Standaard voor Publicatie
Dynamische Parkeerdatal

Standard for Publishing
Dynamic Parking Data
DOCUMENT INFORMATION

Title: Standard for Publishing Dynamic Parking Data
Author(s): Werkgroep SPDP
Version: 2.0
Date: 2014-12-17
File: Standard for the Publication of Dynamic Parking Data v2.0.7.docx
Project: Standaard voor Publicatie Dynamische Parkeerdata

Document versions:

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>2013-11-19</td>
<td>Erwin Gribnau</td>
<td>Initial version</td>
</tr>
<tr>
<td>0.2</td>
<td>2013-11-20</td>
<td>Erwin Gribnau</td>
<td>Detailed HTTP usage</td>
</tr>
<tr>
<td>0.3</td>
<td>2013-11-25</td>
<td>Leontien Balkenende</td>
<td>Added data model and message contents</td>
</tr>
<tr>
<td>0.4</td>
<td>2013-11-25</td>
<td>Erwin Gribnau</td>
<td>Changes to data model</td>
</tr>
<tr>
<td>0.5</td>
<td>2013-11-28</td>
<td>Erwin Gribnau</td>
<td>New document structure</td>
</tr>
<tr>
<td>0.6</td>
<td>2013-12-04</td>
<td>Leontien Balkenende</td>
<td></td>
</tr>
<tr>
<td>0.7</td>
<td>2013-12-12</td>
<td>Kees Koster</td>
<td>Added generic project information &amp; document structure</td>
</tr>
<tr>
<td>0.8</td>
<td>2013-12-20</td>
<td>Henk den Breejen</td>
<td>Changes to data model, overall-view, and description.</td>
</tr>
<tr>
<td>0.9</td>
<td>2014-01-03</td>
<td>Kees Koster</td>
<td>Addition of descriptions and review</td>
</tr>
<tr>
<td>1.0</td>
<td>2014-02-09</td>
<td>Kees Koster</td>
<td>Preparation of the initial release version after processing of review comments</td>
</tr>
<tr>
<td>1.01</td>
<td>2014-10-20</td>
<td>Kees Koster</td>
<td>Rework of examples</td>
</tr>
<tr>
<td>2.0</td>
<td>2014-12-17</td>
<td>Kees Koster</td>
<td>Added on-street parking</td>
</tr>
</tbody>
</table>
CONTENTS

1. Introduction ................................................................. 4
  1.1 Context ........................................................................ 4
  1.2 Purpose of this report .................................................. 4
  1.3 Structure of this report .................................................. 5
  1.4 Reading guide ............................................................. 5
  1.5 Changes with respect to previous versions ..................... 5
    1.5.1 Version 1 ............................................................. 5
    1.5.2 Version 1.01 .......................................................... 5
    1.5.3 Version 2 ............................................................. 6
  1.6 References .................................................................... 6
  1.7 Acknowledgements ....................................................... 6

2. Overall view ................................................................. 7

3. Symbols and Abbreviations ............................................ 9

4. Basic technology choices .............................................. 10
  4.1 Transfer protocol ......................................................... 10
  4.2 Data format ................................................................... 10
  4.3 Roles ............................................................................ 10
  4.4 Authentication ............................................................ 10
  4.5 Identification ............................................................... 10
    4.5.1 Non-random UUID’s ................................................. 10
    4.5.2 ASCII Codes .......................................................... 11
  4.6 Versioning .................................................................... 11

5. Domain model ............................................................... 12
  5.1 Complex data types ..................................................... 12
    5.1.1 DateTime ................................................................ 12
    5.1.2 Polygon ................................................................. 12
    5.1.3 Time .................................................................... 12
    5.1.4 TimeType .............................................................. 12
  5.2 Static data ................................................................. 13
    5.2.1 AccessPoint .......................................................... 14
    5.2.2 Address .................................................................. 15
    5.2.3 ContactPerson ....................................................... 16
    5.2.4 EntryTime ............................................................ 17
    5.2.5 ExitTime ............................................................... 18
    5.2.6 FacilityPaymentMethod .......................................... 19
    5.2.7 IntervalRate .......................................................... 19
    5.2.8 Location ............................................................... 20
    5.2.9 OpeningTime ......................................................... 21
    5.2.10 Operator .............................................................. 22
    5.2.11 ParkingFacilityInformation .................................... 23
    5.2.12 Restriction ............................................................ 24
    5.2.13 SellingPoint ........................................................ 24
    5.2.14 SellingPointPaymentMethod .................................. 25
5.2.15 SpecialDay .......................................................... 30
5.2.16 Specifications ...................................................... 30
5.2.17 Tariff ................................................................. 31
5.2.18 ValidityExtension .................................................. 33
5.2.19 ValidityExtensionRestriction ................................... 34
5.3 Dynamic data ................................................................ 34
5.3.1 ActualStatus .......................................................... 35
5.3.2 ParkingFacilityDynamicInformation ............................ 35
5.4 Index data .................................................................. 36
5.4.1 Location .................................................................. 36
5.4.2 ParkingIndexEntry .................................................. 37

6. Mapping data to JSON .................................................. 39

7. Push protocol ............................................................... 40
7.1 Pushing static data ..................................................... 40
7.1.1 Client request ........................................................ 40
7.1.2 Server response ..................................................... 43
7.2 Pushing dynamic data .................................................. 43
7.2.1 Client push message .............................................. 43
7.2.2 Server response ..................................................... 44

8. Pull protocol ............................................................... 45
8.1 Pulling a list of known parking facilities ......................... 45
8.1.1 Client request ........................................................ 45
8.1.2 Server response ..................................................... 45
8.2 Pulling the static data of a parking facility ...................... 46
8.2.1 Client request ........................................................ 46
8.2.2 Server response ..................................................... 46
8.3 Pulling the dynamic data of a parking facility ................. 46
8.3.1 Client request ........................................................ 46
8.3.2 Server response ..................................................... 47

9. Licensing ................................................................. 48
1. INTRODUCTION

1.1 Context

This document has been prepared by the “Werkgroep Standaard Publicatie Dynamische Parkeerdata”. This specification defines a standard for the publication of dynamic parking data in enclosed parking sites and on-street parking.

The standard has been created for the Dutch market by parking technology providers and parking management companies. The working group was initiated by the Netherlands’ Ministry of Infrastructure and the Environment. The working group consisted of representatives of the following organizations:

- P1 (www.p1.nl)
- Q-Park (www.q-park.nl)
- SKIDATA (www.skidata.nl)
- Technolution (www.technolution.eu)
- Schmit (www.schmit.nl)
- Vialis (www.vialis.nl)
- Imtech (www.imtech.nl)
- RDW (www.rdw.nl)
- Netherlands’ Ministry of Infrastructure and the Environment (www.rijksoverheid.nl/ministeries/ienm)

The standard is published by the Netherlands’ Ministry of Infrastructure and the Environment on the website for the responsible programme:

- http://www.beterbenutten.nl/
  - Parking data can be found in Theme ’ITS’, topic ’Slimmer reizen met ITS’, subtopic ’Open Parkeerdata’.
- https://www.parkeerdatacatalogus.nl
  - On the above location, the index server for the parking data is hosted.

1.2 Purpose of this report

The purpose of the publication of dynamic parking data is to provide information about available parking spaces both in enclosed parking facilities (controlled by barriers) and for on-street parking.

For an accurate overview, some (semi-)static data about the facilities must also be provided. The main purpose of the publication is to provide dynamic data on the actual availability of parking spaces for visitors. Visitors are people accessing the parking facility who are not subscription holders.

An enclosed public parking facility is to be defined as parking spaces, to be accessed by visitors, passing the entry of the facility, at one common set of restrictions/rates and capacity control. Two separated sections with different restrictions and/or rates and capacity control are to be specified as two parking facilities. For example a facility with a lower deck with height limitations and an upper deck without height limitation should be specified as two facilities.

On street parking facilities differ from enclosed public parking facilities. On street parking requires an accurate description of the parking area using polygons. Conceptually, entries and exits to the facilities are less relevant for on street parking.
This technical specification specifies components required for the exchange and use of data in the field of parking. The components include the following:

- Domain model: Data structure and relationships;
- Data content (or data dictionary)
- Communications specification (or communication protocol)
- Licensing for the use of the dynamic parking data

1.3 Structure of this report

This document describes two protocols in one:

1. A protocol to get parking data from parking management systems (PMS’s) to central parking data servers (CPDS’s), called the ‘push protocol’;
2. A protocol to get parking data from CPDS’s to website and app developers, called the ‘pull protocol’.

The standard uses a domain model, which has been based on two sources:

- PRIS domain model
- Draft DATEX II standard for parking information

The domain model is documented in Sparx Systems’ Enterprise Architect. Therefore, the domain model uses XML types. The mapping of XML types to JSON is described in Chapter 6.

1.4 Reading guide

Examples of messages are formatted in a fixed width font, e.g.

```
PUT /parkingdata/v2/dynamic/<UUID>/ HTTP/1.1
Content-Type: application/json
Content-Length: ...
{
  ...
}
```

Within these blocks text marked with < and > and triple dots (...) are entries meant to be replaced by the producing software.

1.5 Changes with respect to previous versions

1.5.1 Version 1

The initial release of the standard was version 1. This was released in January 2014.

1.5.2 Version 1.01

Version 1.01 aimed to correct inconsistencies in the initial release – specifically chapter 2 with the view on the use of the standard and corrections in the JSON examples.

Property ‘durationUntil’ of the IntervalRate (paragraph 5.2.7) gained a new default value (-1 to signify unlimited instead of 99999).
### 1.5.3 Version 2

This version adds properties to the domain model that extend its functionality to on street parking. These extensions are not required properties. The extensions focus on providing (semi-)static data that are collected for the Dutch 'Nationaal Parkeer Register'. Examples are:

- the areaGeometry which describes the area of the on-street parking facility;
- parking extensions and restrictions;
- the concept of sellingPoints (parking ticket machines) and associated payment methods.

Properties EntryTime and Tariff were changed. Both properties contained the weekDay, which was a complex type restricted to the following values: “Mon”, “Tue”, “Wed”, “Thu”, “Fri”, “Sat”, “Sun”. WeekDay has been replaced by dayNames, which is a list of strings defining either the day of the week or a special day as defined by the specialDay property.

It is recommended that in dayNames the following strings are used for the days of the week:

- "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun".

If any other string than one of the above is used, this indicates a SpecialDay, which is expected to be specified in the list of specialDays for the facility.

WeekDay has been removed from the model.

Many properties allow the setting of the start of the validity and the end of the validity of the property. This enables the publisher of the data to provide information that is not the actual situation, but e.g. valid in the future.

Many changes have been made in the standard between versions 1 and 2, but in the static domain model only. In the dynamic domain model, no changes have been made in the model.

### 1.6 References

The standard references the following external sources:

<table>
<thead>
<tr>
<th>Name</th>
<th>Referenced location</th>
</tr>
</thead>
<tbody>
<tr>
<td>GeoJSON</td>
<td><a href="http://geojson.org/geojson-spec.html">http://geojson.org/geojson-spec.html</a></td>
</tr>
<tr>
<td>DATEX II</td>
<td><a href="http://www.datex2.eu">http://www.datex2.eu</a></td>
</tr>
<tr>
<td>DateTime</td>
<td><a href="http://en.wikipedia.org/wiki/Unix_time">http://en.wikipedia.org/wiki/Unix_time</a></td>
</tr>
<tr>
<td>dayNames &amp; validityDays</td>
<td><a href="https://www.ietf.org/rfc/rfc0822.txt">https://www.ietf.org/rfc/rfc0822.txt</a></td>
</tr>
<tr>
<td>Index server</td>
<td><a href="https://www.parkeerdatalocaties.nl">https://www.parkeerdatalocaties.nl</a></td>
</tr>
</tbody>
</table>

### 1.7 Acknowledgements

The standard was prepared with the valuable input of all members of the working group.

The contribution of Leontien Balkenende of Technolution in preparing the UML diagrams of the chosen domain model is gratefully acknowledged.

---

1 Names in the list have been chosen in accordance with the HTTP standard (see IETF RFC 0822, Standard for ARPA Internet Text Messages, paragraph 5.1).
2. **OVERALL VIEW**

There are several options to use the static and dynamic parking data. A generic overview of the data flows from the parking facilities to the end users is shown below in Figure 1.

![Diagram of data flows](image)

**Figure 1. Generic overview of data flows and interfaces of parking data**

The process of publishing parking data as shown in Figure 1 is the following:

- A parking facility publishes two kinds of data, viz.
  - Metadata, i.e. data describing what actual data is published and, importantly also, where the parking data will be published (the URL to the parking data);
  - Parking data, i.e. static or dynamic parking data that is published using the protocol and model as described in this standard.

- The Index server is the (web)server that publishes the metadata of all parking facilities that publish parking data. Through the index server end-users can find where parking facilities are located and on which location (URL) on the Internet the parking data can be found. The Netherlands’ Ministry of Infrastructure and Environment has taken the responsibility to create the Index server. The index server publishes the metadata of parking facilities as described in paragraph 5.4 of this standard. The index server is hosted at [https://www.parkeerdatacatalogus.nl/](https://www.parkeerdatacatalogus.nl/);

- A Central Parking Data Server publishes parking data using the protocol as defined in this standard for static and dynamic parking data (paragraphs 5.2 and 5.2.2 respectively). A CDPS can publish the parking data of one single parking facility or of any number of parking facilities.
This depends on the operator of the facility and how the business processes of the operator are organized;

- The User utilizes the Index Server to find out where parking data can be found on the Internet. The index server returns metadata about parking facilities, amongst other the URL of the CDPS, based on which the end-user can query one or more Central Parking Data Servers. A User could be an app that implements the parking data protocol and that queries CDPS’s for actual parking data. The preferred method for any app builder would be to create a web client under his or her own control, that queries the existing CDPS’s and that serves as the feed of parking data of his/her mobile app. In cases where the user must agree to a license for the use of the parking data, this is probably the only viable option.
3. SYMBOLS AND ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPDS</td>
<td>Central Parking Data Server</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>NPR</td>
<td>“Nationaal Parkeer Register”</td>
</tr>
<tr>
<td>PMS</td>
<td>Parking Management System</td>
</tr>
<tr>
<td>PRIS</td>
<td>Parking Route Information System</td>
</tr>
<tr>
<td>SPDP</td>
<td>“Standaard voor Publicatie Dynamische Parkeerdata”</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Sockets Layer</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>UUID</td>
<td>Universally Unique Identifier</td>
</tr>
</tbody>
</table>
4. BASIC TECHNOLOGY CHOICES

4.1 Transfer protocol
All data will be transferred using an HTTP connection. To query (‘pull’) data, clients execute an HTTP GET request. To push data, clients execute an HTTP PUT request.

4.2 Data format
The content type for all transferred data is ‘application/json’.

4.3 Roles
In the push protocol the PMS is an HTTP client, CPDS is an HTTP server. In the pull protocol the website or app is an HTTP client, CPDS is an HTTP server.

4.4 Authentication
To authenticate clients to servers (can be used for both push and pull), the connection can be secured using SSL. The client is identified and authenticated using HTTP basic authentication.

HTTP basic authentication uses a combination of username and password, which, in this case, is sent through an encrypted connection. It is advisable that the CPDS system enforces the use of strong passwords.

4.5 Identification
Names of parking facilities are not globally unique. Still, pieces of data need to be matched to other data using a form of unique identification. A central parking data server needs to match incoming dynamic data with the already known static data. This protocol chooses to use a UUID\(^2\) to identify parking facilities.

In absence of a central registration party, the source, a PMS, chooses a random UUID for a parking facility once and uses this UUID in all communications thereafter. Using a UUID the chances of two parking facilities having the same UUID are negligible.

If a central registration party is present, this party can provide a UUID for a parking facility or simply validate that there are no UUID clashes.\(^3\)

All popular programming languages support choosing a random UUID.

4.5.1 Non-random UUID’s
Parking facilities not wishing to use a random UUID, may do so by using a scheme that results in a unique identifier.

For this purpose, the scheme that is explained below is recommended.

A UUID is represented in the canonical form as xxxxxxxxx-xxxx-Mxx-Nxx-xxxxxxxxxxx.

---


\(^3\) UUID must be delivered by the parking operator. In NPR it is delivered by the Cities.
For the purpose of the standard, the first two sequences will consist of the zip code of the operator represented by the ASCII code in hexadecimal format. E.g. the zip code “2597JG” will be represented by “32353937-4A47”.

The next two sequences will consist of 4 letters of the name of the operator, also represented by the ASCII code in hexadecimal format. E.g. the name “IenM” will be represented by “4965-6E4D”. If the operator name is shorter than 4 characters, leading hexadecimal code(s) 00 will be used to fill the sequence. I.e. “P1” would be represented by “0000-5031”.

The last remaining sequence will consist of a per operator unique number of the parking facility (12 positions available).

For the IenM operator above, a valid parking facility identifier could be the following UUID: “32353937-4A47-4965-6E4D-000000000001”.

### 4.5.2 ASCII Codes

<table>
<thead>
<tr>
<th>Hex</th>
<th>Sign</th>
<th>Hex</th>
<th>Sign</th>
<th>Hex</th>
<th>Sign</th>
<th>Hex</th>
<th>Sign</th>
<th>Hex</th>
<th>Sign</th>
<th>Hex</th>
<th>Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>SP</td>
<td>30</td>
<td>0</td>
<td>40</td>
<td>@</td>
<td>50</td>
<td>P</td>
<td>60</td>
<td>`</td>
<td>70</td>
<td>p</td>
</tr>
<tr>
<td>21</td>
<td>!</td>
<td>31</td>
<td>1</td>
<td>41</td>
<td>A</td>
<td>51</td>
<td>Q</td>
<td>61</td>
<td>a</td>
<td>71</td>
<td>q</td>
</tr>
<tr>
<td>22</td>
<td>&quot;</td>
<td>32</td>
<td>2</td>
<td>42</td>
<td>B</td>
<td>52</td>
<td>R</td>
<td>62</td>
<td>b</td>
<td>72</td>
<td>r</td>
</tr>
<tr>
<td>23</td>
<td>#</td>
<td>33</td>
<td>3</td>
<td>43</td>
<td>C</td>
<td>53</td>
<td>S</td>
<td>63</td>
<td>c</td>
<td>73</td>
<td>s</td>
</tr>
<tr>
<td>24</td>
<td>$</td>
<td>34</td>
<td>4</td>
<td>44</td>
<td>D</td>
<td>54</td>
<td>T</td>
<td>64</td>
<td>d</td>
<td>74</td>
<td>t</td>
</tr>
<tr>
<td>25</td>
<td>%</td>
<td>35</td>
<td>5</td>
<td>45</td>
<td>E</td>
<td>55</td>
<td>U</td>
<td>65</td>
<td>e</td>
<td>75</td>
<td>u</td>
</tr>
<tr>
<td>26</td>
<td>&amp;</td>
<td>36</td>
<td>6</td>
<td>46</td>
<td>F</td>
<td>56</td>
<td>V</td>
<td>66</td>
<td>f</td>
<td>76</td>
<td>v</td>
</tr>
<tr>
<td>27</td>
<td>'</td>
<td>37</td>
<td>7</td>
<td>47</td>
<td>G</td>
<td>57</td>
<td>W</td>
<td>67</td>
<td>g</td>
<td>77</td>
<td>w</td>
</tr>
<tr>
<td>28</td>
<td>(</td>
<td>38</td>
<td>8</td>
<td>48</td>
<td>H</td>
<td>58</td>
<td>X</td>
<td>68</td>
<td>h</td>
<td>78</td>
<td>x</td>
</tr>
<tr>
<td>29</td>
<td>)</td>
<td>39</td>
<td>9</td>
<td>49</td>
<td>I</td>
<td>59</td>
<td>Y</td>
<td>69</td>
<td>i</td>
<td>79</td>
<td>y</td>
</tr>
<tr>
<td>2A</td>
<td>*</td>
<td>3A</td>
<td>:</td>
<td>4A</td>
<td>J</td>
<td>5A</td>
<td>Z</td>
<td>6A</td>
<td>j</td>
<td>7A</td>
<td>z</td>
</tr>
<tr>
<td>2B</td>
<td>+</td>
<td>3B</td>
<td>;</td>
<td>4B</td>
<td>K</td>
<td>5B</td>
<td>[</td>
<td>6B</td>
<td>k</td>
<td>7B</td>
<td>{</td>
</tr>
<tr>
<td>2C</td>
<td>,</td>
<td>3C</td>
<td>&lt;</td>
<td>4C</td>
<td>L</td>
<td>5C</td>
<td>\</td>
<td>6C</td>
<td>l</td>
<td>7C</td>
<td></td>
</tr>
<tr>
<td>2D</td>
<td>-</td>
<td>3D</td>
<td>=</td>
<td>4D</td>
<td>M</td>
<td>5D</td>
<td>)</td>
<td>6D</td>
<td>m</td>
<td>7D</td>
<td>}</td>
</tr>
<tr>
<td>2E</td>
<td>.</td>
<td>3E</td>
<td>&gt;</td>
<td>4E</td>
<td>N</td>
<td>5E</td>
<td>^</td>
<td>6E</td>
<td>n</td>
<td>7E</td>
<td>~</td>
</tr>
<tr>
<td>2F</td>
<td>/</td>
<td>3F</td>
<td>?</td>
<td>4F</td>
<td>O</td>
<td>5F</td>
<td>_</td>
<td>6F</td>
<td>o</td>
<td>7F</td>
<td>DEL</td>
</tr>
</tbody>
</table>

### 4.6 Versioning

The URL’s for all HTTP requests contain a version number to identify the version of the protocol used. This document describes version 2 of the protocol. URL’s contain ‘v2’ in their paths.

-/v2/dynamic/<UUID>/
-/v2/static/<UUID>/
5. **DOMAIN MODEL**

This chapter describes the domain model for transferring static and dynamic parking facility data. UML diagrams are used to show the structure and details of the model.

Optional attributes and relations are marked with [0..1]. Attributes and relations that can have more than one value are marked with [0..*].

5.1 **Complex data types**

This chapter using some specific complex data types. They are explained below.

5.1.1 **DateTime**

A data type representing the number of seconds since Unix epoch.\(^4\)

5.1.2 **Polygon**

Data type representing polygons for describing areas. Polygons must be built-up as specified in the GeoJSON specifications.

5.1.3 **Time**

Object containing the hour, minutes and seconds on any day.

<table>
<thead>
<tr>
<th>ATTRIBUTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>h: int</td>
</tr>
<tr>
<td>Constraints:</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>m: int</td>
</tr>
<tr>
<td>Constraints:</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>s: int</td>
</tr>
<tr>
<td>Constraints:</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

5.1.4 **TimeType**

Enumeration of strings, specifying a time interval.

<table>
<thead>
<tr>
<th>ATTRIBUTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Days”</td>
</tr>
<tr>
<td>“Hours”</td>
</tr>
<tr>
<td>“Minutes”</td>
</tr>
<tr>
<td>“Seconds”</td>
</tr>
<tr>
<td>“Weeks”</td>
</tr>
</tbody>
</table>

5.2 Static data

Figure 2. Static Data
### 5.2.1 AccessPoint

Specification of parking facility access point (entry and/or exit) information.

<table>
<thead>
<tr>
<th>ATTRIBUTES</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validityStartOfPeriod: DateTime</td>
<td>Indicates the start of the validity period of the access point. No value or null means unlimited.</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
<td></td>
</tr>
<tr>
<td>validityEndOfPeriod: DateTime</td>
<td>Indicates the end of the validity period of the access point. No value or null means unlimited.</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
<td></td>
</tr>
<tr>
<td>alias: string</td>
<td>Name of the access point</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
<td></td>
</tr>
<tr>
<td>isVehicleExit: boolean</td>
<td>Indicates whether it is possible for vehicles to exit the parking facility via the access point</td>
</tr>
<tr>
<td>Multiplicity: ([1..1])</td>
<td></td>
</tr>
<tr>
<td>isVehicleEntrance: boolean</td>
<td>Indicates whether it is possible for vehicles to enter the parking facility via the access point</td>
</tr>
<tr>
<td>Multiplicity: ([1..1])</td>
<td></td>
</tr>
<tr>
<td>isPedestrianExit: boolean</td>
<td>Indicates whether it is possible for pedestrians to exit the parking facility via the access point</td>
</tr>
<tr>
<td>Multiplicity: ([1..1])</td>
<td></td>
</tr>
<tr>
<td>isPedestrianEntrance: boolean</td>
<td>Indicates whether it is possible for pedestrians to enter the parking facility via the access point</td>
</tr>
<tr>
<td>Multiplicity: ([1..1])</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ASSOCIATIONS</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association (direction: Source -&gt; Destination) accessPointAddress</td>
<td>The address of an AccessPoint (Entry/Exit) of a parking facility</td>
</tr>
<tr>
<td>Source: (Class) AccessPoint</td>
<td>Target: (Class) Address</td>
</tr>
<tr>
<td>Cardinality: [1]</td>
<td></td>
</tr>
<tr>
<td>Association (direction: Source -&gt; Destination) accessPointLocation</td>
<td>The location of an AccessPoint. Because the accessPointLocation has a validity, it is possible in the domain model for an AccessPoint to have multiple locations. It is the</td>
</tr>
</tbody>
</table>
ASSOCIATIONS

<table>
<thead>
<tr>
<th>Source: (Class) AccessPoint</th>
<th>Target: (Class) Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cardinality: [0..*]</td>
<td></td>
</tr>
</tbody>
</table>

Association (direction: Source -> Destination) accessPoints
List of access points (entrances and/or exits) to the parking facility.
Source: (Class) ParkingFacilityInformation Target: (Class) AccessPoint
Cardinality: [0..*]

5.2.2 Address
Specification of address information

<table>
<thead>
<tr>
<th>ATTRIBUTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>country: string</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>Country of the address</td>
</tr>
</tbody>
</table>

| province: string |
| Multiplicity: ([0..1]) |
| Province of the address |

| city: string |
| Multiplicity: ([1..1]) |
| City of the address |

| streetName: string |
| Multiplicity: ([1..1]) |
| Street name of address (for postal address this could also be PO Box or Postbus) |

| houseNumber: string |
| Multiplicity: ([0..1]) |
| House number of address (for postal address this could also be the PO Box or Postbus number) |

| zipcode: string |
| Multiplicity: ([0..1]) |
| Zip code of the address |

| emailAddresses: string |
| Multiplicity: ([0..*]) |
| List of E-mail addresses linked to the specific address |
### ATTRIBUTES

- **phoneNumber**
  - Type: string
  - Multiplicity: ([0..*])
  - List of phone numbers of the specific address

### ASSOCIATIONS

- **administrativeAddresses**
  - Association (direction: Source -> Destination)
  - List of administrative addresses of the operator of the parking facility
  - Source: (Class) Operator
  - Target: (Class) Address
  - Cardinality: [0..*]

- **postalAddress**
  - Association (direction: Source -> Destination)
  - Postal address of the operator of the parking facility. There can be only one postal address.
  - Source: (Class) Operator
  - Target: (Class) Address
  - Cardinality: [0..1]

- **accessPointAddress**
  - Association (direction: Source -> Destination)
  - The address of an AccessPoint (Entry/Exit) of a parking facility
  - Source: (Class) AccessPoint
  - Target: (Class) Address
  - Cardinality: [1]

### 5.2.3 ContactPerson

Information about the contact person of a parking facility

### ATTRIBUTES

- **validityStartOfPeriod**
  - Type: DateTime
  - Multiplicity: ([0..1])
  - Indicates the start of the validity period of the contact person. No value or null means unlimited.

- **validityEndOfPeriod**
  - Type: DateTime
  - Multiplicity: ([0..1])
  - Indicates the end of the validity period of the contact person. No value or null means unlimited.

- **firstName**
  - Type: string
  - Multiplicity: ([0..1])
  - First name of the contact person

- **name**
  - Type: string
  - Multiplicity: ([1..1])
  - Name of the contact person
**Attributes**

- **position**: string
  - Multiplicity: \([0..1]\)
  - Function of the contact person

- **emailAddress**: string
  - Multiplicity: \([0..1]\)
  - E-mail address of the contact person

- **faxNumber**: string
  - Multiplicity: \([0..1]\)
  - Fax number of the contact person

- **phoneNumber**: string
  - Multiplicity: \([0..1]\)
  - Telephone number of the contact person

- **public**: boolean
  - Multiplicity: \([0..1]\)
  - Indicates whether the contact details of the contact person can be made public or not

- **availableAllDay**: boolean = false
  - Multiplicity: \([0..1]\)
  - This property indicates whether or not the ContactPerson can be contacted 24 hrs per day. By default this is false.

**Associations**

- **Association (direction: Source -> Destination) emergencyContacts**
  - List of emergency contacts for the facility. Generally, the emergency contacts can be contacted 24 hrs per day, for real emergencies or for access to the facility outside opening hours.
  - Source: (Class) ParkingFacilityInformation
  - Target: (Class) ContactPerson
  - Cardinality: \([0..*]\)

- **Association (direction: Source -> Destination) contactPersons**
  - List of contact persons for the parking facility.
  - Source: (Class) ParkingFacilityInformation
  - Target: (Class) ContactPerson
  - Cardinality: \([0..*]\)

**5.2.4 EntryTime**

Specification of entry time(s) of a parking facility

**Attributes**

- **validityStartOfPeriod**: DateTime
ATTRIBUTES

Multiplicity: ([0..1])
Indicates the start of the validity period of the entry time. No value or null means unlimited.

validityEndOfPeriod: DateTime
Multiplicity: ([0..1])
Indicates the end of the validity period of the entry time. No value or null means unlimited.

dayNames: string
Multiplicity: ([1..*])
A list of the days of the week that the entry time is valid. This could be any of the days of the week or a specialDay.

It is recommended that the following strings are used for the days of the week:

- “Mon”, “Tue”, “Wed”, “Thu”, “Fri”, “Sat”, “Sun”.\(^5\)

If any other string than one of the above is used, this indicates a SpecialDay, which is expected to be specified in the list of specialDays for the facility.

enterFrom: Time
Multiplicity: ([0..1])
Start time (of day) from which it is possible to enter the parking facility

enterUntil: Time
Multiplicity: ([0..1])
End time (of day) until which it is possible to enter the parking facility

ASSOCIATIONS

Association (direction: Source -> Destination) entryTimes
List of entry times

Source: (Class) OpeningTime
Target: (Class) EntryTime
Cardinality: [0..*]

5.2.5 ExitTime
Specification of exit time(s) of a parking facility

ATTRIBUTES

validityStartOfPeriod: DateTime
Multiplicity: ([0..1])
Indicates the start of the validity period of the exit time. No value or null means unlimited.

\(^5\) Names in the list have been chosen in accordance with the HTTP standard (see IETF RFC 0822, Standard for ARPA Internet Text Messages, paragraph 5.1).
## ATTRIBUTES

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Data Type</th>
<th>Multiplicity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validityEndOfPeriod</td>
<td>DateTime</td>
<td>([0..1])</td>
<td>Indicates the end of the validity period of the exit time. No value or null means unlimited.</td>
</tr>
<tr>
<td>dayNames</td>
<td>string</td>
<td>([1..*])</td>
<td>A list of the days of the week that the exit time is valid. This could be any of the days of the week or a specialDay. It is recommended that the following strings are used for the days of the week:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• “Mon”, “Tue”, “Wed”, “Thu”, “Fri”, “Sat”, “Sun”⁶</td>
</tr>
<tr>
<td>exitFrom</td>
<td>Time</td>
<td>([0..1])</td>
<td>Start time (of day) from which it is possible to exit from the parking facility</td>
</tr>
<tr>
<td>exitUntil</td>
<td>Time</td>
<td>([0..1])</td>
<td>End time (of day) from which it is possible to exit from the parking facility</td>
</tr>
</tbody>
</table>

## ASSOCIATIONS

<table>
<thead>
<tr>
<th>Association</th>
<th>Direction</th>
<th>Source</th>
<th>Target</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association (direction: Source -&gt; Destination) exitTimes</td>
<td></td>
<td>(Class) OpeningTime</td>
<td>(Class) ExitTime</td>
<td>[0..*]</td>
</tr>
</tbody>
</table>

### 5.2.6 FacilityPaymentMethod

Specification which methods can be used to pay at the parking facility

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Data Type</th>
<th>Multiplicity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validityStartOfPeriod</td>
<td>DateTime</td>
<td>([0..1])</td>
<td>Indicates the start of the validity period of the payment method. No value or null means unlimited.</td>
</tr>
<tr>
<td>validityEndOfPeriod</td>
<td>DateTime</td>
<td>([0..1])</td>
<td>Indicates the end of the validity period of the payment method. No value or null means unlimited.</td>
</tr>
</tbody>
</table>

⁶ Names in the list have been chosen in accordance with the HTTP standard (see IETF RFC 0822, Standard for ARPA Internet Text Messages, paragraph 5.1).
**ATTRIBUTES**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
</table>
| unlimited | Method: string  
Multiplicity: ([1..1])  
Description of the payment method  
The following standard payment methods should at the minimum be recognized: |
| - "Coins",  
- "Banknotes",  
- "Maestro",  
- "VPay",  
- "MasterCard",  
- "AMEX",  
- "Visa" |
| It is allowed to use other payment method names, however these might not be recognized by users of the data. |
| atPayStation: boolean  
Multiplicity: ([1..1])  
Indicates whether this payment method is available at the pay stations |
| atExit: boolean  
Multiplicity: ([1..1])  
Indicates whether this payment method is available at the exits |

**ASSOCIATIONS**

Association (direction: Source -> Destination) paymentMethods  
List of accepted payment methods  
Source: (Class) ParkingFacilityInformation  
Target: (Class) FacilityPaymentMethod  
Cardinality: [0..*]

**5.2.7 IntervalRate**

Specification of the rate per interval for the parking facility

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
</table>
| validityStartOfPeriod: DateTime  
Multiplicity: ([0..1])  
Indicates the start of the validity period of the interval rate. No value or null means unlimited. |
| validityEndOfPeriod: DateTime  
Multiplicity: ([0..1])  
Indicates the end of the validity period of the interval rate. No value or null means unlimited. |
### ATTRIBUTES

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Multiplicity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>charge</td>
<td>float</td>
<td>([1..1])</td>
<td>Rate in euros for the interval</td>
</tr>
<tr>
<td>chargePeriod</td>
<td>integer</td>
<td>([1..1])</td>
<td>Specification of the length of the period for the given charge</td>
</tr>
<tr>
<td>durationFrom</td>
<td>integer</td>
<td>([1..1])</td>
<td>Interval start indicator (minimum 0)</td>
</tr>
<tr>
<td>durationUntil</td>
<td>integer</td>
<td>([1..1])</td>
<td>Interval end indicator (-1 means unlimited)</td>
</tr>
<tr>
<td>durationType</td>
<td>TimeType</td>
<td>([0..1])</td>
<td>Time specification for this interval, of type TimeType (i.e. could be any of “Seconds”, “Minutes”, “Hours”, “Days” or “Weeks”)</td>
</tr>
</tbody>
</table>

### ASSOCIATIONS

<table>
<thead>
<tr>
<th>Association</th>
<th>Source: (Class)</th>
<th>Target: (Class)</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association (direction: Source -&gt; Destination) intervalRates</td>
<td>Tariff</td>
<td>IntervalRate</td>
<td>[1..*]</td>
</tr>
</tbody>
</table>

### 5.2.8 Location

Specification of location data

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Multiplicity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validityStartOfPeriod</td>
<td>DateTime</td>
<td>([1..1])</td>
<td>Indicates the start of the validity period of the location. No value or null means unlimited.</td>
</tr>
<tr>
<td>validityEndOfPeriod</td>
<td>DateTime</td>
<td>([0..1])</td>
<td>Indicates the end of the validity period of the location. No value or null means unlimited.</td>
</tr>
<tr>
<td>coordinatesType</td>
<td>string = WGS84</td>
<td>([1..1])</td>
<td>Type of coordinate system, “WGS84”</td>
</tr>
</tbody>
</table>
## Attributes

- **latitude**: float
  - Multiplicity: ([1..1])
  - Latitude of coordinate in decimal degrees

- **longitude**: float
  - Multiplicity: ([1..1])
  - Longitude of coordinate in decimal degrees

## Associations

- **Association (direction: Source -> Destination) sellingPointLocation**
  - The location of a SellingPoint. Because the sellingPointLocation has a validity, it is possible in the domain model for a SellingPoint to have multiple locations. It is the responsibility of the data owner that the content of the data can be clear to the user.
  - Source: (Class) SellingPoint
  - Target: (Class) Location
  - Cardinality: [0..*]

- **Association (direction: Source -> Destination) locationForDisplay**
  - Location of the parking facility for displaying on maps
  - Source: (Class) ParkingFacilityInformation
  - Target: (Class) Location
  - Cardinality: [0..1]

- **Association (direction: Source -> Destination) accessPointLocation**
  - The location of an AccessPoint. Because the accessPointLocation has a validity, it is possible in the domain model for an AccessPoint to have multiple locations. It is the responsibility of the data owner that the content of the data can be clear to the user.
  - Source: (Class) AccessPoint
  - Target: (Class) Location
  - Cardinality: [0..*]

- **Association (direction: Source -> Destination) locationForDisplay**
  - Location of the parking facility for displaying on maps
  - Source: (Class) ParkingIndexEntry
  - Target: (Class) Location
  - Cardinality: [0..1]

### 5.2.9 OpeningTime

Specification of the opening time(s) of a parking facility

## Attributes

- **startOfPeriod**: DateTime
  - Multiplicity: ([0..1])
  - Start date and time of the opening time period. No value or null means unlimited.

- **endOfPeriod**: DateTime
ATTRIBUTES

Multiplicity: ([0..1])
End date and time of the opening time period. No value or null means unlimited.

- openAllYear: boolean
  Multiplicity: ([0..1])
  Indicates whether the parking facility is open all year

- periodName: string
  Multiplicity: ([0..1])
  Name of opening time period

- exitPossibleAllDay: boolean
  Multiplicity: ([0..1])
  Indicates whether it is possible to exit the parking facility all day. Implicitly, when it is not possible to exit all day, exiting is possible in the same times as entry (as specified in the list of entry times).

ASSOCIATIONS

- Association (direction: Source -> Destination) entryTimes
  List of entry times
  Source: (Class) OpeningTime
  Target: (Class) EntryTime
  Cardinality: [0..*]

- Association (direction: Source -> Destination) exitTimes
  List of exit times
  Source: (Class) OpeningTime
  Target: (Class) ExitTime
  Cardinality: [0..*]

- Association (direction: Source -> Destination) openingTimes
  List of availability hours to enter by car
  Source: (Class) ParkingFacilityInformation
  Target: (Class) OpeningTime
  Cardinality: [0..*]

5.2.10 Operator
Specification of operator data

ATTRIBUTES

- validityStartOfPeriod: DateTime
  Multiplicity: ([0..1])
  Indicates the start of the validity period of the operator. No value or null means unlimited.
ATTRIBUTES

- **validityEndOfPeriod**: DateTime
  
  Multiplicity: ([0..1])

  Indicates the end of the validity period of the operator. No value or null means unlimited.

- **name**: string
  
  Multiplicity: ([1..1])

  Name of the operator of the parking facility

- **url**: string
  
  Multiplicity: ([0..1])

  Web site address of the operator of the parking facility

ASSOCIATIONS

- **Association (direction: Source -> Destination) administrativeAddresses**
  
  List of administrative addresses of the operator of the parking facility

  **Source**: (Class) Operator | **Target**: (Class) Address

  **Cardinality**: [0..*]

- **Association (direction: Source -> Destination) postalAddress**
  
  Postal address of the operator of the parking facility. There can be only one postal address.

  **Source**: (Class) Operator | **Target**: (Class) Address

  **Cardinality**: [0..1]

- **Association (direction: Source -> Destination) operator**
  
  Responsible operator of the parking facility

  **Source**: (Class) ParkingFacilityInformation | **Target**: (Class) Operator

  **Cardinality**: [0..1]

5.2.11 ParkingFacilityInformation

Container for all information about a parking facility.

ATTRIBUTES

- **validityStartOfPeriod**: DateTime
  
  Multiplicity: ([0..1])

  Indicates the start of the validity period of the parking facility. No value or null means unlimited.

- **validityEndOfPeriod**: DateTime
  
  Multiplicity: ([0..1])

  Indicates the end of the validity period of the parking facility. No value or null means unlimited.
ATTRIBUTES

- **name**: string
  Multiplicity: ([1..1])
  Name of the parking facility

- **identifier**: string
  Multiplicity: ([1..1])
  UUID of the parking facility

- **description**: string
  Multiplicity: ([0..1])
  Text field containing a description of the parking facility

- **limitedAccess**: boolean
  Multiplicity: ([0..1])
  Indicates whether authentication is necessary to request data for this parking facility. This field should only be used when sending parking facility data to servers that redistribute data to others.

ASSOCIATIONS

- **Association (direction: Source -> Destination) specifications**
  Specifications and limitations of the parking facility. Specifications may have a limited validity and the data source can publish multiple specifications indicating the validity of the published data.
  Source: (Class) ParkingFacilityInformation
  Target: (Class) Specifications
  Cardinality: [0..*]

- **Association (direction: Source -> Destination) locationForDisplay**
  Location of the parking facility for displaying on maps
  Source: (Class) ParkingFacilityInformation
  Target: (Class) Location
  Cardinality: [0..1]

- **Association (direction: Source -> Destination) accessPoints**
  List of access points (entrances and/or exits) to the parking facility.
  Source: (Class) ParkingFacilityInformation
  Target: (Class) AccessPoint
  Cardinality: [0..*]

- **Association (direction: Source -> Destination) specialDays**
  List of special days applicable for the parking facility
  Source: (Class) ParkingFacilityInformation
  Target: (Class) SpecialDay
  Cardinality: [0..*]

- **Association (direction: Source -> Destination) tariffs**
  List of tariffs applicable for parking at the parking facility
<table>
<thead>
<tr>
<th>ASSOCIATIONS</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Source: (Class) ParkingFacilityInformation</td>
<td>Target: (Class) Tariff</td>
<td>Cardinality: [0..*]</td>
</tr>
<tr>
<td>Association (direction: Source -&gt; Destination) paymentMethods</td>
<td>List of accepted payment methods</td>
<td></td>
</tr>
<tr>
<td>Source: (Class) ParkingFacilityInformation</td>
<td>Target: (Class) FacilityPaymentMethod</td>
<td>Cardinality: [0..*]</td>
</tr>
<tr>
<td>Association (direction: Source -&gt; Destination) parkingRestrictions</td>
<td>List of restrictions applicable for parking at the parking facility</td>
<td></td>
</tr>
<tr>
<td>Source: (Class) ParkingFacilityInformation</td>
<td>Target: (Class) Restriction</td>
<td>Cardinality: [0..*]</td>
</tr>
<tr>
<td>Association (direction: Source -&gt; Destination) sellingPoints</td>
<td>List of points where tickets for the parking facility can be obtained</td>
<td></td>
</tr>
<tr>
<td>Source: (Class) ParkingFacilityInformation</td>
<td>Target: (Class) SellingPoint</td>
<td>Cardinality: [0..*]</td>
</tr>
<tr>
<td>Association (direction: Source -&gt; Destination) emergencyContacts</td>
<td>List of emergency contacts for the facility. Generally, the emergency contacts can be contacted 24 hrs per day, for real emergencies or for access to the facility outside opening hours.</td>
<td></td>
</tr>
<tr>
<td>Source: (Class) ParkingFacilityInformation</td>
<td>Target: (Class) ContactPerson</td>
<td>Cardinality: [0..*]</td>
</tr>
<tr>
<td>Association (direction: Source -&gt; Destination) contactPersons</td>
<td>List of contact persons for the parking facility.</td>
<td></td>
</tr>
<tr>
<td>Source: (Class) ParkingFacilityInformation</td>
<td>Target: (Class) ContactPerson</td>
<td>Cardinality: [0..*]</td>
</tr>
<tr>
<td>Association (direction: Source -&gt; Destination) openingTimes</td>
<td>List of availability hours to enter by car</td>
<td></td>
</tr>
<tr>
<td>Source: (Class) ParkingFacilityInformation</td>
<td>Target: (Class) OpeningTime</td>
<td>Cardinality: [0..*]</td>
</tr>
<tr>
<td>Association (direction: Source -&gt; Destination) operator</td>
<td>Responsible operator of the parking facility</td>
<td></td>
</tr>
<tr>
<td>Source: (Class) ParkingFacilityInformation</td>
<td>Target: (Class) Operator</td>
<td>Cardinality: [0..1]</td>
</tr>
<tr>
<td>Association (direction: Source -&gt; Destination) parkingExtensions</td>
<td>List of extensions of the right to park at this parking facility at other parking facilities</td>
<td></td>
</tr>
</tbody>
</table>
ASSOCIATIONS
Source: (Class) ParkingFacilityInformation  Target: (Class) ValidityExtension
Cardinality: [0..*]

5.2.12 Restriction
Specification of parking restrictions

<table>
<thead>
<tr>
<th>ATTRIBUTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>validityStartOfPeriod: DateTime</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>Indicates the start of the validity period of the restrictions. No value or null means unlimited.</td>
</tr>
<tr>
<td>validityEndOfPeriod: DateTime</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>Indicates the end of the validity period of the restrictions. No value or null means unlimited.</td>
</tr>
<tr>
<td>validityDays: string</td>
</tr>
<tr>
<td>Multiplicity: ([1..*])</td>
</tr>
<tr>
<td>A list of day names for which the restriction is valid. This could be any of the days of the week but also a specialDay.</td>
</tr>
<tr>
<td>It is recommended that the following strings are used for the days of the week:</td>
</tr>
<tr>
<td>• “Mon”, “Tue”, “Wed”, “Thu”, “Fri”, “Sat”, “Sun”.</td>
</tr>
<tr>
<td>If any other string than one of the above is used, this indicates a SpecialDay, which is expected to be specified in the list of specialDays for the facility.</td>
</tr>
<tr>
<td>validityFromTime: Time</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>The time of day from which the restriction is valid.</td>
</tr>
<tr>
<td>validityUntilTime: Time</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>The time of day until which the restriction is valid.</td>
</tr>
<tr>
<td>parkingProhibited: boolean</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>Indicates whether or not parking is prohibited during the validity of the restriction</td>
</tr>
<tr>
<td>durationRestriction: boolean</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>Indicates whether or not a restriction of the parking duration is applicable</td>
</tr>
</tbody>
</table>

---

7 Names in the list have been chosen in accordance with the HTTP standard (see IETF RFC 0822, Standard for ARPA Internet Text Messages, paragraph 5.1).
### ATTRIBUTES

- **maximumDuration**: long
  *Multiplicity: ([0..1])*

Indicates the number of minutes of the maximum parking duration

- **minimumParkingInteruption**: long
  *Multiplicity: ([0..1])*

Indicates the number of minutes between two subsequent times when a vehicle is parked or payment is made

### ASSOCIATIONS

- **Association (direction: Source -> Destination) parkingRestrictions**

  List of restrictions applicable for parking at the parking facility

  **Source**: (Class) ParkingFacilityInformation  
  **Target**: (Class) Restriction  
  **Cardinality**: [0..*]

#### 5.2.13 SellingPoint

A selling point is a point where the user of a parking facility can get a parking ticket and/or pay for a ticket.

### ATTRIBUTES

- **validityStartOfPeriod**: DateTime
  *Multiplicity: ([0..1])*

  Indicates the start of the validity period of the selling point. No value or null means unlimited.

- **validityEndOfPeriod**: DateTime
  *Multiplicity: ([0..1])*

  Indicates the end of the validity period of the selling point. No value or null means unlimited.

- **sellingPointID**: string
  *Multiplicity: ([1..1])*

  Identifier of the SellingPoint

- **sellingPointDescription**: string
  *Multiplicity: ([0..1])*

  A text description of the selling point

### ASSOCIATIONS

- **Association (direction: Source -> Destination) sellingPointLocation**

  The location of a SellingPoint. Because the sellingPointLocation has a validity, it is possible in the domain model for a SellingPoint to have multiple locations. It is the responsibility of
ASSOCIATIONS

the data owner that the content of the data can be clear to the user.

<table>
<thead>
<tr>
<th>Source: (Class) SellingPoint</th>
<th>Target: (Class) Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cardinality: [0..*]</td>
<td></td>
</tr>
</tbody>
</table>

Association (direction: Source -> Destination) sellingPointPaymentMethods
List of SellingPointPaymentMethods available at the selling point

<table>
<thead>
<tr>
<th>Source: (Class) SellingPoint</th>
<th>Target: (Class) SellingPointPaymentMethod</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cardinality: [0..*]</td>
<td></td>
</tr>
</tbody>
</table>

Association (direction: Source -> Destination) sellingPoints
List of points where tickets for the parking facility can be obtained

<table>
<thead>
<tr>
<th>Source: (Class) ParkingFacilityInformation</th>
<th>Target: (Class) SellingPoint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cardinality: [0..*]</td>
<td></td>
</tr>
</tbody>
</table>

5.2.14 SellingPointPaymentMethod
Payment method definition for selling points

ATTRIBUTES

<table>
<thead>
<tr>
<th>attribute</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>validityStartOfPeriod</td>
<td>DateTime</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
<td></td>
</tr>
<tr>
<td>Indicates the start of the validity period of the selling point payment information. No value or null means unlimited.</td>
<td></td>
</tr>
<tr>
<td>validityEndOfPeriod</td>
<td>DateTime</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
<td></td>
</tr>
<tr>
<td>Indicates the end of the validity period of the selling point payment information. No value or null means unlimited.</td>
<td></td>
</tr>
<tr>
<td>method</td>
<td>string</td>
</tr>
<tr>
<td>Multiplicity: ([1..1])</td>
<td></td>
</tr>
<tr>
<td>Description of the payment method</td>
<td></td>
</tr>
</tbody>
</table>

The following standard payment methods should at the minimum be recognized:

- “Coins”
- “Banknotes”
- “Maestro”
- “VPay”
- “MasterCard”
- “AMEX”
- “Visa”

It is allowed to give other payment method names, however these might not be recognized by users of the data.
ASSOCIATIONS

Association (direction: Source -> Destination) sellingPointPaymentMethods
List of SellingPointPaymentMethods available at the selling point
Source: (Class) SellingPoint        Target: (Class) SellingPointPaymentMethod
Cardinality: [0..*]

5.2.15 SpecialDay
Defines the relation between a special day (name) to a specific date

ATTRIBUTES

specialDayName: string
Multiplicity: ([1..1])
The name of the special day which is defined. E.g. “Christmas Day” or “Market Day”.

specialDayDates: DateTime
Multiplicity: ([1..*])
A list of the dates on which the special day name is applicable

ASSOCIATIONS

Association (direction: Source -> Destination) specialDays
List of special days applicable for the parking facility
Source: (Class) ParkingFacilityInformation        Target: (Class) SpecialDay
Cardinality: [0..*]

5.2.16 Specifications
Miscellaneous specifications of a parking facility

ATTRIBUTES

validityStartOfPeriod: DateTime
Multiplicity: ([0..1])
Indicates the start of the validity period of the specifications. No value or null means unlimited.

validityEndOfPeriod: DateTime
Multiplicity: ([0..1])
Indicates the end of the validity period of the specifications. No value or null means unlimited.

capacity: integer
Multiplicity: ([0..1])
Total number of available parking spaces in the parking facility
## ATTRIBUTES

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Multiplicity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chargingPointCapacity</td>
<td>integer</td>
<td>[0..1]</td>
<td>Total number of available parking spaces in the parking facility equipped with a charging point</td>
</tr>
<tr>
<td>disabledAccess</td>
<td>boolean</td>
<td>[0..1]</td>
<td>Indicates whether there is access for disabled people</td>
</tr>
<tr>
<td>minimumHeightInMeters</td>
<td>float</td>
<td>[0..1]</td>
<td>Lowest height in the facility in meters</td>
</tr>
<tr>
<td>usage</td>
<td>string</td>
<td>[0..1]</td>
<td>Indicates what type of parking facility this concerns. E.g. on-street parking, parking garage, Park&amp;Ride, etc.</td>
</tr>
<tr>
<td>areaGeometry</td>
<td>Polygon</td>
<td>[0..*]</td>
<td>An array of polygons defining the geometry of the parking facility. This is especially useful (and generally only offered) for on-street parking. Polygons must be built-up as specified in the GeoJSON specifications.</td>
</tr>
</tbody>
</table>

## ASSOCIATIONS

<table>
<thead>
<tr>
<th>Specification</th>
<th>Source Class</th>
<th>Target Class</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association</td>
<td>ParkingFacilityInformation</td>
<td>Specifications</td>
<td>[0..*]</td>
</tr>
</tbody>
</table>
**ATTRIBUTES**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Multiplicity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>endOfPeriod</td>
<td>DateTime</td>
<td>([0..1])</td>
<td>End date and time of the validity of the period for a tariff. No value or null means unlimited.</td>
</tr>
<tr>
<td>periodName</td>
<td>string</td>
<td>([0..1])</td>
<td>Name of tariff period</td>
</tr>
<tr>
<td>maximumDayCharge</td>
<td>float</td>
<td>([0..1])</td>
<td>Maximum charge for a day's parking</td>
</tr>
<tr>
<td>tariffDescription</td>
<td>string</td>
<td>([0..1])</td>
<td>Textual description of the tariff period</td>
</tr>
<tr>
<td>validityDays</td>
<td>string</td>
<td>([1..*])</td>
<td>A list of day names for which the tariff is valid. This could be any of the days of the week but also a specialDay.</td>
</tr>
</tbody>
</table>

It is recommended that the following strings are used for the days of the week:

- "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun".  

If any other string than one of the above is used, this indicates a SpecialDay, which is expected to be specified in the list of specialDays for the facility.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Multiplicity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validityFromTime</td>
<td>Time</td>
<td>([0..1])</td>
<td>Start time (of day) of the tariff</td>
</tr>
<tr>
<td>validityUntilTime</td>
<td>Time</td>
<td>([0..1])</td>
<td>End time (of day) of the tariff</td>
</tr>
</tbody>
</table>

**ASSOCIATIONS**

<table>
<thead>
<tr>
<th>Association</th>
<th>Source</th>
<th>Target</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association (direction: Source -&gt; Destination) intervalRates</td>
<td>(Class) Tariff</td>
<td>(Class) IntervalRate</td>
<td>[1..*]</td>
</tr>
</tbody>
</table>

---

8 Names in the list have been chosen in accordance with the HTTP standard (see IETF RFC 0822, Standard for ARPA Internet Text Messages, paragraph 5.1).
ASSOCIATIONS

Association (direction: Source -> Destination) tariffs
List of tariffs applicable for parking at the parking facility
Source: (Class) ParkingFacilityInformation Target: (Class) Tariff
Cardinality: [0..*]

5.2.18 ValidityExtension
Specification of extensions of parking rights to other parking facilities from a parking facility.

ATTRIBUTES

validityStartOfPeriod: DateTime
Multiplicity: ([0..1])
Indicates the start of the validity period of the extension. No value or null means unlimited.

validityEndOfPeriod: DateTime
Multiplicity: ([0..1])
Indicates the end of the validity period of the extension. No value or null means unlimited.

extensionParkingFacilityIdentifier: string
Multiplicity: ([0..1])
The identifier of the parking facility to which the parking is extended

extensionParkingFacilityOperator: string
Multiplicity: ([0..1])
The operator of the parking facility to which the parking is extended

extensionUsage: string
Multiplicity: ([0..1])
Indicates what type of parking facility this extension refers to. E.g. on-street parking, parking garage, Park&Ride, etc.

ASSOCIATIONS

Association (direction: Source -> Destination) validityRestrictionOnExtension
List of restrictions to the extension of parking rights.
Source: (Class) ValidityExtension Target: (Class) ValidityExtensionRestriction
Cardinality: [0..*]

Association (direction: Source -> Destination) parkingExtensions
List of extensions of the right to park at this parking facility at other parking facilities
Source: (Class) ParkingFacilityInformation Target: (Class) ValidityExtension
Cardinality: [0..*]
5.2.19 **ValidityExtensionRestriction**  
Defines the restrictions applicable to parking extensions.

<table>
<thead>
<tr>
<th>ATTRIBUTES</th>
</tr>
</thead>
</table>
| restrictionDayNames: string  
Multiplicity: ([0..*]) |
| An list of the days that the restriction of the extension is valid. This could be a day of the week or a specialDay.  
It is recommended that the following strings are used for the days of the week:  
If any other string than one of the above is used, this indicates a SpecialDay, which is expected to be specified in the list of specialDays for the facility. |
| restrictionFromTime: Time  
Multiplicity: ([0..1])  
The time from which the restriction of the extension is valid. |
| restrictionUntilTime: Time  
Multiplicity: ([0..1])  
The time until which the restriction of the extension is valid. |

<table>
<thead>
<tr>
<th>ASSOCIATIONS</th>
</tr>
</thead>
</table>
| List of restrictions to the extension of parking rights.  
Source: (Class) ValidityExtension  
Target: (Class) ValidityExtensionRestriction  
Cardinality: [0..*] |

5.3 **Dynamic data**

![Figure 3. Dynamic data](image)

[^9]: Names in the list have been chosen in accordance with the HTTP standard (see IETF RFC 0822, Standard for ARPA Internet Text Messages, paragraph 5.1).
### 5.3.1 ActualStatus
Specification of the actual status of a parking facility.

<table>
<thead>
<tr>
<th>ATTRIBUTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>![chargePointVacantSpaces: integer]</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>Number of vacant parking spaces with a charge point</td>
</tr>
<tr>
<td>![full: boolean]</td>
</tr>
<tr>
<td>Multiplicity: ([1..1])</td>
</tr>
<tr>
<td>Indicates whether the facility currently has vacant spaces for visitors or not</td>
</tr>
<tr>
<td>![lastUpdated: DateTime]</td>
</tr>
<tr>
<td>Multiplicity: ([1..1])</td>
</tr>
<tr>
<td>Timestamp of last update</td>
</tr>
<tr>
<td>![open: boolean]</td>
</tr>
<tr>
<td>Multiplicity: ([1..1])</td>
</tr>
<tr>
<td>Indicates whether the facility is currently open for visitors</td>
</tr>
<tr>
<td>![parkingCapacity: integer]</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>Number of parking spaces for area, facility of assigned parking. This capacity may be dynamic, e.g. when a dynamic allocation is made for subscription holders. This number therefore signifies the total number of parking spaces (occupied and non-occupied) for visitors.</td>
</tr>
<tr>
<td>![statusDescription: string]</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>Textual explanation for the current actual status</td>
</tr>
<tr>
<td>![vacantSpaces: integer]</td>
</tr>
<tr>
<td>Multiplicity: ([0..1])</td>
</tr>
<tr>
<td>Number of vacant spaces for area, facility or assigned parking</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ASSOCIATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Association (direction: Source -&gt; Destination) facilityActualStatus]</td>
</tr>
<tr>
<td>Actual status of parking facility</td>
</tr>
<tr>
<td>Source: (Class) ParkingFacilityDynamicInformation</td>
</tr>
<tr>
<td>Cardinality: [1]</td>
</tr>
</tbody>
</table>

### 5.3.2 ParkingFacilityDynamicInformation
Container for all dynamic information about a parking facility.
5.4 Index data

Figure 4. Index data

5.4.1 Location
Specification of location data

ATTRIBUTES

✓ validityStartOfPeriod: DateTime
Multiplicity: ([0..1])
Indicates the start of the validity period of the location. No value or null means unlimited.

✓ validityEndOfPeriod: DateTime
Multiplicity: ([0..1])
Indicates the end of the validity period of the location. No value or null means unlimited.
**ATTRIBUTES**

- coordinatesType: string = WGS84
  - Multiplicity: ([1..1])
  - Type of coordinate system, “WGS84”

- latitude: float
  - Multiplicity: ([1..1])
  - Latitude of coordinate in decimal degrees

- longitude: float
  - Multiplicity: ([1..1])
  - Longitude of coordinate in decimal degrees

**ASSOCIATIONS**

- **Association (direction: Source -> Destination) sellingPointLocation**
  - The location of a SellingPoint. Because the sellingPointLocation has a validity, it is possible in the domain model for a SellingPoint to have multiple locations. It is the responsibility of the data owner that the content of the data can be clear to the user.
  - Source: (Class) SellingPoint
  - Target: (Class) Location
  - Cardinality: [0..*]

- **Association (direction: Source -> Destination) locationForDisplay**
  - Location of the parking facility for displaying on maps
  - Source: (Class) ParkingFacilityInformation
  - Target: (Class) Location
  - Cardinality: [0..1]

- **Association (direction: Source -> Destination) accessPointLocation**
  - The location of an AccessPoint. Because the accessPointLocation has a validity, it is possible in the domain model for an AccessPoint to have multiple locations. It is the responsibility of the data owner that the content of the data can be clear to the user.
  - Source: (Class) AccessPoint
  - Target: (Class) Location
  - Cardinality: [0..*]

- **Association (direction: Source -> Destination) locationForDisplay**
  - Location of the parking facility for displaying on maps
  - Source: (Class) ParkingIndexEntry
  - Target: (Class) Location
  - Cardinality: [0..1]

5.4.2 ParkingIndexEntry

Container for index data of a parking facility

**ATTRIBUTES**

- name: string
### Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Multiplicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of the parking facility</td>
<td>string</td>
<td>[1..1]</td>
</tr>
<tr>
<td>UUID of the parking facility</td>
<td>string</td>
<td>[1..1]</td>
</tr>
<tr>
<td>limitedAccess</td>
<td>boolean</td>
<td>[1..1]</td>
</tr>
<tr>
<td>staticDataUrl</td>
<td>string</td>
<td>[1..1]</td>
</tr>
<tr>
<td>dynamicDataUrl</td>
<td>string</td>
<td>[1..1]</td>
</tr>
</tbody>
</table>

#### Attributes Description

- **identifier**: string
  - Multiplicity: [1..1]
  - Indicates whether authentication is necessary to request data for this parking facility.
- **staticDataUrl**: string
  - Multiplicity: [1..1]
  - URL that must be used to retrieve the static data of the parking facility.
- **dynamicDataUrl**: string
  - Multiplicity: [1..1]
  - URL that must be used to retrieve the dynamic data of the parking facility.

### Associations

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Class) ParkingIndexEntry</td>
<td>(Class) Location</td>
<td>[0..1]</td>
</tr>
</tbody>
</table>

#### Association Description

- **locationForDisplay**: Association (direction: Source -> Destination)
  - Location of the parking facility for displaying on maps.
# 6. MAPPING DATA TO JSON

JSON is a simple data serialization format, having only a few basic types. This chapter contains the mapping of the types in the data model to JSON.

<table>
<thead>
<tr>
<th>Type</th>
<th>Mapping to JSON</th>
<th>Restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>number</td>
<td></td>
</tr>
<tr>
<td>integer</td>
<td>number</td>
<td>Only integers used</td>
</tr>
<tr>
<td>boolean</td>
<td>boolean literal 'true' or 'false'</td>
<td></td>
</tr>
<tr>
<td>DateTime</td>
<td>number</td>
<td>Only integers used</td>
</tr>
<tr>
<td>string</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>TimeType</td>
<td>string</td>
<td></td>
</tr>
</tbody>
</table>
7. **PUSH PROTOCOL**

A PMS pushes parking facility data to a CPDS by issuing a HTTP PUT request. Two request types are needed:

1. one to push static data from the PMS to the CPDS;
2. one to push dynamic data from the PMS to the CPDS.

Typically, CPDS systems will want to authenticate incoming requests. Authentication is done using the mechanism mentioned in paragraph 4.4.

7.1 **Pushing static data**

7.1.1 **Client request**

7.1.1.1 **Message content**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>parkingFacility</td>
<td>ParkingFacilityInformation</td>
</tr>
</tbody>
</table>

7.1.1.2 **JSON example**

PUT /parkingdata/v2/static/<UUID>/ HTTP/1.1  
Content-Type: application/json  
Content-Length: ...

```
{  
  "parkingFacilityInformation": {  
    "description": "Delft, Phoenixgarage",  
    "identifier": "637bcfc1-3fd6-4204-b8c8-af9db2699661",  
    "name": "Phoenixgarage",  
    "accessPoints": [  
      {  
        "alias": "Phoenixstraat",  
        "isPedestrianEntrance": true,  
        "isPedestrianExit": true,  
        "isVehicleEntrance": true,  
        "isVehicleExit": true,  
        "accessPointAddress": {  
          "city": "Delft",  
          "country": "Nederland",  
          "streetName": "Phoenixstraat"  
        },  
        "accessPointLocation": [  
          {  
            "coordinatesType": "WGS84",  
            "latitude": 52.010781,  
            "longitude": 4.354725  
          }  
        ]  
      },  
      {  
        "alias": "Oude Delft",  
        "isPedestrianEntrance": true,  
        "isPedestrianExit": false,  
        "isVehicleEntrance": false,  
        "isVehicleExit": false,  
        "accessPointAddress": {  
          "city": "Delft",  
          "country": "Nederland",  
          "streetName": "Oude Delft"  
        }  
      }  
    ]  
  }  
}
```
"accessPointLocation": {
  "coordinatesType": "WGS84",
  "latitude": 52.010979,
  "longitude": 4.35639
}

"operator": {
  "name": "GemeenteDelft",
  "url": "www.delft.nl",
  "postalAddress": {
    "city": "Delft",
    "country": "Nederland",
    "houseNumber": "78",
    "streetName": "Postbus",
    "zipcode": "2600 ME"
  },
  "administrativeAddresses": [
    {
      "city": "Delft",
      "country": "Nederland",
      "emailAddresses": [
        "info@delft.nl"
      ],
      "houseNumber": "16",
      "phoneNumbers": [
        "14015",
        "015 260 22 22"
      ],
      "province": "ZH",
      "streetName": "Phoenixstraat",
      "zipcode": "2611 AL"
    }
  ],
  "paymentMethods": [
    {
      "atExit": true,
      "atPaystation": true,
      "method": "Visa"
    },
    {
      "atExit": false,
      "atPaystation": true,
      "method": "Coins"
    }
  ],
  "openingTimes": {
    "startOfPeriod": 1388534400,
    "endOfPeriod": 1419984000,
    "exitPossibleAllDay": true,
    "openAllYear": true,
    "periodName": "Hele jaar",
    "entryTimes": {
      "enterFrom": {
        "h": 0,
        "m": 0,
        "s": 0
      },
      "enterUntil": {

{ "enterFrom": { "h": 0, "m": 0, "s": 0 }, "enterUntil": { "h": 23, "m": 59, "s": 59 }, "dayNames": [ "Fri", "Sat", "Sun" ] }, "tariffs": [ { "startOfPeriod": 1388534400, "endOfPeriod": 1419984000, "maximumDayCharge": 24, "periodName": "...", "tariffDescription": "...", "validityDays": [ "Mon", "Tue", "Wed", "Thu", "Fri" ], "validityFromTime": { "h": 0, "m": 0, "s": 0 }, "validityUntilTime": { "h": 23, "m": 59, "s": 59 }, "intervalRates": [ { "charge": 0.2, "chargePeriod": 10, "durationFrom": 0, "durationTo": 180, "durationType": "Minutes" } ] }]
7.1.2 Server response
The server can respond with one of the following HTTP Responses:

- Status-Code 200 ('Ok') if the data was accepted by the server;
- Status-Code 400 ('Bad request') if the server finds the request incorrect;
- Status-Code 401 ('Unauthorized') if the client is not authorized to push this data.

7.2 Pushing dynamic data

7.2.1 Client push message

7.2.1.1 Message content

```
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>ActualStatus</td>
</tr>
</tbody>
</table>
```

7.2.1.2 JSON example

PUT /parkingdata/v2/dynamic/<UUID>/ HTTP/1.1
Content-Type: application/json
Content-Length: ...

```
{  
  "parkingFacilityDynamicInformation": {  
    "description": "Delft, Phoenixgarage",  
    "identifier": "637bcf1c-3fd6-4204-b8c8-af9db2699661",  
    "name": "Phoenixgarage",  
    "facilityActualStatus": {  
      "lastUpdated": 1386166308,  
      "statusDescription": "...",  
      "open": true,  
      "full": false,  
```
"parkingCapacity": 250,
"vacantSpaces": 123,
"chargePointVacantSpaces": 0
}
}
}

7.2.2 Server response
The server can respond with one of the following HTTP Responses:

- Status-Code 200 (‘Ok’) if the data was accepted by the server;
- Status-Code 400 (‘Bad request’) if the server finds the request incorrect.
- Status-Code 401 (‘Unauthorized’) if the client is not authorized to push
  this data.
8. PULL PROTOCOL

A website or app pulls static parking facility data from the CPDS by issuing a HTTP GET request. Three request types are needed:

1. one to get a list of known parking facilities and the location (URL’s) of the associated static and dynamic data;
2. one to get the static data of a parking facility;
3. one to get the dynamic data of a parking facility.

8.1 Pulling a list of known parking facilities

The purpose of this message is to allow website and app developers to get a list of all known facilities, filter it using a geographical filter, and then only pull the data of the facilities of interest.

The response also contains URL’s where the static and dynamic data can be obtained. This allows for separation of indexing servers and data servers and for separation of static data servers and dynamic data servers.

The response also contains an field (‘limitedAccess’) to indicate whether accessing the static and dynamic data of that facility requires authentication.

8.1.1 Client request

GET /parkingdata/v2/ HTTP/1.1
Content-Type: application/json

8.1.2 Server response

The server can respond with one of the following HTTP Responses:

- Status-Code 200 (‘Ok’) if the request was accepted by the server and;
- Status-Code 400 (‘Bad request’) if the server finds the request incorrect;
- Status-Code 401 (‘Unauthorized’) if the client is not authorized to get this data.

8.1.2.1 Message content

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>parkingFacilities</td>
<td>ParkingIndexEntry</td>
</tr>
</tbody>
</table>

8.1.2.2 JSON example of response

HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: ...
{
    "parkingIndexEntry": [
        {
            "name": "Delft, Phoenixgarage",
            "identifier": "637bcf1c-3fd6-4204-b8c8-af9db2699661",
            "limitedAccess": false,
            "staticDataUrl": "http://...",
            "dynamicDataUrl": "http://...",
            "locationForDisplay": {
                "coordinatesType": "WGS84",
                "latitude": 52.010781,
                "longitude": -4.311768,
            }
        }
    ]
}
8.2 Pulling the static data of a parking facility

This request can be subject to authentication. Authentication is done using the mechanism mentioned in paragraph 4.4.

8.2.1 Client request

GET <URL OBTAINED FROM LIST> HTTP/1.1
Content-Type: application/json

8.2.2 Server response

The server can respond with one of the following HTTP Responses:

- Status-Code 200 ('Ok') if the request was accepted by the server and;
- Status-Code 400 ('Bad request') if the server finds the request incorrect.
- Status-Code 401 ('Unauthorized') if the client is not authorized to get this data;
- Status-Code 404 ('Not found') if the server has no static data for this parking facility.

8.2.2.1 Message content of response

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>parkingFacility</td>
<td>ParkingFacilityInformation</td>
</tr>
</tbody>
</table>

8.2.2.2 JSON example

See example for pushing static data.

8.3 Pulling the dynamic data of a parking facility

This request can be subject to authentication. Authentication is done using the mechanism mentioned in paragraph 4.4.

8.3.1 Client request

GET <URL OBTAINED FROM LIST> HTTP/1.1
Content-Type: application/json
8.3.2 Server response
The server can respond with one of the following HTTP Responses:
- Status-Code 200 ('Ok') if the request was accepted by the server and;
- Status-Code 400 ('Bad request') if the server finds the request incorrect.
- Status-Code 401 ('Unauthorized') if the client is not authorized to get this data;
- Status-Code 404 ('Not found') if the server has no static data for this parking facility.

8.3.2.1 Message content of response

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>ActualStatus</td>
</tr>
</tbody>
</table>

8.3.2.2 JSON example
See example for pushing dynamic data.
9. LICENSING

Data providers may elect to provide the dynamic parking data under license. Therefore, authentication is an integral part of the transport protocol. This standard does not prescribe which license model is the most appropriate for a data provider, nor does it set requirements for the registration procedures employed.