will also have noticed that – by convention - refund amounts are expressed as negative numbers.

1.5.3 Check transaction on NPP

A service provider can also check if a previously transmitted transaction record resides on the NPP:

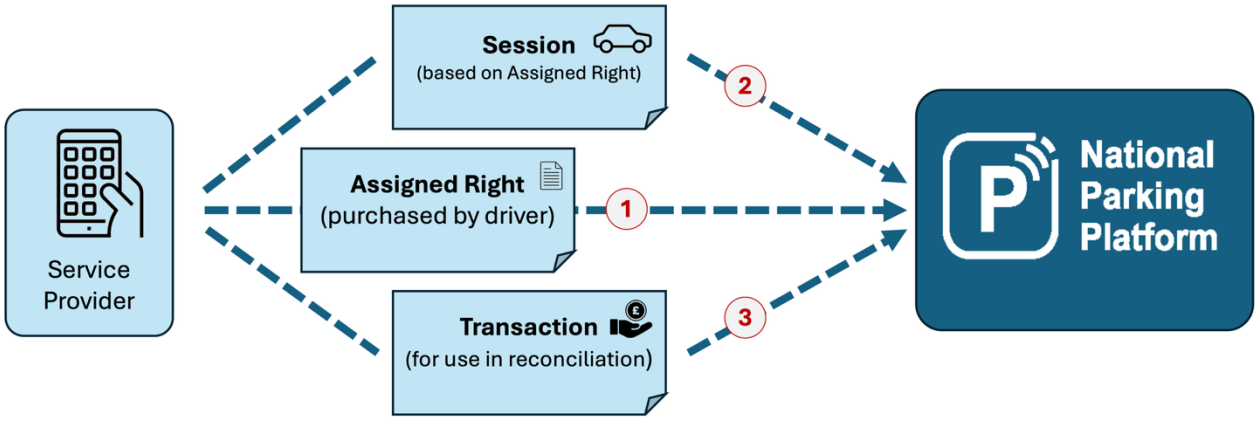
```
GET /v4/parking/reconciliation/transactions/UNIQUEPROVIDERGENERATEDTRANSACTIONID1
```

This call will return a copy of the transaction.

1.5.4 Recap: Pay-on-Arrival Transaction Components

The following diagram illustrates the various components of a typical pay-on-arrival transaction:

1. The driver purchases a (one-time-use) parking right. Its APDS representation (assigned right) is sent to the NPP.
2. As it is a pay-on-arrival use case with immediate use of the obtained parking right, the matching session record is sent to the NPP.
3. The corresponding financial transaction with all relevant details is sent to the NPP.



1.6 Close-out Message

Use case: a service provider posts the close-out message for the current reconciliation period. It defines which previously submitted transactions (payments and refunds) shall be included in the consolidated reconciliation report.

The close-out message consists of

- some header data (identifiers and reconciliation period information),
- a list with the ids of all (previously sent) transactions that shall be included in the report
- optional notes from the provider that shall be included in the report for the operator

The following table shows the data elements expected by the NPP to be found in the close-out message:

Date Element Name	Description
Provider ID	service provider identifier
Operator ID	parking operator identifier
Submittal ID	unique identifier of this close-out submittal
Submittal Status	status of the submittal (one of: confirmed, revoked)
Period Start Time	start time of reconciliation period
Period End Time	end time of reconciliation period
Period Name	name of the reconciliation period (e.g. "November 2025")
Transaction IDs	list of all transaction ids to be included in the report
Provider Notes	optional notes/comments from the provider to be made available to the operator
Provider Notes Format	text format of the notes text (one of: plain, html, markdown, default: plain)

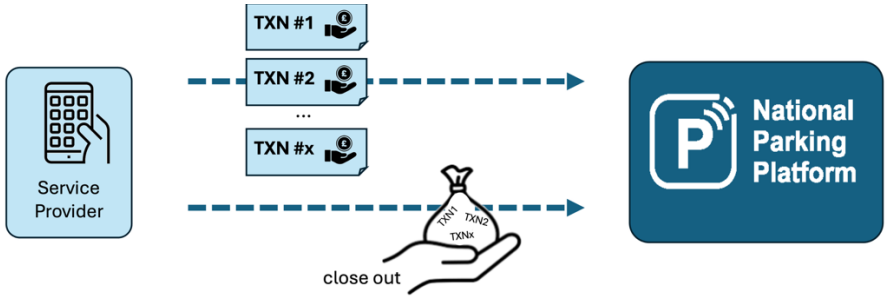
1.6.1 Example Close-out Message

The following example shows how to compile and send a close-out message:

```
POST /v4/parking/reconciliation/submittals

{
	"providerId": "PROVIDER1",
	"operatorId": "OPERATOR27",
	"submittalId": "PROVIDERGENERATEDSUBMITTALID1",
	"periodStartTime": "2025-06-01T00:00:00.000Z",
	"periodEndTime": "2025-06-30T23:59:59.999Z",
	"periodName": "June 2025",
	"transactionIds": [
		"UNIQUEPROVIDERGENERATEDTRANSACTIONID1",
		"UNIQUEPROVIDERGENERATEDTRANSACTIONID2",
		// more transaction ids
	],
	"providerNotesText": "Important notes to the operator: ...",
	"providerNotesFormat": "plain"
}
```

The NPP expects to receive the close-out message from every provider for every operator. Once all close-out messages for a particular operator have been received, the NPP starts compiling the consolidated reconciliation report. The report will then be made available to the operator. An example reconciliation report is presented in section 3.6.

The following diagram illustrates the flow of individual transactions over the course of the reconciliation period with the close-out message then concluding it:



1.6.2 Rejection of close-out messages

When the NPP receives a close-out message, it is validated. If the submitted message doesn't pass the inspection, it is rejected with a corresponding error message. The following example shows a situation where a close-out message includes a transaction id that is unknown to the NPP, the submittal is therefore rejected.

```
POST /v4/parking/reconciliation/submittals

{
    "submittalId": "UNIQUE_SUBMITTAL_ID",
    "providerId": "NPPDEMO",
    "operatorId": "TESTCOUNCIL",
    "periodStartTime": "2025-06-01T00:00:00Z",
    "periodEndTime": "2025-06-30T23:59:59Z",
    "periodName": "June 2025",
    "transactionIds": [ "KNOWN_TRANSACTION_ID", "UNKNOWN_TRANSACTION_ID"],
    "providerNotesText": "Please note...",
    "providerNotesFormat": "plain"
}
```

The NPP rejects the submittal:

```
HTTP 409 (CONFLICT)

{
    "code": 409,
    "status": "Conflict",
    "message": "submittal UNIQUE_SUBMITTAL_ID references unknown transactions",
    "ids": [ "UNKNOWN_ID_UNKNOWN_TRANSACTION_ID"]
}
```

A close-out message is declined if one of the following errors is detected:

- Reference to unknown provider
- Mismatch between provider reference in payload and provider tied to API user
- Unknown operator reference
- Key constraint issue (duplicate submittal id)
- Close-out message referring to transactions that were not previously sent
- Various period start/end checks for plausibility, gaps and overlap

1.6.3 Revocation of a previous close-out message

In rare (!) instances, it might become necessary to revoke an already submitted close-out message and send a new, adjusted close-out message. To do this, a service provider will have to send a corresponding revocation request to the NPP:

```
DELETE /v4/parking/reconciliation/submittals/PROVIDERGENERATEDSUBMITTALID1
```

This command will revoke the close-out message. The NPP will not delete it, but set its status from "confirmed" to "revoked". Once this has been confirmed, the service provider can send a revised close-out message as shown in paragraph 1.6.1..

2. Connected Supplier Use Cases

In this first version of the document, enforcement system providers are the only category of connected suppliers.

2.1 Check Parking Rights (Pull Mode)

The enforcement system actively fetches parking right information. The search results can be narrowed down using appropriate filter criteria. For enforcement purposes, this will typically be the following query parameters:

- place - the location id of the parking location currently being monitored
- credential_id - the VRM of the vehicle being checked
- end_after - the earliest expiration timestamp of the rights to be returned

The NPP will return a paginated list of all matching assigned right records. Below is a sample request and response:

```
GET /v4/parking/rights/assigned
      ?place=CARPARK1&credential_id=TST001&end_after=1750166111
```

(query parameters shown in the second line for better readability)

```
{
      "meta": {
            "referenceInstant": 1750166111,
            "offset": 0,
            "pageSize": 200,
            "total": 1
      },
      "data": [

            {
                  “id”: “d0a16d11-9804-4c9e-bba2-99a8a6d95dfb”,
                  “version”: 1,
                  “rightHolder”: {
                        “credentials”: [
                              {
                                    “type”: “licensePlate”,
                                    “identifier”: {
                                          “id”: “TST001”, // the right holders VRM
                                          “className”: “UKNumberPlate”
                                    },
                                    “issuer”: [
                                          {
                                                “language”: “en”,
                                                “string”: DVLA”
                                          }
                                    ]
                              }
                        ]
                  },
                  “assignedRightIssuer”: {
                        “id”: “PROVIDER1”,          // the id of the service provider
                        “version”: 1
                  },
                  “rightSpecification”: {
```

```
                        “id”: “CARPARK1-RIGHT1”,
                        “version”: 1,
                        "hierarchyElements": [
                                {
                                        "id": "7591001",
                                        "version": 1,
                                        "name": [
                                                {
                                                        "language": "en",
                                                        "string": "Lord Street"
                                                }
                                        ]
                                },
                                {
                                        "id": "7582442",
                                        "version": 1,
                                        "name": [
                                                {
                                                        "language": "en",
                                                        "string": "Victoria Street"
                                                }
                                        ]
                                }
                        ],
                },
                “issuanceTime”: “2025-05-20T10:02:00Z”, // the time the right became valid
                “expiry”: “2025-05-20T11:02:00Z”, // the time the right expires
                “issueMethod”: “electronic”,
                “monetaryValue”: {
                        “taxIncluded”: true,
                        “value”: {
                                “currencyType”: “GBP”,
                                “currencyValue”: 2
                        }
                },
                “payments”: [
                        {
                                “id”: “23d45a6d-947b-4932-ac8b-0d97ec3a328d”,
                                “version”: 1,
                                “dateCollected”: “2025-05-20T10:01:00Z”,
                                “startPeriodCovered”: “2025-05-20T10:02:00Z”,
                                “endPeriodCovered”: “2025-05-20T11:02:00Z”,
                                “paymentLines”: [
                                        {
                                                “id”:“403511a6-3e79-40ab-851b-da45b4ea8b34”,
                                                “version”: 1,
                                                “value”: {
                                                        “currencyType”: “GBP”,
                                                        “currencyValue”: 2
                                                },
                                                “idCode”: “PARKING-FEE-VAT20”,
                                                “identifierId”:“PROVIDER1-PAYMENT-REF-0001”,
                                                “paymentType”: “payment”
                                        }
                                ]
                        }
                ]
        }
    ]
}
```

The *end_after* query parameter can also be used to for instance query recently expired assigned rights. Timestamps in query parameters are always specified as epoch seconds.

Important note: per the specification, the referenced right specification(s) only provide a reference to the applicable hierarchy elements (parking locations), i.e. id and version. As a convenience feature, it is possible to activate the additional provision of the location name, helping the caller to avoid another call to the inventory API. Please indicate this to the NPP team in case you want this option activated.

2.2 Parking Right (Push Notification)

The enforcement system (subscribed to the NPP push service) receives a notification of a new or updated parking right. The contents is the same as in the previous example, but the information will be pro-actively pushed to the system's webhook endpoint. The custom http request header named "X-Event-Type" will indicate the nature of the data event.

```
POST {provider-specified webhook endpoint}
X-Event-Type: AssignedRightCreated
```


```
{
	"id": "d0a16d11-9804-4c9e-bba2-99a8a6d95dfb",
	"version": 1,
	"rightHolder": {
		"credentials": [
			{
				"type": "licensePlate",
				"identifier": {
					"id": "TST001",
					"className": "UKNumberPlate"
				},
				"issuer": [
					{
						"language": "en",
						"string": DVLA"
					}
				]
			}
		]
	},
	"assignedRightIssuer": {
		"id": "PROVIDER1",
		"version": 1
	},
	"rightSpecification": {
		"id": "CARPARK1-RIGHT1",
		"version": 1,
		"hierarchyElements": [
			{
				"id": "7591001",
				"version": 1,
				"name": [
					{
						"language": "en",
						"string": "Lord Street"
					}
				]
			},
			{
				"id": "7582442",
				"version": 1,
				"name": [
					{
						"language": "en",
						"string": "Victoria Street"
					}
				]
			}
		],
	},
	"issuanceTime": "2025-05-20T10:02:00Z",
	"expiry": "2025-05-20T11:02:00Z",
	"issueMethod": "electronic",
```

```
        “monetaryValue”: {
                “taxIncluded”: true,
                “value”: {
                        “currencyType”: “GBP”,
                        “currencyValue”: 2
                }
        },
        “payments”: [
                {
                        “id”: “23d45a6d-947b-4932-ac8b-0d97ec3a328d”,
                        “version”: 1,
                        “dateCollected”: “2025-05-20T10:01:00Z”,
                        “startPeriodCovered”: “2025-05-20T10:02:00Z”,
                        “endPeriodCovered”: “2025-05-20T11:02:00Z”,
                        “paymentLines”: [
                                {
                                        “id”: “403511a6-3e79-40ab-851b-da45b4ea8b34”,
                                        “version”: 1,
                                        “value”: {
                                                “currencyType”: “GBP”,
                                                “currencyValue”: 2
                                        },
                                        “idCode”: “PARKING-FEE-VAT20”,
                                        “identifierId”: “PROVIDER1-PAYMENT-REF-0001”,
                                        “paymentType”: “payment”
                                }
                        ]
                }
        ]
}
```

2.3 Sessions (Pull Mode)

The enforcement system actively fetches session information. The search results can be narrowed down using appropriate filter criteria. For enforcement purposes, this will typically be the following query parameters:

- place - the location id of the parking location currently being monitored
- credential_id - the VRM of the vehicle being checked
- end_after - the earliest expiration timestamp of the rights to be returned

The NPP will return a paginated list of all matching session records. Below is a sample request and response:

```
GET /v4/parking/sessions?place=CARPARK1&credential_id=VRM123&end_after=1750166111
```

```
{
	"meta": {
		"referenceInstant": 1750166111,
		"offset": 0,
		"pageSize": 200,
		"total": 1
	},
	"data": [
		{
			"id": "SESSION1",
			"version": 1,
			// …details of session 1
		}

	]
}
```

Depending on the filters provided, the result may contain multiple sessions. See the next section for an example of session details. The *end_after* query parameter can also be used to for instance query recently expired sessions. Timestamps in query parameters are always specified as epoch seconds.

2.4 Sessions (Push Notification)

The enforcement system (subscribed to the NPP push service) receives a notification of a new or updated session. Data will be pro-actively pushed to the system's webhook endpoint. The custom http request header named "X-Event-Type" will indicate the nature of the data event.

```
POST {provider-specified webhook endpoint}
X-Event-Type: SessionCreated
```

```
{
	"id": "PROVIDER-GENERATED-SESSION-ID-1",
	"version": 1,
	"actualStart": "2025-05-20T10:02:00Z",  // the start of the parking session
	"actualEnd": "2025-05-20T11:02:00Z",    // the end of the parking session
	"initiator": {
		"id": "PROVIDER1",
		"version": 1
	},
	"identifiedCredentials": [
		{
			"type": "licensePlate",
			"identifier": {
				"id": "TST001",     // the parking vehicle's VRM
				"className": "UKNumberPlate"
			},
			"issuer": [
				{
					"language": "en",
					"string": DVLA"
				}
			]
		}
	],
	"hierarchyElement": {
		"id": "CARPARK1",          // location id of where the vehicle is parked
		"version": 1,
		"name": [
			{
				"language": "en",
				"string": "Car Park No 1"
			}
		]
	},
	"segments": [
		{
			"id": "PROVIDER-GENERATED-SESSION-ID-1-SEGMENT-1",
			"version": 1,
			"actualStart": "2025-05-20T10:02:00Z",
			"actualEnd": "2025-05-20T11:02:00Z",
			"assignedRight": {
				"id": "NEW-PARKING-RIGHT-1",
				"version": 1
			},
			"validationType": [
				"licensePlate"
			]
		}
	],
	"identifiedVehicle": {          // details of the parked vehicle (if available)
		"make": "AUDI",
		"color": "BLUE",
		"country": "GB"
	}
}
```

Important note: per the specification, the session record only provides a reference to the hierarchy element (parking location where the vehicle is parked), i.e. id and version. As a convenience feature, it is possible to activate the additional provision of the location name, helping the caller to avoid another call to the inventory API. Please indicate this to the NPP team in case you want this option activated.

3. Operator Use Cases

Parking Operators (currently mostly Local Authorities) are responsible for providing inventory information to the NPP. Inventory information is broken down into

- Location/Place data
- Right Specifications applicable to one or more location(s)
- Tariffs referenced by Right Specifications

Updates to the inventory information can be pro-actively pulled by NPP users. Subscribers to the corresponding notification topics will receive updates to their pre-registered webhook endpoints. Topics to subscribe to for this purpose are:

- PlaceCreated
- PlaceUpdated
- PlaceDeleted
- RightSpecificationCreated
- RightSpecificationUpdated
- RightSpecificationDeleted
- RateCreated
- RateUpdated
- RateDeleted

3.1 Upload New Rate Table

A parking operator uploads a new tariff. The corresponding signage will look like this:

Test Council CARPARK1
Long Stay Standard Day Tariff

Up to 1 hour	£ 2.00
Up to 2 hours	£ 3.00
Up to 3 hours	£ 4.00
Up to 4 hours	£ 5.00
Up to 5 hours	£ 6.00
Up to 6 hours	£ 7.00
Up to 24 hours	£ 8.00
Maximum stay	24 hours

This is the payload sent to the NPP representing this tariff:

```
POST /v4/parking/rates
```


```
{
    "id": "7a93c824-f648-4808-ba85-4255468a431c",
    "version": 1,
    "rateTableName": [
        {
            "language": "en",
            "string": "Long Stay Standard Day"
        }
    ],
    "rateLineCollections": [
        {
            "applicableCurrency": "GBP",
            "collectionSequence": 0,
            "minTime": "PT1H",
            "maxTime": "PT24H",
            "rateLines": [
                {
                    "sequence": 0,
                    "description": [
                        {
                            "language": "en",
                            "string": "up to 1 hour"
                        }
                    ],
                    "rateLineType": "incrementingRate",
                    "value": 2.0,
                    "incrementPeriod": "PT1H",
                    "usageCondition": "once"
                },
                {
                    "sequence": 1,
                    "description": [
                        {
                            "language": "en",
                            "string": "up to 2 hours"
                        }
                    ],
                    "rateLineType": "incrementingRate",
                    "value": 1.0,
                    "incrementPeriod": "PT1H",
                    "usageCondition": "once"
                },
                {
                    "sequence": 2,
                    "description": [
                        {
                            "language": "en",
                            "string": "up to 3 hours"
                        }
                    ],
                    "rateLineType": "incrementingRate",
                    "value": 1.0,
                    "incrementPeriod": "PT1H",
                    "usageCondition": "once"
```

```
                },
                {
                    "sequence": 3,
                    "description": [
                        {
                            "language": "en",
                            "string": "up to 4 hours"
                        }
                    ],
                    "rateLineType": "incrementingRate",
                    "value": 1.0,
                    "incrementPeriod": "PT1H",
                    "usageCondition": "once"
                },
                {
                    "sequence": 4,
                    "description": [
                        {
                            "language": "en",
                            "string": "up to 5 hours"
                        }
                    ],
                    "rateLineType": "incrementingRate",
                    "value": 1.0,
                    "incrementPeriod": "PT1H",
                    "usageCondition": "once"
                },
                {
                    "sequence": 5,
                    "description": [
                        {
                            "language": "en",
                            "string": "up to 6 hours"
                        }
                    ],
                    "rateLineType": "incrementingRate",
                    "value": 1.0,
                    "incrementPeriod": "PT1H",
                    "usageCondition": "once"
                },
                {
                    "sequence": 6,
                    "description": [
                        {
                            "language": "en",
                            "string": "up to 24 hours"
                        }
                    ],
                    "rateLineType": "incrementingRate",
                    "value": 1.0,
                    "incrementPeriod": "PT18H",
                    "usageCondition": "once"
                }
            ],
            "relativeTimes": true
        }
    ],
    "availability": "public",
    "rateResponsibleParty": "Test Council"
}
```

In a next step, the newly created tariff (APDS: rate table) can be referenced in one or more right specifications (with the right specification being effective in one or more locations). For details, see 3.3, 3.4, 3.5.

3.2 Upload New Location

A parking operator uploads a new location record to the NPP. Both, Service Providers and Connected Suppliers must take note of this new location.

```
POST /v4/parking/places
```

```
{
    "id": "7591001",
    "version": 1,
    "type": "place",
    "layer": 1,
    "name": [
        {
            "language": "en",
            "string": "Lord Street"
        }
    ],
    "indicativePointLocation": {
        "type": "Point",
        "coordinates": [
            -2.146692,
            54.298062
        ]
    },
    "characteristics": {
        "spacesTotal": 217,
        "structureType": "onStreet"
    },
    "contacts": [
        {
            "organisationName": [
                {
                    "language": "en",
                    "string": "Test Council"
                }
            ],
            "type": "operator",
            "emailCommonData": [
                {
                    "address": "enquiries@testcouncil.gov.uk",
                    "typeCode": "customerService"
                }
            ]
        }
    ],
    "rightSpecifications": [
        {
            "id": "7591001-RIGHT1",
            "version": 1
        }
    ],
```

```
    "paymentMethods": [
        {
            "paymentMode": [
                "payOnEntry"
            ]
        }
    ],
    "placeStreetAddress": {
        "postCode": "CD10 3AC",
        "city": [
            {
                "language": "en",
                "string": "Parking City"
            }
        ],
        "addressLines": [
            {
                "type": "street",
                "order": 0,
                "text": "24 Main Town Street"
            }
        ]
    }
}
```

3.3 Update Location

A parking operator updates an existing location record. Existing location records will typically be updated in case

- characteristics change (e.g. number of spaces)
- right specifications are added/removed (e.g. in preparation of a next period tariff)

The Operator will send a corresponding update command to the NPP (HTTP PUT).

```
PUT /v4/parking/places/7591001
```


```
{
    "id": "7591001",
    "version": 1,
    "type": "place",
    "layer": 1,
    "name": [
        {
            "language": "en",
            "string": "Lord Street"
        }
    ],
    "indicativePointLocation": {
        "type": "Point",
        "coordinates": [
            -2.146692,
            54.298062
        ]
    },
    "characteristics": {
        "spacesTotal": 200,
        "structureType": "onStreet"
    },
    "contacts": [
        {
            "organisationName": [
                {
                    "language": "en",
                    "string": "Test Council"
                }
            ],
            "type": "operator",
            "emailCommonData": [
                {
                    "address": "enquiries@testcouncil.gov.uk",
                    "typeCode": "customerService"
                }
            ]
        }
    ],
    "rightSpecifications": [
        {
            "id": "7591001-RIGHT1",
            "version": 1
        }
```

```
    ],
    "paymentMethods": [
        {
            "paymentMode": [
                "payOnEntry"
            ]
        }
    ],
    "placeStreetAddress": {
        "postCode": "CD10 3AC",
        "city": [
            {
                "language": "en",
                "string": "Parking City"
            }
        ],
        "addressLines": [
            {
                "type": "street",
                "order": 0,
                "text": "24 Main Town Street"
            }
        ]
    }
}
```


In the above example, the number of spaces in the car park changed to 200.

3.4 Upload New Right Specification

A Parking Operator uploads a new right specification (e.g. charging hours). This can for instance happen when a Local Authority posts a new tariff tied to a hosting new right specification.

```
POST /v4/rights/specs
```

```
{
	"id": "NEWRIGHT",
	"version": 1,
	"description": [{ "language": "en", "string": "Winter Tariff"}],
	"validity": {
		"validityStatus": "definedByValidityTimeSpec",
		"validityTimeSpecification": {
			"overallStartTime": "2025-11-01T00:00:00Z"
		}
	},
	"rateEligibility": [
		{
			"id": "NEWELIGIBILITY",
			"version": 1,
			"rateTable": {
				"id": "WINTERTARIFF",
				"version": 1
			}
		}
	],

	// further right specification details
}
```

In the example, the new right specification references a new (winter time) tariff, and it will become effective in the future (November of the year). This will most likely also require an update to an existing – effective – right specification for the curren period. This is shown in the next section 3.5.

3.5 Update Right Specification

A parking operator updates an existing right specification. In the previous example (3.4), a new right specification was posted. It had an effective date of November 1st. Hence, the currently active right specification will have to expire by then. The corresponding update command will look like this:

```
PUT /v4/rights/specs/CURRRENTRIGHT

{
	"id": "CURRENTRIGHT",
	"version": 1,
	"description": [{ "language": "en", "string": "Spring/Summer Tariff"}],
	"validity": {
		"validityStatus": "definedByValidityTimeSpec",
		"validityTimeSpecification": {
			"overallStartTime": "2025-03-01T00:00:00Z",
			"overallEndTime": "2025-10-31T23:59:59Z"
		}
	},
	"rateEligibility": [
		{
			"id": "CURRENTELIGIBILITY",
			"version": 1,
			"rateTable": {
				"id": "SPRINGSUMMERTARIFF",
				"version": 1
			}
		}
	],

	// further right specification details
}
```

As you can see, the orginally open-ended, currently active right specification is now updated to expire right before the new right specification (see 3.4) kicks in.

3.6 Reconciliation Report

Use case: an operator retrieves monthly reconciliation information.

When the current reconciliation period ends, the NPP generates a consolidated reconciliation report across all service providers and across all locations of each parking operator. The input data for this report are taken from the collected transactions and the SP close-out messags (see paragraphs 1.5 and 1.6). Once the report compilation is complete, it can be retrieved by an operator via the reconciliation API. The following example shows how to request a reconciliation report for the month of June 2025:

```
GET /v4/parking/reconciliation/reports?year=2025&month=6&includeDetails=yes
```

The NPP will respond with a payload representing the machine-readable version of the consolidated reconciliation report.

```
{
	"id": "UNIQUEREPORTID1",
	"created": "2025-07-02T06:00:00Z",
	"periodStart": "2025-06-01T00:00:00.000Z",
	"periodEnd": "2025-06-30Z23:59:59.999Z",
	"periodName: "June 2025",
	"summary": {
		"summaryByProvider": [
			{
				"providerId": "PROVIDER1",
				"providerName": "Service Provider 1",
				"transactionsQuantity": 217,
				"parkingFeeTotalIncVAT": 1234.40,
				"parkingFeeTotalExVAT": 1111,
				"parkingFeeTotalVAT": 123.40,
				"refundsQuantity": 2,
				"refundsTotalIncVAT": 12,
				"refundsTotalExVAT": 10,
				"refundsTotalVAT": 2,
				"providerCommissionTotalIncVAT": 37.03,
				"providerCommissionTotalExVAT": 30.86,
				"providerCommissionTotalVAT": 6.17,
				"remittanceTotal": 1184.47
			},
			{
				"providerId": "PROVIDER2",
				"providerName": "Service Provider 2",
				"transactionsQuantity": 434,
				"parkingFeeTotalIncVAT": 2468.80,
				"parkingFeeTotalExVAT": 2222,
				"parkingFeeTotalVAT": 234.80,
				"refundsQuantity": 4,
				"refundsTotalIncVAT": 24,
				"refundsTotalExVAT": 20,
				"refundsTotalVAT": 4,
				"providerCommissionTotalIncVAT": 74.06,
```

```
                        "providerCommissionTotalExVAT": 61.72,
                        "providerCommissionTotalVAT": 12.34,
                        "remittanceTotal": 2368.94
                }
        ],
        "totals": {
                "providerId": "ALL",
                "providerName": "Total",
                "transactionsQuantity": 651,
                "parkingFeeTotalIncVAT":3703.20,
                "parkingFeeTotalExVAT": 3333,
                "parkingFeeTotalVAT": 370.20,
                "refundsQuantity": 6,
                "refundsTotalIncVAT": 36,
                "refundsTotalExVAT": 30,
                "refundsTotalVAT": 6,
                "providerCommissionTotalIncVAT": 111.09,
                "providerCommissionTotalExVAT": 92.58,
                "providerCommissionTotalVAT": 18.51,
                "remittanceTotal": 3553.41

        }
},
"summaryByLocation": [
        {
                "locationId": "CARPARK1",
                "costCode": "COSTCODECARPARK1",
                "vatPercentage": 20,
                "providerId": "PROVIDER1",
                "providerName": "Service Provider 1",
                "transactionsQuantity": 2,
                "parkingFeeTotalIncVAT": 12,
                "parkingFeeTotalExVAT": 10,
                "parkingFeeTotalVAT": 2,
                "refundsQuantity": 0,
                "refundsTotalIncVAT": 0,
                "refundsTotalExVAT": 0,
                "refundsTotalVAT": 0
        },
        // same information for this location for all service providers
        {
                "locationId": "CARPARK2",
                // same information for the next location
        }
],
"individualTransactions": [
        {
                "providerId": "PROVIDER1",
                "operatorId": "OPERATOR27",
                "transactionId": "UNIQUEPROVIDERGENERATEDTRANSACTIONID1",
                "transactionType": "payment",
                "transactionTime": "2025-07-17T17:23:02Z",
                "transactionReference": null,
                "locationId": "CARPARK1",
                "vatRate": 20,
                "totalAmount": 6,
                "netAmount": 5,
                "vatAmount": 1,
                "commissionRate": 2.5,
                "commissionVatRate": 20,
                "commissionTotalAmount": 0.18,
                "commissionNetAmount": 0.15,
```

```
                "commissionVatAmount": 0.03,
                "remittanceTotal": 5.82
            }
            // all transactions with an id mentioned in the close-out message
        ]
}
```

Please note that the individual transactions will only be included in the report if the "includeDetails" flag is set to "yes".

4. Shared Use Cases

4.1 Read List of Organisations on NPP

An NPP user reads the list of organisations (contacts) known to the NPP. The endpoint offers filtering by contact type, e.g. "serviceProvider", "operator".

Use cases could be

- An enforcement system retrieving the list of registered service providers via

  `GET /v4/parking/contacts?type=serviceProvider`

- A service provider wants to get a list of all active operators on the platform. For this, they would call

  `GET /v4/parking/contacts?type=operator`

Sample response:

```
{
    "meta": {
        "referenceInstant": 1753883082,
        "offset": 0,
        "pageSize": 200,
        "total": 6
    },
    "data": [
        {
            "id": "NPPDEMO",
            "version": 1,
            "organisationName": [
                {
                    "language": "en",
                    "string": "Demo Provider"
                }
            ],
            "type": "serviceProvider"
        },
// all other records
    ]
}
```

4.2 Read List of User-defined/Project-defined Code Lists

An NPP user retrieves the list of project-specific code lists known to the NPP.

```
GET /v4/parking/codelists
```


```
[
    {
        "id": "user_types",
        "version": 1,
        "creator": {
            "id": "NPPDEMO",
            "version": 1,
            "className": "Creator"
        },
        "locator": "https://staging-api.npp.org.uk/v4/parking/codelists/user_types",
        "userDefinedCodeListEntries": [
            {
                "entryIndex": 0,
                "definedValue": "blueBadge",
                "entryDescription": "blue badge holder"
            },
            {
                "entryIndex": 1,
                "definedValue": "bonusCard",
                "entryDescription": "bonus card holder"
            },
            {
                "entryIndex": 2,
                "definedValue": "nhsWorker",
                "entryDescription": "NHS Worker"
            },
            {
                "entryIndex": 3,
                "definedValue": "employee",
                "entryDescription": "Employee"
            }
        ]
    }
]
```


Note: once you know the URL of a particular code list (see “locator” above), you can always read the most recent version of this list using it.

5. APDS Concepts in the NPP Context

5.1 Validity of Right Specifications and Rates

APDS allows the specification of validity information in several places. Below is a corresponding extract from the official APDS specification v4.0:

- **right specification**
  - validity[mandatory] as an overall period with
    - overall start time and overall end time
    - valid periods
    - exception periods

    "Right specification validity for the associated RightSpecification"

- **rate table**
  - validity as an overall period with
    - overall start time and overall end time
    - valid periods
    - exception periods

    "Rate table validity for the associated RateTable"

    Note: if the rate table has no validity specification, it will inherit it from its parent right specification. If both, the right specification and the rate table specify a validity, the rate table's validity will be
    - the intersection of both for the valid periods and
    - the join of both for the exception periods

- **rate line collection**
  - startValidUsagePeriod[mandatory] (date/time)
    "the start time for the validity of this rate line collection"
  - endValidUsagePeriod[optional] (date/time)
    "the end time for the validity of this rate line collection"

- **rate line**
  - durationStart[optional]
    "If used, indicates the start time for the applicability of the specific rate line, generally with respect to the start of the parking or other mobility session. e.g. the start of a time-based tier charge rate."
  - durationEnd[optional]
    "If used, indicates the end time for the applicability of the specific rate line, generally with respect to the start of the parking or other mobility session. e.g. the start of a time-based tier charge rate."
  - incrementPeriod[optional]
    "the time period for incrementing the rate line charge. If set to the same as the duration of the period between the durationStart and durationEnd the increment will occur once per period, i.e. a flat rate time-based tier charge rate."

- usageCondition[optional]
  one of: fixedDuration[1], fixedNumber[2], once, unlimited
  "indicates conditions on the use of this rate line"
  1) *"check Rate Usage Duration Limitation field"*
  2) *"check Rate Usage Count Limitation field"*

Validity

The APDS specification offers different values for the *Validity.validityStatus* field:

- *planned*
- *active*
- *suspended*
- *definedByValidityTimeSpec*

Per convention, the NPP will only use *definedByValidityTimeSpec*.

RightSpecification

Right Specifications will always have a defined *Validity*. Apart from that, APDS allows to also specify an *expiry* for a right specification. Convention: the NPP will only use the (mandatory) *Validity* and not the (optional) *expiry*.

5.2 Eligibility

In APDS, right specifications define the rules for the applicability of certain usage conditions (e.g. particular rates). In the previous section, you learned about time-based constraints that can be specified in form of a validity definition.

Other eligibility criteria are available to further restrict the usage by additional conditions. The following types of eligibility criteria (expressed through corresponding qualifications) are currently used by the NPP:

- **Emissions + PropulsionEnergyType**
  Used to specify emission-based rate availability criteria
- **UserQualifications**
  Used to limit availability to particular user groups (see also “Code Lists”)
- **VehicleTypes**
  Used to limit availability to certain types of vehicles
- **LinkedRightSpecification**
  Used to specify constraints beyond session boundaries (e.g. “no return” policies)
- **FreeToPark**
  Used define times where a parking facility can be used free of charge. Shortcut to avoid creation of empty/zero value rate tables.

APDS defines more types of qualifications, but they are currently not used in the NPP context.

5.3 Interpreting Tariff Information (Rate Tables, Right Specifications)

5.3.1 RateTable

In APDS, a *RateTable* provides pricing information, broken down to individual steps if applicable. This section looks at selected examples to provide further insights into this concept. The *RateTable* itself holds some high level information such as

- unique identifier
- overall availability
- responsible party (typically the operator)
- overall validity

In addition to this, the *RateTable* holds one or more *RateLineCollection*s. A *RateLineCollection* again holds some higher level information (applicable VAT rate, maximum duration of stay) and a list of the individual tariff steps/increments. Optionally, A *RateLineCollection* can have a validity. This can be used to post future tariff information in form of a new (not yet effective) RateLineCollection which will at sometime then replace the currently active *RateLineCollection*.

Let’s look at a concrete example:


```
{
	"id": "UNIQUE_RATE_ID",
	"version": 1,
	"rateTableName": [{ "language": "en", "string": "example rate table"}],
	"rateResponsibleParty":
		{ "id": "DEMOOPERATOR", "version": 1, "className": "Operator"},
	"availability": "public",
	"validity": {
		"validityStatus": "definedByValidityTimeSpec",
		"validityTimeSpecification": {
			"overallStartTime": "2025-01-01T00:00:00Z",
			"validPeriods": [
				{
					"recurringDayWeekMonthPeriod": [
						{
							"applicableDay": [
								"monday",
								"tuesday",
								"wednesday",
								"thursday",
								"friday",
								"saturday",
								"sunday"
							]
						}
					],
					"recurringTimePeriodOfDay": [
						{
							"startTimeOfPeriod": "07:00:00",
							"endTimeOfPeriod": "23:00:00"
						}
					]
				}
			]
		}
	},
	"rateLineCollections": [
		{
			"collectionSequence": 0,
			"applicableCurrency": "GBP",
			"maxTime": "PT5H",
			"taxIncluded": true,
			"taxRate": 20,
			"rateLines": [
				{
					"sequence": 0,
					"description":
						[{ "language": "en", "string": "up to 30 minutes"}],
					"rateLineType": "flatRateTier",
					"durationStart": "00:00",
					"durationEnd": "00:30",
					"incrementPeriod": "PT30M",
					"value": 2.0,
					"usageCondition": "once"
				},
				{
					"sequence": 1,
					"description":
						[{ "language": "en", "string": "up to 1 hour"}],
					"rateLineType": "flatRateTier",
					"durationStart": "00:30",
					"durationEnd": "01:00",
					"incrementPeriod": "PT30M",
					"value": 1.5,
					"usageCondition": "once"
				},
```

```
                        {
                                "sequence": 2,
                                "description":
                                        [{ "language": "en", "string": "2-5 hours"}],
                                "rateLineType": "incrementingRate",
                                "durationStart": "01:00",
                                "durationEnd": "05:00",
                                "incrementPeriod": "PT1H",
                                "value": 1.0,
                                "usageCondition": "unlimited"
                        }
                ]
        }
    ]
}
```


This *RateTable* provides a lot of information.

- Like (currently) all tariffs on the NPP, it is publicly available.
  availability
- It is controlled by the operator identified by "DEMOOPERATOR".
  rateResponsibleParty
- It is effective every day from 7am to 11pm.
  validity.validityTimeSpecification.validPeriods
- It is subject to 20% VAT (most likely used at an off-street location).
  rateLineCollections[0].taxRate
- The operator allows a max. stay of 5 hours.
  rateLineCollections[0].maxTime
- The first half hour costs £2, the second half hour costs £1.5. Any subsequent hour (up to the max stay) is £1.
  rateLineCollections[0].rateLines

Now, let's look at the individual *RateLines*. They are placed in a specific order via their *sequence* property. The first increment is defined as:

"sequence": 0	defines the order of processing the individual rate lines
"description": [...]	a description understandable for humans
"rateLineType": "flatRateTier"	the type of RateLine (one of: *flatRate, incrementingRate, flatRateTier*); in our case, "flatRateTier" indicates a fixed price for a specific time-based tier, here the first 30 minutes
"durationStart": "00:00"	the (relative!) start of the time-based tier
"durationEnd": "00:30"	the (relative!) end of the time-based tier
"incrementPeriod": "PT30M"	the time period for incrementing the charge within this tier; in this example only one increment fits into the boundaries defined by *durationStart* and *durationEnd*, hence the *flatRateTier* type which is a specialisation of the *incrementingRate* type which you will see later in this example.
"usageCondition": "once"	in this particular case, it is redundant, but the *usageCondition* here indicates that only one increment is allowed within this *RateLine*.
"value": 2	the price per increment within the *RateLine*

The second *RateLine* is very similar and should be clear from the above explanations. Let's look at the third *RateLine* which is different.

"*sequence*": 2	defines the order of processing the individual rate lines
"*description*": [...]	a description understandable for humans
"*rateLineType*": "incrementingRate"	the type of RateLine (one of: *flatRate, incrementingRate, flatRateTier*); in our case, "incrementingRate" indicates that the defined charge is incurred incrementally (per 1-hour step)
"*durationStart*": "01:00"	the (relative!) start of this time-based tier
"*durationEnd*": "05:00"	the (relative!) end of this time-based tier, so a total of 4 hours
"*incrementPeriod*": "PT1H"	the time period for incrementing the charge within this tier; due to the max stay of 5 hours and the start/end definition in this example, a total of 4 1-hour increments can be applied.
"*usageCondition*": "unlimited"	the *usageCondition* indicates the repeatability of increments within this *RateLine*.
"*value*": 1	the price per increment within the *RateLine*

Per the APDS spec, a *durationStart/durationEnd* pair and an *incrementPeriod* can occur individually or in combination (depending on the use case).

The signage for this tariff example might look like this:

Parking Mon-Sun 7am – 11pm

up to 30 minutes	£ 2.00
up to 1 hour	£ 3.50
up to 2 hours	£ 4.50
up to 3 hours	£ 5.50
up to 4 hours	£ 6.50
up to 5 hours	£ 7.50

Max Stay 5 hours

Please note how the communicated totals are calculated by applying the individual increment charges (per rate line, per applied increment). APDS sometimes allows different representations for the same scenario. So, the 3rd rate line in the example above could also be described via four distinct 1-hour rate lines of type *flatRateTier*.

5.3.2. Right Specifications

In APDS, *RightSpecifications* define constraints / eligibility criteria for the applicability of a *RateTable*. Examples for this are:

- tariff transitions
- "no return after..." policies
- emission-based charging
- shortcuts for "free parking"

5.3.2.1 Tariff Transition

Whilst the APDS rate tables define costs for parking at specific times, it is often necessary to define policies for handling exceeding of tariff boundaries. This is done via the “rateTransition” ruleset in the right specification. It has the following properties:

- **overpaymentPolicy** (one of *rateEndCutOff*, *fullTimePurchased*, *creditCarryOver*)
- **followThroughAllowed**: indicates if a tariff-follow-through is allowed
- **prepaymentAllowed**: indicates whether a parking right can be purchased for a charging period that has not started yet
- **reservationAvailable**: indicates whether the operator offers pre-booking at the corresponding location(s)

Here is an example:

```
{
      "overpaymentPolicy": "rateEndCutOff",
      "followThroughAllowed": false,
      "prepaymentAllowed": false,
      "reservationAvailable": false
}
```

"overpaymentPolicy": "rateEndCutOff" *"overpaymentPolicy"*: "fullTimePurchased" *"overpaymentPolicy"*: "creditCarryOver"	Defines the overpayment policy: • The parking rights ends at the end of the charging hours, so the motorist that paid at 10.45pm only gets 15 minutes parking time and has to leave by 11pm. This is known as "rateEndCutOff" • The motorist gets the full parking time. So, the motorist who paid at 10.45pm gets to stay until 11.45pm. This is known as "fullTimePurchased" • If parking were unrestricted during the night hours, there might be a third option; that the "unused portion" of the time purchased can be used towards the next time the tariff is in force. This is known as "creditCarryOver"
"followThroughAllowed": true "followThroughAllowed": false	This is a specific type of overpayment that is only applicable if a tariff contains rates that follow on from each other but can be used in combination with other types of overpayment. If e.g. a motorist has paid enough to buy time on the daytime and evening rate, the parking operator’s policy could be that: • The motorist can stay for the entire period (Follow-through is “Allowed”) • The motorist is not allowed to stay for the entire period (in which case they should have been prevented from making the payment). This case is “No Follow”.
"prepaymentAllowed": false	Specifies whether it is possible to purchase a parking right for a charging period that has not yet started.
"reservationAvailable": false	Indicates whether a space can be reserved. Most of the current NPP locations do not offer this or at least do not give a guarantee.

Combinations

The operator-defined *overpaymentPolicy* and the *followThroughAllowed* settings theoretically allow a total of 12 permutations (combination of possible values and/or missing values). However:

A *followThroughAllowed* policy shall only be missing in case it is not applicable/relevant (i.e. when rates do not follow on from each other with a certain charging hours group. The same applies to the *overpaymentPolicy* established by the operator. It shall only be missing in case it is not applicable (e.g. in free tariffs or permit parking). In all other instances, the operator shall make sure to provide this information. Typically, over-payment and follow-through will be mutually exclusive. If there are rates following on from each other and the operator explicitly allows a follow-through, there will be no overpayment situation. If follow-through is explicitly not allowed, the overpayment policy will provide further guidance, i.e. how long the motorist can park and when he will have to leave. In case of *rateEndCutOff*, he will have to leave by the end of the current validity. In case of *fullTimePurchased*, he will be able to stay as llong as he has paid for. The *creditCarryOver* policy relates to a different scenario (with e.g. unrestricted night times between rates).

Should an operator (by mistake) not have adhered to the above, the following behaviour shall be the convention:

- *overpaymentPolicy* has a default of *creditCarryOver*
- *followThroughAllowed* has a default of *yes* (in case there are rates following on from each other)

5.3.2.2 No Return Policies

A "no return before ..." policy makes it necessary to interconnect (potential) sessions. In APDS, this is done via the *linkedRightSpecification* qualification in the *eligibility* package. The example below shows how a "no return within 60 minutes" policy would be represented in the inventory API:

```
{
    "id": "RIGHT_SPEC_ID",
    "version": 1,
    "type": "oneTimeUseParking",
    "rateEligibility": [
        {
            "id": "ELIGIBILITY_1",
            "version": 1,
            "rateTable": {
                "id": "STANDARD_RATE_ID",
                "version": 1
            },
            "eligibility": {
                "qualifications": [
                    {
                        "linkedRightSpecification": {
                            "qualifyingRightSpecification": {
                                "id": "RIGHT_SPEC_ID",
                                "version": 1
                            },
                            "assignedRightTimeRelative": {
                                "earliestStartRelative": "PT1H"
                            }
                        }
                    }
                ]
            }
        }
    ],
    "validity": {
        "validityStatus": "active"
    }
}
```

In other words: a driver is only eligible to use the standard day rate (STANDARD_RATE_ID) if the last use of the (same) *RightSpecification* was not less than 1 hour ago.

5.3.2.3 Extension Policies

The configuration spreadsheet currently used to coordinate location settings with all operators has a column "Extensions". It allows three possible values:

- *None*: session extensions are not allowed or not applicable
- *Restart*: extensions charged at initial tariff, e.g. customer buys 1st hour on arrival, then buys a second hour 50 minutes after arrival at same fee as 1st hour
- Cumulative: extensions charged at rate for full time parked, e.g. customer buys 1st hour on arrival, then buys a second hour 50 minutes after arrival. Customer is charged difference between fee for 1 hour and 2 hours.

This is how those three scenarios are represented in the *NPP Inventory API*:

None

This is expressed via an eligibility using a qualification of type *linkedRightSpecification*. It defines itself as qualifying *Right Specification*, indicating that it can earliest be re-used the next day.


```
"eligibility": {
    "eligibilityName": [
        {
            "language": "en",
            "string": "extensions: None"
        }
    ],
    "qualifications": [
        {
            "linkedRightSpecification": {
                "qualifyingRightSpecification": {
                    "id": "4e435976b9f9ac3d788f45ea09e0f231-2",
                    "version": 1
                },
                "assignedRightTimeRelative": {
                    "metaType": "RelativeTimes",
                    "earliestStartRelative": "next",
                    "unit": "day"
                }
            }
        }
    ]
}
```


Restart

This again is expressed via an eligibility using a qualification of type *linkedRightSpecification*. It defines itself as qualifying *Right Specification*, indicating that it can (and by convention must) be re-used the same day.


```
"eligibility": {
    "eligibilityName": [
        {
            "language": "en",
            "string": "extensions: Restart"
        }
    ],
    "qualifications": [
        {
            "linkedRightSpecification": {
                "qualifyingRightSpecification": {
                    "id": "4e435976b9f9ac3d788f45ea09e0f231-0",
                    "version": 1
                },
                "assignedRightTimeRelative": {
                    "metaType": "RelativeTimes",
                    "earliestStartRelative": "current",
                    "unit": "day"
                }
            }
        }
    ]
}
```


Cumulative

By convention, this is the default, obviously respecting overpayment and follow-through rules.

5.3.2.4 Emission-based Charging

Guidance on emission-based charging and how this is represented in the inventory API can be found in the APDS refresher section on eligibility (see here).

5.3.2.5 Shortcut for "Free Parking"

Operators also post *RightSpecifications* where parking is free (either for everyone or for selected user groups). In this case, there is no need to publish an "empty" or zero value *RateTable*. Instead, free parking can be indicated via a corresponding qualification in the *RightSpecification*.

```
{
      "id": "UNIQUE_RIGHT_SPEC_ID",
      "type": "oneTimeUseParking",
      "rateEligibility": [
         {
              "id": "917e3535-cd57-4cb2-9398-8b592d479cfc-5",
              "version": 1,
              "eligibility": {
                  "qualifications": [
                      {
                          "freeToPark": true
                      }
                  ]
              }
      ],

      // ... remaining details
}
```

As you can see, the *rateEligibility* does not reference any *RateTable* and instead indicates free parking via the eligibility.

5.4 Restrictions

There can be times where a location is not available for parking or temporary access limitations apply:

Restriction/Status	Meaning	Comment
No Parking	A parking location is not available for parking during the times specified in the Hrs Group, although there is no physical means of preventing parking at the location.	Usually applies to on-street (though could also be used for a pay on arrival off street location where there is no facility for restricting access)
Closed – no parking	A parking location is physically closed by a gate or other barrier during the times specified in the Hrs Group. Vehicles **are not** allowed to remain in the location when it is closed.	Will apply to an off-street location. Vehicles remaining in the location during closed hours may be subject to a fine or other sanction and will not be retrievable, or retrieval will be charged for (the NPP will not provide details of sanctions)
Closed – no access	A parking location is physically closed by a gate or other barrier at the times specified in the Hrs Group. Vehicles **are** allowed to remain in the location when it is closed (subject to a tariff).	Will apply to an off-street location. Vehicles entering before the location is closed may remain in the location during closed hours but will not be retrievable, or retrieval will be charged for during the closed hours (the NPP will not provide details of retrieval costs).
Closed – exit only	A parking location is physically closed by a gate or other barrier preventing entry at the times specified in the Hrs Group. However, vehicles can exit the location during closed hours.	Will apply to an off-street location. Vehicles entering before the location is closed may remain in the location during closed hours and can exit at any time.
Unrestricted	There are no parking restrictions or tariff at the location during the times specified in the Hrs Group	

The APDS standard uses the following instruments to differentiate the above scenarios:

- No parking → expressed *via Place.operatingRestrictions ("noParking")*
- Location “closed” status → expressed via the exceptionPeriods in *Place.times.operatingTime*
- Location “closed” status → expressed via the exceptionPeriods in *Place.times.operatingTime*
- No entry possible → expressed via *Place.times.accessAndEgress.entranceOpenTime*
- No exit possible → expressed via *Place.times.accessAndEgress.exitOpenTime*
- Physical means for preventing access → *Place.characteristics.accessControlled = true*

Per convention, gaps in the specification are interpreted as “unrestricted”.

(Examples of the different scenarios can be found in the GitHub repository)

5.5 Push Notification Service

Depending on their push notification service subscription (managed in coordination with the NPP team), push API users will receive notifications for different types of events. To do this, they must provide the NPP with corresponding call-back endpoints to which new/updated information will be sent.

This webhook mechanism is

- mandatory for *Service Providers* to receive inventory information in a timely manner (in addition, they shall sporadically query the NPP to make sure nothing was missed)
- optional for *Enforcement Systems* to receive assigned right and session information (alternatively, they can actively query the NPP)

5.5.1 Event Types

Distinguishing the type of event received can be done in two ways:

- Setting up separate endpoints by event type
- Checking the custom http request header "X-Event-Type"

Both is possible, but the latter is the recommended approach. Below is the list of possible event types to be received:

- PlaceCreated *(body will contain records of type Place)*
- PlaceUpdated *(body will contain records of type Place)*
- PlaceDeleted *(body will contain records of type Place)*
- RightSpecificationCreated *(body will contain records of type RightSpecification)*
- RightSpecificationUpdated *(body will contain records of type RightSpecification)*
- RightSpecificationDeleted *(body will contain records of type RightSpecification)*
- RateCreated *(body will contain records of type RateTable)*
- RateUpdated *(body will contain records of type RateTable)*
- RateDeleted *(body will contain records of type RateTable)*
- AssignedRightCreated *(body will contain records of type AssignedRight)*
- AssignedRightUpdated *(body will contain records of type AssignedRight)*
- AssignedRightDeleted *(body will contain records of type AssignedRight)*
- SessionCreated *(body will contain records of type Session)*
- SessionUpdated *(body will contain records of type Session)*
- SessionDeleted *(body will contain records of type Session)*
- ContactCreated *(body will contain records of type ContactPoint)*
- ContactUpdated *(body will contain records of type ContactPoint)*
- ContactDeleted *(body will contain records of type ContactPoint)*
- SystemEvents *(body will contain records of type SystemAlert) – future use*

The range of subscribed event types will depend on your NPP role (service provider, operator, connected supplier).

5.5.2 Webhook Endpoint(s)

The NPP Push Notification Service requires you to expose one ore more endpoints for callback purposes. Whenever a data event that you are subscribed to occurs, you will receive a corresponding notification to your endpoint(s). The Push Service is available for both, Inventory API events as well as Assigned Right / Session events (mainly used by enforcement systems).

Currently, the subscription process is a manual one. You provide the NPP team with:

- endpoint URL
- authentication method
- authentication credentials
- list of event types to be notified of

5.5.3 Authentication

While NPP test environments support, but do not require authentication by the NPP (in its role as an HTTP client), this is mandatory for the production environment. You have a choice between

- API Key based authentication and
- OAUTH2 authentication

5.5.3.1 API Key

For the API key based authentication, you will have to provide the NPP team with either the name of a custom header to hold the actual API key / access token or an auth type to be used in the standard *Authorization* header. In addition, you'll have to pre-share the actual API key with us.

Examples for the two variants:

- `Authorization: ProviderApiKey {yourApiKey}`
- `Provider-Specific-Header: {yourApiKey}`

5.5.3.2 OAUTH2

As an alternative, you can offer OAUTH2 based authentication. In that case, you'll have to provide

- the URL of the endpoint where fresh access tokens can be obtained
- the client id and
- the client secret created by you for the NPP push client

The NPP will then create new access tokens as needed (we'll always use a *grant_type* of *client_credentials*).

5.5.4 Example Push Notification

```
POST
X-CLIENT-VERSION: NPP Notification Service 1.2
X-CORRELATION-ID: 097b36ba-ce52-4edd-9123-ac8d20b9acfb
X-EVENT-TYPE: RightSpecificationUpdated
{
	"id": "f1e7af08-eada-4b5e-8d81-1e29a11006f0",
	"version": 1,
	"type": "oneTimeUseParking",
	"description": [{ "language": "en", "string": "Standard Day"}],
	"issuer": { "id": "TESTCOUNCIL", "version": 1, "className": "Operator"},
	"hierarchyElements": [
		{ "id": "5621001", "version": 1},
		{ "id": "5621002", "version": 2}
	],
	// remaining right specification details
}
```

Please note that all notifications are **POST**ed.

5.5.5 Retry on Notification Failure

If sending a push notification to your webhook endpoint fails, the NPP will retry (currently up to three times), with increasing waiting times in between. After the maximum number of configured attempts, the NPP will give up, and the information will have to be retrieved in one of your sporadic active enquiries.

Whilst the NPP currently logs returned error payloads, they're never processed/evaluated. During the introductory phase of the live system inventory API, the NPP may pull error logs upon request.

5.5.6 Order of Events

The order of data events may vary depending on the concrete use case. Hence, there is currently no guaranteed order of events. As long as the platform has control, it will however attempt to choose a sensible order.

5.5.6.1 Sequence Examples

- The NPP will always send a *ContactCreated* event first to publish a **new operator**'s identity before then sending place, rate table and right specification data.

- When an operator publishes a **new future tariff**, the order will be:

  - *RateUpdated* (to set an expiration date for the currently active rate table) if required
  - *RateCreated* (to publish the new future rate table)
  - *RightSpecificationUpdated / RightSpecificationCreated* (to define the new or updated rate eligibility criteria and validity)
  - *PlaceUpdated* (to reference the updated/new right specification)

- When an operator introduces a **new location**, the sequence will typically be

    - *RateCreated* (in case the new location uses a new tariff)
    - *RightSpecificationCreated / RightSpecificationUpdated* (to define a new right specification or add a rate eligibility to an existing one)
    - *PlaceCreated* (to publish the actual location details)

- If you combine the above into a scenario where a **new operator** is onboarded with 100 locations, the order will be

    - *ContactCreated*
    - 100 x *RateCreated* (or whatever the number of tariffs is)
    - 100 x *RightSpecificationCreated* (or whatever the number of right specs is)
    - 100 x *PlaceCreated*

5.5.6.2 Event Pace

The temporal proximity of push notifications depends on the actual use case (singular update or bulk upload) and the tools used for this (interactive editor with "publish record now" button, or bulk upload tool). Push messages belonging to a particular bulk upload will be sent with at least 100ms waiting time in between.

5.6 Versioning

The APDS standard offers an optional versioning mechanism. In the context of the NPP pilot, versioning has not been used. In favour of a more robust process for inventory data updates, this will however change. The essence of the versioning mechanism – as used for the NPP – is to have an unambigiuous indicator (the version number) of the state of uniquely identifiable resource (via its id). This means that

- Whenever an existing resource (e.g. place, right specification, rate table) changes, its version must be incremented by 1.
- An API consumer must never run into a situation where two copies of a resource differ even though both, id and version are identical.

5.6.1 Example

Let's look at an example (can also be found in the GitHub repository) which is an updated version of a previous example not using versioning.

5.6.1.1 Initial Situation

An operator has an existing, effective configuration:

- Place with id CAR-PARK-1 (and version 1)
- Right Specification RS-CP-1 (version 1), applicable to CAR-PARK-1
- Rate Table RT-CURRENT-TARIFF (version 1) referenced by Right Specification RS-CP-1

In the original example, the RT-CURRENT-TARIFF had an initial validity

- starting in the past,
- but open-ended

The use case looked at a scenario where the operator decided to replace the currently active tariff with a new one to become effective in the future.

CAR-PARK-1

```
{
    "id": "CAR-PARK-1",
    "version": 1,
    "rightSpecifications": [{"id": "RS-CP-1", "version": 1}]
    // location details
}
```

RS-CP-1

```
{
    "id": "RS-CP-1",
    "version": 1,
    "hierarchyElements": [{"id": "CAR-PARK-1", "version": 1}],
    "rateEligibility": [{"rateTable": {"id":"RT-CURRENT-TARIFF","version":1}}]
    // right spec details
}
```

RT-CURRENT-TARIFF

```
{
    "id": "RT-CURRENT-TARIFF",
    "version": 1,
    "validity": {
        "validityStatus": "definedByValidityTimeSpec",
        "validityTimeSpecification": {
            "overallStartTime": "2025-01-01T00:00:00Z"
        }
    }
    // remaining rate table details
}
```

5.6.1.2 Required Steps (Inventory-API-wise)

The original example listed the following required steps:

1. Set an expiration for the currently active tariff
2. Publish the new future tariff with a validity starting when the current one expires
3. Add the new tariff to the right specification

The steps remain valid with the introduction of versioning. There is however a required 4th and last step.

Step 1: RT-CURRENT-TARIFF Expiration

In a first step, the operator defines an expiration date for the currently active tariff by updating its validity accordingly.

PUT /v4/parking/rates/RT-CURRENT-TARIFF

```
{
    "id": "RT-CURRENT-TARIFF",
    "version": 2, ← incremented version
    "validity": {
        "validityStatus": "definedByValidityTimeSpec",
        "validityTimeSpecification": {
            "overallStartTime": "2025-01-01T00:00:00Z",
            "overallEndTime": "2026-03-01T00:00:00Z" ← defined expiration
        }
    }
}
```

Step 2: Publish new RT-NEW-TARIFF Rate Table

In a next step, the operator publishes the new tariff which will become active in the future (once the current tariff expires). It starts with an initial version of 1.

POST /v4/parking/rates

```
{
    "id": "RT-NEW-TARIFF",
    "version": 1, ← initial version 1
    "validity": {
        "validityStatus": "definedByValidityTimeSpec",
        "validityTimeSpecification": {
            "overallStartTime": "2025-01-01T00:00:00Z", ← validity start
                                                          (in alignment with expiry
                                                           of RT-CURRENT-TARIFF)
        }
    }
    // remaining rate table details
}
```

Step 3: Update Right Specification to include new Tariff

In a next step, the operator updates the current right specification to include the new tariff. This leads to a version number increment for the right specification.

PUT /v4/parking/rights/specs/RS-CP-1

```
{
    "id": "RS-CP-1",
    "version": 2, ← updated version 1
    "rateEligibility": [

        {"rateTable": {"id":"RT-CURRENT-TARIFF","version":2}},← updated version
                                                                (with added expiration)

        {"rateTable": {"id":"RT-NEW-TARIFF","version": 1}} ← new rate table
                                                             (taking over in the future)
    ]
    // remaining right spec details
}
```

Step 4: Update Place to reference updated Right Specification

In a last step, the operator then updates the place to reference the updated right specification (version 2).

PUT /v4/parking/places/CAR-PARK-1

```
{
    "id": "CAR-PARK-1",
    "version": 2, ← new version (due to right spec reference change)
    "rightSpecifications": [
        {
            "id": "RS-CP-1",
            "version": 2 ← updated right specification (with new tariff added)
        }
    ]
    // remaining place details
}
```

5.6.1.3 Summary

Before	After
Place (id: CAR-PARK-1, version: 1)	Place (id: CAR-PARK-1, version: **2**)
Right Specification (id: RS-CP-1, version: 1)	Right Specification (id: RS-CP-1, version: **2**)
Rate Table (id: RT-CURRENT-TARIFF, version: 1)	Rate Table (id: RT-CURRENT-TARIFF, version: **2**)
	Rate Table (id: RT-NEW-TARIFF, version: 1)

6. Important Concepts outside the APDS Context

6.1 Authorisation (Roles and Permissions)

The NPP uses a combination of RBAC (role-based access control) and ABAC (attribute-based access control). This chapter provides an overview of the implemented controls.

6.1.1 Roles

6.1.1.1 Atomic Roles

The NPP currently defines the following atomic roles:

Role	Comments
INVENTORY_CONSUMER	read inventory information (locations, right specifications, tariffs, code lists)
INVENTORY_PROVIDER	publish inventory information (locations, right specifications, tariffs, code lists)
RIGHT_CONSUMER	read assigned right information
RIGHT_PROVIDER	create/update and send assigned right information
SESSION_CONSUMER	read session information
SESSION_PROVIDER	create/update and send session information

6.1.1.2 Role Groups

Atomic rules are then grouped as follows:

- OPERATOR
- SERVICE_PROVIDER
- ENFORCEMENT_PROVIDER

The the following table shows the atomic roles included in each one of the above role groups:

Role Group	Included Atomic Roles
OPERATOR	INVENTORY_CONSUMER, INVENTORY_PROVIDER, RIGHT_CONSUMER, RIGHT_PROVIDER, SESSION_CONSUMER, SESSION_PROVIDER
SERVICE_PROVIDER	INVENTORY_CONSUMER, RIGHT_CONSUMER, RIGHT_PROVIDER, SESSION_CONSUMER, SESSION_PROVIDER
ENFORCEMENT_PROVIDER	INVENTORY_CONSUMER, RIGHT_CONSUMER, SESSION_CONSUMER

6.1.2 Attribute-based Access Control

In addition to the role-based access control, further – attribute-based – constraints apply.

- An Operator can only create and update inventory information that relates to locations within their own NPP-assigned range of location codes.
- Users with the RIGHT_PROVIDER/SESSION_PROVIDER roles can only modify assigned right/session records that they created.
- Users with the SERVICE_PROVIDER group role can only read assigned right/session data created by themselves.
- Users with the ENFORCEMENT_PROVIDER group role can read session data across all service providers.
- Users with the ENFORCEMENT_PROVIDER group role can only read data relating to locations that they have been contracted for (and subscribed to).

7. Required Conventions

7.1 Need for Conventions

The *NPP* is based on the *APDS* standard data model and the *APDS*-defined API. There are however situations where the implementing parties need to agree upon certain conventions to ensure a mutually-aligned interpretation of data. The same applies to the configuration of an operator's inventory (locations, tariffs, charging hours). Conventions answer questions like

- Which set of enumeration values to use (or not use), where *APDS* defines more than needed by the *NPP*
- Handling of edge cases, e.g. boundaries of time ranges
- Which set of classes and endpoints to use (the *NPP* e.g. currently does not make use of the *APDS Observation* and *Quotes* domains)
- Which set of possible eligibility criteria to use

7.2 Identified Required Conventions

This section provides an overview of the already agreed-upon conventions in the *NPP* context.

7.2.1 Point Location

Shall be the car park's entrance or centre of street (by length) or centre of parking bay.

7.2.2 Validity End Times

If determined as same as start of next validity, always 0.001 minute before stated time. E.g. validity 1 is 10:00 to 18:00, validity 2 is 18:00 to 22:00 shall be interpreted as validity 1 from 10:00 to 17:59.999 and validity 2 from 18:00 to 22:00.

7.2.3 Describing non-chargeable Periods

The use of zero cost tariffs for this purpose is deprecated. Non-chargeable periods will instead be represented

- explicitly via a corresponding `freeToPark` setting in *Right Specifications*
- implicitly via a lack of applicable *Operating Restrictions* and/or *Right Specifications*

7.2.4 Rounding

VAT Amounts

VAT amounts shall be rounded to the nearest 1p by rounding up any amount of 0.5p and higher and rounding down any amount lower than 0.5p as defined in "VATREC12030".

Commission Amounts

Commission amounts shall be rounded to the nearest 0.1p by rounding up any amount of 0.05p and higher and rounding down any amount lower than 0.05p.

7.2.5 Eligibility

All eligibility qualifications are ANDed.

7.2.6 Eligibility User Types

Defined as *APDS* user-defined code list. The *NPP* URL for project-specific user types is: `/v4/parking/codelists/user_types`. Currently, only the `blueBadge` and `payOnDeparture` types are used.

7.2.7 Energy Source Types

The *APDS* model defines energy source types that are not used within the *NPP* context. The *NPP* convention for energy source type references is this:

- For petrol-powered vehicles, the *NPP* only differentiates between ***petrol*** and ***petrolBatteryHybrid***. Specialisations concerning octane and lead are not used, they all fall under the ***petrol*** categorisation.
- The same applies to the Diesel energy source. The *NPP* only differentiates between ***diesel*** and ***dieselBatteryHybrid***. Hence, ***biodiesel*** counts as ***diesel***.
- The *NPP* only uses ***lpg***, and ***liquidGas*** counts as ***lpg***.
- Fully electric vehicles are categorised as ***battery***.
- The ***unknown*** enum is used in all situations where the energy source is either unknown/uncategorised or cannot be otherwise categorised (using the set of *APDS*-defined enumeration values).
- There is currently no *NPP* use case where ***all*** would be required.
- The ***other*** category is not used, because it creates contextual dependencies.

7.2.8 Vehicle Types

The *APDS* model defines an extensive list of vehicle types not all of which are used in the *NPP* context. The following table lists the *APDS* vehicle type enumeration values used for the NPP along with an information of implicitly included *APDS* enumeration values and a mapping to the nearest DVLA category equivalent.

NPP category in use	included additional APDS categories (i.e. not explicitly used)	nearest DVLA category
agriculturalVehicle		T
bicycle		
bus	articulatedBus, articulatedTrolleyBus, trolleyBus	M3
car	carOrLightVehicle, fourWheelDrive, largeCar, passengerCar, smallCar	M1
carWithCaravan	vehicleWithCaravan	
carWithTrailer	vehicleWithTrailer	
heavyGoodsVehicle	heavyDutyTransporter, heavyGoodsVehicleWithTrailer, heavyVehicle, highSidedVehicle, largeGoodsVehicle, longHeavyLorry, lorry, tanker	N3
minibus		M2
motorcycle	moped, motorScooter, twoWheeledVehicle	L1, L3
motorcycleWithSideCar		L4
motorhome		M1 "motor caravan"
threeWheeledVehicle		L5
trailer		O
van		N1

The following *APDS*-defined categories are <u>not</u> considered at all:
constructionOrMaintenanceVehicle, metro, other, tram, unknown, vehicleWithCatalyticConverter, vehicleWithoutCatalyticConverter, withEvenNumberedRegistrationPlates, withOddNumberedRegistrationPlates

7.2.9 Order of Priority

There is always a remaining risk of slight inconsistencies in the inventory. To provide guidance on how to handle such situations, the following order of priority has been defined:

1. `{AnyItem}.version` (the most recent valid version of a resource is always the one to be used, i.e. the one that has the same id and the highest version and is valid)
2. `Place.operatingRestrictions` (operating restrictions defined for a parking location)
3. `RightSpecification.validity` (validity as specified in the *hours groups* definition)
4. `RightSpecification.eligibility` (specified eligibilities)
5. `RateTable.validity` (validity as specified in the *tariff* - if any)
6. `RateTable.rateLineCollections.maxTime` (max. duration of stay)
7. `RateTable.rateLineCollections.rateLines` (increments in the tariff)

Any higher-level element (according to this order) overrules subsequent items.

Annexes

Annex 1: Postman Collection

This Developer Guide has an accompanying PostMan collection with sample requests. Like the document itself, this collection will be continuously enhanced and adapted (based on user feedback). You can find it on the NPP Developer Exchange GitHub: https://github.com/national-parking-platform/npp-developer-exchange/tree/main/postman.

(if you don't have access, send a message to development@npp.uk.net)

Annex 2: Document Management

Date	Version	Author(s)	Description
15/07/2025	1.0	Markus Schneider	Initial Version
17/07/2025	2.0	Markus Schneider	Add Reconciliation Section (for both, service providers and operators)
22/07/2025	2.1	Markus Schneider, Tim Jansen	• Add diagram for flow of assigned rights, sessions, and corresponding transactions • Add "operatorDefinedReference" to show representation of cost codes • Add "responsibilityRoleAssignments" to show who operates a parking location • Correct attribute name in example payloads (emailCommonData) • Add new /contacts endpoint
26/07/2025	2.2	Markus Schneider	• Elaborate on authorisation (roles, permissions, RBAC, ABAC) • Add /codelists endpoint to inventory API • Additional information on existing reconciliation API (close-out message errors)
31/07/2025	2.3	Markus Schneider	• Elaborate on validity of right specifications, rate tables, and rate line collections • Rate examples with comments • Right examples with comments
07/08/2025	2.4	Markus Schneider	• Use case details for enforcement
12/08/2025	2.5	Markus Schneider	• Correction of validity requirements for right specifications and rate tables • List of validity checks for close-out message
20/08/2025	2.6	Tim Jansen	• More information on emission-based charging • Additional details and conventions converning validity
10/09/2025	2.7	Keith Williams, Markus Schneider	• “no parking” and other operating restrictions
19/09/2025	2.8	Tim Jansen	• Fixed enum name (comparison operator)
25/09/2025	2.9	Moritz Drathen	• Optional provision of location name in right specification (convenience feature for enforcement systems)
08/10/2025	2.10	Moritz Drathen	• List of possible push notification event types

12/11/2025	2.11	Moritz Drathen	• Update list of possible push notification event types
10/12/2025	2.12	Tim Jansen	• Elaborate on rate transition combinations • Add info on session extensions
04/02/2026	2.13	Markus Schneider	• Elaborate on webhook mechanism
18/02/2026	2.14	Markus Schneider	• Fix typo in section 3.5 (update right example)
23/02/2026	2.15	Markus Schneider	• Explain fuel type differentiation in more detail (update of section 1.1.5.2, new section 1.1.5.3 with convention)
25/02/2026	2.16	Markus Schneider	• New chapter 7 on required conventions, especially "order of priority"
05/03/2026	2.17	Moritz Drathen	• New section 5.6 on versioning
05/03/2026	2.18	Moritz Drathen	• Updated conventions (order of priority)
09/04/2026	2.19	Markus Schneider	• Updated conventions (vehicle types)
14/04/2026	2.20	Markus Schneider	• Correction in sequence examples 5.5.6.1