Information Model

Version 4.1 Release

Updated January 14, 2026

Contents

Introduction

The Alliance for Parking Data Standards (APDS) is managing the creation of a consensus - built international parking and kerbside data specification to establish a common language for data concepts and definitions in the parking, transportation, and mobility sectors that will facilitate seamless integration, compatibility, and communication between parking entities, the automotive sector, IT developers, services, and map and app providers, as well as other stakeholders. In addition, APDS has proposed these specifications to ISO as the basis of a future ISO global standard.

APDS Technical Documentation

The APDS Technical Documentation describes a set of data concepts and definitions that public and private parking owners, operators, service providers can adopt and use as a common language to facilitate the communication of data between themselves and other sectors.

The specification comprises a set of documents that define the various data related to parking and kerbside management operations and show the relation of these data to each other. The documentation includes the following documents and information:

- Information Model Document:
  - List of data namespaces, packages, classes, attributes and relationships
  - Defined lists of acceptable entries for certain data elements
  - Identified references to other standards, as appropriate.
  - Relationship diagram showing how the data is organized and expected to be sent
- Use Case Document: example use cases for applying the data specification.
- Data Dictionary Document: Terminology and definitions for the data elements

The present document is the Information Model Document.

This document seeks to define common international definitions of data concepts, and their relationship for parking operations and other related activities. These definitions have been drawn from several existing sources of information and existing specifications and standards, including:

- Technical documentation supplied by the International Parking & Mobility Institute
- Technical documentation drawn from the Dutch National Parking Register
- Technical documentation supplied by the British Parking Institute
- CEN DATEX II Standards, in the CEN 16157 series

The data concepts defined in this document are modelled using UML (Unified Modeling Language) in a form that substantively aligns with the modelling approach defined in Part 1 of the CEN DATEX 16157 series (EN 16157-1:2018). Adaptations to this approach are defined in Annex B.

Release Changes

Changes between Release v3.0 and v4.0

APDS Release 3.0 documents were formally released June 2021. This document is Release 4.0 of the Information Model; modifications have been made within the following elements:

Place

- Identified Area specializations.
- Added Pedestrian Access Facility to identify pedestrian access and exit points in a Place.
- Added detail to Organization and Contacts.

Rates

- Added User Defined Code list to support Vehicle_type and Vehicle loads (passenger, hazardous waste, cargo, etc.) to Qualifications.
- Add Emissions as a data class to Qualifications.

Right

- Moved financial transaction data from Session to AssignedRight data class.

Changes between Release v4.0 and v4.1

A small set of minor enhancements:

- Enhancement to the Observation domain model - the addition of classes "Confidence" and "CharacterConfidence", and additional attributes within the "Image" class.
- Bug correction - Attribute "emailAddress" added to the class "EMailCommonData".
- Bug correction - Attribute "status" added to the class "RefillPoint".
- Enhancement – Introduction of a generic extension mechanism.
- Enhancement – Introduction of an extension to the modelling of emissions based conditions to support a variety of different conditions that have been seen in actual deployments (such as engine capacity, CO_2 tailpipe output, vehicle weight).

Data Model

Data Model Overview

This specification facilitates the sharing of basic information between entities and systems. This includes map services, online marketing and aggregator services, event ticketing platforms, transit and transportation agencies, and other firms, organisations or individuals that have a need to know the location of parking and other mobility related services and general information about the operation.

Figure 1 shows the data domains in this document. Each data domain defines a specific set of data concepts that logically can be grouped.

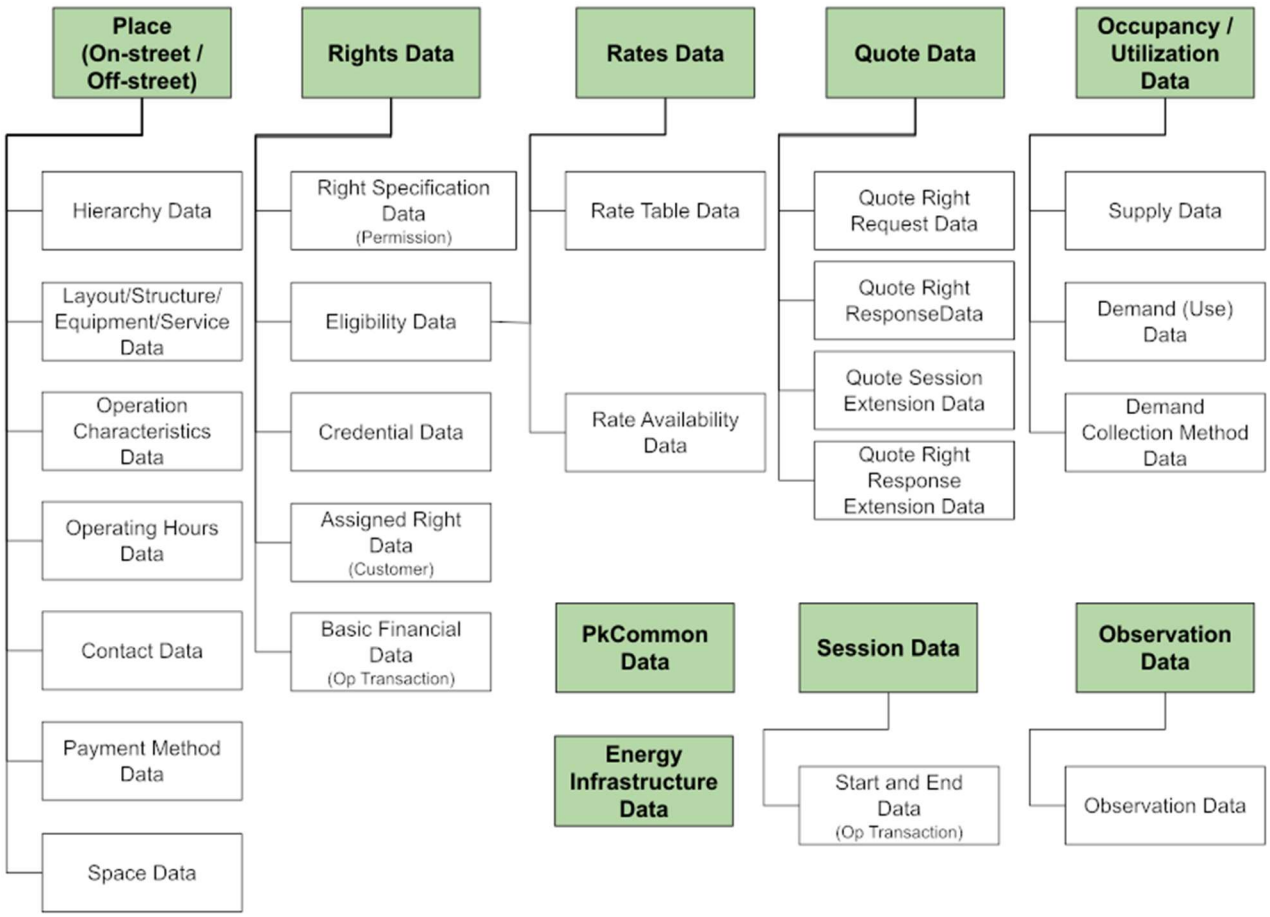

Figure 1 - Data Domains

To assist navigation and understanding of the specification in this document, the overall model is separated into data domains, which can be seen in UML packages in Figure 2. The domains focus on functional groups of data concepts and relationships. Each of the domains is specified in its own namespace. In addition, the structure of the data model supports the definition of common concepts that are referenced and reused across several domains. These common concepts are specified in the "PkCommon" domain.

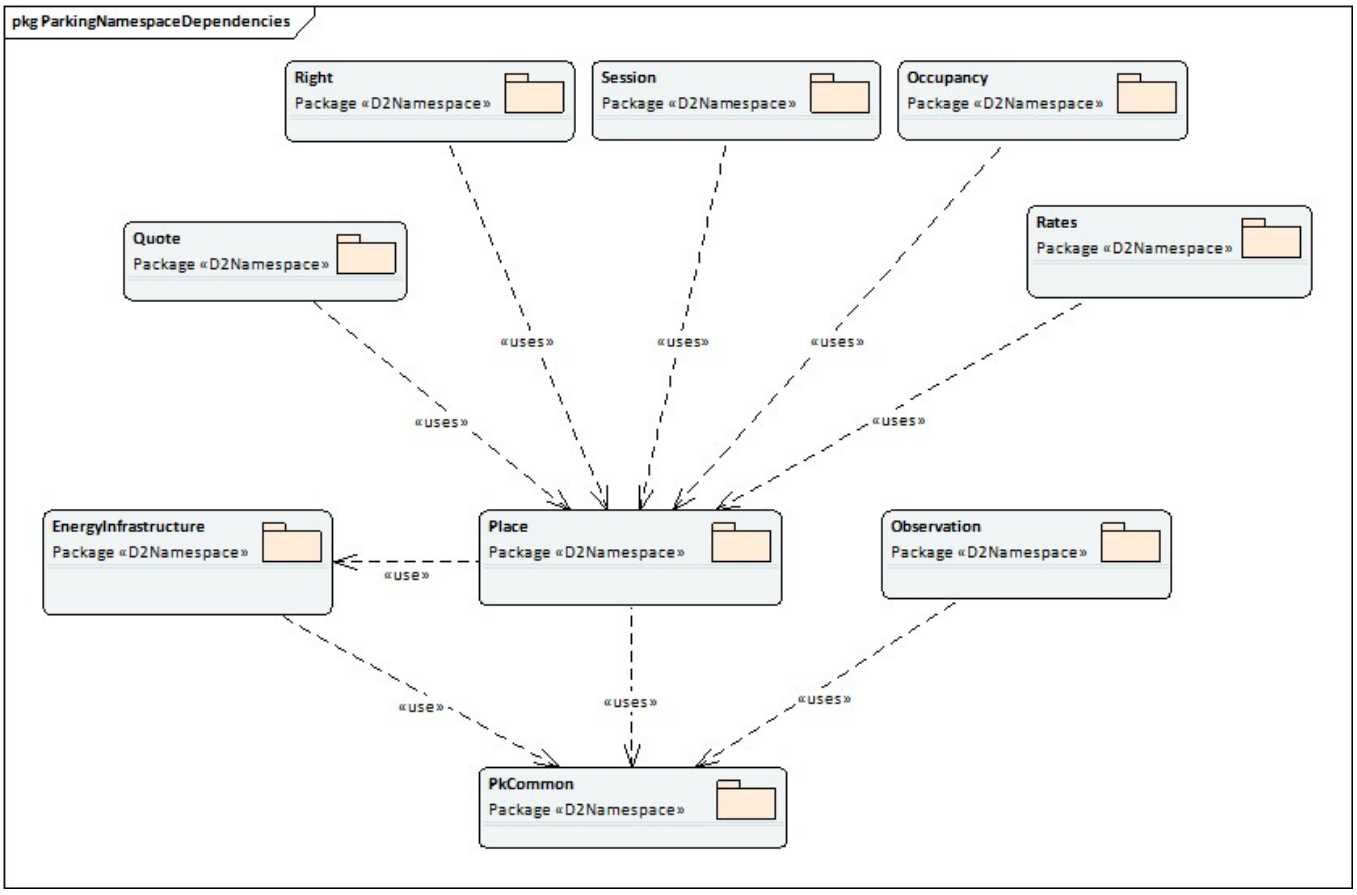

Figure 2 – Namespace Dependencies

Figure 2 illustrates the dependency between namespaces.

These data domains, as characterised by the namespaces, are combined in various manners to support specific use cases that exist in parking and mobility operations. As examples:

- The operations and systems to support automated valet parking rely on this document's data domains of place, rate, quote, session, right, and occupancy.

- The operation and systems to support kerbside management will rely on all data domains in this document to support data sharing.

Place Data Concepts

Place Hierarchy

This clause describes concepts used in the place hierarchy, which are defined in the <Place> namespace of the data model.

Note: Location, time and contact/organisation concepts are defined in the <PkCommon> namespace of this data model.

The specification defines a method to build a hierarchy of place records in on-street, off-street, and zone environments. Each instance of a place record corresponds to an instance of a class within the place hierarchy. This enables a parking or other type of operation to identify a location or zone in a simple manner with a single hierarchy element or to breakdown a place into a multi-layered hierarchy that identifies discrete parking enclosures or defined areas to communicate operating hours, space counts, operating restrictions, location, rights and associated pricing and occupancy.

The hierarchy supports the ability of lower-level place hierarchy class instances (child records) to inherit specific instance data from a higher-level place hierarchy class instance (a parent record) to simplify the amount of data shared. The place hierarchy also enables lower-level place hierarchy class instances to document variations in specific attributes from their parent, higher-level place hierarchy class instance.

The specification also enables other operation types, not directly related to parking, to define a place into discrete operating enclosures. This may include a defined on-street area for managing delivery services or a sidewalk area enabled for bike or e-scooter placement.

The hierarchy allows a Data Distributing Party to decide the appropriate level of detail to send to a Data Receiving Party via messaging protocols. In addition, it is not necessary to build multi-layer hierarchies of Place data. Simple data needs can be represented without using the multiple layers available in the data specification.

Each of the component PLACE data concepts is mentioned in this clause, but further details and descriptions follow in subsequent clauses. These component data concepts are Place (6.2.1.4), SubplaceElement (6.2.1.5), IdentifiedArea (6.2.1.6), Space (6.2.1.7), Campus, PedestrianAccess (6.2.2.1), VehicularAccess (6.2.2.2), SpecificArea (6.2.2.3), and SupplementalFacility (6.2.2.4).

Place is a term introduced in the specification to define where a vehicle may park, stand, rest, or briefly transit to allow a person to change modes of transport (i.e., taxi drop-off/pickup, ride share drop-off/pickup, valet stand, etc.). Place is instantiated via the Place class. Place can also be used to define entry and exit roadways, driveways, and acceleration/deceleration zones for vehicles as well as pedestrian access points. **Place supports both on-street and off-street operating environments**. Place also defines specific areas to be defined for mobility or other related uses such as bike storage, e-scooter enclosures, etc. where it is useful to share operating parameters or assign RightSpecifications, Rates, or other data domains.

A Place is an aggregation of instances of SubplaceElement, IdentifiedArea and Space. In this specification, the lowest mandatory class instance to define a Place is the IdentifiedArea. Note: Space is lower data concept instance that can exists below IdentifiedArea data concept instance, but it is not required.

An aggregation of IdentifiedAreas can create an instance of a SubplaceElement or a Place. Specific attributes are associated to each of the four specialization types of IdentifiedAreas.

An aggregation of instances of SubplaceElements can be linked to an instance of a higher level SubplaceElement or a Place. This allows a data provider to share as much or as little detail as necessary to support their operation.

At the highest level of the place hierarchy, an aggregation of instances of Place can be linked to an instance of a Campus. An instance of Campus is not a required within an implemented place hierarchy.

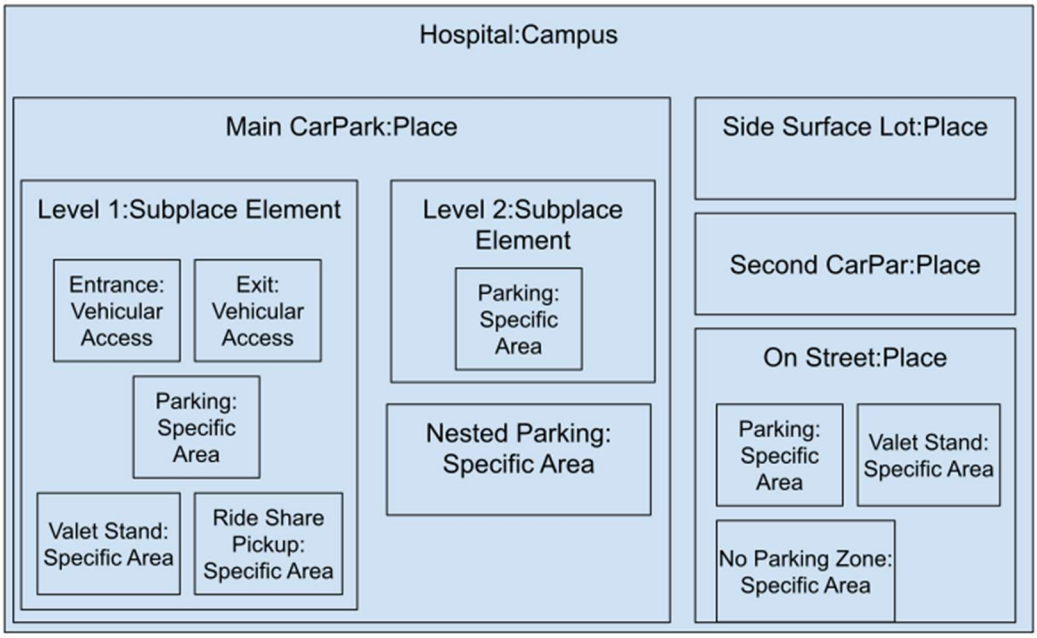

Figure 3 – Place Hierarchy Example – User-Oriented

Figure 3 illustrates how the instances of the various place hierarchy data concepts are combined to construct instances of nested parking hierarchy data concepts within one campus as a use case example.

To support parking, mobility or kerbside management operations, the IdentifiedArea class can be refined, through specialization, as one of four specialization types: PedestrianAccess, VehicularAccess, SpecificArea, and SupplementalFacility.

PedestrianAccess class is a specialization of IdentifiedArea class that enables a place to share coherent, relevant information about pedestrian entry and exit access points to a Place. This allows an entity to identify operating hours for the pedestrian access points and access requirements such as credentials or restrictions for each access point.

VehicularAccess class is a specialization of IdentifiedArea class that enables a Place to share coherent, relevant information about entry lane, exit lane, and similar type of vehicular access points to a Place. The VehicularAccess class is used to define entry and exit to a specific facility or is used to provide a more macro description of an entry to a facility by reference to Road and RoadNodes.

- Road: Allows an organizational entity to identify the primary road a place exists on and describe routes to the Place
- RoadNode: Allows an organizational entity to identify nearby major intersections of roads

SpecificArea class is a specialization of the IdentifiedArea class that denotes a specific (generally geographically bounded) area in a place that has a common operating purpose and common characteristics. Examples of common operating purpose include reserved parking area, EV parking, bike parking, delivery zone, loading zone, etc. The SpecificArea class describes the physical components of a Place.

For clarity, when defining the use of a SpecificArea, authorized activities to perform in an IdentifiedArea are defined by the assignment of a RightSpecification and include activities such as valet parking, disabled parking, delivery zone, etc. The assignment and authorization to perform a use in a SpecificArea occurs via the assignment of a RightSpecification. As an example, valet parking is associated to an IdentifiedArea with a SpecificArea class via the RightSpecification which assigns valet parking to a specific person and place.

SupplementalFacility is a specialization of IdentifiedArea class that enables a Place to identify the type and location of equipment and services available within the Place. The SupplementalFacility class is used to define restroom/toilet, shower, food concession services, etc. It is also used to identify specific equipment available in a Place, such as bike storage, electric chargers, payment stations, etc. This IdentifiedArea specialization defines physical structures and equipment in the place as well as services available.

The IdentifiedArea data concept collects general operating information such as operating hours, operating restrictions, space information, and payment information. If the data is absent at the specific instance of an IdentifiedArea, it is assumed the data for this information exists at a higher level in the hierarchy, encapsulated in either instances of Place or any level instance of SubplaceElement above the instance of IdentifiedArea. This allows for customization of operations at lower levels while relying on default data from higher levels in the Place hierarchy.

A Place is synonymous with the structure or area a consumer associates with where a vehicle parks or a mobility service is delivered. It can be an entire parking structure or an aggregation of streets supporting on-street parking (also sometimes called a zone). General operating information such as operating hours, operating restrictions, space information, payment information, etc. is associated to a Place and any parts of the hierarchy underneath it as appropriate.

Place is a core component of this model as it defines a hierarchy that supports the identification of portions of locations that may be related to parking and other types of operations. Use of this hierarchy enables data suppliers to provide a structured mechanism to refer to related zonal and place concepts with an ordered hierarchy.

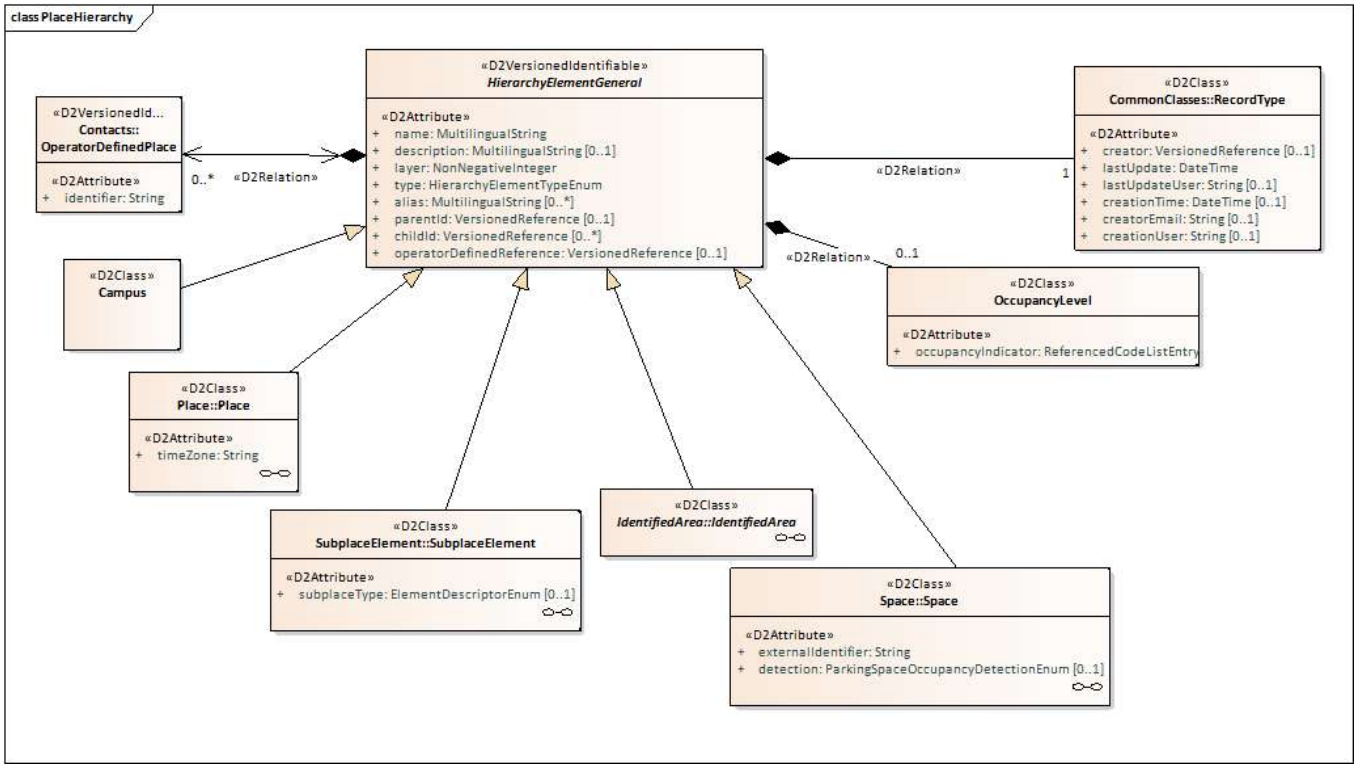

Figure 4 – Place Hierarchy

Figure 4 shows concepts within the place hierarchy:

- HierarchyElementGeneral – a generalised component of a Place hierarchy that forms one element in the tree-like hierarchy. This forms a reusable block of the hierarchy, with relations to its parent element (if one exists) and any child elements. Each Place in the hierarchy shall have a name and may support a free-text description and an operator/property owner defined reference (e.g., location number/identifier).

- There are five specializations of the Place hierarchy (HierarchyElementGeneral class), which conceptually defines different scales of hierarchy elements. From largest to smallest, they are:

  - Campus – the highest level in the hierarchy, it typically defines a large facility (such as a university campus, or an airport), or a large geographic zone (such as a city or a town), which may contain numerous places that can be logically reported together. A campus combines and encompasses a number of Places that can be logically reported together. Different entities sharing data may create their own aggregation of Places. Thus, a Place may appear in different campuses if a receiving party receives data from different sending parties. Example: a parking operator may group five (5) places together that it operates and call it a campus to reflect the five (5) operations in a city. Three of the same places may be associated with a property management firm that defined a separate campus with the three places.
  - Place – a place or location used for parking, loading, unloading, standing, or some other mobility or transport related activity. Place typically identifies a parking structure, surface lot or on street parking zone.
  - SubplaceElement – a sub-division of a Place for the convenience of the operator to segment the Place and identify varied uses for parking and mobility related operations or other purposes.

- IdentifiedArea – an identifiable discrete bounded geographic zone that shares common characteristics and is used for parking and mobility related operations or other purposes. IdentifiedAreas are segmented into four specializations: PedestrianAccess class, VehicularAccess class, SpecificArea class, and SupplementalFacilityArea class.
- Space – a single space for parking or other mobility related purposes, usually designed for one vehicle, which may, but not necessarily, be denoted by painted or another road surface marker.

These specialisations of place hierarchy (HierarchyElementGeneral)class (i.e., Campus, Place, SubplaceElement, IdentifiedArea and Space) support both the notion of different scales within the hierarchy, but also support different forms of attribution, which are described later in this document.

The Place hierarchy shall be defined “top-down” with the highest layer being numbered 0 (zero) and each subsequent layer being numbered by incremental increasing integers.

The Place hierarchy can best be considered as a tree-like structure which starts from one common root element which will either be an instance of Place or, if used, Campus. Navigating down through the layers of the hierarchy, irrespective of which branch is taken, each subsequently layer sub-divides the previous. Each branch of hierarchy "tree" shall terminate in either an instance of IdentifiedArea or Space. Each branch of the hierarchy shall contain one instance of Place, and no more than one instance of IdentifiedArea. Note: A single instance of Place may support multiple branches, thus an instance of Place can have multiple instances of IdentifiedArea associated to it. If an instance of Campus is present, it shall only occur once in each branch of the hierarchy. See Figure 5, which provides an illustrative example of elements of a place hierarchy combine.

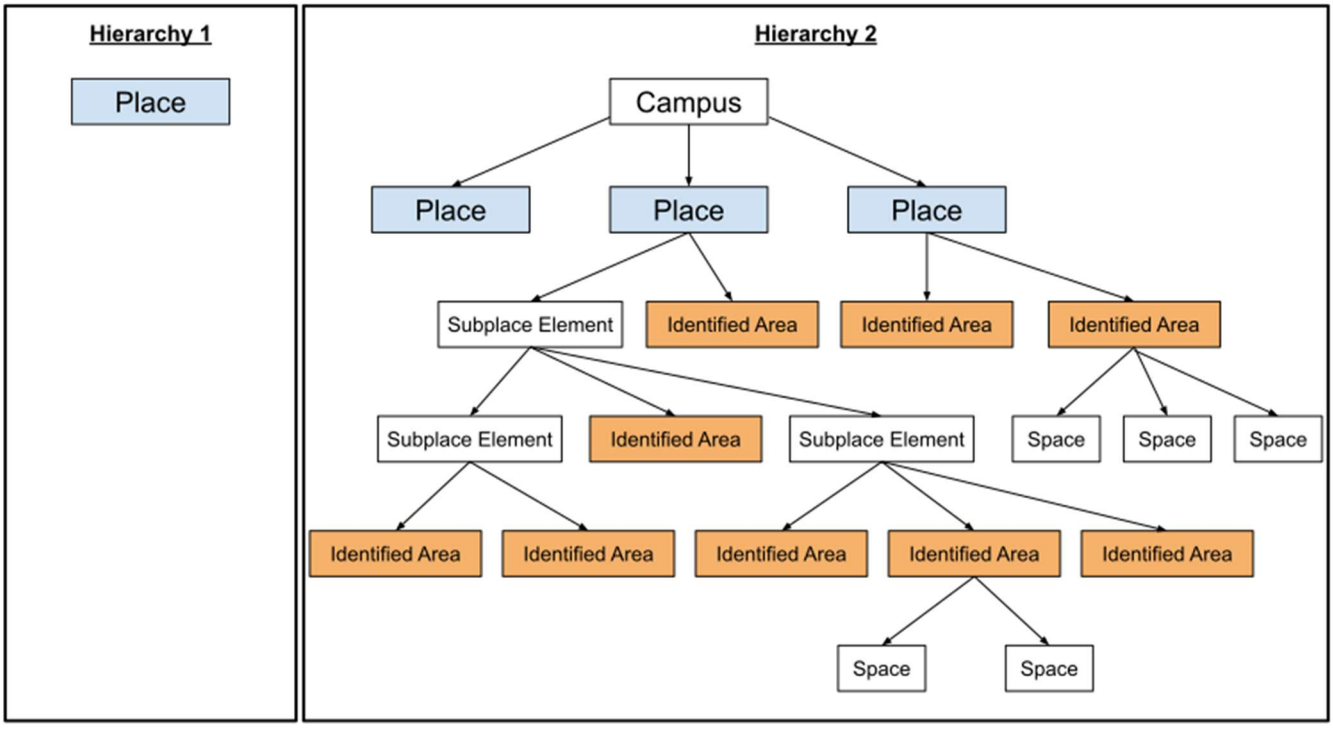

Figure 5 – Examples of Place Hierarchy

When describing the homogeneous characteristic for an entire place, the tree structure may terminate with the instance of Place, (see Hierarchy 1 in Figure 5) with no further child hierarchy elements (including IdentifiedArea).

If the place is sub-divided, each "branch" of the hierarchy tree shall terminate in one instance of IdentifiedArea, and the IdentifiedArea class shall appear only once in each branch of the hierarchy tree. Only instances of the Space class may exist at the levels of a hierarchy below the occurrence of IdentifiedArea in each branch of the hierarchy tree. Every branch of the hierarchy tree may use instances of the data concepts Campus, Place, SubplaceElement, IdentifiedArea and Space in the order stated. Instances of SubplaceElement may be repeated in any branch of the hierarchy, so long as it conforms to the order in the stated sequence.

Location Hierarchy Example

Top down:

- Layer 0 – Campus (Airport)
- Layer 1 – Place (Short Stay Car Park) {others might be "long stay", "valet meet and greet", etc.}
- Layer 2 – SubplaceElement (Orange Level or 3rd Floor)
- Layer 3 – IdentifiedArea (Row J) - Type SpecificArea, with additional attributes
- Layer 4 – Space (74)

As a reminder, it is not necessary to build multi-layer hierarchies of Place data. Simple Place data needs can be represented without using the multiple layers available in the data specification.

Place:

Figure 6 provides a UML class diagram for Place.

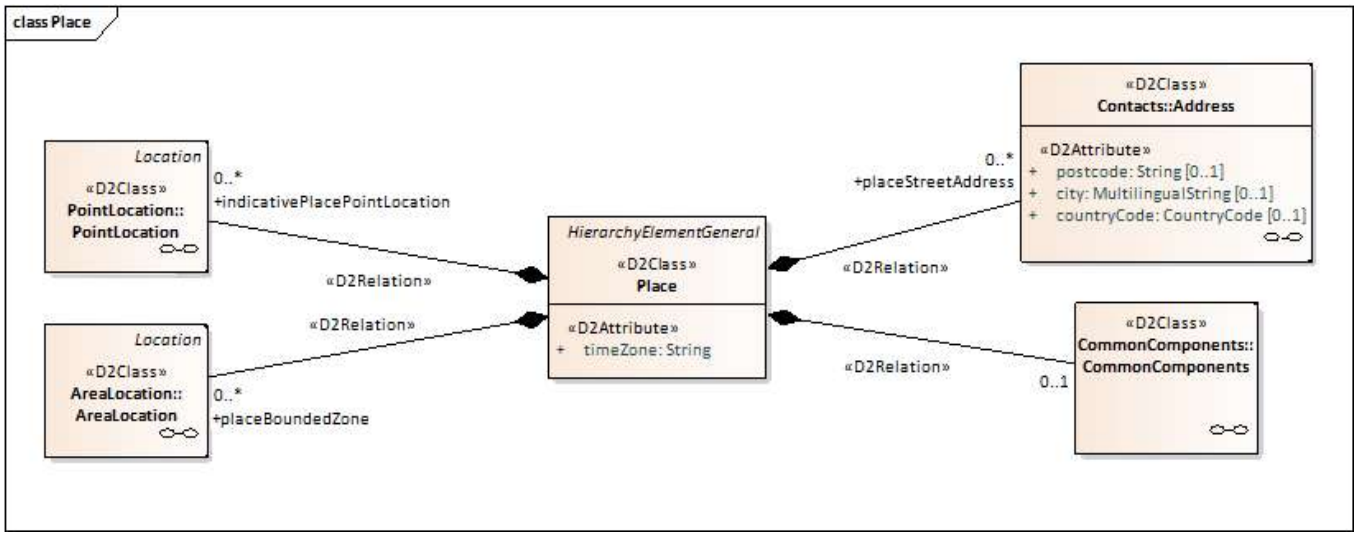

Figure 6 – Place

An instance of Place can have zero, one or several point locations that represent the notional point spatial location of the place. Additionally, an instance of Place can have zero, one or several area locations that represent the notional bounding polygon or zone for the place. An instance of Place can have zero, one or several street addresses. An instance of Place can access CommonComponents classes which support a wide range of optional attribution for contacts (which includes addresses), times (operating times, entrance opening times, exit opening times), location data, marketing content references, payment methods, operating restrictions, colour association (where parking facility elements are colour coded) and other characteristics (such as operating mode, type of the parking structure, access control, etc.)

Instances of IdentifiedArea and Space enable additional classes beyond the CommonComponents to be associated. IdentifiedArea is an abstract class, see Figure 8, which can be realised in four implementable specializations: PedestrianAccess, VehicularAccess, SpecificArea, and SupplementalFacility.

SubplaceElement and Space have a user defined code list related to occupancy level. This user defined code list enables a Place to define codes for various occupancy levels and the user-defined definition of each occupancy level. As an example, a place or space may identify occupancy levels using user-defined colour codes and define each colour code as follows:

- User Defined Code 1: Green
- User Defined Code 1 Definition: Occupancy utilization is less than 80% of Supply
- User Defined Code 2: Yellow
- User Defined Code 2 Definition: Occupancy utilization is less than 85% of Supply
- User Defined Code 3: Red
- User Defined Code 3: Definition: Occupancy utilization is equal to or greater than 95% of Supply

SubplaceElement

Figure 7 provides a UML class diagram for SubplaceElement.

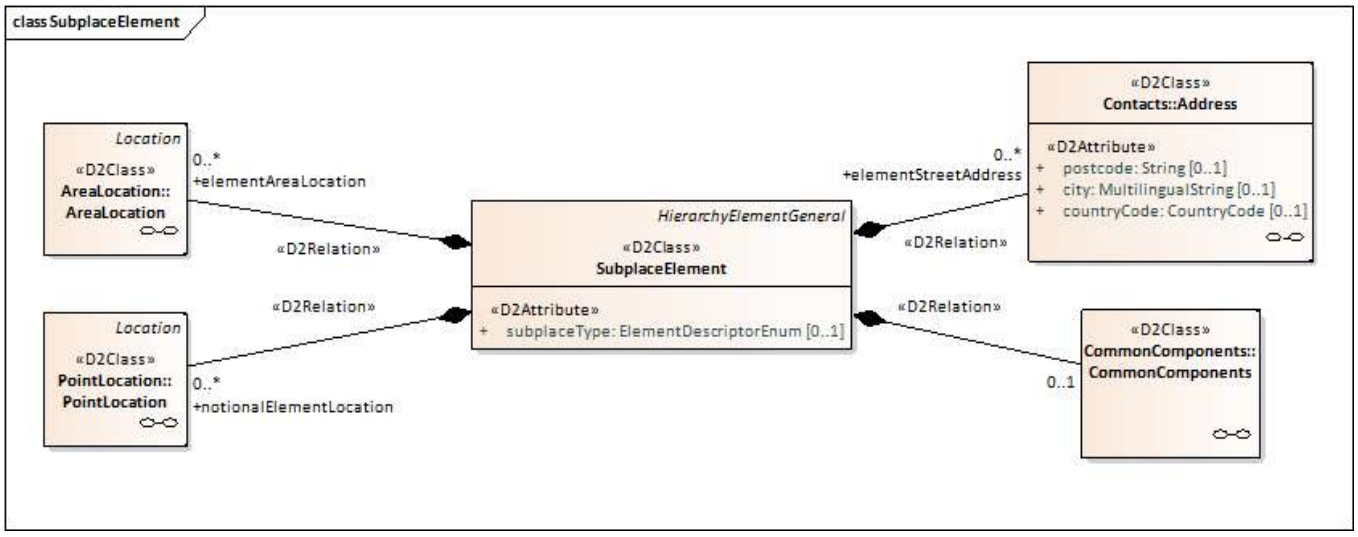

Figure 7 – SubplaceElement

IdentifiedArea

Figure 8 provides a UML class diagram for IdentifiedArea.

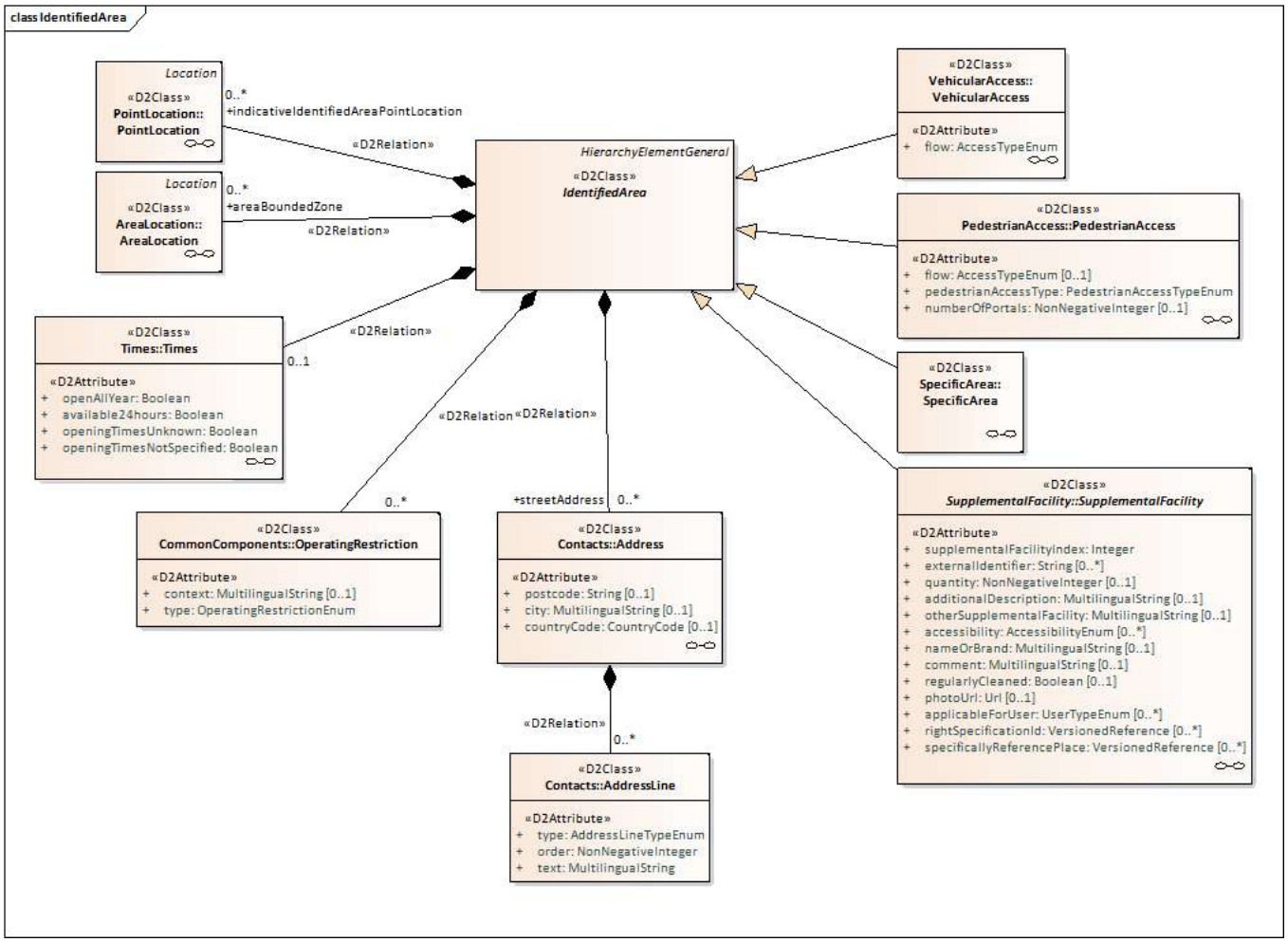

Figure 8 – IdentifiedArea

Space

Figure 9 provides a UML class diagram for Space.

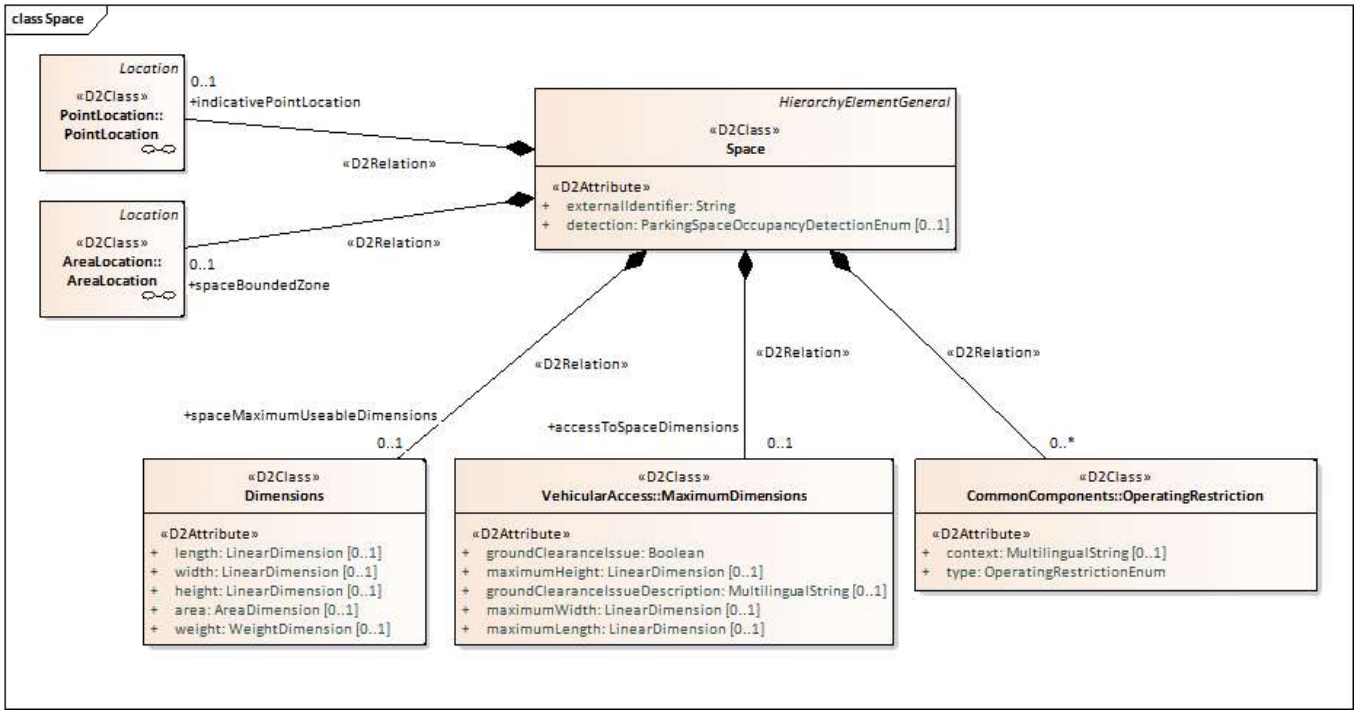

Figure 9 – Space

IdentifiedArea Sub-Types

Pedestrian Access

Figure 10 provides a UML class diagram for PedestrianAccess.

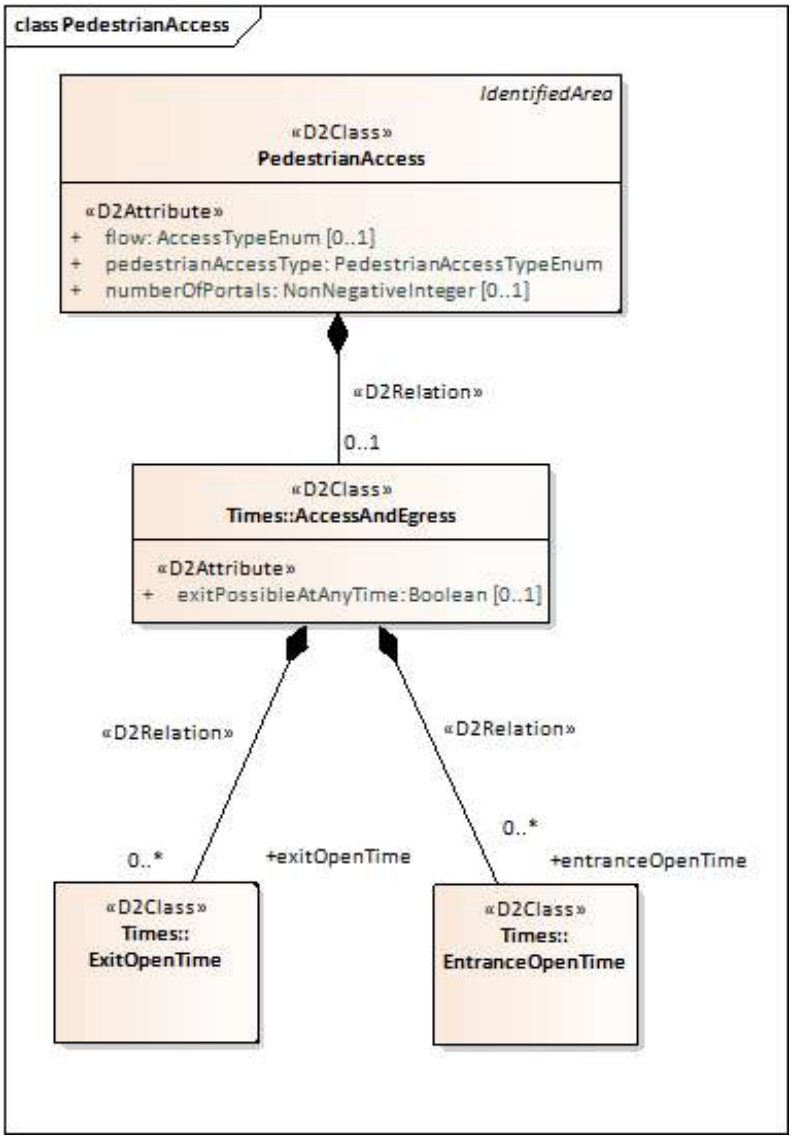

Figure 10 – PedestrianAccess class diagram

Note: This model only supports the modelling of vehicular and pedestrian accesses at the present time. Other forms of access may be introduced during a later revision if a stakeholder requirement is identified.

PedestrianAccessType enumerations (PedestrianAccessTypeEnum) for PedestrianAccess support the definition of characteristics for access in and out from a place or part thereof respectively (denoted by the pedestrianAccessType attribute, using the PedestrianAccessType enumerations). PedestrianAccess enables an entity to identify the quantity of each access type as well as the operating times the PedestrianAccess is available for use.

Vehicular Access

Figure 11 provides a UML class diagram for VehicularAccess.

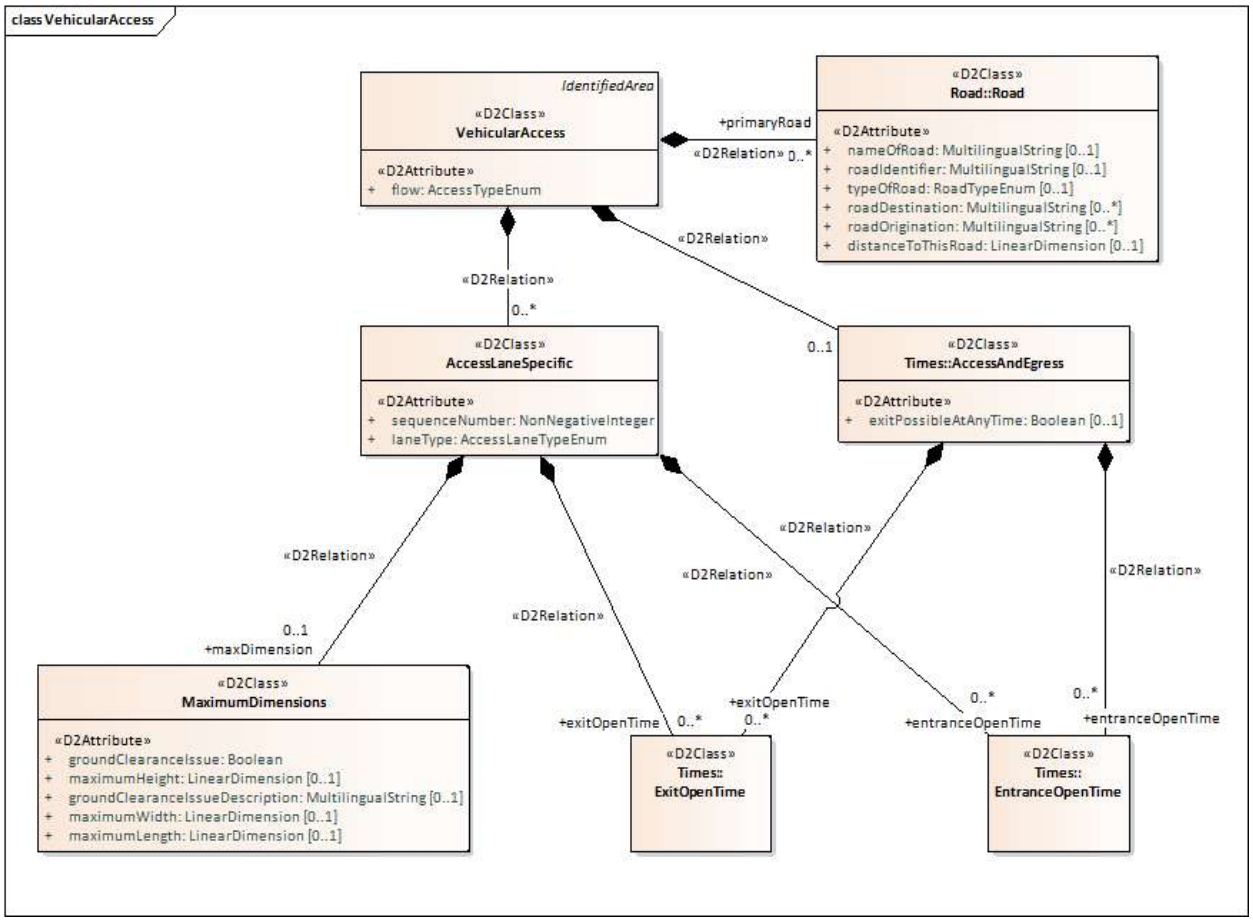

Figure 11 – VehicularAccess class diagram

AccessType enumerations (AccessTypeEnum) for VehicularAccess support the definition of characteristics for the how vehicles flow to, from and both to and from a place or part thereof respectively (denoted by the flow attribute, using the AccessType enumerations). Each of the AccessLaneType enumerations (AccessLaneTypeEnum) may support the definition of characteristics for each lane within the instance of VehicularAccess. Preferred practice gives a sequence number to each lane left to right when entering the facility.

VehicularAccess also supports a macro view of access by enabling an entity to reference access or approaches to a facility by defining primary roads near the place and/or identifying primary routes from a highway and/or nearby significant crossroads to define approximate location of the vehicular access point. This is accomplished via instances of Road and RodeNode classes, see Figure 12.

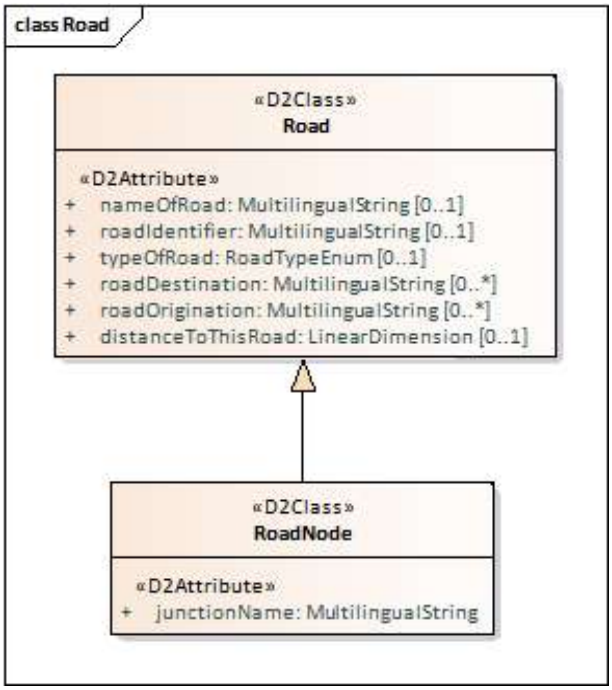

Figure 12 – Road class diagram

Specific Area

Figure 13 provides a UML class diagram for SpecificArea.

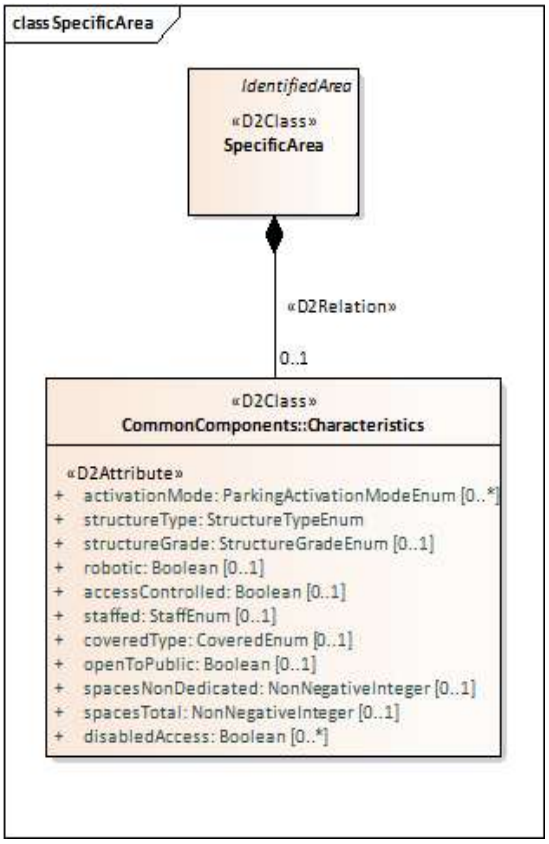

Figure 13 - SpecificArea class diagram

SpecificArea class is a specialization of IdentifiedArea that denotes a specific geographic area in a place that has a common physical purpose of use and common characteristics. Examples of common physical purpose of use include parking area, delivery zone, ride share pickup zone, and valet. SpecificArea data concept describes the physical components of a Place. SpecificArea specialization of IdentifiedArea is used to assign Rightspecifications.

Supplemental Facility

Figure 14 provides a UML class diagram for SupplementalFacility.

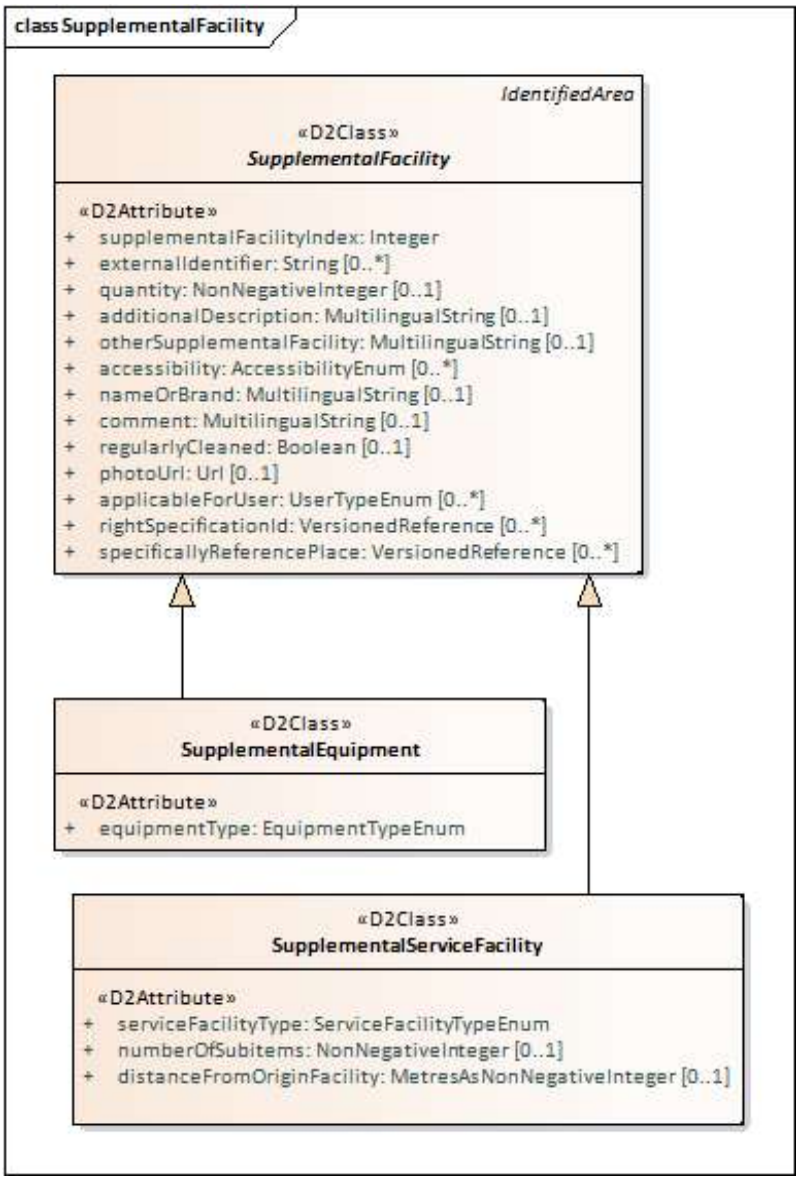

Figure 14 - Supplemental Facility class diagram

SupplementalFacility is a specialization of IdentifiedArea data concept that denotes a specific geographic area in a Place that has specific services and equipment available. This may include showers, meeting rooms, restrooms/toilets, lockers, and other services.

The SupplementalFacility specialization relies on the RightSpecification to define eligible users or vehicles of the defined equipment and services. Within the SupplementalFacility specialization, further classes are available to detail Equipment (SupplementalEquipment) and Services (SupplementalServcesFacility). In the SupplementalEquipment class, an entity can identify specific type of equipment and quantity available; this can include electric charging stations, payment stations, bike storage, etc.

Additional details for specific equipment can be further defined in future revisions of the specification. The data model currently includes a data class for ElectricChargingPoints which allows for more detail to be provided for one-to-many charging stations. The SupplementalServiceFacility class similarly allows a Place to identify various services available in the IdentifiedArea. Services include restroom, shower, food store, and many other services.

Common Components

CommonComponents represent a collection of common attribution that may be applied at several points in the Place hierarchy. CommonComponents provides optional support for a range of attribution including times (operating times, entrance opening times, exit opening times) [defined in the <PkCommon> namespace], marketing content references, payment methods, operating restrictions, colour association (where parking facility elements are colour coded) and other characteristics (such as operating mode, type of the parking structure, disabled parking characteristics, access control, etc.).

Figure 15 provides a UML class diagram for CommonComponents.

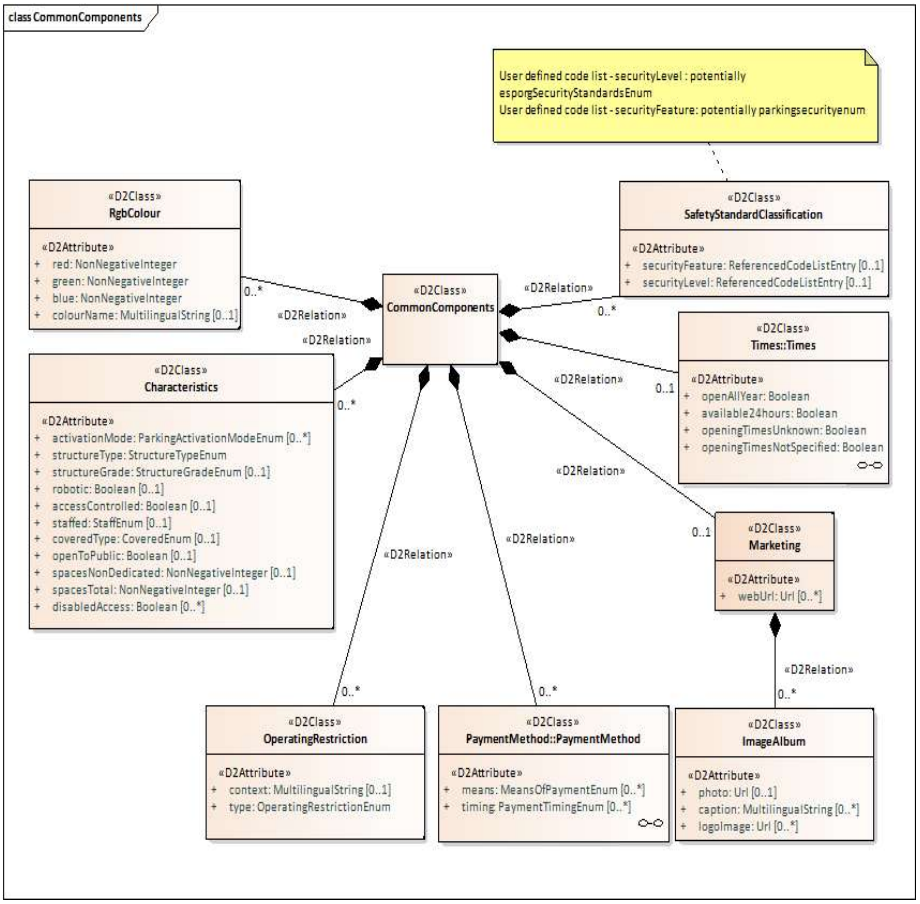

Figure 15 - CommonComponents class diagram

PaymentMethod

Figure 16 provides a UML class diagram for PaymentMethod.

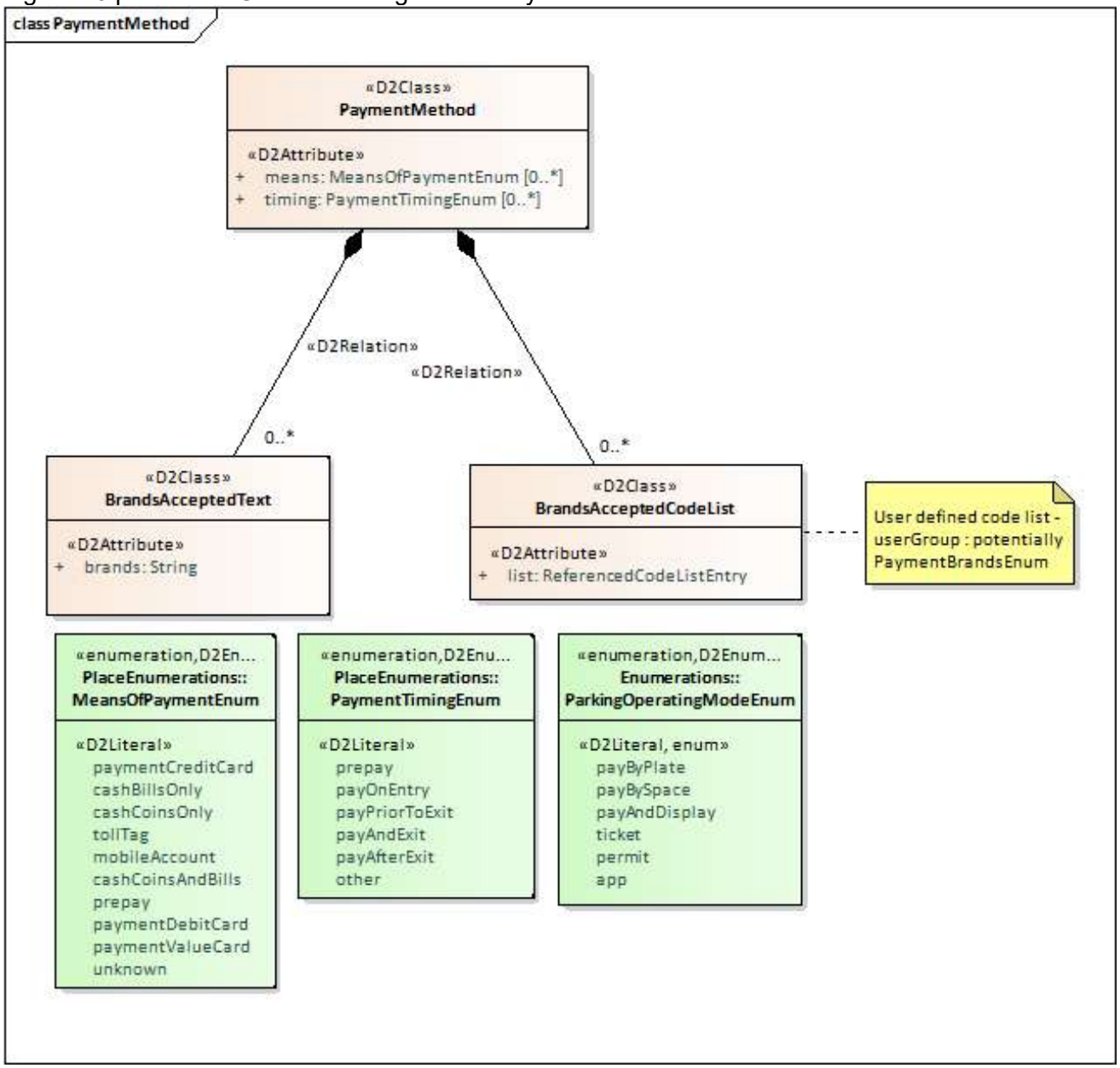

Figure 16 - PaymentMethod class diagram

Basic Elements (in Place)

The following concepts define data types, enumerations and common classes defined and used specifically in the <Place> namespace. More widely used data concepts, data types, enumerations and common classes are defined in Section 6.5 – The PkCommon Domain.

Data Types (for Place)

No specific data types are defined in <Place> namespace. For general data types, see Section 6.5 – The PkCommon domain.

Enumerations (for Place)

The following enumerations are defined in the <Place> namespace, see Figure 17.

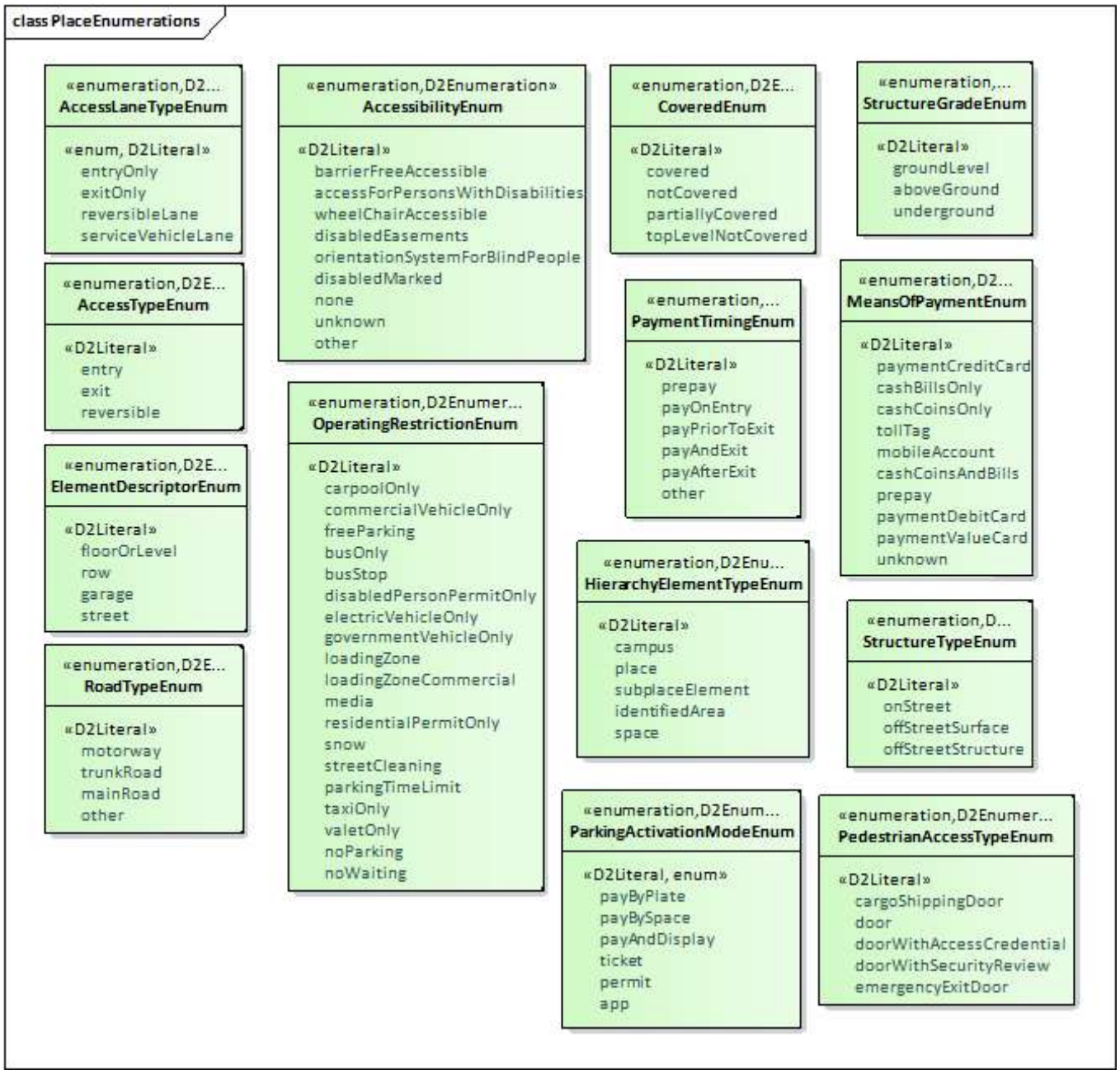

Figure 17 - Enumerations in the Place domain

External Codelists (for Place)

An external codelist exists for <Place> which enables a place owner to define a custom list of occupancy levels as a code with accompanying definitions for each occupancy level code. For general external code lists, see Section 6.5 – The PkCommon domain. No specific external code lists are defined in <Place>. For general external code lists, see Section 6.5 – The PkCommon domain.

Occupancy

Introduction to Occupancy

This section describes concepts used to define occupancy supply and demand for a specified place.

- Supply shows space inventory allocated for use (i.e., for parking, delivery, etc.),
- Demand shows the usage of space inventory.

These are defined in the <Occupancy> namespace of the data model.

Supply

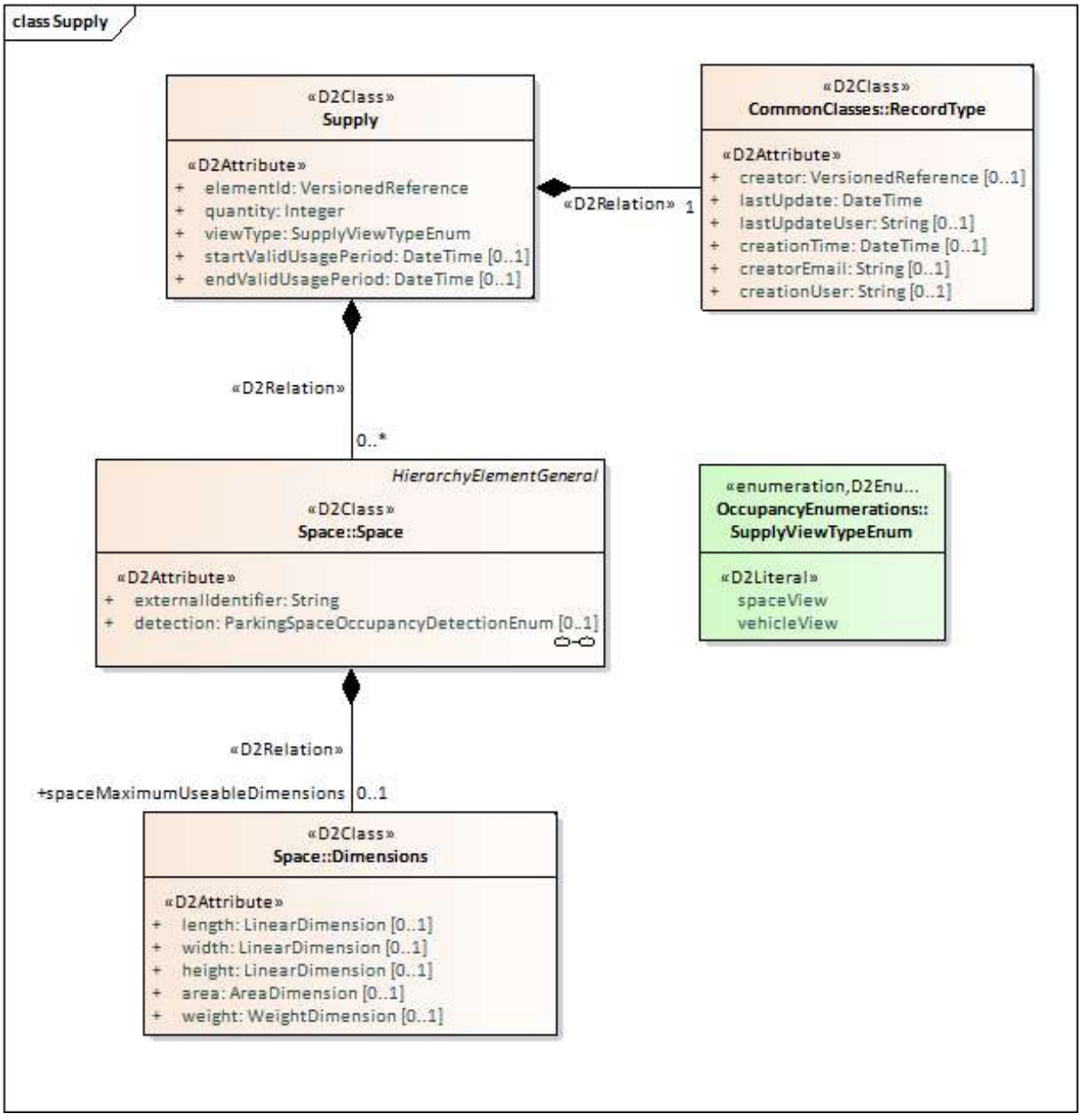

Figure 18 - Supply class model

Figure 18 shows the supply side of the <Occupancy> namespace. Through the HierarchyElementReference class, the Supply data class are linked to an identifiable element within the Place hierarchy. The Supply class defines the supply quantity as an integer, relevant for a defined period. If the Supply.startValidUsagePeriod attribute is not set, the SupplyQuantity is considered to be relevant from the present time until the Supply.endValidUsagePeriod date/time. The SupplyViewType attribute enables identification of different perspectives of specifying supply:

- spaceView - means that each space is physically marked and identifiable.
- vehicleView - means that vehicle capacity in an area is estimated based on calculating a capacity via a defined method. (e.g., kerb length divided by a length of vehicle, or the holding capacity of a specific parking area). The Distributing Party decides on the appropriate means to calculate the vehicleView (e.g., length of vehicle to use).

The Supply class can be linked to one-to-many identified Spaces whose characteristics can be defined through the Dimensions class which provides maximum useable dimensions for a space.

Demand

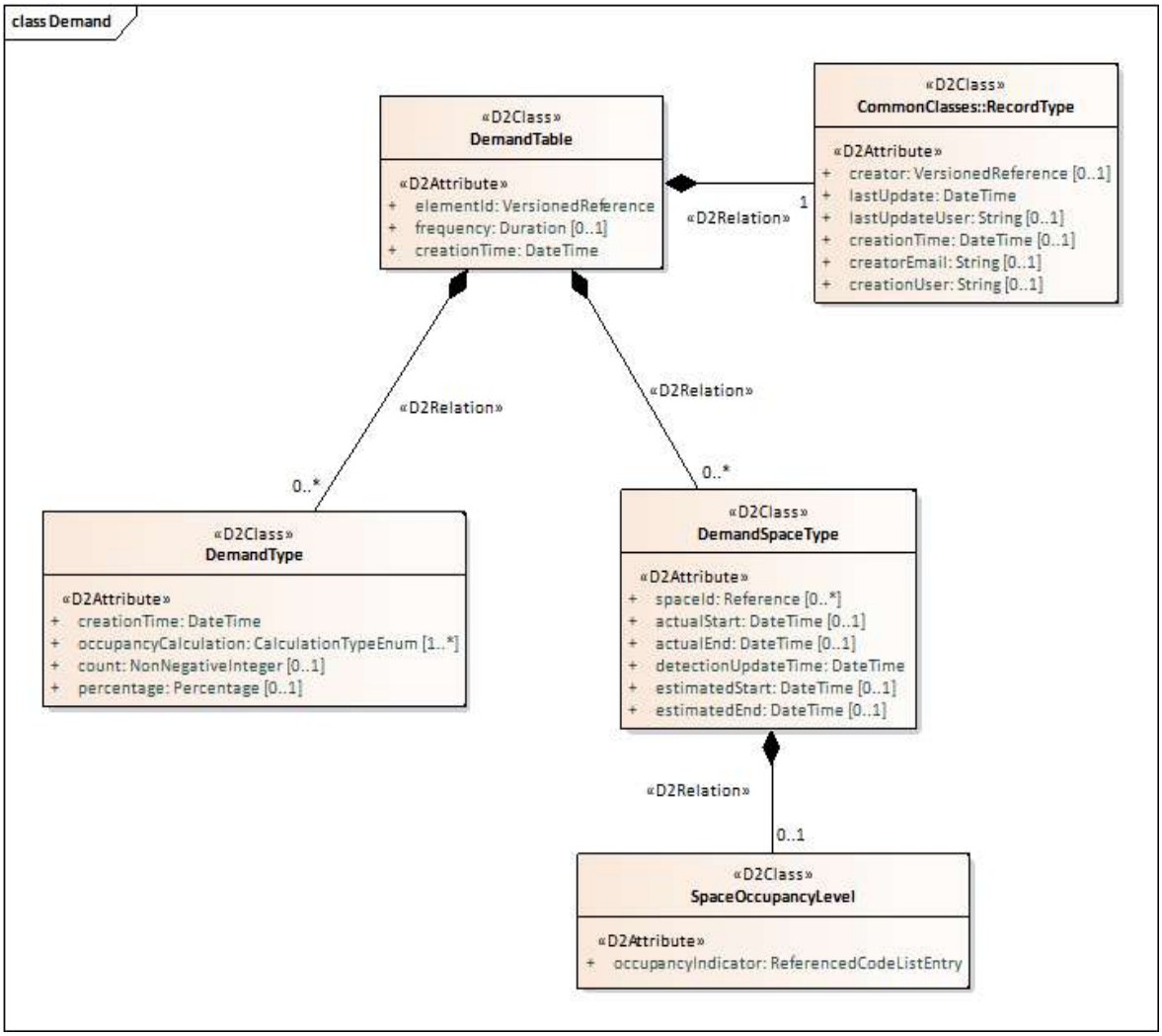

Figure 19 - Demand class model

Figure 19 shows the demand side of the <Occupancy> namespace. Through the HierarchyElementReference class, the DemandTable class are linked to an identifiable element of the Place hierarchy. Demand being the quantity of available capacity in use.

The DemandType class indicates the timestamp of the associated demand characteristics, and the frequency that the demand characteristics are updated.

The DemandType class provides the numeric value, or the percentage of this demand compared to the total supply and its specific date/time stamp of when the value or percentage is determined. Only one numeric value or percentage shall be provided. The DemandType class also supports identification of how the demand is calculated through the occupancyCalculation attribute.

The DemandSpaceType class enables the demand to be defined relative to a specific identified Space. This can be used to indicate historical demand with the actualStart and actualEnd being used to provide actual times of occupancy. Current demand uses either the actualStart or estimatedStart to define the known or estimated start of the Session respectively. Future demand can be defined by use of the estimatedStart and estimatedEnd to define the estimated Session.

Enumerations (for Occupancy)

The following enumerations are defined in the <Occupancy> namespace – see Figure 20.

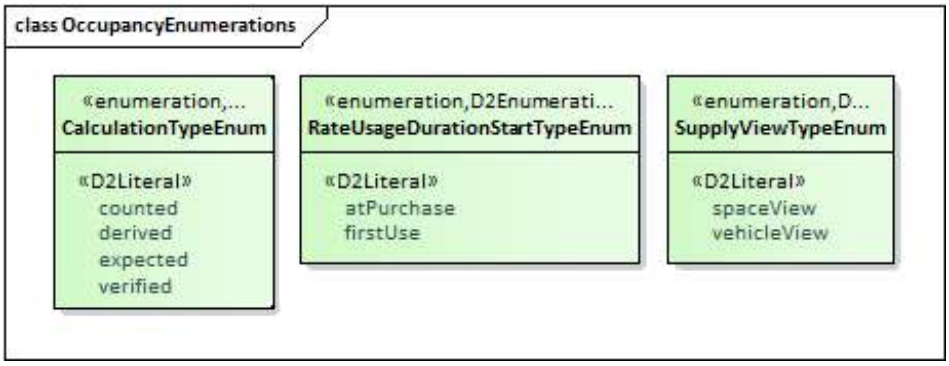

Figure 20 - Enumerations in the Occupancy Domain

External Codelists (for Occupancy)

No specific external code lists are defined in <Occupancy>. For general external code lists, see Section 6.5 – The PkCommon domain.

Rates

Introduction to Rates

This section describes concepts used to define rates and tariffs for a specified Place element. Rates can be associated to a variety of activities including parking, delivery, loading, unloading, crossing an access point, etc. These are defined in the <Rates> namespace of the data model.

Rate Tables

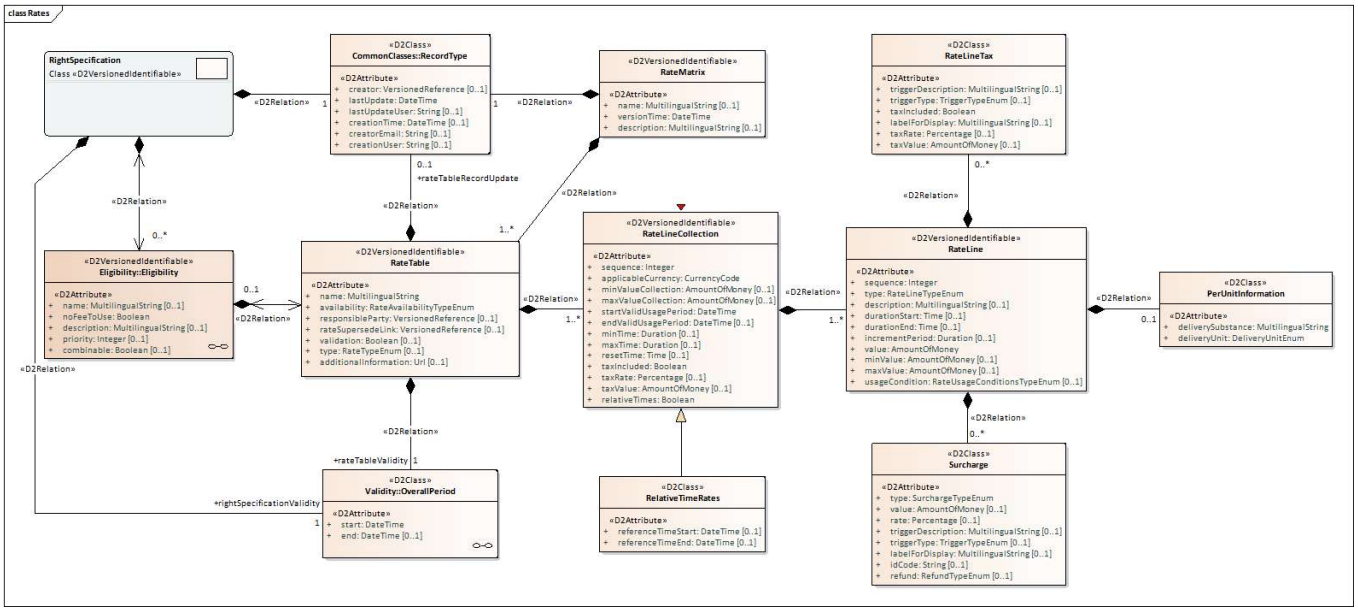

Figure 21 - Rates domain class diagram

Figure 21 details the data structure related to <Rates>, which are used to convey information concerning tariffs and fee schedules relating to an identifiable element of the Place hierarchy through use of the HierarchyElementReference.

A RateTable represents a set of charges that are applied to a single RightSpecification for parking or other operating activities (e.g., delivery permits, rideshare access, etc.) at the Place. Examples of a RateTable are a weekday charge rate scale in a public, multi-storey car park or evening rates and weekend rates.

The Validity class associated to the Rate table defines two important time periods related to a Rate Table:

- overallPeriod.start and overallPeriod.end identifies when the Rate table and associated attributes are valid for usage. This enables an entity to schedule the release of new ratetables and communicate when current ratetables will expire. If the overallPeriod.end is not set, the RateTable is considered to be valid until it is replaced

- period.startOfPeriod and period.endOfPeriod define when the fees and tariffs defined in the RateTable are available during a specific period to be applied to a Session.

To support the transmission of a RateTable that may contain multiple charging elements; a RateTable contain one to many RateLineCollection(s). An example could be a RateTable that

contains a flat rate fee for a reservation, plus a tiered time-based rate structure for charging during the Session. Each RateLineCollection represents one of those charging elements. A RateLineCollection is constructed of one-to-many RateLine.

The RateLine concept is flexible and supports a range of different characterisations, which include:

Flat rate – where the Rateline is active, the applied fee charge is a flat rate, unrelated to the duration or timing of the parking or other type of Session; an example is a flat rate reservation fee for a Session. Flat rate RateLine are defined by use of defining a value, but no use of the durationStart, durationEnd or incrementingPeriod attributes.

Flat rate tier – where the applied fee charged is charged in full if the parking or other type of Session indicates that that specific RateLine is active. Example: in the second hour of a Session the fee is $1.00 (USD) – for any part of that hour. For a flat rate within a tier, the RateLine defines the time boundary of the tier by use of the durationStart and durationEnd attributes and the value attribute to define the charge amount. If used, the incrementingPeriod attribute, in this case, shall be the same as the period between the durationStart and durationEnd (i.e., there is one increment). Charging is assumed to occur at the start of each increment.

IncrementingRate – where the applied fee charged is related to the duration of the specific tier activated by the parking or other type of Session. This charge type supports a RateTable that applies for short incrementing periods or time-based small increments of charge. Example: in the second hour of a Session, charging is done at a rate of 0.05€ every 3 minutes. For an incrementing rate within a tier, the RateLine defines the time boundary of the tier by use of the durationStart and durationEnd attributes. The incrementingPeriod and the value attribute indicate the charge amount of each increment (e.g., 0.05€ each 3 minutes. Charging is assumed to occur at the start of each increment.

Per Unit – where the applied fee charged is based on a per unit of measure where the unit of measure is user defined. The user defined unit must be specific and explicit in definition. This user defined unit of measure could be kWh to collect fees for electric vehicle charging or it could be a passenger count to support a fee structure similar to $5 per passenger in a car.

Under most circumstances the start and end of charging periods are fixed and relative to local time (e.g., between 8am and 5pm weekdays). In some instances, the charging period and related tiers may be relevant to a specific event. The reference of a charging period time to another event is indicated by the use of the relativeTimes set to TRUE in the RateLineCollection and the use of the RelativeTimeRates class. All times are defined relative to the referenceTimeStart, which is the start time of the reference event

The applicable currency is defined in the RateLineCollection.

Individual RateLine support the identification of whether tax is applicable within the defined RateTable or applied in addition to the defined Rate. The value of tax, if included, can be specified as either a monetary amount or a percentage rate. Taxes may also be applied to a RateLineCollection in a similar manner. It is common practice for taxes to be applied at the RateLine level – for example the application of Value Added Tax (VAT) in Europe which is added to a basic parking fee and declared in the cost of the parking to the end user.

A RateLineCollection indicates whether the child RateLine are a chargeable tariff or represent a surcharge, which may be partially or fully refundable.

A RightSpecification is linked to Eligibility without reference to a RateTable when no RateTable is applicable (i.e., the RightSpecification has eligibility constraints, but no cost is associated).

Eligibility

Each RateTable is applicable to a singular set of criteria and a RightSpecification. The specification of the qualifying criteria is specified using Eligibility, see Figure 22. Typically, Eligibility is defined at the RightSpecification, however if within a single RightSpecification access to specific RateTable requires further qualification, additional Eligibility criteria are defined at the RateTable.

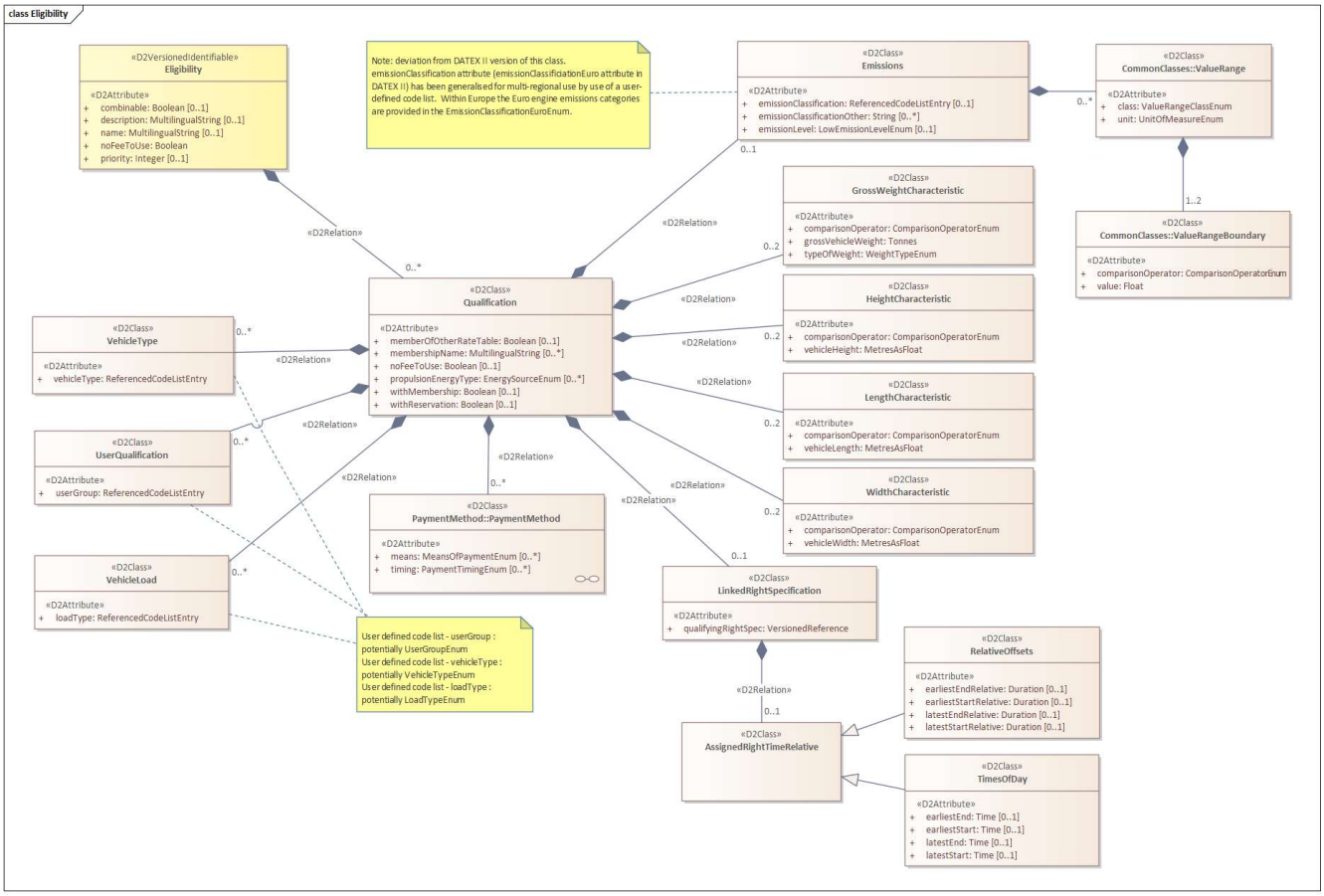

Figure 22 - Eligibility class diagram

Eligibility is specified as a collection of individual Qualifications. A Qualification is specified as a test with any of the attributes in the Qualification class set. In addition, Qualifications can be specified for other criteria including vehicle's emissions classes, maximum and/or minimum values for gross vehicle weight, load types (passenger, goods, hazardous materials), heights, widths, and length measures.

Eligibility may be related or defined by membership. Typically, the Qualification in this case is defined by the withMembership attribute set to TRUE and the membershipName attribute set to the names of the relevant memberships (e.g., J-Park frequent users club, shoppers, cinema attendees, event ticket holders, military, industry specific vehicle, etc.). Eligibility may also be based on a user defined codelist. This allows an entity to define specific lists relevant to their operations to be shared. User defined code list are available for Emissions, UserGroup, VehicleLoad and VehicleType.

Furthermore, Eligibility may be defined in relation to a specified active use of a RightSpecification (linkedRightSpecification attribute). The attribute linkedRightSpecification identifies the RightSpecification, when used in an AssignedRight associated to a Segment, qualifies the user to an additional RightSpecification or RateTable (i.e., an active use of a RightSpecification, such as parking a car). This attribute enables an entity to define a RateTable that is eligible for use if a RightSpecification with a specific RateTable has been used. (Example: because a parker paid to park this morning in Zone 1, the parker is eligible to park in the afternoon in zone 1 or Zone 2.)

Eligibility may be specified on a time-restricted basis (AssignedRightTimeRelative) with respect to an existing held RightSpecification (which is identified via possession of an AssignedRight). AssignedRightTimeRelative identifies the duration of time that a specific AssignedRight can be referenced to access a RateTable via the RelativeOffsets class (example, Rightholder is eligible to use a new specific RateTable if they have used a RightSpecification within 2 hours of requesting access to the new Ratetable. The time constraints are defined in the AssignedRightRelative and / or specific times of day the offset). When access to the qualification criteria exists can be defined via TimesOfDay class (example Rightholder is eligible to use a new specific RateTable if they have used a RightSpecification within 2 hours of requesting access to the new RateTable between 8am and 2pm.)

For emissions-like conditions, additional condition modelling capability has been incorporated. The ValueRange class enables different forms of condition and unit of measure to be defined. Linked to the ValueRange class the ValueRangeBoundary class may optionally be defined, with up to two instances, to define the permissible range by use of upper and lower bounds of the ValueRange.

Enumerations (for Rates)

The following enumerations are defined in the <Rates> namespace – see Figure 23.

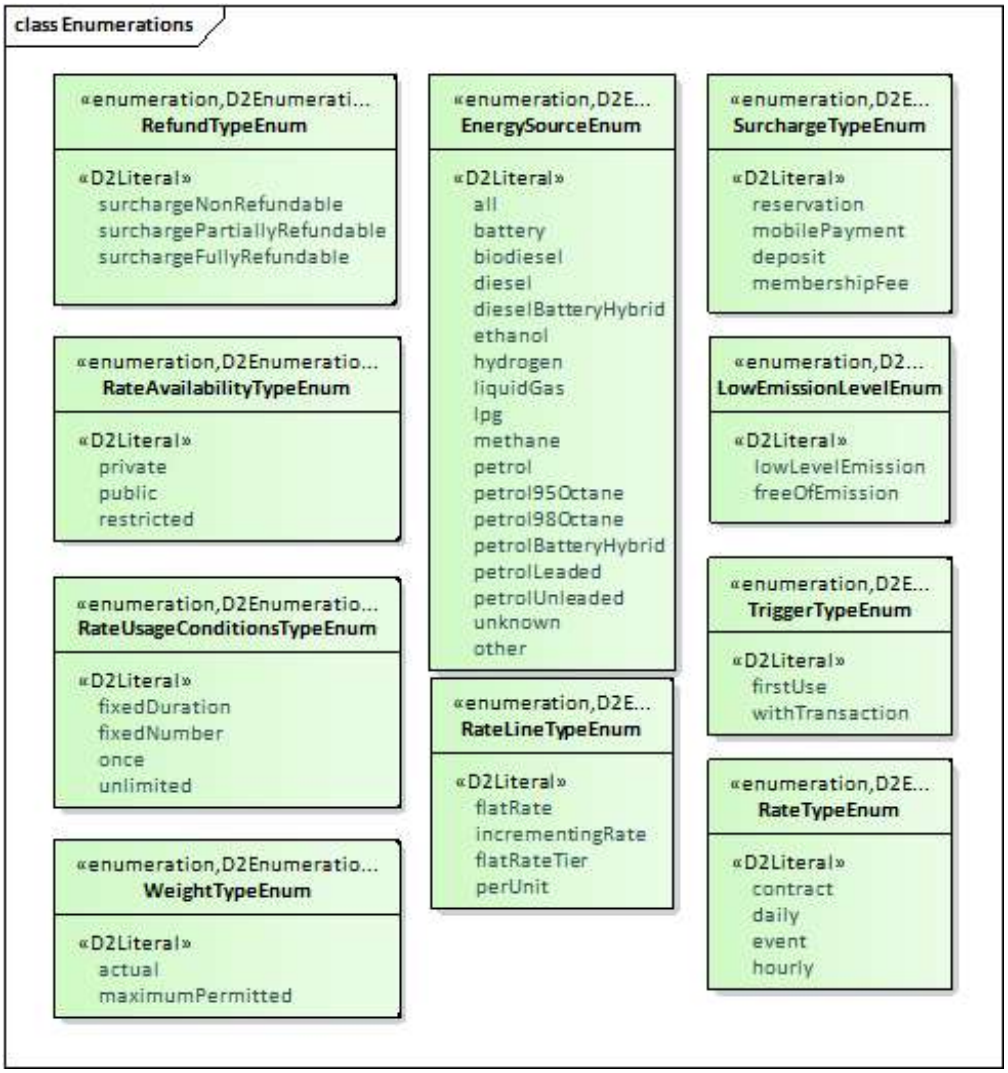

Figure 23 - Enumerations in the Rates Domain

External Codelists (for Rates)

Multiple external code lists exist for <Rates> which enables a place owner to define a custom list for the following:

- Emissions: associate to a country or geographic standard list of emission codes for vehicles
- UserGroup: define a list of relevant user groups for the place that define qualification criteria for a RateTable or RightSpecification.
- VehicleType: define a list of relevant vehicle type for the Place that define qualification criteria for a RateTable or RightSpecification.
- VehicleLoad: define a list of relevant vehicle load for the Place that define qualification criteria for a RateTable or RightSpecification.

For general external code lists, see Section 6.5 – The PkCommon domain. Also see Annex D.

Right

The data concepts with the <Right> domain is presented in Figure 24. Figure 25, Figure 26 and Figure 27 provide sub-models for the <Right> domain.

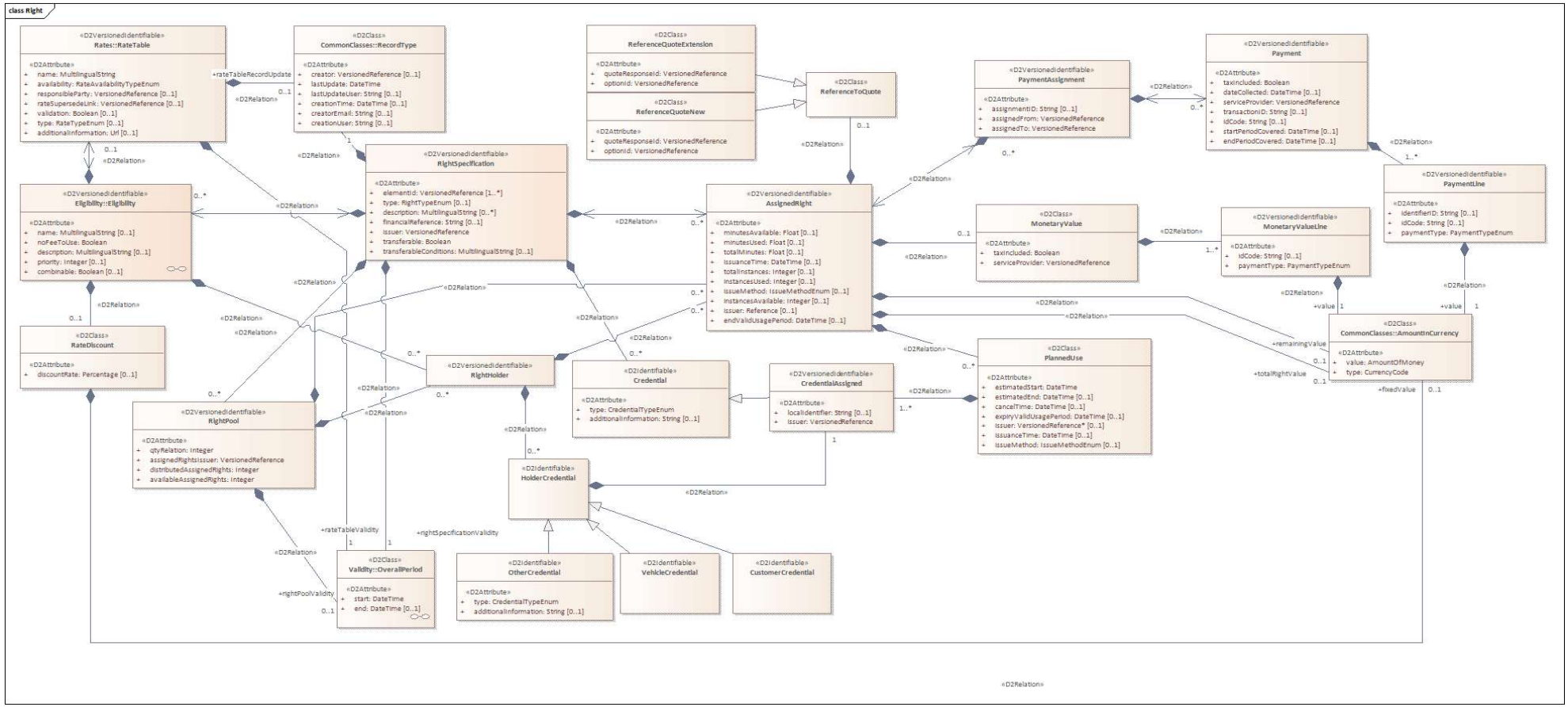

Figure 24 - Right domain model (overview)

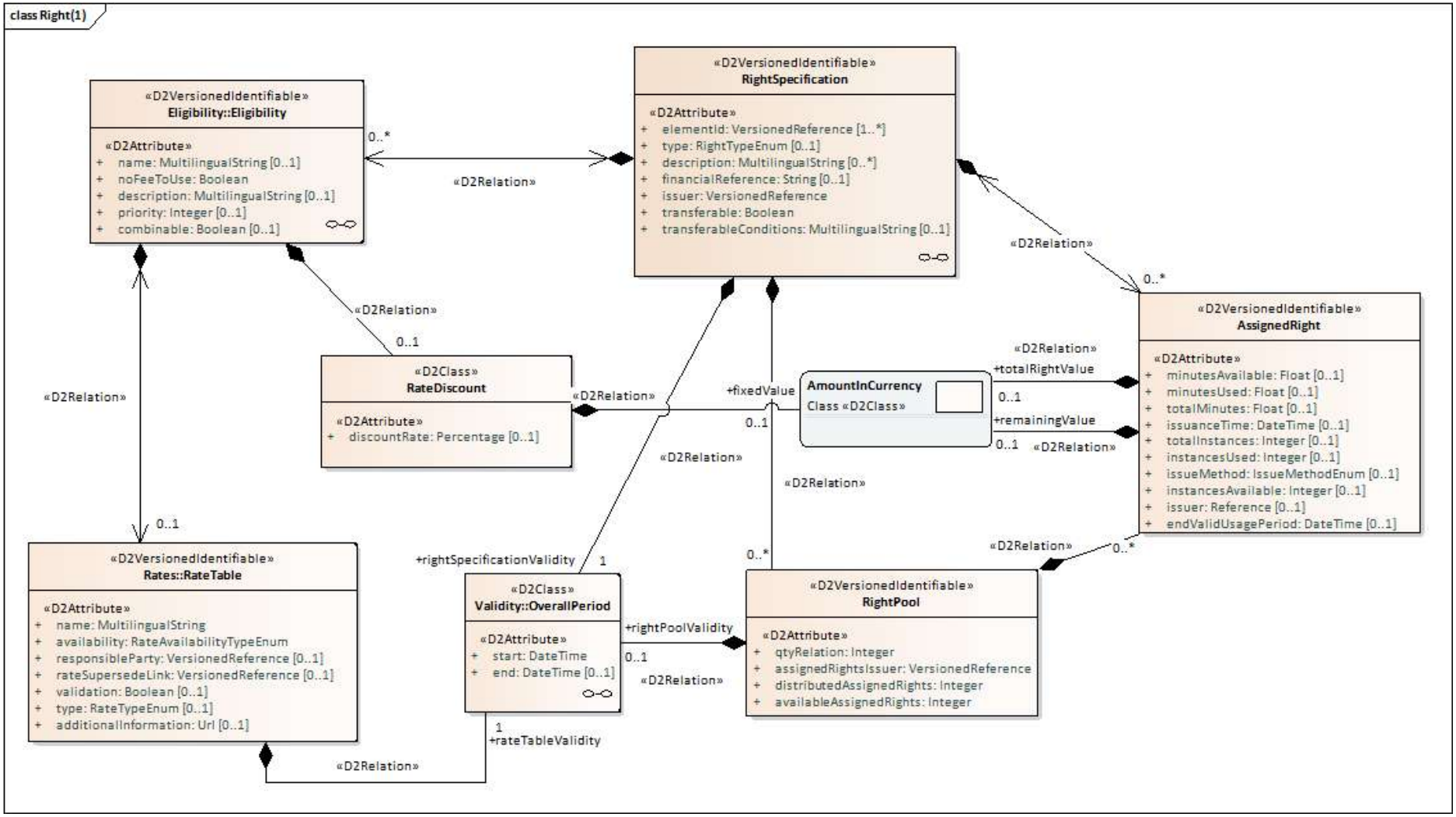

Figure 25 - Right domain model – submodel 1

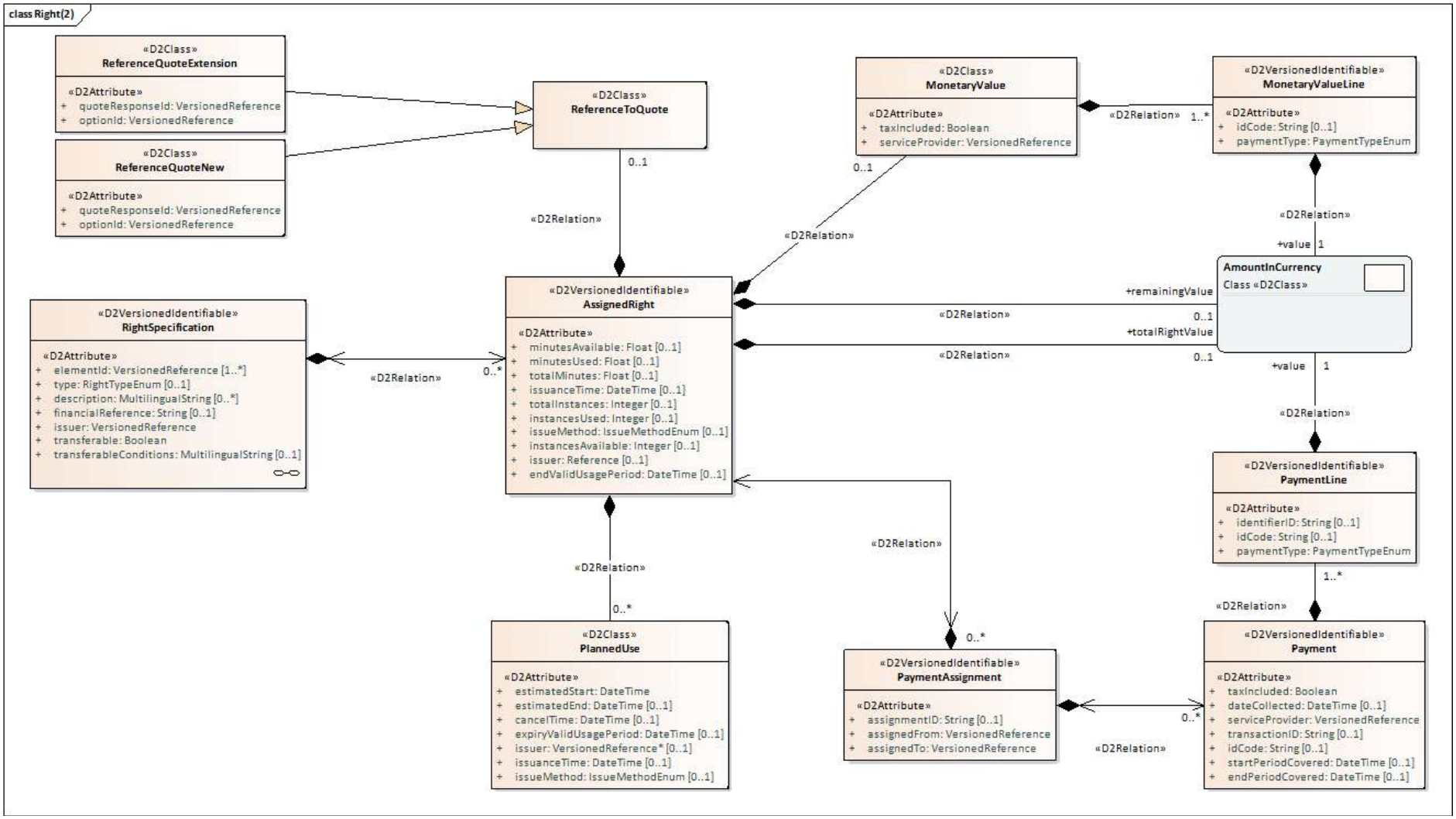

Figure 26 - Right domain model – submodel 2

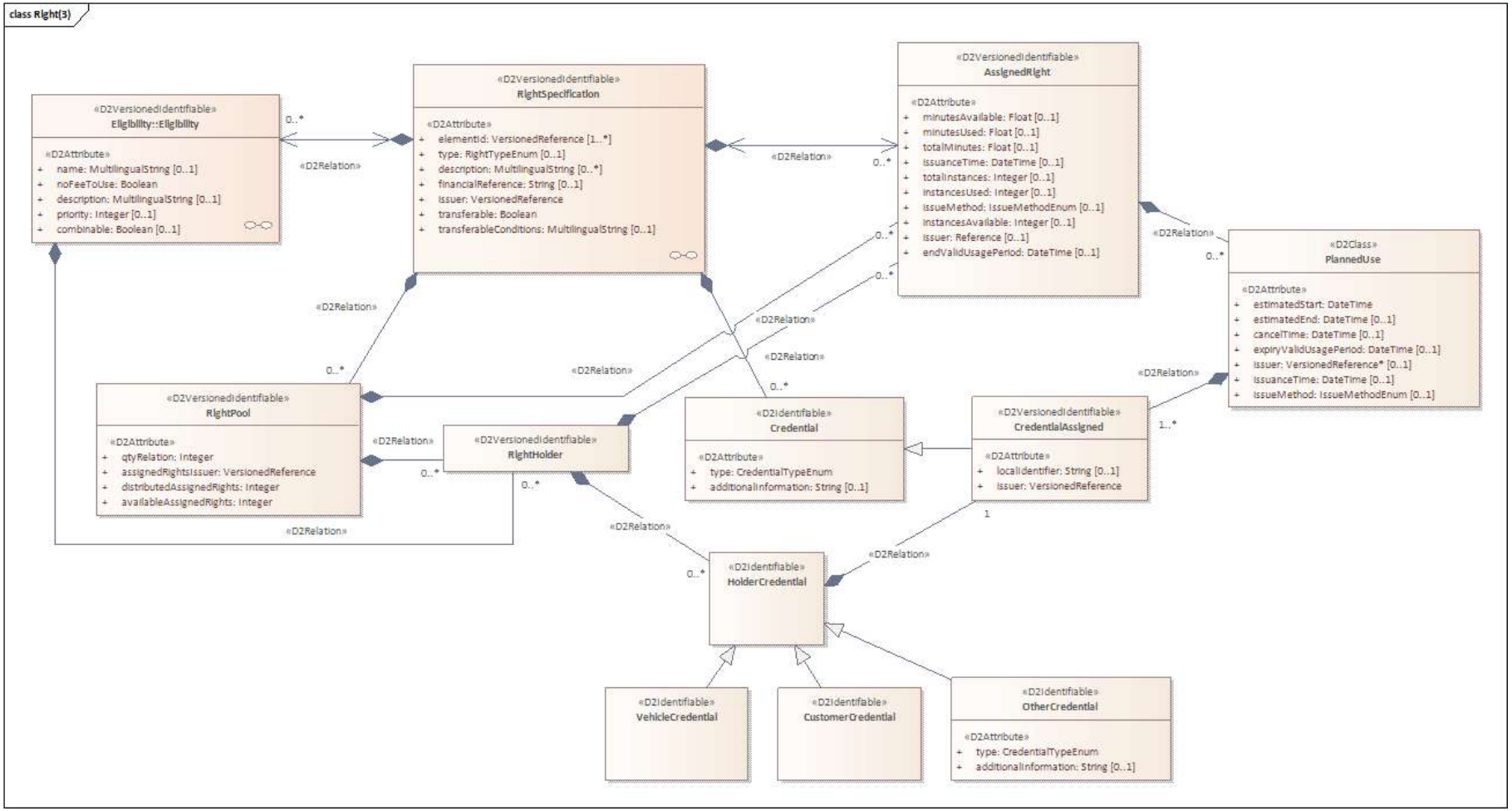

Figure 27 - Right domain model – submodel 3

Figure 28 provides an illustration of key data attributes to RightSpecification.

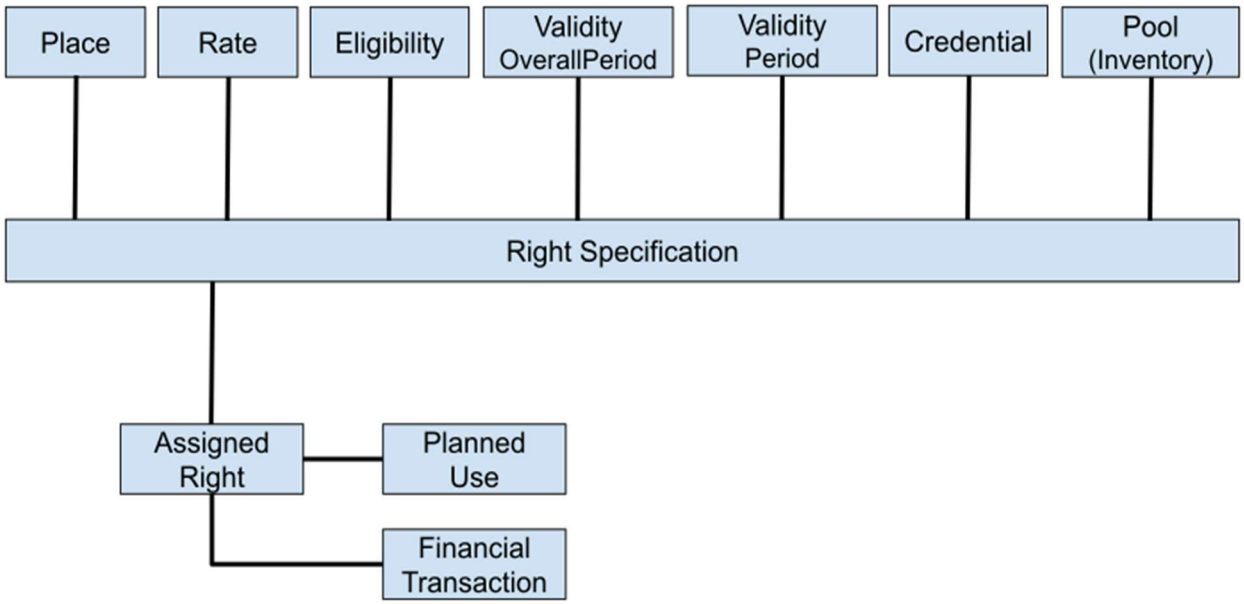

Figure 28 - Simplified RightSpecification and key data relationships

These are defined in the <Right> namespace of the data model. The <Right> domain includes the data concepts that define how a facility / place owner or manager authorizes the use (e.g., park, delivery, pick up, etc.) of a specific Place to various users or vehicles via a Credential at a specific set of RateTable(s), if a RateTable applies.

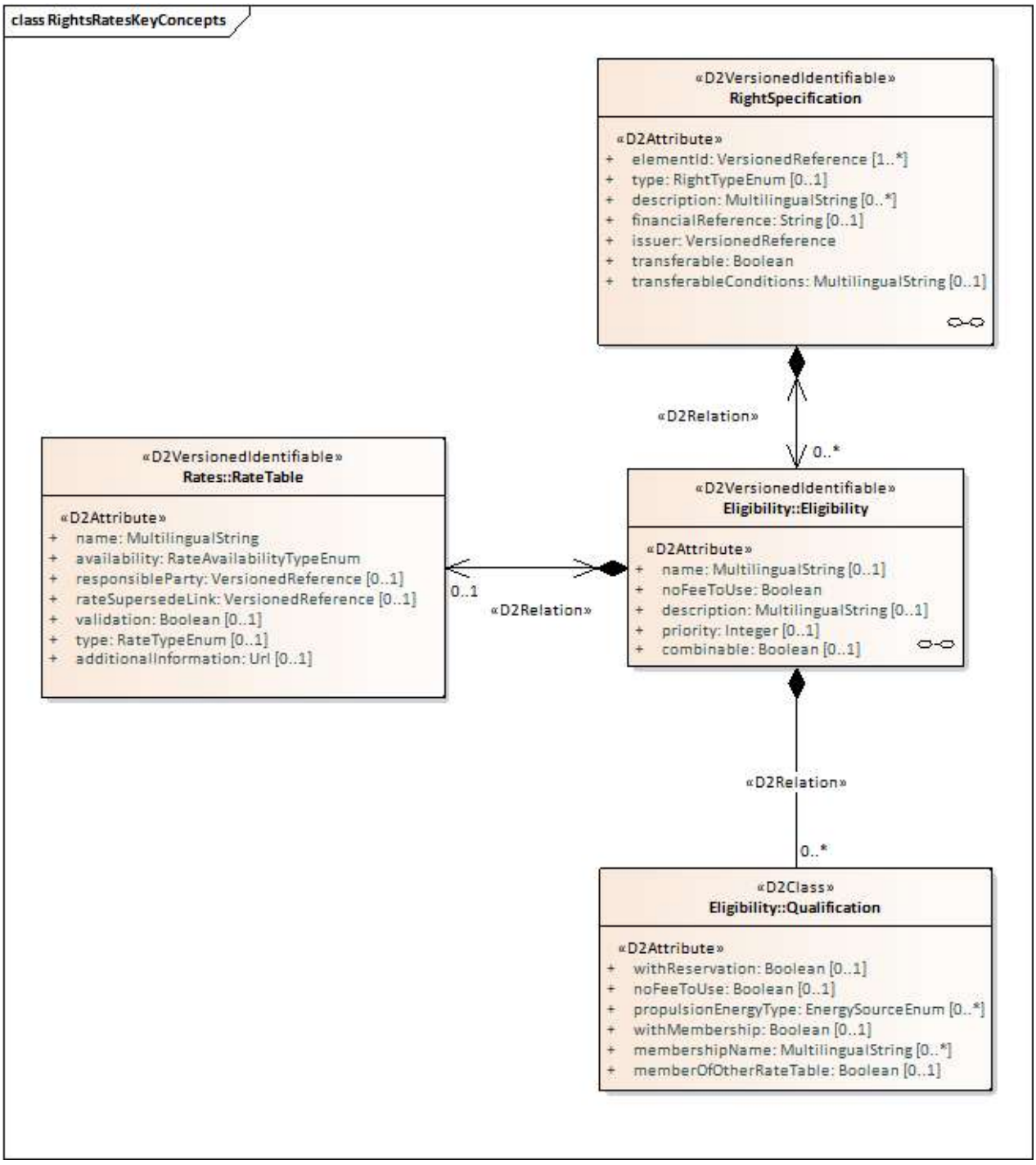

Figure 29 - Right - Rates key data concepts

Figure 29 shows the relation of Rightspecification, RateTable and Eligibility. To use RightSpecification and a related RateTable a user or vehicle shall meet the eligibility criteria for both the RightSpecification and the RateTable. A Rightspecification defines the permission that is being granted to a vehicle or user to perform a specific activity (park, deliver, pick up, etc.) or to use a specific service in an IdentifiedArea. The RateTable defines how the RightSpecification is priced. Eligibility defines the qualification criteria to use a RightSpecification and RateTable combination.

Note: A RightSpecification might be available to many, while specific RateTables associated to the RightSpecification maybe limited based on specific qualifications. Example: A RightSpecification to park in a facility may have the following set of RateTables:

- Public RateTable: qualification is limited to passenger vehicles
- Resident RateTable: qualification is limited to passenger vehicles and users that are members of the Resident group (i.e., live in the building)

The RightSpecification defines the operating parameters for parking or related mobility activities (delivery, pick-up/drop-off, electric vehicles only, etc.)

The RightSpecification is best described as the template of a right (permission to do a specific action) as defined by the place owner. A RightSpecification is granted to a specific RightHolder by an entity identified by the AssignedRight.issuer attribute. The AssignedRight.issuer can be the place owner, a reservation service, or other entities. When identified as the AssignedRight.issuer, the entity is authorized to sell or distribute the RightSpecification on behalf of the place owner.

When a RightSpecification is granted to a specific RightHolder, an AssignedRight is created. The AssignedRight includes the information from the RightSpecification as well as specific information related to the RightHolder (expiration of the AssignedRight, number of uses, etc.). In some cases, an AssignedRight can include the ability to perform a specific RightSpecification multiple times (example, Rightholder has prepaid for five (5) parking events). When a specific, future use of the AssignedRight is initiated, a PlannedUse is generated.

RightSpecification

- Has a unique identification within a place.
- Has a description of the permission (rightspecification) being granted.
- Has an expiration: the date/time when the permission is no longer valid for any user as defined by the place owner.
- Has a creator: the entity, typically the place owner that defines and authorizes the permission.
- Has authorized credential types to identify methods of proof of holding an AssignedRight.

Data associations to a RightSpecification or AssignedRight:

1. Place: defined earlier in the document. Place defines where a RightSpecification is valid. A RightSpecification can be associated to multiple Places. When multiple Places are associated to a RightSpecification, each place owner needs to authorize the RightSpecification and reference the proper RightSpecification IDs when sharing data.

2. Eligibility: This defines the type of vehicle, customer, or other qualification criteria that is able to access the RightSpecification. Eligibility may be associated to being a member of a group (office employee, resident, etc.), a vehicle type (electric car, Truck, passenger car, etc.), or use of a RightSpecification in a previous Session. Eligibility, in addition to defining the qualifications to access a RightSpecification, defines the Qualifications requirements to access specific RateTables. Within the Eligibility class, the ability to combine instances of RateTable within a RightSpecification is defined via a Yes / No (Boolean) definition. Either all instances of RateTable associated to a RightSpecification with a common Eligibility can be combined or they cannot. If Eligibility has the Combinable flagged YES, then instances of RateTable can be combined in a Session or Segment.

3. Rate Table: RateTable is a specific rate structure that defines how a RightSpecification is priced. A RightSpecification can have multiple instances of RateTable associated to it as

long as the instances of RateTable all apply to the same Eligibility requirements defined at the RightSpecification. Actual pricing in a RateTable may vary based on time of day.

4. Right Pool: shares the number of instances of AssignedRight that are available for use, are currently in use, or intended for use within a specific RightSpecification in specific date/time periods by a specific AssignedRight.issuer. A specific calendar (i.e., March 23, March 25, April 7) can be defined to represent the number of instances of AssignedRight available for distribution via the Validity class attributes Rightpool.activeStart and Rightpool.activeEnd or a recurring time period can be defined (Monday - Friday, Saturday-Sunday) via RelativeValidity.

5. The Validity class associated to the RightSpecification table defines two important time periods related to a RightSpecification.

    a. overallPeriod.start and overallPeriod.end identifies when the RightSpecification and associated attributes are valid for usage. This enables an entity to schedule the release of new RightSpecification(s) and communicate when current RightSpecifications will expire. If the RightSpecification.overallPeriod.end is not set, the RightSpecification is considered to be valid until it is replaced
    b. period.startOfPeriod and period.endOfPeriod define when the RightSpecification are available during a specific period to be applied to a Session. In association with RightPool, these attributes define the specific number of instances of RightSpecification that are available for use, in use, or intended for use in specific date/time periods. A RightSpecification for monthly parking for a business employee may have RightSpecification.Period.startOfPeriod and RightSpecification. endOfPeriod valid Monday through Friday from 6am to 9pm. While a resident may have a RightSpecification with RightSpecification.Period.startOfPeriod and RightSpecification.Period.endOfPeriod of Sunday through Saturday 24 hours a day.

6. RightHolder: this is a specific entity [e.g., individual, corporation, and vehicle] that is issued a RightSpecification by a place owner or AssignedRight.issuer via an AssignedRight. The RightSpecification, along with additional data related to the RightHolder, is contained in the AssignedRight class. Associated to a RightHolder are the individuals and/or vehicles able to access the AssignedRight.

    a. A RightHolder may have multiple vehicles associated to one or more credentials
        i. Example: a person that has access to more than one vehicle and uses them interchangeably.
    b. A RightHolder may have multiple users associated to one or more credentials
        i. Example: a company that provides RightSpecification to its employees under one contract, or that issues validations to customers to discount their parking.
        ii. Example: a family share a defined number of credentials

7. AssignedRight: when a place owner or an AssignedRight.issuer (authorized issuer) grants a RightSpecification to a specific person or entity, an AssignedRight is created. The AssignedRight includes the key data from the RightSpecification as well as specific data related to the RightHolder's use of the RightSpecification which includes:
    a. Approved credential,
    b. Expiration date and time of AssignedRight,
    c. Rate(s)
    d. Valid number of uses, minutes or some other value describing the quantity of use.
    e. Additional data on the entity that issued the RightSpecification to the RightHolder. The issuer can be the place owner, place manager, or an AssignedRight.issuer - 3rd party entity authorized by the place owner.

f. Additional data about when the AssignedRight was issued to the RightHolder
g. AssignedRight has one to multiple PlannedUse(s).
h. AssignedRight has financial data defined in two data concepts: MonetaryValue and Payment
    i. Monetary Value: the calculated total value of the AssignedRight based on duration, number of uses, and RateTable value. This is the expected total value of the AssignedRight.
    ii. Payment: the completed actual payments by an entity for the AssignedRight. A payment can be the transfer of funds between two entities (i.e., a payment) or a validation. A validation is a transaction where one entity has the ability to reduce or agrees to fund a portion of an AssignedRight for a specific entity.

8. PlannedUse: When an AssignedRight is "activated" for a future use, a PlannedUse class instance is created. The PlannedUse defines the proposed time to use the AssignedRight in the future and identifies the credential of the AssignedRight.

The specification defines data models to share financial transaction data as defined in the MonetaryValue and Payments classes associated to a specific AssignedRight as described above. The specification also supports sharing financial transaction data between entities for funds disbursement related to the aggregation of multiple AssignedRight(s) transactions.

Monetary Value

MonetaryValue allows an entity to share the expected value of a transaction (a specific AssignedRight) in addition to relevant attributes to further define the transaction. The additional attributes include

- AssignedRight.issuanceDate – the date and time the AssignedRight was issued and the monetary value was defined
- MonetaryValue.taxIncluded – Boolean attribute that denotes if tax is embedded in the fee presented to the AssignedRightholder or if the taxes are explicitly identified as a separate charge.
- serviceProvider – the serviceprovider presenting and collecting the MonetaryValue
- MonetaryValueLine – a data class that enables transaction details to be accumulated based on the type of payment (i.e., base fee, taxes, surcharges, etc.)
    - base fee - expected value to be received by place owner or operator (i.e., amount of calculated RateTable based on Session times)
    - tax – expected value of various taxes as defined in rate table
    - surcharges – expected value of surcharges as defined in rate table
    - discount – expected value of discounts or validations to be applied
- currency – the currency used to define the monetary value

Payment

Payment allows an entity to share the actual payment values received for a transaction. Attributes related to Payment include:

- dateCollected – the date the payment was collected
- serviceProvider – the serviceprovider collecting the monetaryvalue
- transactionID – a reference number related to the transaction
- idCode – a user defined field to identify additional details to the transaction such as Customer Account number or G/L account number
- periodCovered – note field to describe the period a payment covers (e.g. $100 for monthly parking in August)
- PaymentLine - a data class that enables transaction details to be accumulated based on the type of payment (i.e., base fee, taxes, surcharges, etc.)
  - base fee - value received by place owner or operator (i.e., amount of calculated RateTable based on Session times)
  - tax –value of various taxes as defined in rate table
  - surcharges –value of surcharges as defined in rate table
  - discount –value of discounts or validations to be applied
- currency – the currency used to define the monetary value

Using PaymentAssignment and Payment, an entity can document financial transactions related to the disbursement of aggregated transaction amounts to various entities and provide reference to the individual AssignedRight transactions included in the funds disbursement to facilitate reconciliation activities.

PaymentAssignment includes the following attributes:
- assignmentID: identification of the specific transaction for the aggregated funds disbursement
- assignedFrom: the entity sending the aggregated funds disbursement
- assignedTo: the entity receiving the aggregrated funds disbursement
- AssignedRights: the matrix of AssignedRight Ids that constitute the aggregated funds disbursement
  - Payment: as defined above

An example of the use of PaymentAssignment is when a PLACE owner disburses funds to a taxing entity for tax payments and service providers for their services fees. Each disbursement represents the aggregate amount for multiple AssignedRights

Example: PLACE collects the following payments on the listed AssignedRights

	AssignedRight #1	AssignedRight #2	AssignedRight #3	AssignedRight #4	
Base Fee	$8.36	$12.23	$10.05	$3.00	$33.64
Taxes / VAT	$1.64	$2.77	$1.95	$1.00	$7.36
Surcharge (payable to Reservation.com)	$1.00	$1.50	$1.50		$4.00
Surcharge (payable to phone.com)				$0.35	$0.35
Total	$11.00	$16.50	$13.50	$4.35	$45.35

The Placeowner makes payments to the taxing authority and its two Service providers

PaymentAssignment
Assignment ID: 1234
Assignedfrom: PLACEowner
Assignedto: TaxingAuthority
AssignedRights: AssignedRight#1 , AssignedRight#2, AssignedRight#3, AssignedRight#4
Payment
Taxincluded: N
Datecollected: October 5, 2021
PaymentLine:
PaymentType: Tax
AmountinCurrency
CurrencyValue: 7.36
Currencycode: US$

Assignment ID: 3456
Assignedfrom: PLACEowner
Assignedto: reservation.com
AssignedRights: AssignedRight#1 , AssignedRight#2, AssignedRight#3, AssignedRight#4
Payment
Taxincluded: N
Datecollected: October 5, 2021
PaymentLine:
PaymentType: Tax
AmountinCurrency
Currencyvalue: 4.00
Currencycode: US$

Assignment ID: 3458
Assignedfrom: PLACEowner
Assignedto: phone.com
AssignedRights: AssignedRight#1 , AssignedRight#2, AssignedRight#3, AssignedRight#4
Payment
Taxincluded: N
Datecollected: October 5, 2021
PaymentLine:
PaymentType: Tax
AmountinCurrency
Currencyvalue: 0.35
Currencycode: US$

F

financial data will be expanded in future releases of this document.

Enumerations (for Right)

The following enumerations are defined in the <Right> namespace, see Figure 30.

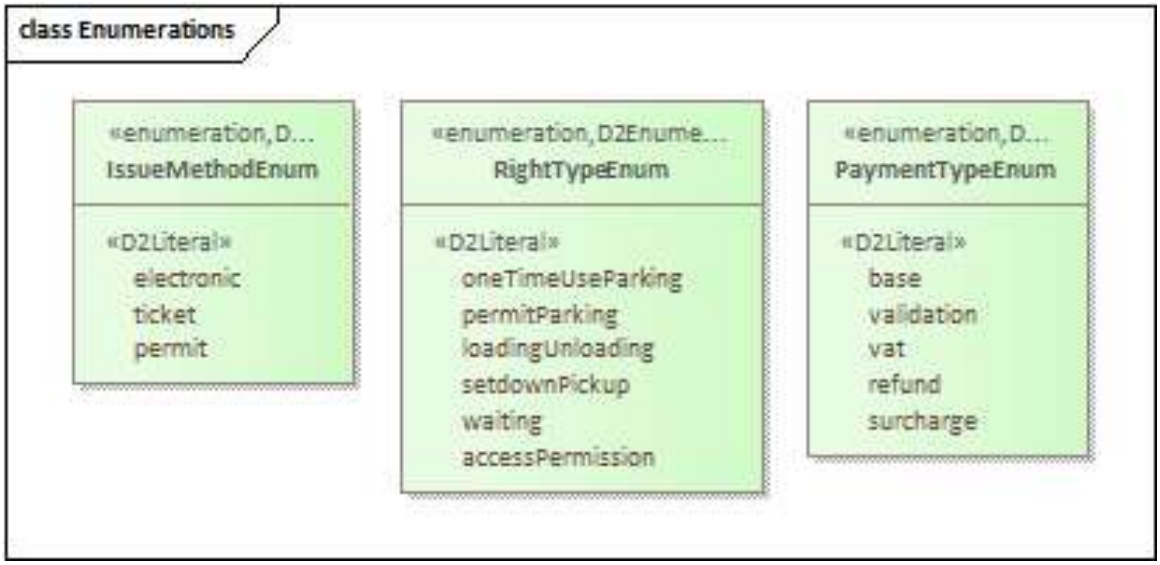

Figure 30 - Enumerations in the Right Domain

External Codelists (for Right)

No specific external code lists are defined in <Right>. For general external code lists, see Section 6.5 – The PkCommon Domain.

Session

The data concepts with the <Session> domain are presented in Figure 31.

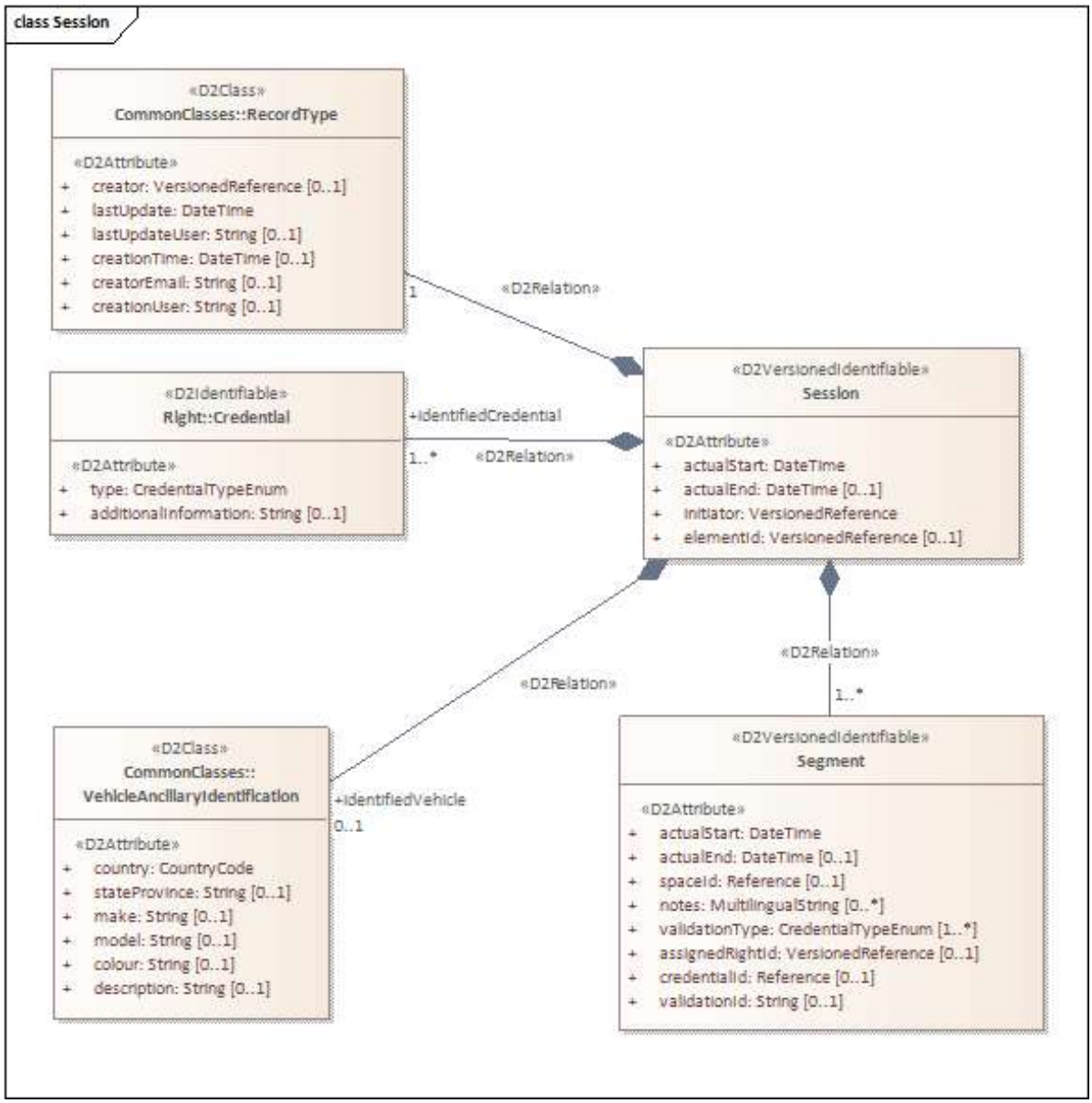

Figure 31 - Session domain model

These data concepts are defined in the <Session> namespace of the data model.
Session includes the data concepts that document the actual act of parking, or other use of an AssignedRight such as delivery, pickup, etc. A Session captures the ACTUAL use of an AssignedRight and includes start time, end time, credential, and other relevant data related to an operational transaction. A Session is not used for future activities.

A Session is broken into one or multiple Segments.

Segments enable a single Session to capture and report changes in RateTable and AssignedRight during a Session.

A Segment can only have one AssignedRight and RateTable associated to it.

Session

- Has a unique identification within a Place.
- Has a start time and end time.
- Has at least one Segment.
- Is associated to a Place.

Segment

- Is associated to a Session.
- Is associated to an AssignedRight.
- Is associated to a specific RateTable via the AssignedRight.
- Has a start and end time.
- Has a Credential via the AssignedRight.
- Has version control on the AssignedRight (i.e., a Segment starts with one AssignedRight and then is updated to a new AssignedRight at the end of the Segment, replacing the original).
- Has a Place reference or Space ID.
- Can be associated to one or multiple Observations.

Enumerations (for Session)

No specific enumerations defined in <Session>.

External Codelists (for Session)

No specific external code lists are defined in <Session>. For general external code lists, see Section 6.5 – The PkCommon Domain.

Observation

The data concepts with the <Observation> domain are presented in Figure 32.

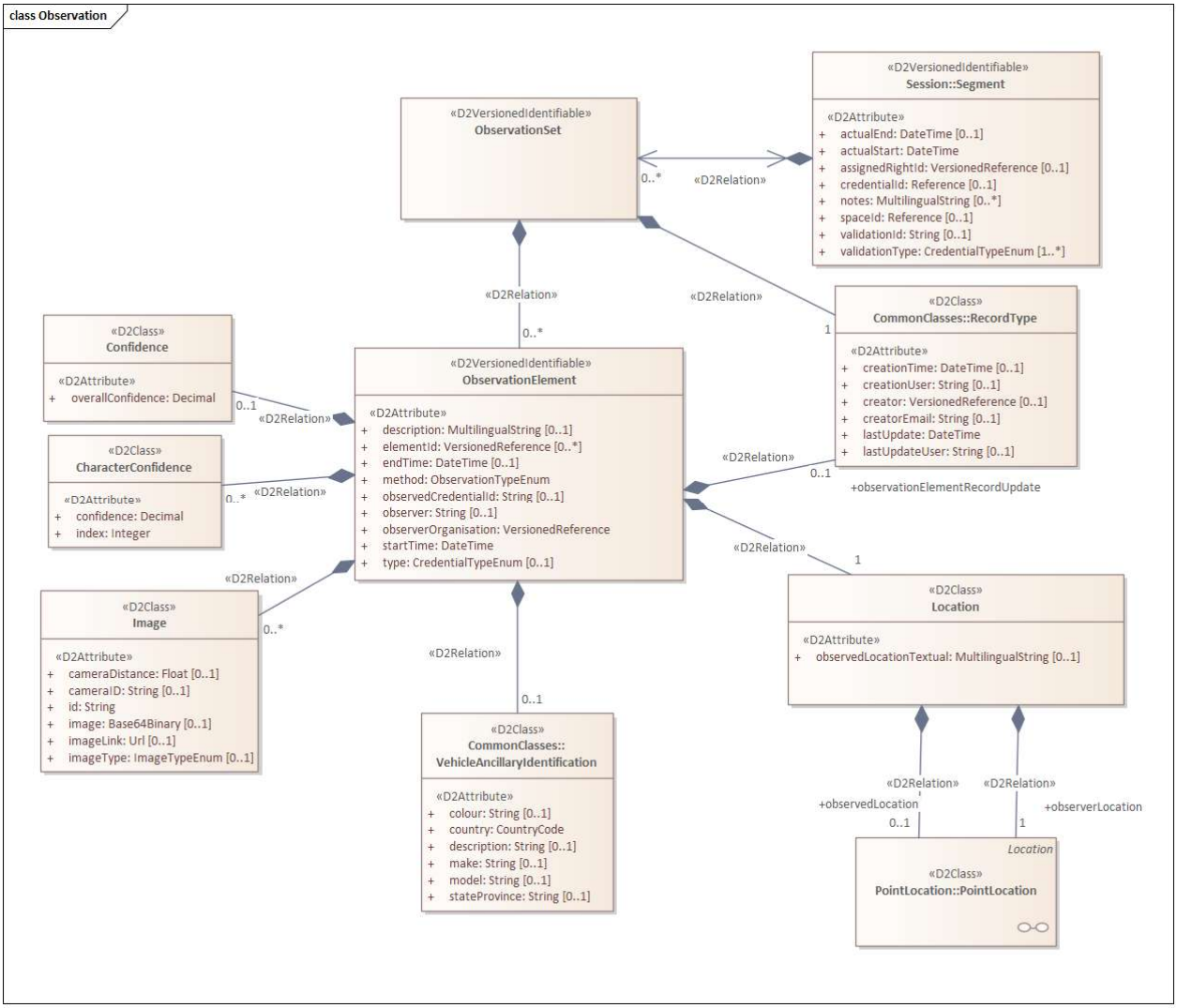

Figure 32 - Observation domain model

These data concepts are defined in the <Observation> namespace of the data model.

Observation data includes the data concepts that document the Observation of an entity, individual, vehicle, etc. in a Place performing an action. Observations can be visual or electronic (e.g., ALPR read, RFID read, etc.).

Observation:

- has a date/time when the observation is made.
- has a location where the observation is made.
- has an association to a credential observation.

Credential Observation:

- has a reference identification of the Observation.
- has a method of observation (ALPR, visual, ticket, RFID, etc.)?
- has an observer name (who made the observation).
- has a credential type observed (license plate, tag, hang tag).
- has the credential identification observed.

Enumerations (for Observation)

The following enumerations are defined in the <Observation> namespace, see Figure 33.

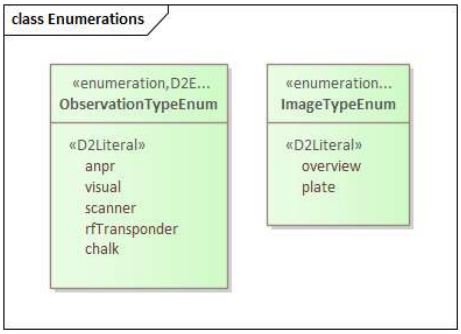

Figure 33 - Enumerations in the Observation domain

External Codelists (for Observation)

No specific external code lists are defined in <Observation>. For general external code lists, see Section 6.5 – The PkCommon Domain.

Quote

This clause describes the concepts used to define a Quote. There are two data structures to support the use of Quotes:

1) A QuoteRequest – a request for quote prices on RightSpecification(s) where Eligibility requirements are met for a specific time range. In some cases, a specific Place may be defined.
2) A QuoteResponse – a response to a QuoteRequest which provides the option(s) available to a QuoteRequest that meet Eligibility requirements. Each option in a QuoteResponse is versionable.

Note: "QuoteRequest" and "QuoteResponse" are abstract super-types (i.e., they define classes of requests and responses, and are not directly instantiated themselves).

A Quote is a request for a firm price and confirmation of availability of a specific RightSpecification, at a specific time, by a specific user. A specific Place may also be specified in the request and is required in the response. A QuoteRequest should initiate a QuoteResponse being received which includes relevant option(s) that the meet the criteria of the QuoteRequest and typically includes an expiration date/time on the QuoteResponse for the recipient to act (i.e., purchase) to obtain/or purchase an AssignedRight.

A QuoteResponse confirms the requester meets Eligibility requirements and provides the requester with a price for use of a requested RightSpecification. QuoteRequests and QuoteResponses are used between systems to allows users to evaluate a purchase of a RightSpecification.

QuoteResponses are not authorizations to use a RightSpecification and do not create an AssignedRight or assign Credentials. A QuoteResponse needs to be converted into an AssignedRight by the issuer to provide the acquiring user the right or permission to perform a specific action, as defined in the RightSpecification in a specific Place.

A QuoteRequest is not the proper means to request general RateTable or RightSpecification details about a Place. Other data structures exist to support messaging to retrieve these details. The purpose of a QuoteRequest is to enable a user to inquire about the purchase or acquisition of a specific RightSpecification at a specific time for a specific Place.

These concepts are defined in the <Quote> namespace of the data model.

QuoteRightRequest - Request for a new transaction

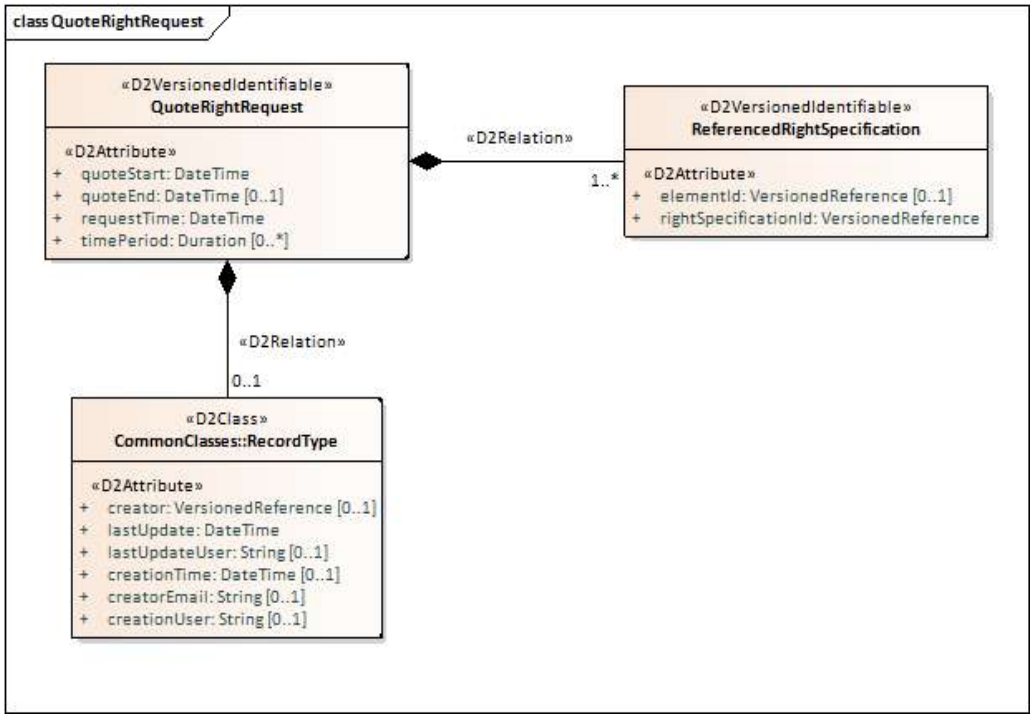

Figure 34 - QuoteRightRequest class diagram

A QuoteRightRequest class, see Figure 34, (which is versioned identifiable) references one or several RightSpecification (which is versioned identifiable). A QuoteRightRequest can include the relevant RightSpecification(s) for the desired Place to be defined. This implies that the QuoteRightRequest initiator has a current list of available RightSpecification for the Place. NOTE: A QuoteRightRequest can include requests for multiple RightSpecification in a single request.

A QuoteRightRequest includes the following data concepts in its data structure:

1) Desired RightSpecification(s) to check availability and confirm Eligibility
   a) A QuoteRightRequest can include several RightSpecification(s)
   b) At a minimum, one RightSpecification is required
2) Desired Place(s)
   a) A QuoteRightRequest can include several Place(s). Place is optional.
3) Targeted period for intended use of a relevant RightSpecification as defined by options:
   a) A quoteStart [1] and a quoteEnd [1]
   b) A quoteStart with one or several timePeriod durations – which supports the idea of retrieving quotations for a range of periods with a defined start time (e.g., from 09:30 am for a period of 15 minutes [and also for a period of 30 minutes, 1 hour, 3 hours, etc.).
   c) A QuoteRightRequest with quoteStart that is not defined then defaults to the time of the QuoteRightRequest received.
   d) Constraint: either quoteEnd or timePeriod shall be used.
4) Submission time of the QuoteRightRequest.
   a) requestTime is the date and time in UTC, the sending party sends QuoteRightRequest submission.
5) Known Eligibility Requirements.
   a) Certain RightSpecification and RateTable require defined Eligibility requirements be met to be accessible by a requestor. To access Eligibility controlled RightSpecifications and RateTables, the Eligibility requirements must be included in the QuoteRightRequest.

The QuoteRightRequest can be optionally qualified with attribution using the sub-model under Eligibility (although this may not be necessary as the requesting part can collect this information prior by seeking available RightSpecification(s)).

QuoteRightResponse – Response to a QuoteRightRequest for a new Transaction

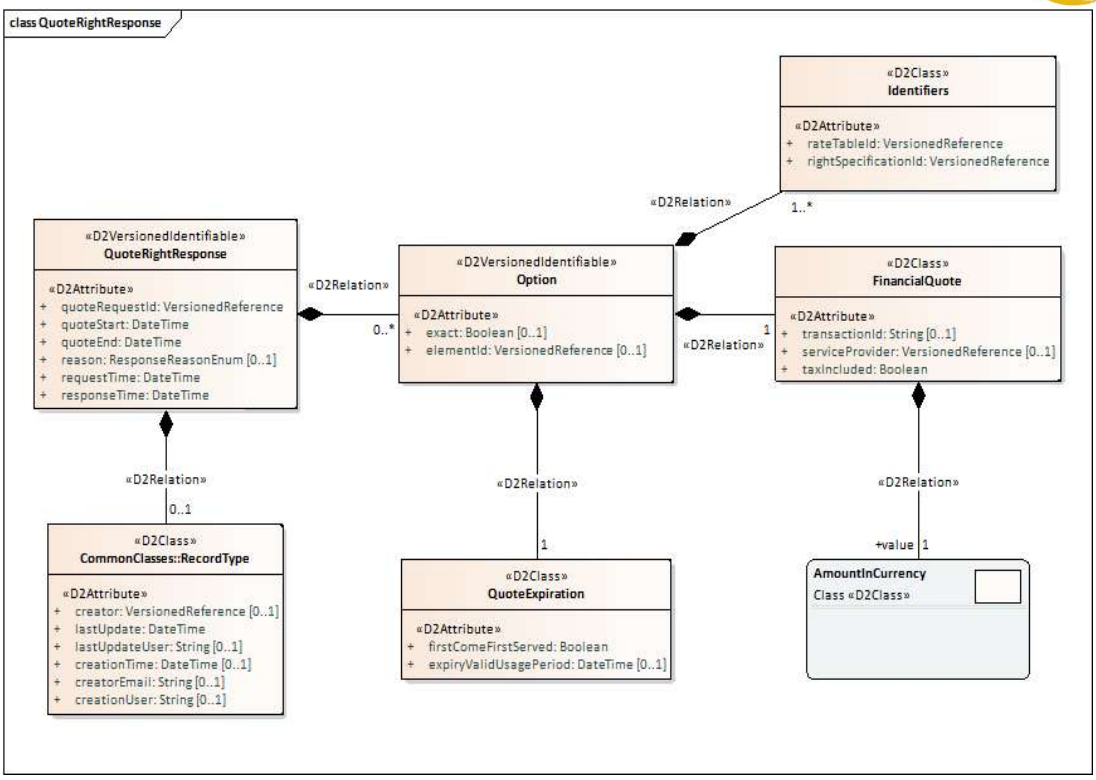

Figure 35 - QuoteRightResponse class diagram

A QuoteRightResponse class, see Figure 35, (which is versioned identifiable) is the documented response to a QuoteRightRequest and includes a time-period covered for the intended new future Session. This class documents the Option(s) presented to a requester of a QuoteRightRequest. An Option lists the specific RightSpecification(s) and RateTable(s) that meet the requirements defined in the QuoteRightRequest and for which the requester meets Eligibility requirements. When a QuoteRightResponse includes more than one possibility, the possibilities are defined as Option(s) and each Option is versioned identifiable.

A QuoteRightResponse includes the following data concepts:

- quoteRequestId – versioned identification of the QuoteRightRequest. This provides traceability between systems on the original request and the response.
- RequestTime – The time in UTC that the request from the requester was recorded/registered by the responder.
- ResponseTime – The time in UTC the response is sent by issuer to the requester.
- quoteStart – the targeted start date and time for the RightSpecification
- quoteEnd – the targeted end date and time for the RightSpecification
- Option: a specific aggregation of RightSpecification, FinancialQuote and QuoteExpiration in response to a QuoteRightRequest. Each Option relates to one available RightSpecification. Each Option is version identifiable. A specific place optionally may be defined using the ElementId attribute. Multiple Option(s) might exist in a QuoteRightResponse.
- A reason attribute is provided to optionally define a qualification to the QuoteRightResponse. This is expected to be used optionally in cases when no response Option(s) are given to provide an explanation. Possible reason includes:
    - Does not meet Eligibility requirements (vehicle, group membership, etc.)
    - RightSpecification not available during time requested
    - Inventory not available during requested time
    - No matching specification

An Option provides the details of the RightSpecification and RateTable that meet QuoteRightRequest conditions. The Option also includes

- FinancialQuote – this represents the total value of the desired RightSpecification based on an eligibility-met RateTable for the defined target date and times at the desired Place. The intent of the FinancialQuote is to provide enough information on RightSpecification value for a requester to decide on purchase, but not enough detail to complete a transaction (i.e., detailed tax and surcharge breakdowns, RateLine(s), RateLineCollection(s)). A value of zero (not null) in the FinancialQuote is acceptable.
- QuoteExpiration. This defines how long the QuoteRightResponse and Option is valid. There are two possibilities:
    - FirstcomeFirstServed: a Boolean value of Y/N shows that the availability and Option and thus FinancialQuote are available to the requester on a first come first served basis. There is no assurance that the advertised RightSpecification will be available based on actual demand.
    - ExpiresDateTime: defines a specific date and time that the Option and thus the FinancialQuote is valid. If not given, means the Option does not expire. This option allows a requestor time to complete a transaction to secure an AssignedRight.

QuoteSessionExtensionRequest – Request for an extension of an existing Session

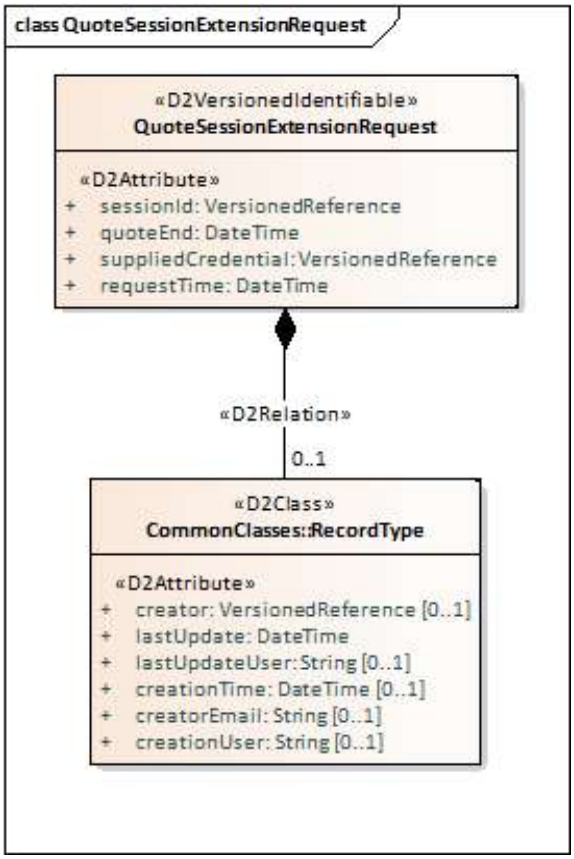

Figure 36 - QuoteSessionExtensionRequest class diagram

A QuoteSessionExtensionRequest class, see Figure 36, (which is versioned identifiable) references a specific, active Session in the request. This version of a request is a shortened version of the QuoteRightRequest used for a new transaction.
For a QuoteSessionExtensionRequest on an existing Session, the following attributes are required:

- SessionId
- quoteEnd – this is the new end time of the requested RightSpecification
- requestTime – the time in UTC that the request was sent by requester.
- supplied Credential – the current Credential in use for the current Session. A credential can be a ticket number, permit number, license plate, or other credential associated to the current AssignedRight.

The Eligibility criteria is assumed to be the same as supplied when the Session was established. If the criteria or the status of the user has changed, this is not considered an as extension of the existing Session.

The SessionId will provide reference to AssignedRight and other necessary information on the RightSpecification.

QuoteSessionExtensionResponse – Response to a QuoteSessionExtensionRequest for an extension of an existing Session

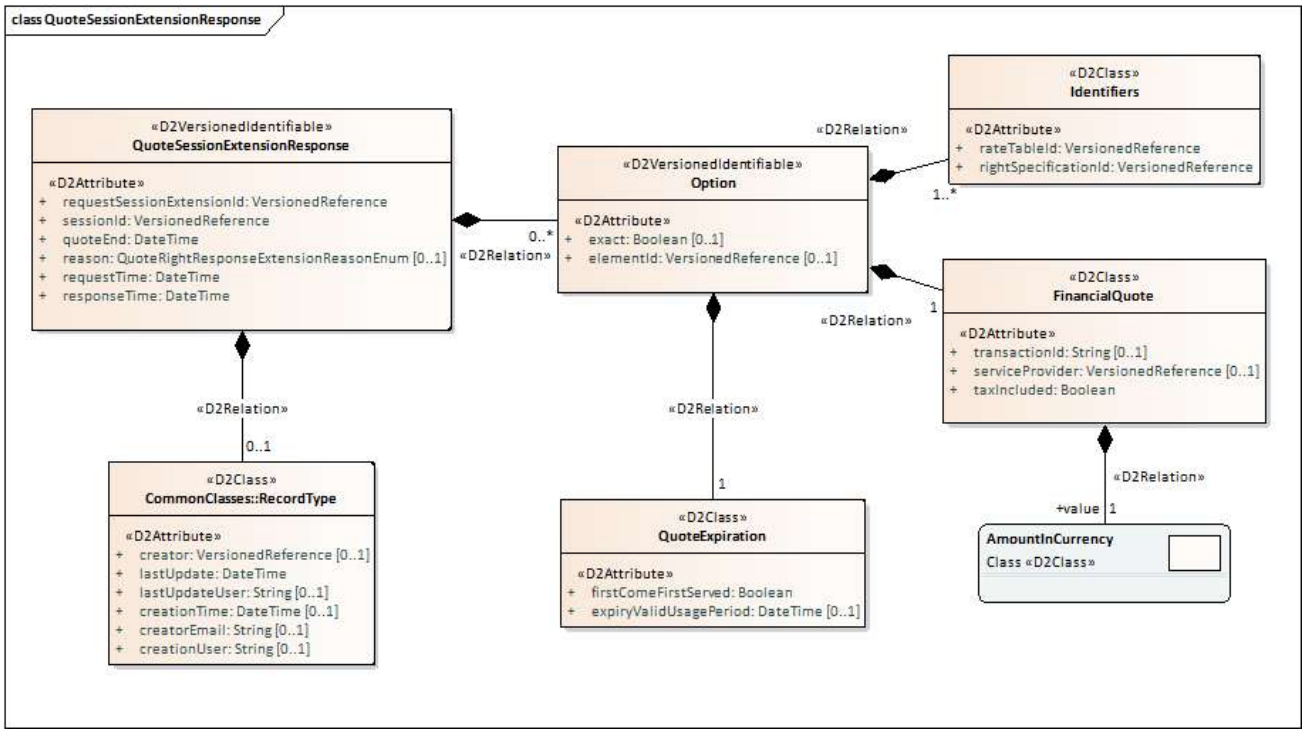

Figure 37 - QuoteSessionExtensionResponse class diagram

A QuoteSessionExtensionResponse class, see Figure 37, (which is versioned identifiable) is the documented response to a QuoteSessionExtensionRequest for the extension of an existing Session. This version of a request is similar to the QuoteRightResponse for a new transaction with the following changes.

A QuoteSessionExtensionResponse includes the following data concepts;

- requestSessionextensionID - versioned identification of the QuoteSessionExtension. This provides traceability between systems on the original QuoteRightRequest and the QuoteRightResponse.
- requestTime - The time in UTC that the request from the requester was recorded/registered by the responder.
- responseTime – The time in UTC that the response was sent by responder.
- quoteEnd – the targeted NEW end date and time for the RightSpecification
- Option - a specific aggregation of RightSpecification, FinancialQuote and QuoteExpiration in response to a QuoteRightRequest. Each Option relates to one available RightSpecification. Each Option is versionidentifiable. A specific Place optionally may be defined using the ElementId attribute. Multiple Options might exist in a QuoteSessionExtensionResponse.
    - Note: given this is an extension to an existing Session, it can be assumed that the referenced Place (elementId) will be the same as for the existing Session. The quoted RightSpecificationId may be the same as the RightSpecification for the existing Session.
- reason - provided to optionally define a qualification to the QuoteSessionExtensionResponse. This is expected to be used optionally in cases when no response OPTIONS are given to provide an explanation. Possible reason includes:

    - Existing Session not found
    - No EXTENSION possible (Not permitted per the RightSpecification)
    - Incorrect Credential provided
    - Inventory not available during requested time
- exact - The "exact" attribute indicates whether the QuoteSessionExtensionResponse full covers the requested period of stay.

An OPTION provides the details of the RightSpecification and RateTable that meet the QuoteSessionExtensionResponse conditions. The Option also includes

- FinancialQuote – this represents the total value of the desired RightSpecification based on an eligibility-met RateTable for the defined target date and times at the desired Place. The intent of the FinancialQuote is to provide enough information on RightSpecification value for a requester to decide on purchase, but not enough detail to complete a transaction (i.e., detailed tax and surcharge breakdowns, RateLine(s), RateLineCollection(s). A value of zero (not null) in the FinancialQuote is acceptable.
- QuoteExpiration. This defines how long the QuoteSessionExtensionResponse and Option is valid. There are two options:
    - FirstcomeFirstServed: a Boolean value of Y/N shows that the availability and Option and thus FinancialQuote are available to the requester on a first come first served basis. There is no assurance that the advertised RightSpecification will be available if all supply has been consumed.
    - ExpiresDateTime: defines a specific date and time that the Option and thus the FinancialQuote is valid. If not given, means the Option does not expire. This option allows a requestor time to complete a transaction to secure an AssignedRight.

A QuoteSessionExtensionResponse optionally can have zero option(s) provided – the semantics of this are that there are no options available to the QuoteSessionExtensionResponse.

Enumerations (for Quote)

The following enumerations are defined in the <Quote> domain, see Figure 38.

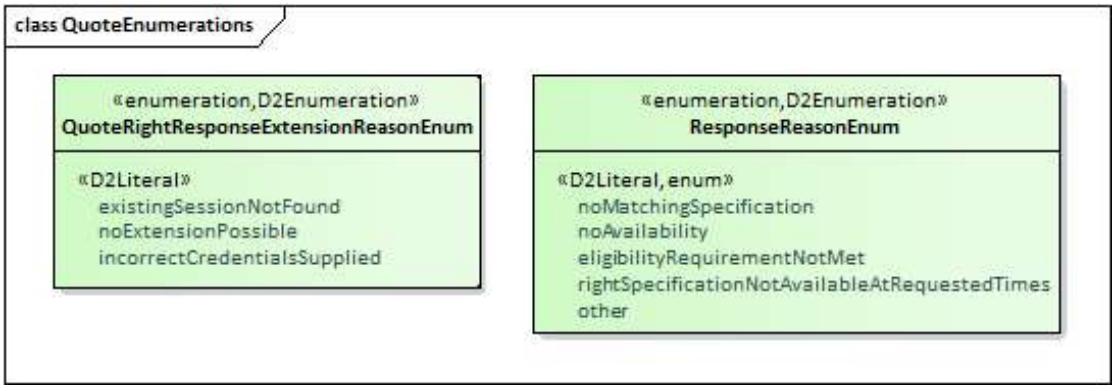

Figure 38 - Enumerations in the Quote domain

External Codelists (for Quote)

No specific external code lists are defined in <Quote>. For general external code lists, see Section 6.5 – The PkCommon Domain.

Common Elements

This clause describes concepts that are shared and commonly referenced by several of the domains of the overall data model. In general, these define elemental data types, commonly used enumerations and enumeration lists, location, contact/address and time concepts and some common classes that behave like complex data types. These are defined in the <PkCommon> namespace of the data model.

Note: where appropriate, more specific usage data types and enumerations may be defined within a specific domain of the data model, such as within the <Place> or <Rates> domains. Figure 39 shows packages and classes defined in the PkCommon domain.

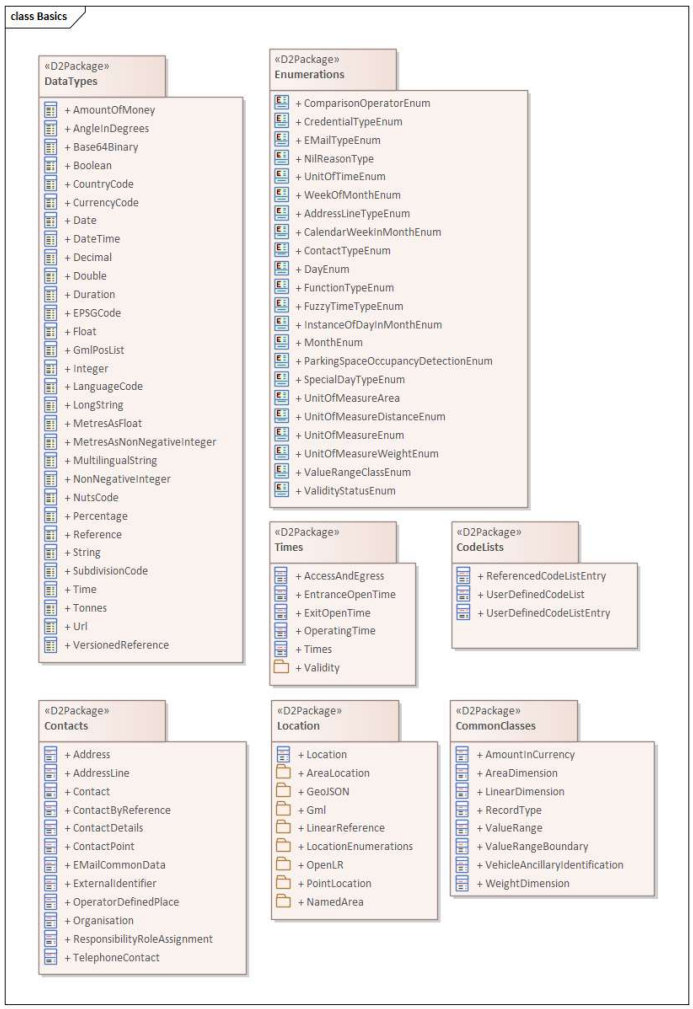

Figure 39 - Common – Basics

DataTypes (General)

Figure 40 shows data types defined in the PkCommon domain.

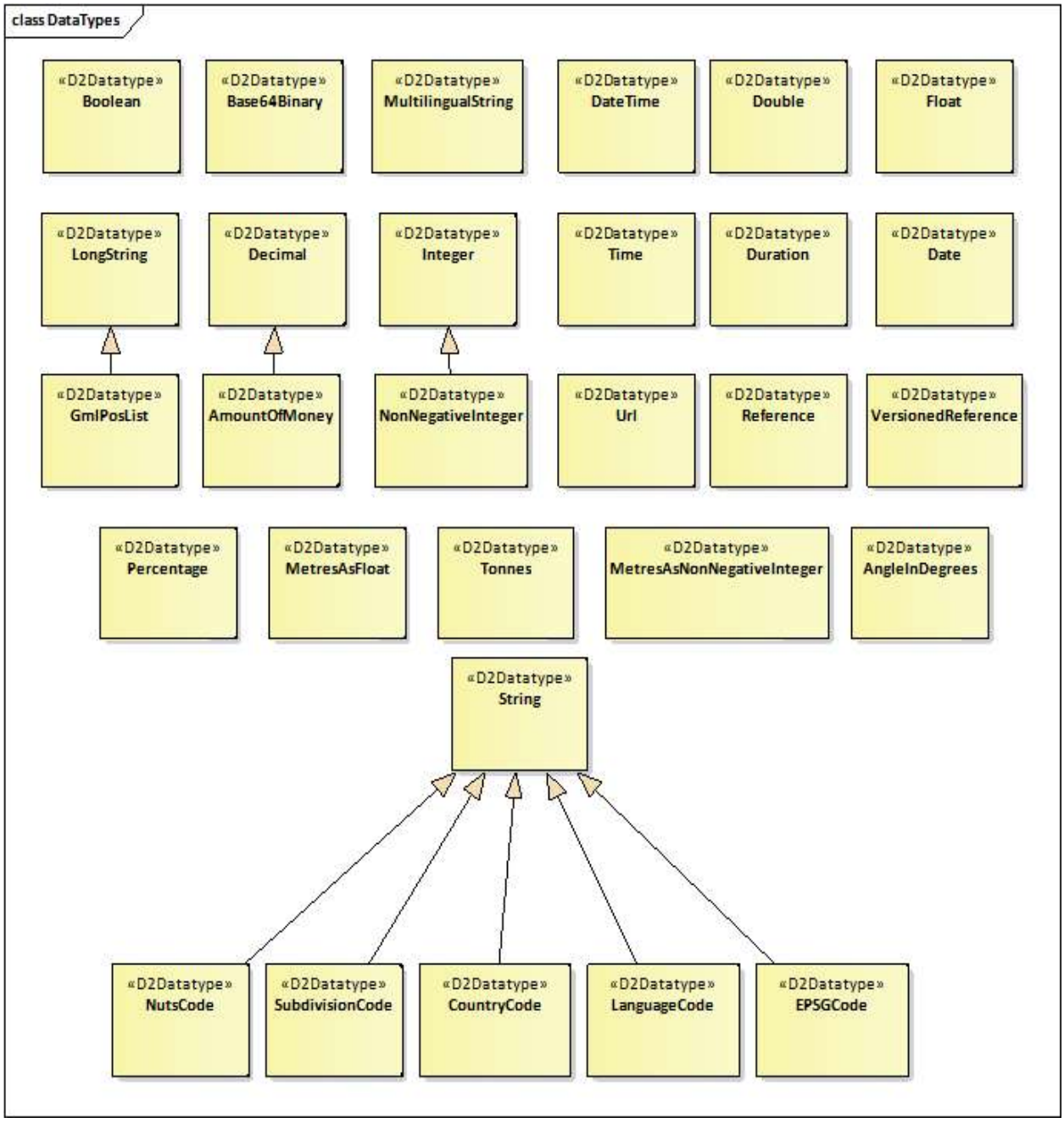

Figure 40 - Data Types in the Common (PkCommon) domain

Enumerations (General)

Figure 41 shows enumerations defined in the PkCommon domain.

Figure 41 - Enumerations in the Common (PkCommon) domain

Common Classes

These common classes define some complex structures that, in general, operate in a similar manner to data types, see Figure 42.

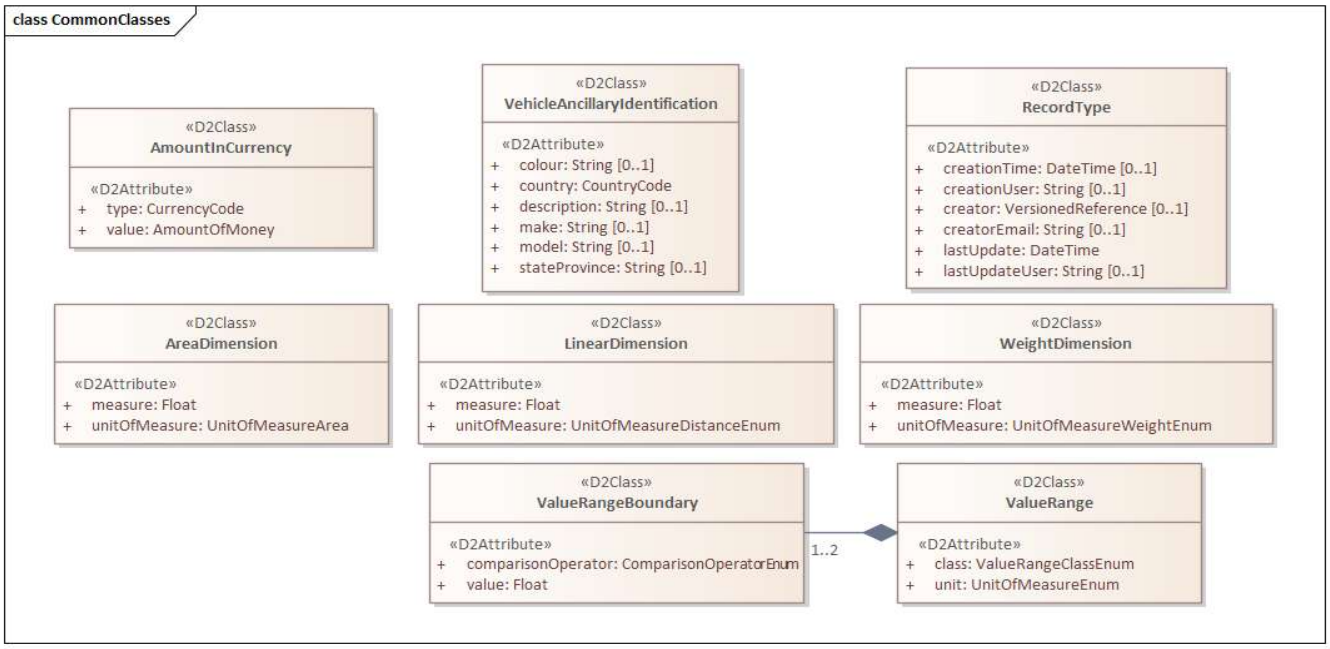

Figure 42 - Common Classes in the Common (PkCommon) domain

External Code Lists

These specifications are intended for global application. In some instances, a well-known defined list of terms or concepts that may apply in one geographical region may not have relevance elsewhere, or a specific data supplier may have their own view about, for example, occupancy levels should be described and banded. To support some flexibility, in a limited number of cases across the specification as a whole the use of a user-defined external code list is permitted. In each case the specification provides an example code list which may be preferred in a specific region or user community. See Annex D for examples.

Figure 43 provides a structure to define and reference user-defined code lists, and then reference specific entries within a specific code list.

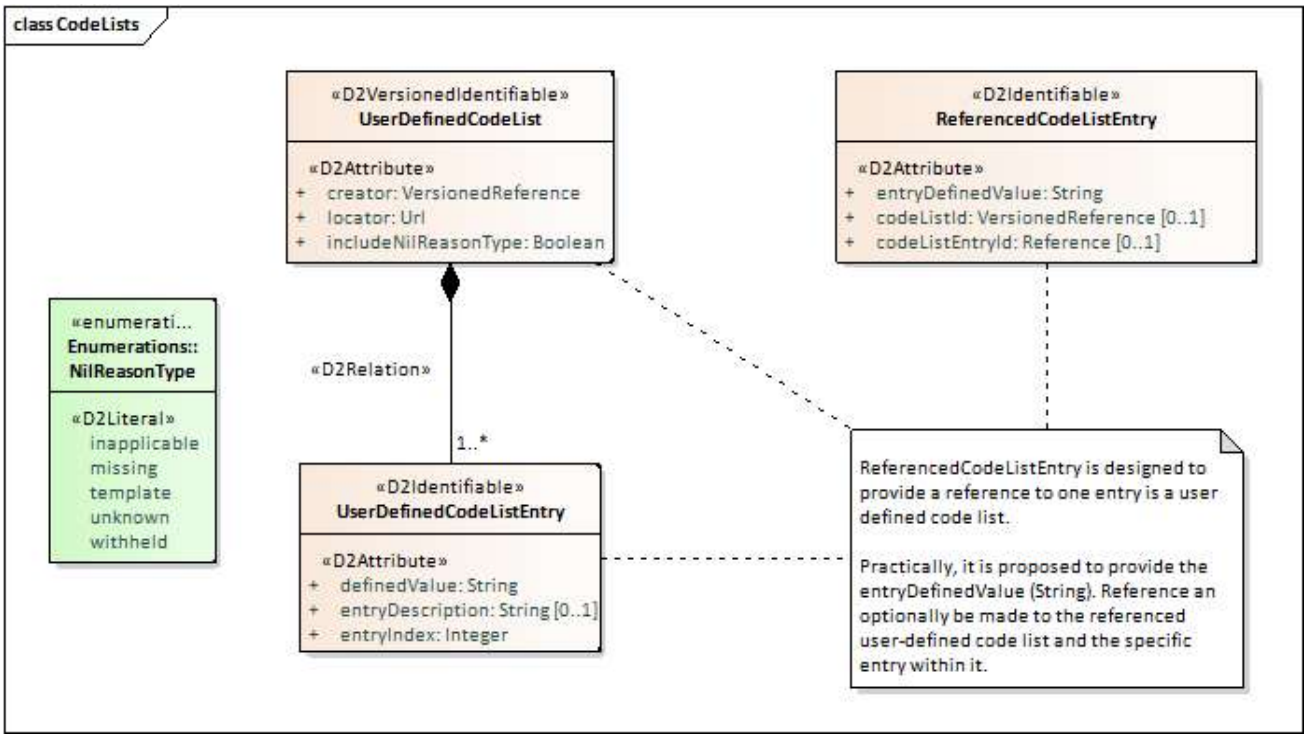

Figure 43 - Classes to support the definition of external user-defined code lists

Organisation, Contacts and Address

Organisation

Figure 44 shows a UML class diagram for organisation.

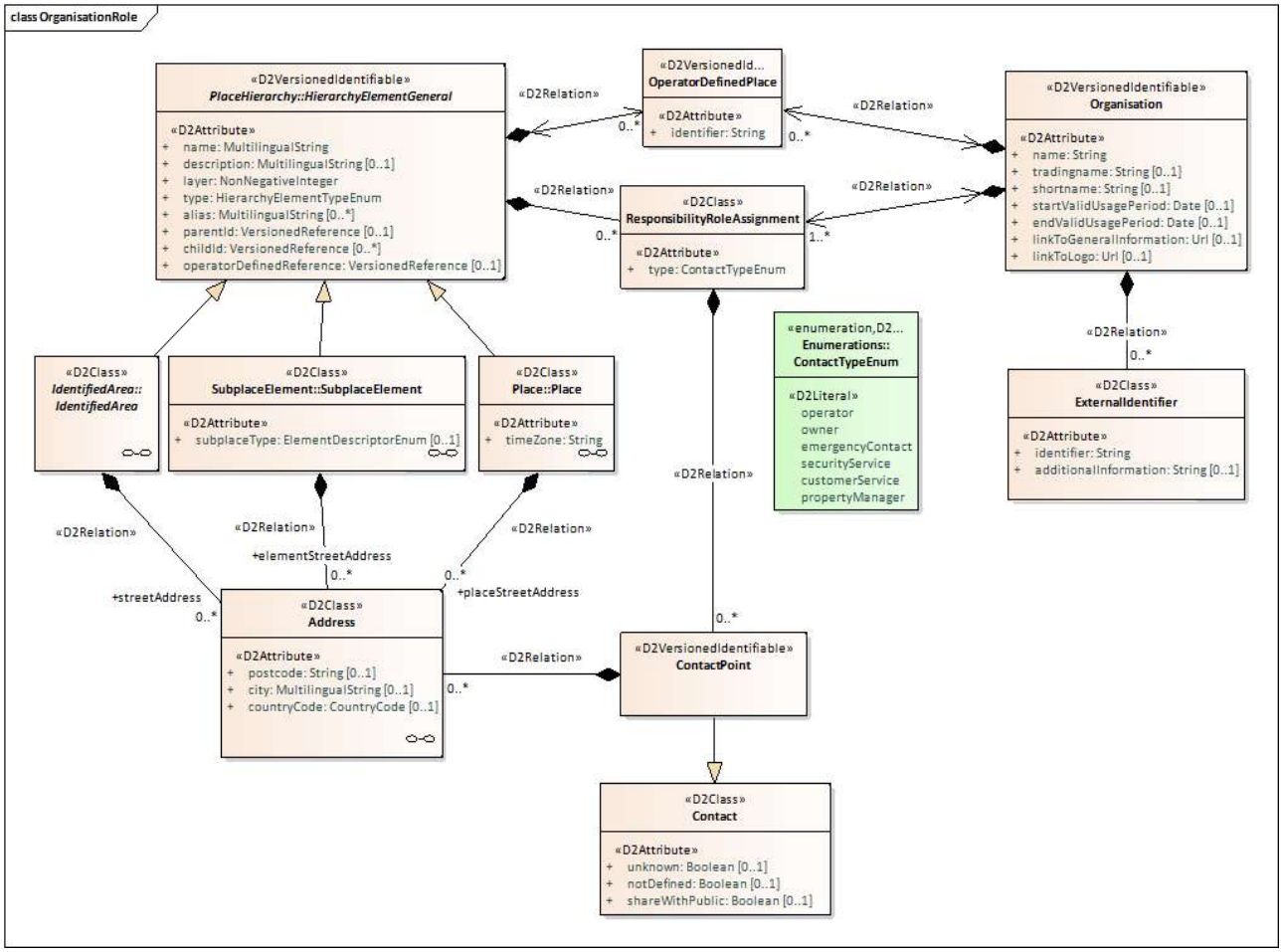

Figure 44 - Organisation class model

Contacts

Figure 45 shows a UML class diagram for Contact.

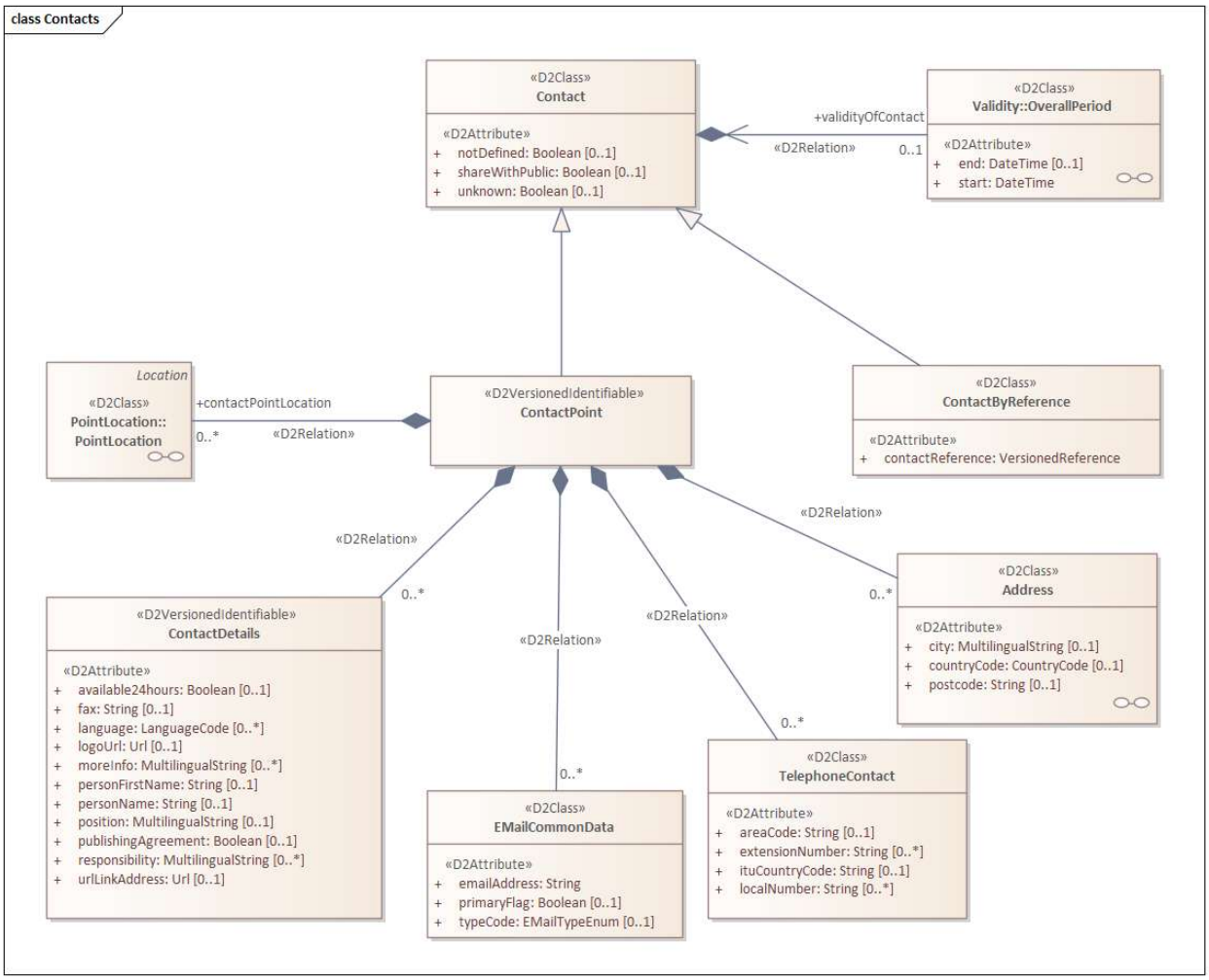

Figure 45 - Contact class model

Address

Figure 46 shows the concepts for a delivery or street address are defined from a profile of an existing ISO standard for this purpose, ISO-19160-4.

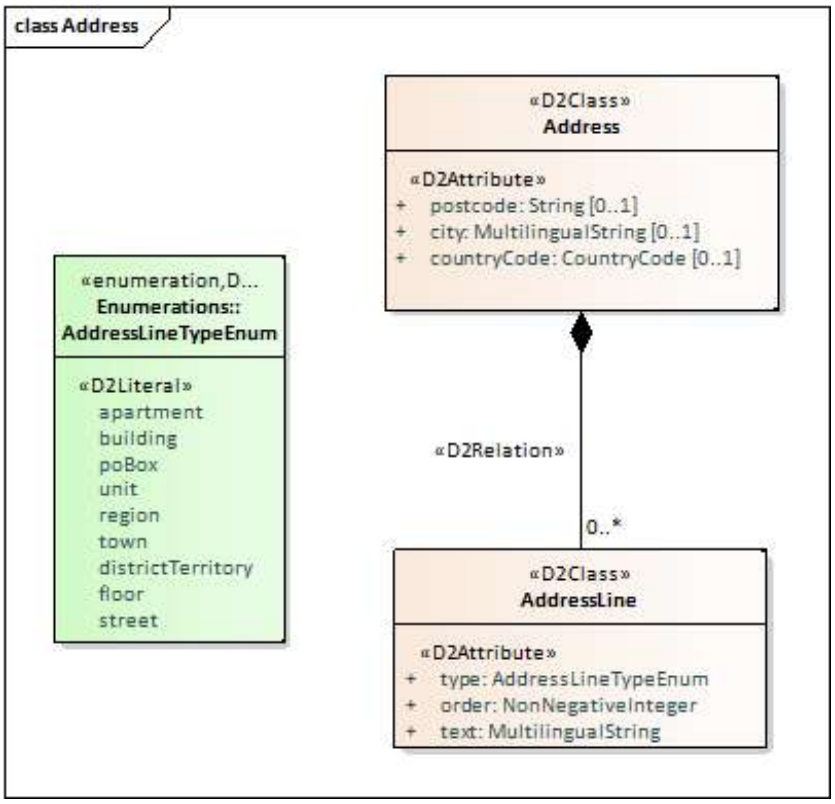

Figure 46 - Address class model

Addressing is a complex topic, with significant regional and national variations used. Addressing data models that attempt to fully model any form of postal or delivery address are inherently complex and not easy to interpret and therefore it is common practice to provide a simplified model that supports the definition of most postal or delivery address (street addresses). On initial inspection of the simple model above some user may think some critical concepts are missing, such as US States. These have been addressed by the generators of the ISO19160-4 standard by extensive notes that are not repeated fully here. By way of example, the following can be found in ISO19160-4:

- For Region (that can be seen in the AddressLineTypeEnum):
- Element indicating the name of the area within or adjacent to the town in which a delivery point is located, or via which it is accessed.
- EXAMPLE Hamlet, Estate, Sector, Arrondissement, Conjunto, Colonia Juarez, Kebele 4, Moo 11

NOTE 1: This element appears in the delivery point specification segment. It comprises sub-elements for levels of district, each with positions, with a type and indicator for each level and position.

NOTE 2: A district may be a commonly known name for an area, or it may be an area assigned for postal or administrative purposes. A district or sector may be one of several areas with a similar naming structure that may include a type and indicator structure.

NOTE 3: “District” is used by some countries for high level administrative divisions that are mapped to element region while element district is reserved for sub-divisions of town.

For districtTerritory (that can be seen in the AddressLineTypeEnum):

Element specifying the geographic or administrative area of the country in which the town is situated.

NOTE 1: This element appears in the delivery point specification segment. It comprises sub-elements for levels of region, each with positions, with a type and indicator for each level and position.

NOTE 2: Regions are generally related to administrative rather than to postal geography.

Examples include French Departments, German Länder, British Counties and American States.

Describing Locations

Introduction on Location, and Location Referencing

Nearly all parking and mobility applications need a description of the location of features (physical objects, restrictions, events, etc.) in a spatial context, both in absolute relationship to the surface of the Earth, and in relation to other features. One definition of the term "location referencing” or geolocation would be "description of an identifiable geographic place"; others include "label(s) which is assigned to a location", or "means to provide information that allows a system to identify accurately a location".

Many location references may be applicable for the features within a hierarchy. Some examples include:

- An x, y coordinate (longitude, latitude) that defines the notional position of the Place as a whole for coarse representation on a small-scale map of, say, the Place within a Campus.
- A bounded polygon which represents the physical extent of a Place.
- An x, y coordinate (longitude, latitude) that marks the location of each Vehicular Access to support SatNav navigation services for routing to the entrances of a Place.
- An x, y coordinate (longitude, latitude) of a specific parking space to support find-my-car applications.

The UML classes, attributes and relationship defined below are very largely drawn from existing recognised standards published by ISO and/or CEN. In some cases, to make these approaches globally applicable some concepts have been extended.

Conformance to this document allows a data supplier entity to define which location referencing method and coordinate referencing system it is using based, by reference to entries in the EPSG register[1].

Location

Location is an abstract class, see Figure 47. Location class model that groups together the different location referencing methods, which in turn provide several ways to define:

- a Point Location, with a single x, y coordinate.
- a Linear Location, which indicates a location with a length, such as the location on a stretch of road.
- an Area Location, which defines a polygon, which defines the extent of a feature, which is often used in map representation.

Each element of the place hierarchy contains both a point and an area location which serves different purposes for the different elements. This can be seen for Place (Figure 6), SubplaceElement (Figure 7), IdentifiedArea (Figure 8) and Space (Figure 9).

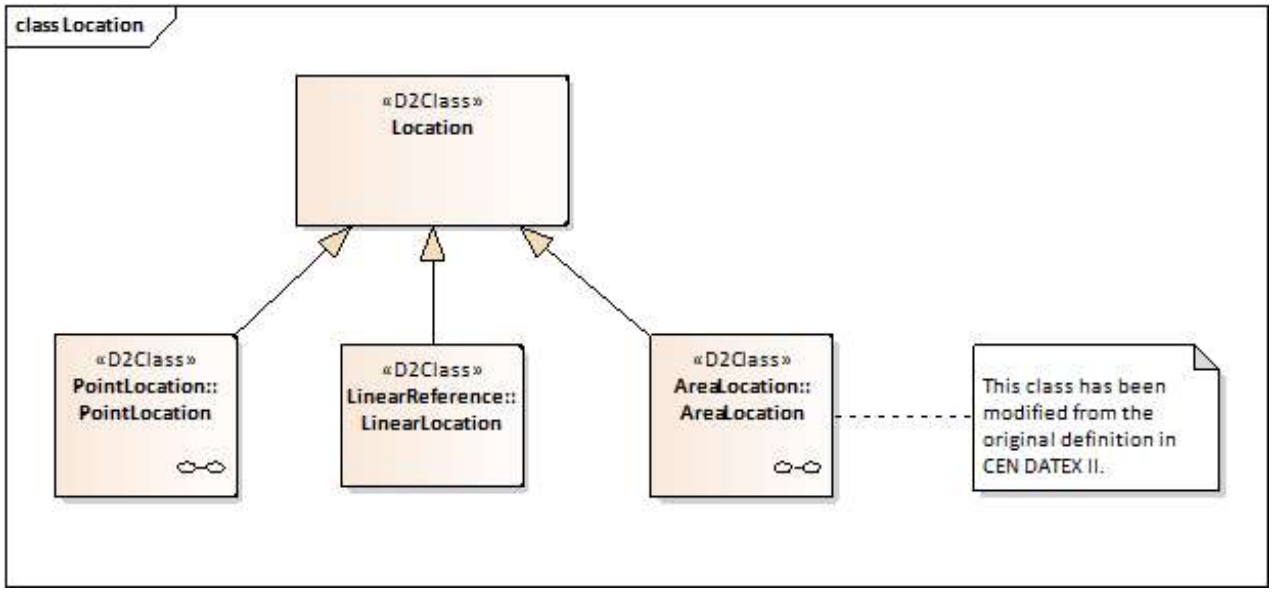

Figure 47 - Location class model

Linear Location is not used in this Specification but has been included for completeness. No further details of how to define a linear location area given.

AreaLocation

Figure 48 shows a UML class diagram for AreaLocation.

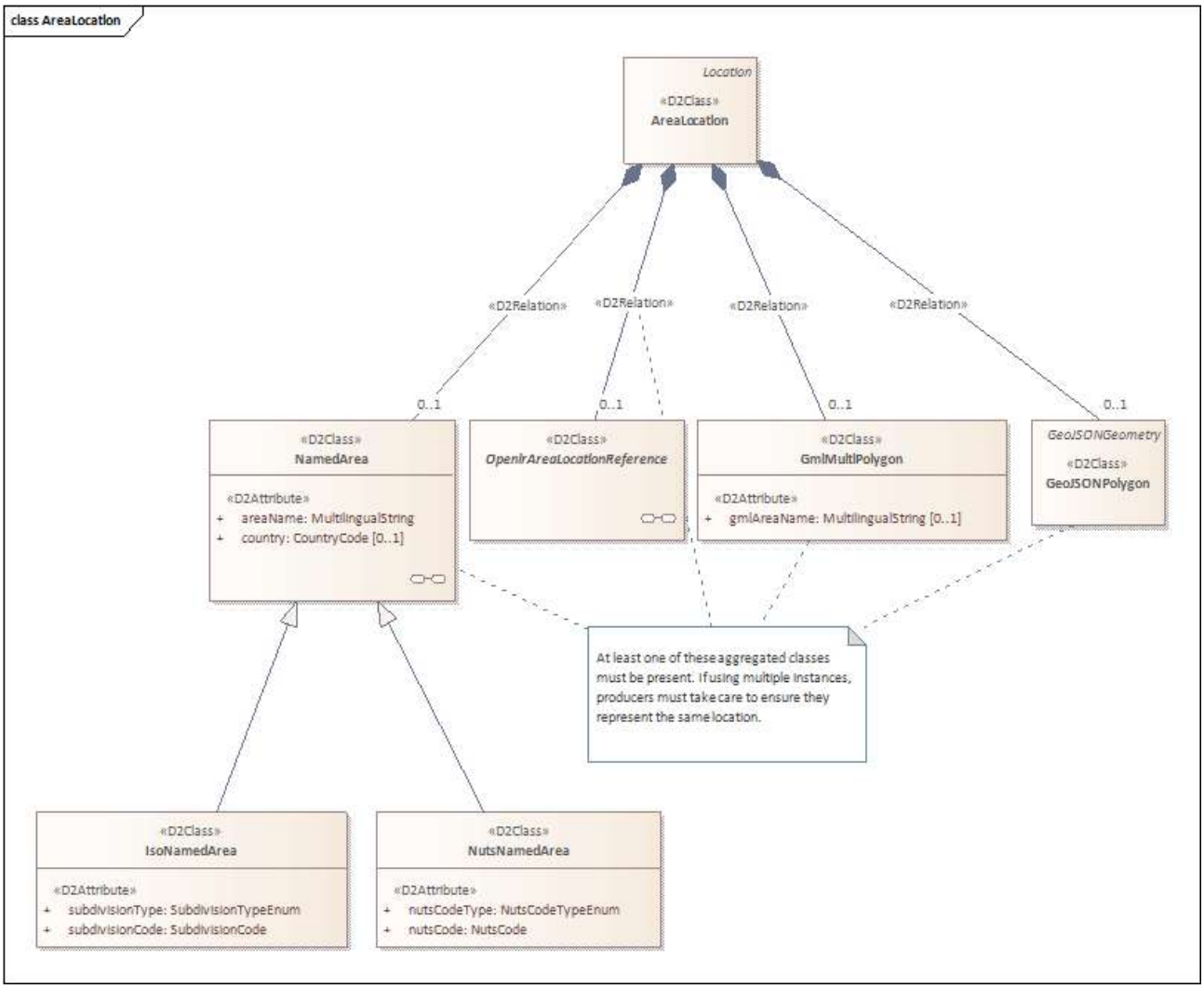

Figure 48 - AreaLocation class model

LinearLocation

Figure 49 shows a UML class diagram for LinearReference.

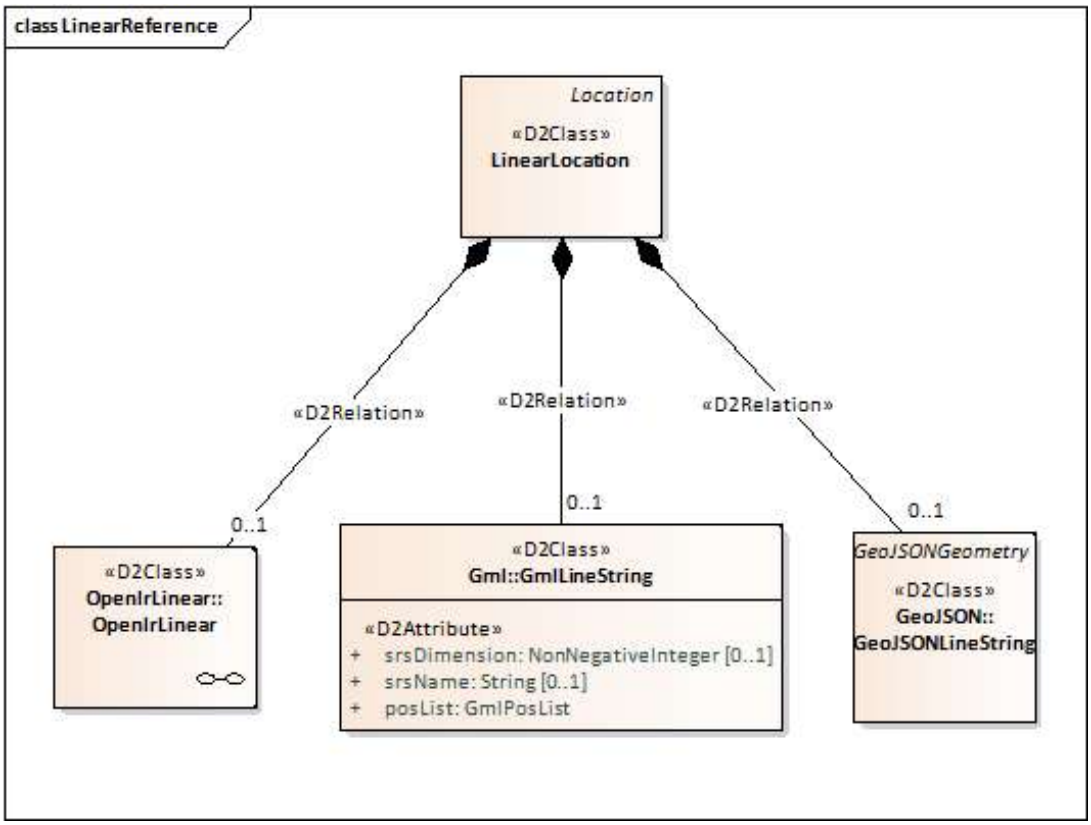

Figure 49 - LinearLocation class model

PointLocation

Figure 50 shows a UML class diagram for PointLocation.

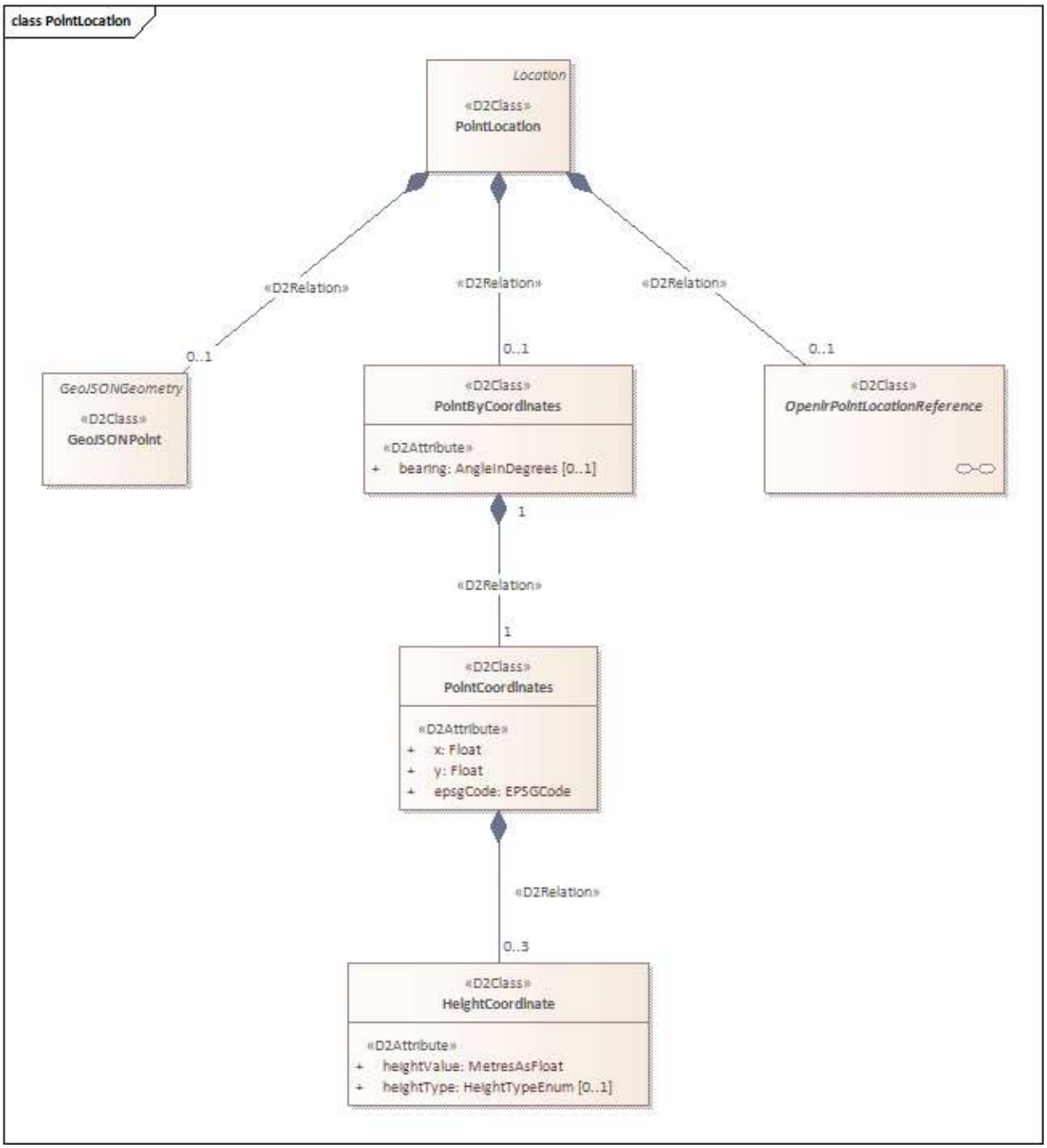

Figure 50 - PointLocation class model

Gml

Figure 51 shows a UML class diagram for GML.

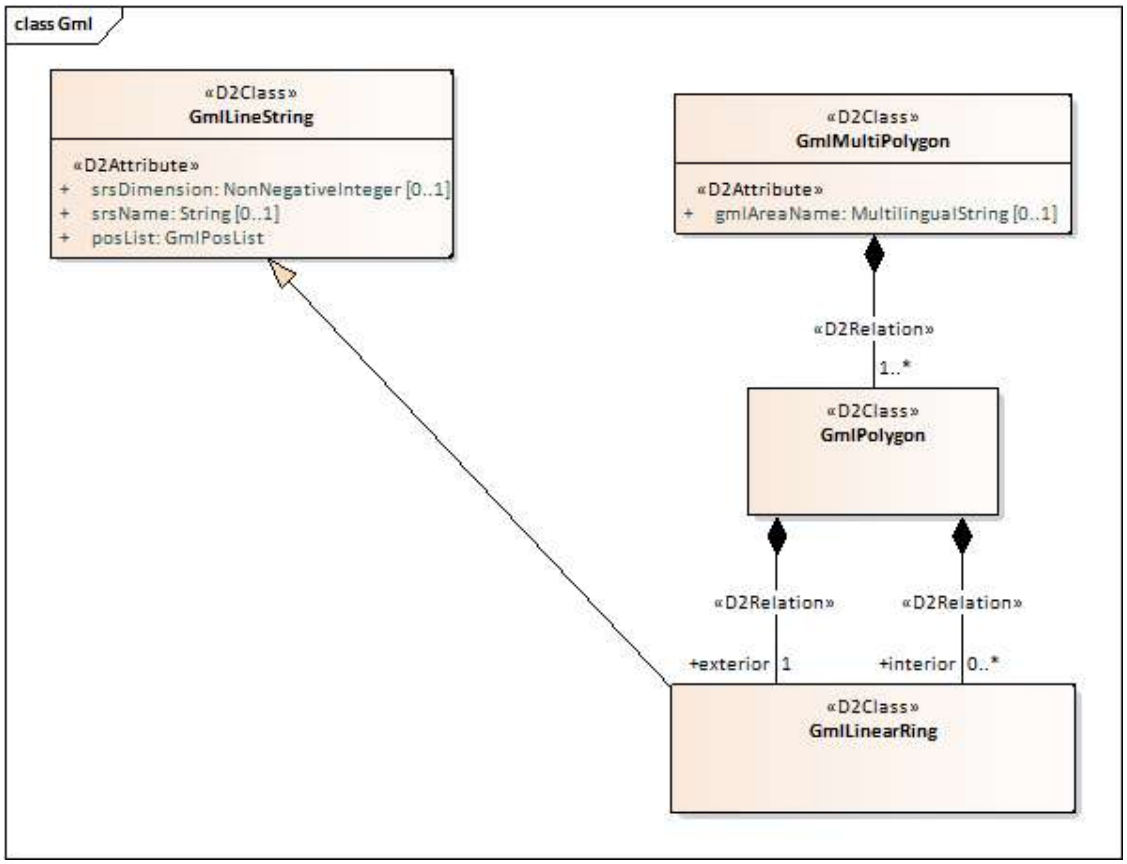

Figure 51 - GML model

GeoJSON

Figure 52 shows the GeoJSON model

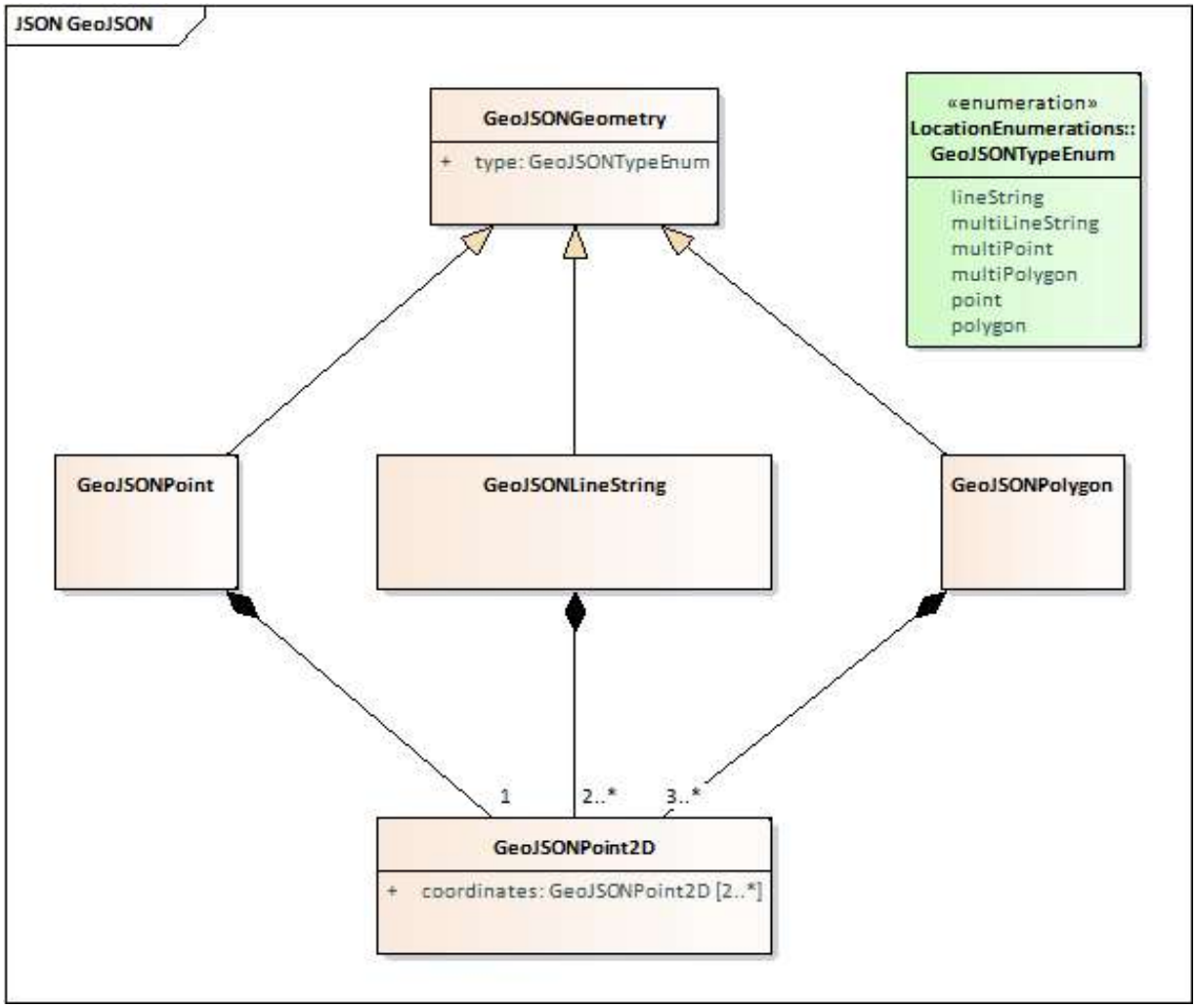

Figure 52 - GeoJSON model

OpenLR

OpenLR is a widely used license-free location referencing approach used in the ITS/Highways sector. Profiles of it appear in several ISO and CEN standards. For more information see: http://www.openlr.org/ . Figure 53 shows a UML class diagram for OpenLR.

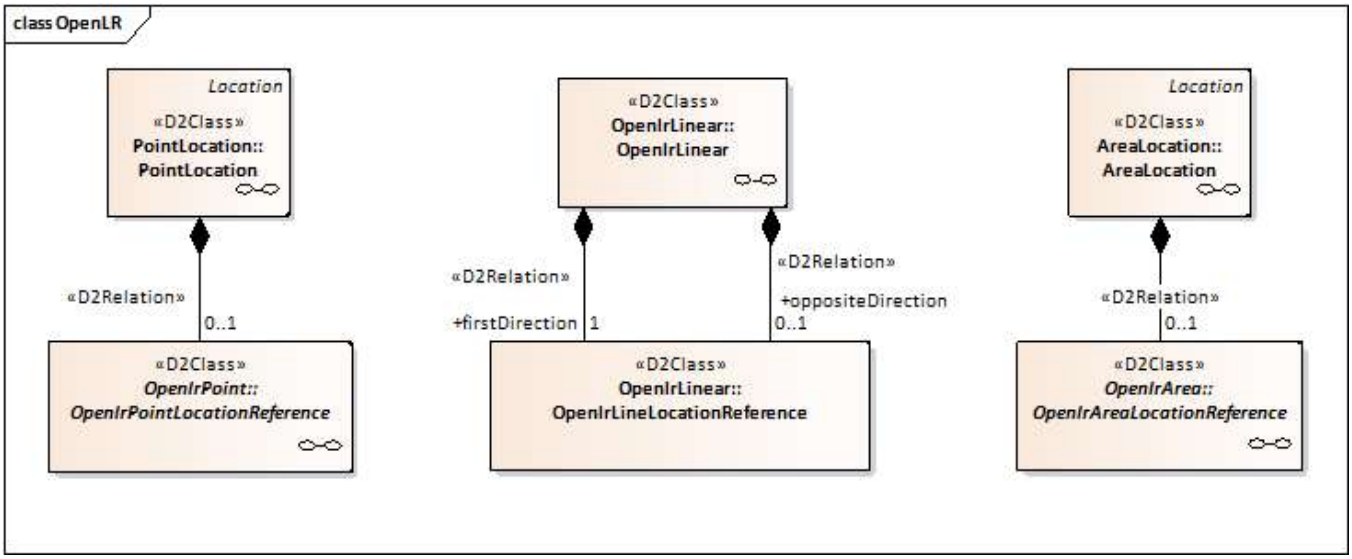

Figure 53 - OpenLR class model

Figure 54 shows a UML class diagram for OpenLR Area data concepts.

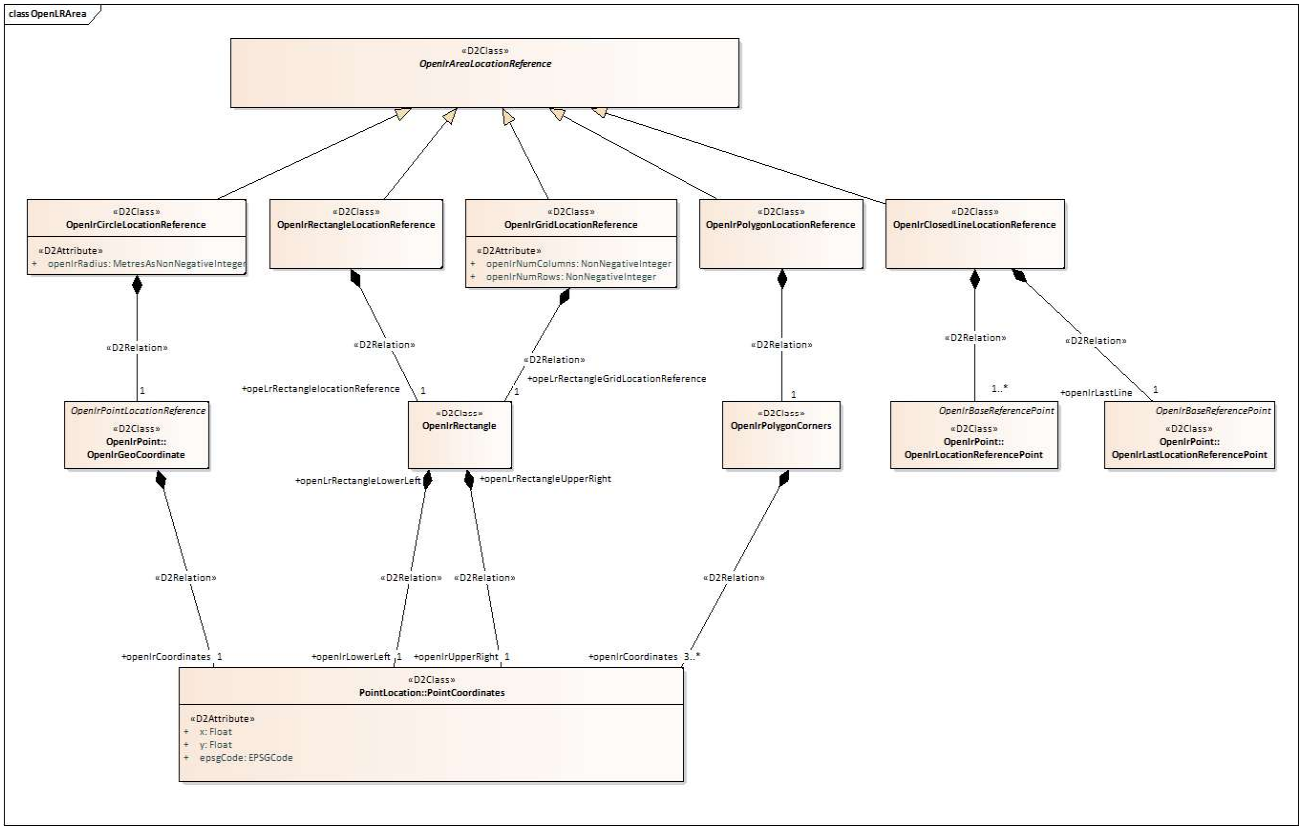

Figure 54 - OpenLRArea class model

Figure 55 shows a UML class diagram for OpenLR Linear data concepts.

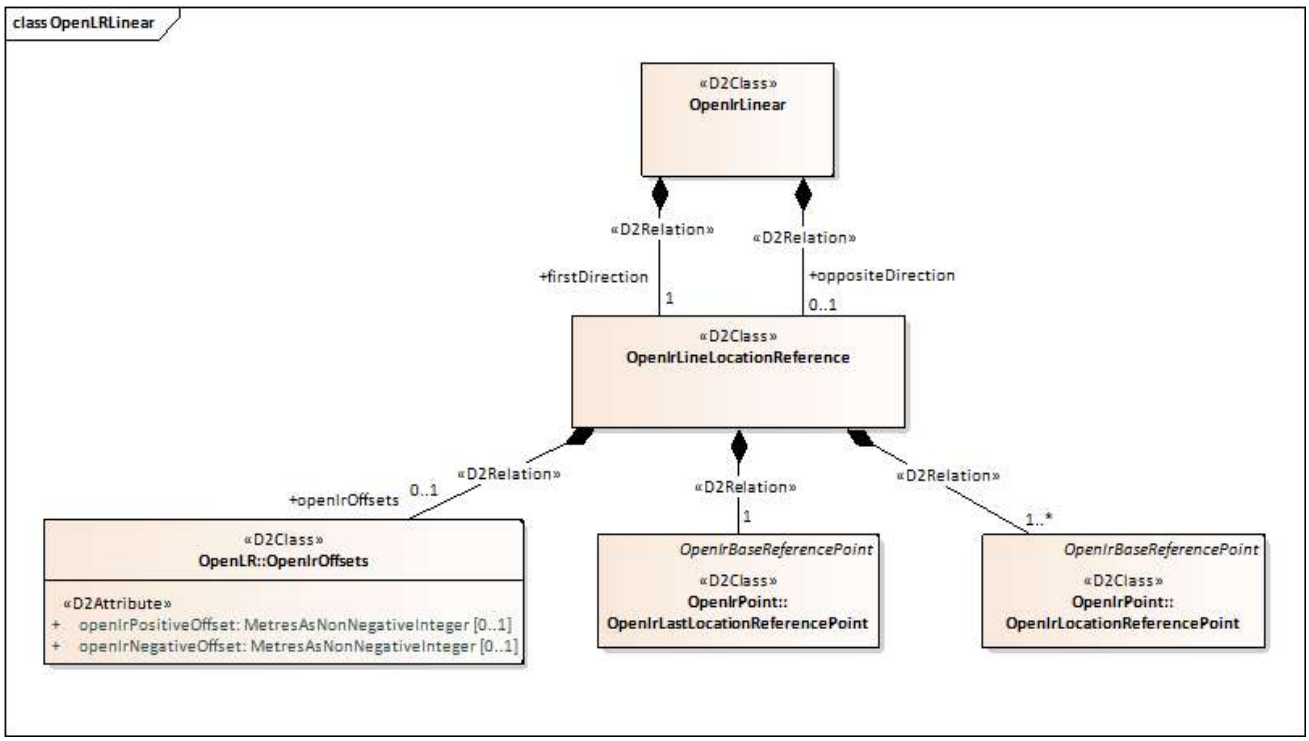

Figure 55 - OpenLRLinear class model

Figure 56 shows a UML class diagram for OpenLR Point data concepts.

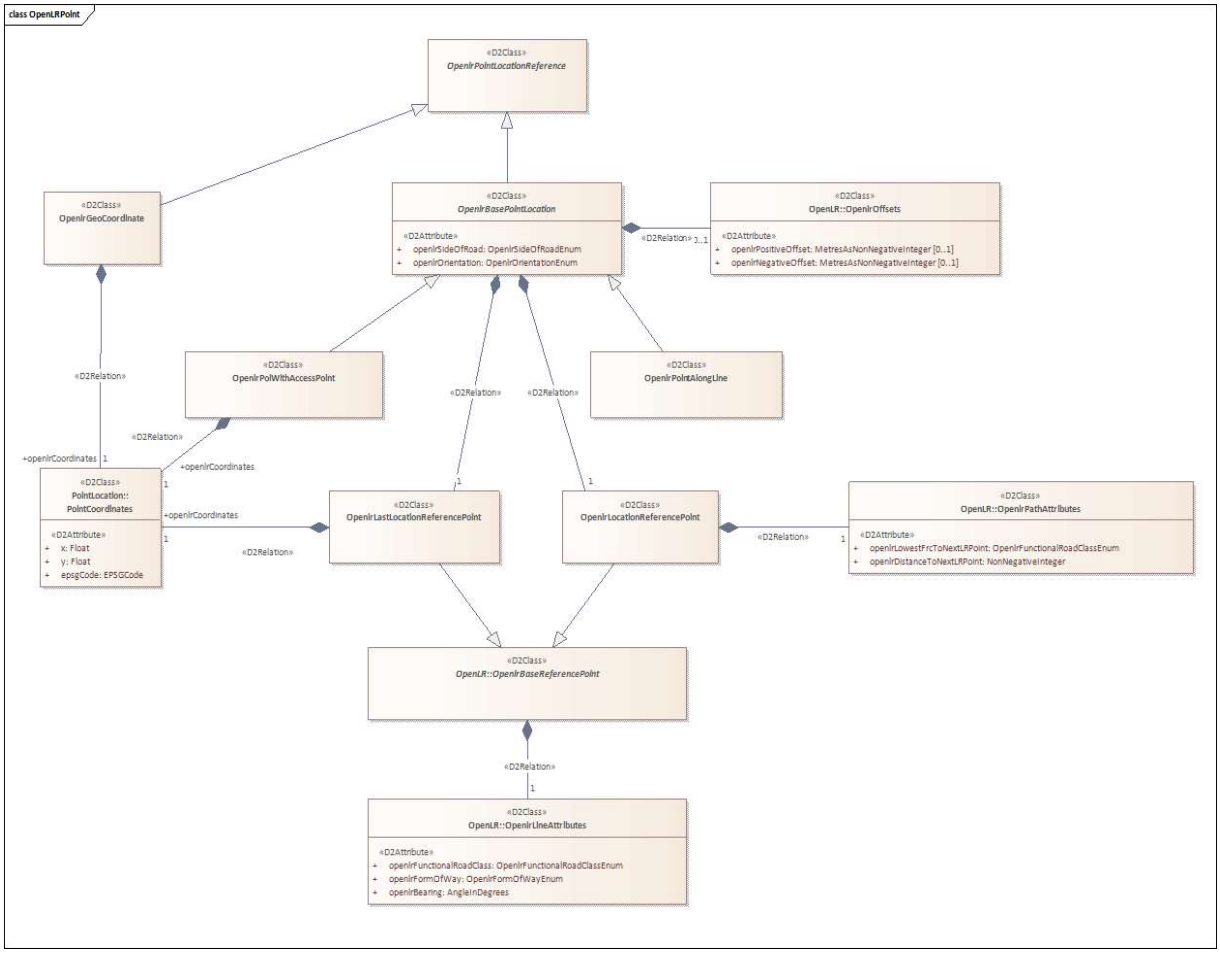

Figure 56 - OpenLRPoint class model

Data Types (for Location)

There are no specific data types for the Location package.

Enumerations (for Location)

The following enumerations are defined in the Location package, see Figure 57.

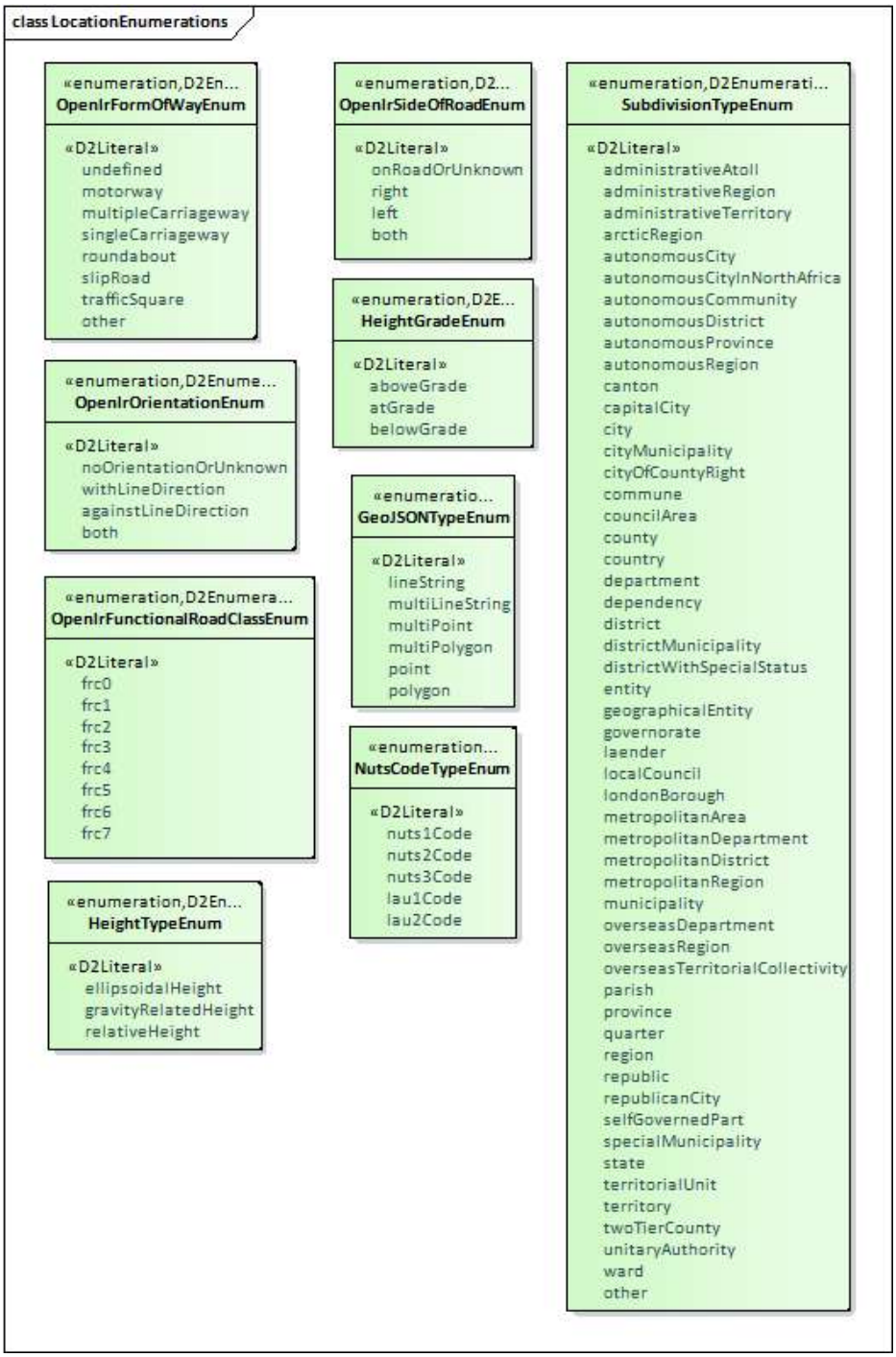

Figure 57 - Enumerations in the Location package

External Codelists (for Location)

No specific external code lists are defined in the Location packages. For general external code lists, see Section 6.5 – The Common Domain.

Times

Figure 58 shows a UML class diagram for Times data concepts, and Figure 59 shows Validity date concepts. Figure 60 shows the relationship between Validity, Rates, and RightSpecification.

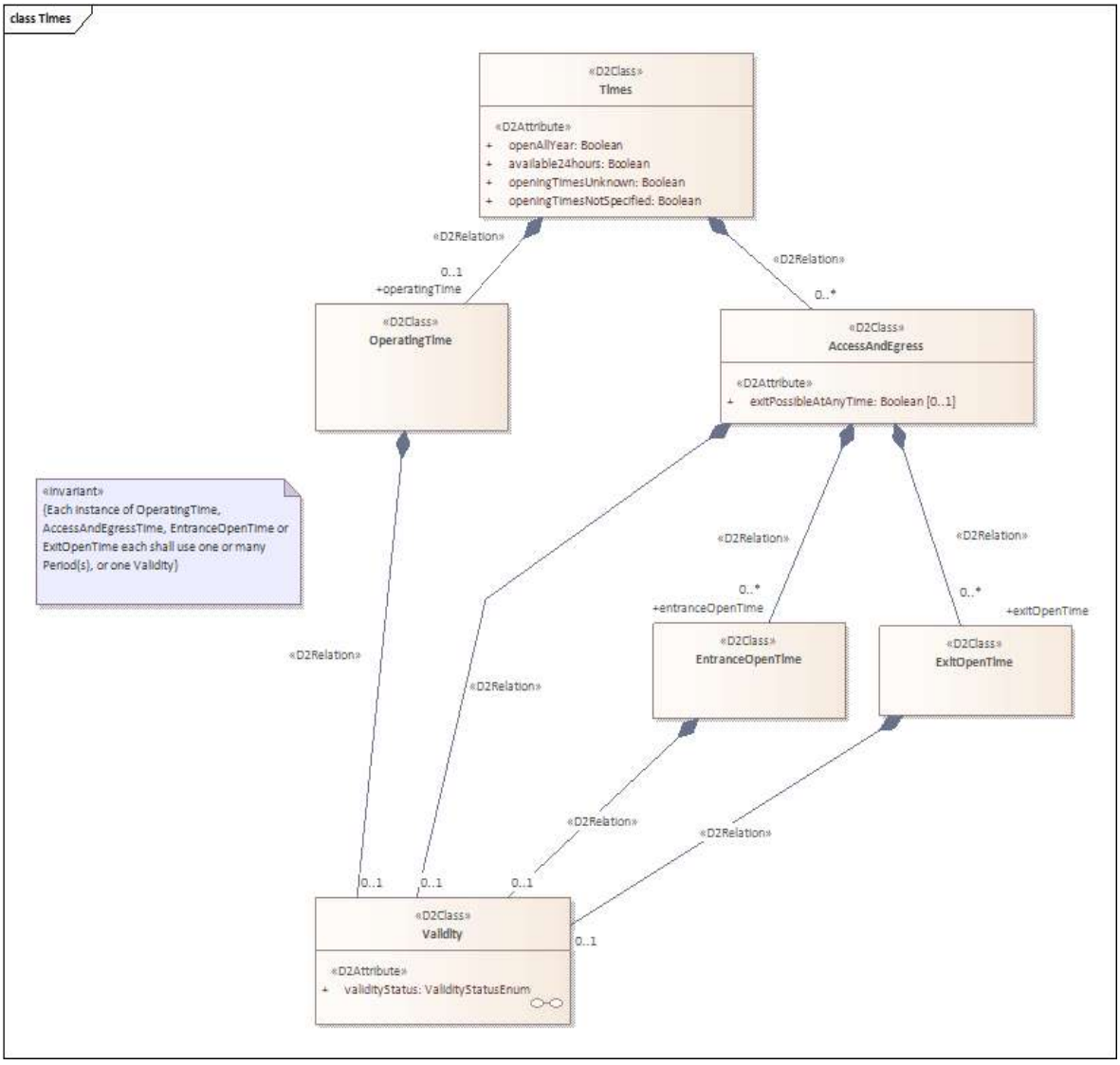

Figure 58 - Times class model

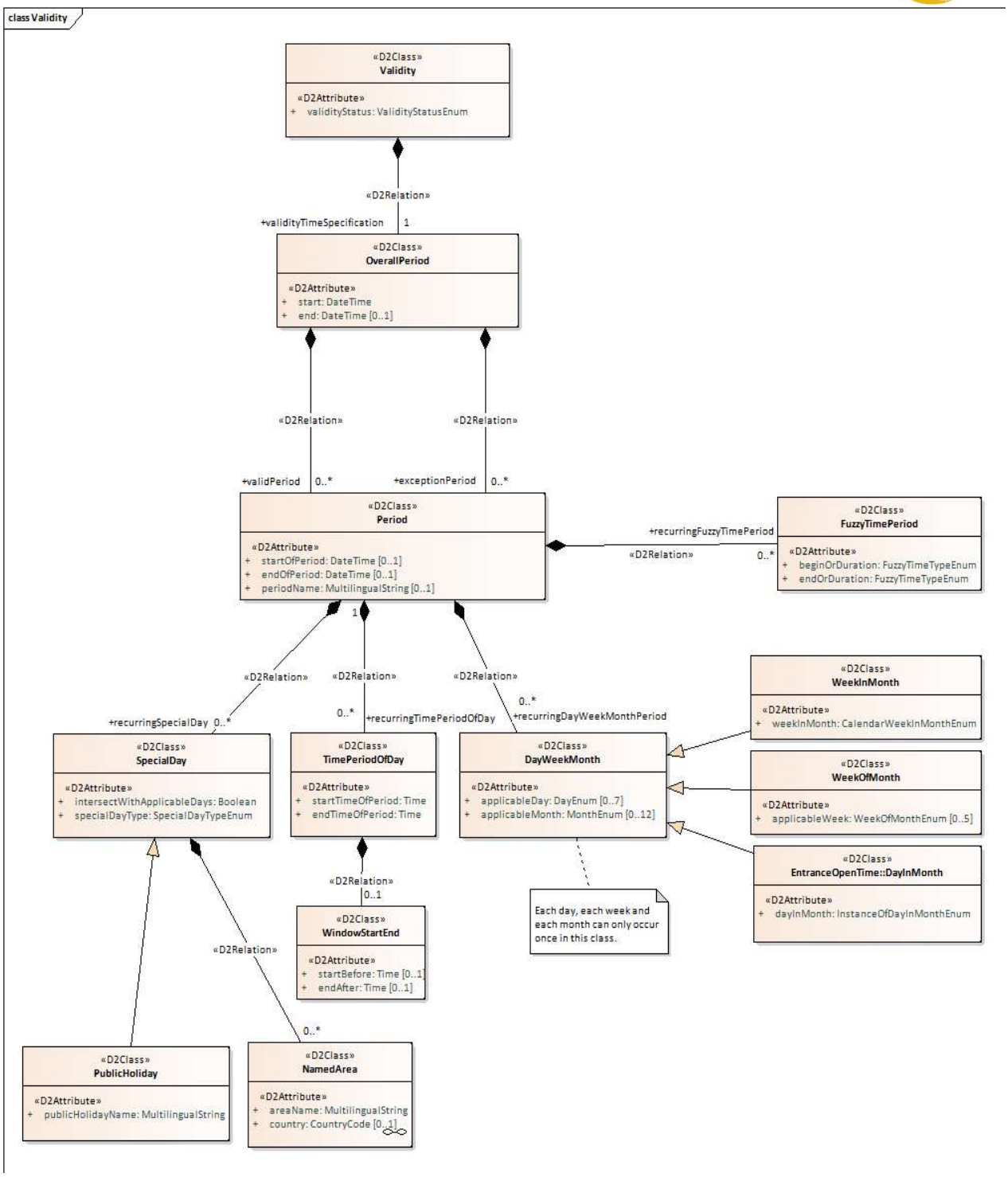

Figure 59 - Validity class model

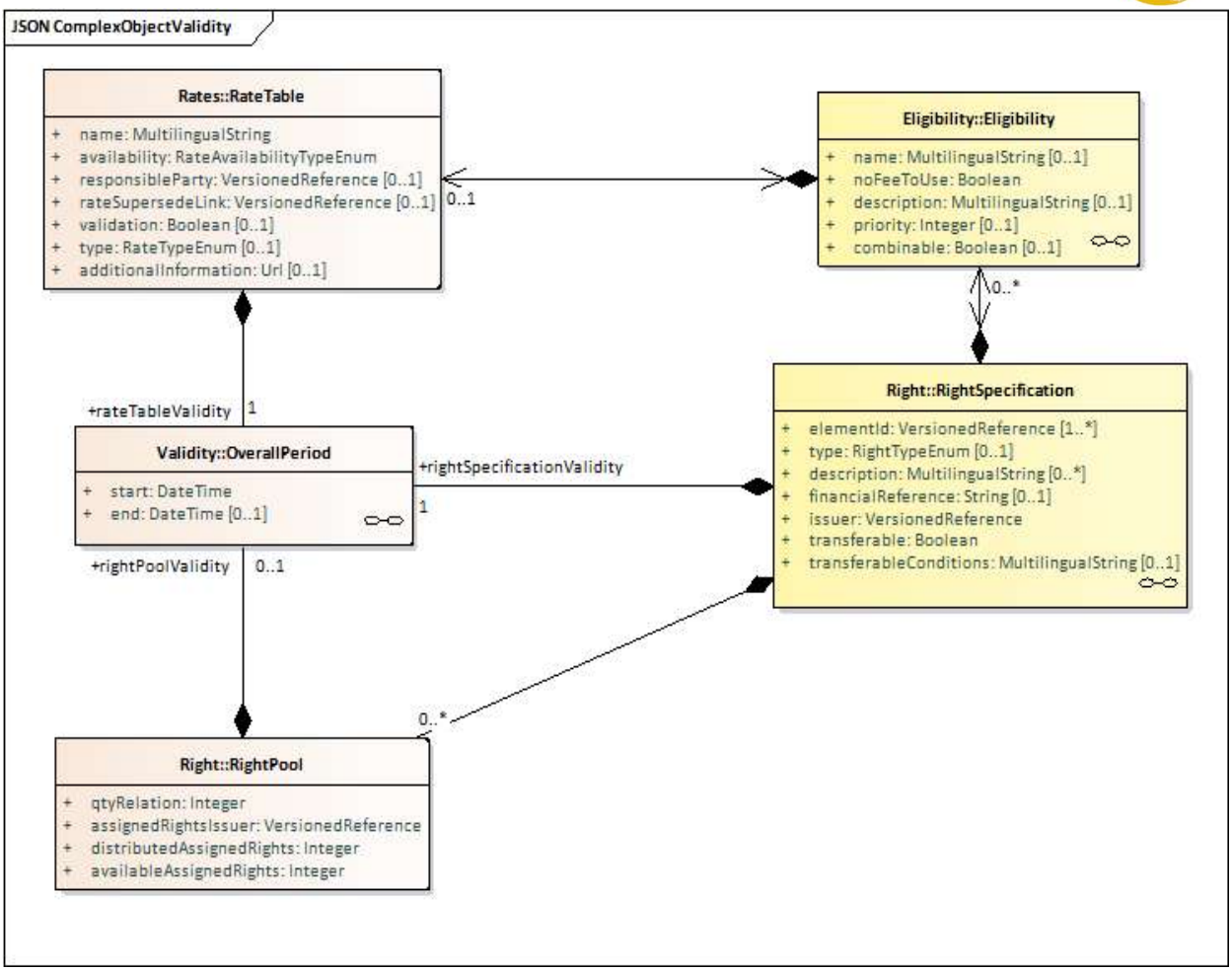

Figure 60 - Validity relationship to RightSpecification and Rates

Note 1: In this document, by convention, the start time is considered to be inclusive within a specified period; and end time indicates the end of a defined period but is considered to not be included within the specified period.
Example – An on-street parking session has 2 segments (10:03-14:00 and 14:01-15:45). If observed at 15:45:30, the second segment is considered to have already ended.

This approach must take account of the time resolution used for recording of parking sessions, observations, etc. within the source system. If times are only recorded to a one-minute resolution, in the example above 15:44 is considered to be within the second segment; and 15:45 is considered to be beyond the end of the second segment. If times are recorded to the second resolution the start time is, by default considered to start at second 00 of the minute and ending occurs at second 59 of the penultimate minute.

Note 2: all date times shall be specified in a format compliant to RFC3339 (Internet Date/Time Format).

Facilities

This describes a supporting data domain relating to the description of services and equipment related to elements of the Place hierarchy. These are specified as a package within the <Place> namespace of the data model.

Data Types (for Facilities)

There are no specific data types defined in Facilities.

Enumerations (for Facilities)

The following enumerations are defined in the Facilities package, see Figure 61.

Figure 61 - Enumerations in the Facilities package

External Codelists (for Facilities)

No external codelist are defined for Facilities. For general external code lists, see Section 6.5 – The Common domain.

Energy Infrastructure

This clause describes a support data domain relating to energy infrastructure and specifically electric vehicle charging. These are defined in the <EnergyInfrastructure> namespace of the data model. A UML class diagram for energy infrastructure is presented in Figure 62.\

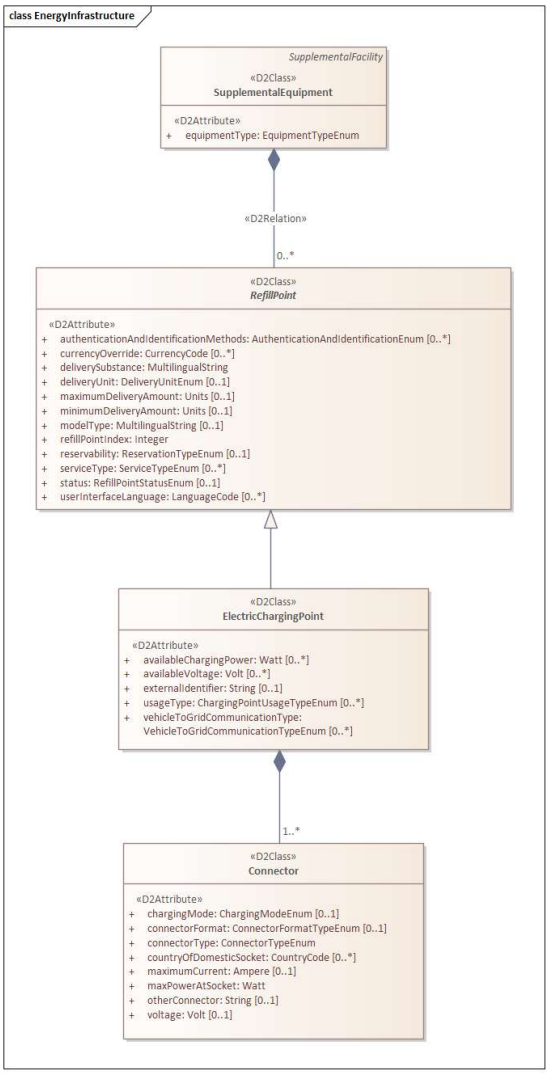

Figure 62 - Class model for the Energy Infrastructure domain

Data Types (for Energy Infrastructure)

The following data types are defined in <Energy Infrastructure>, see Figure 63.

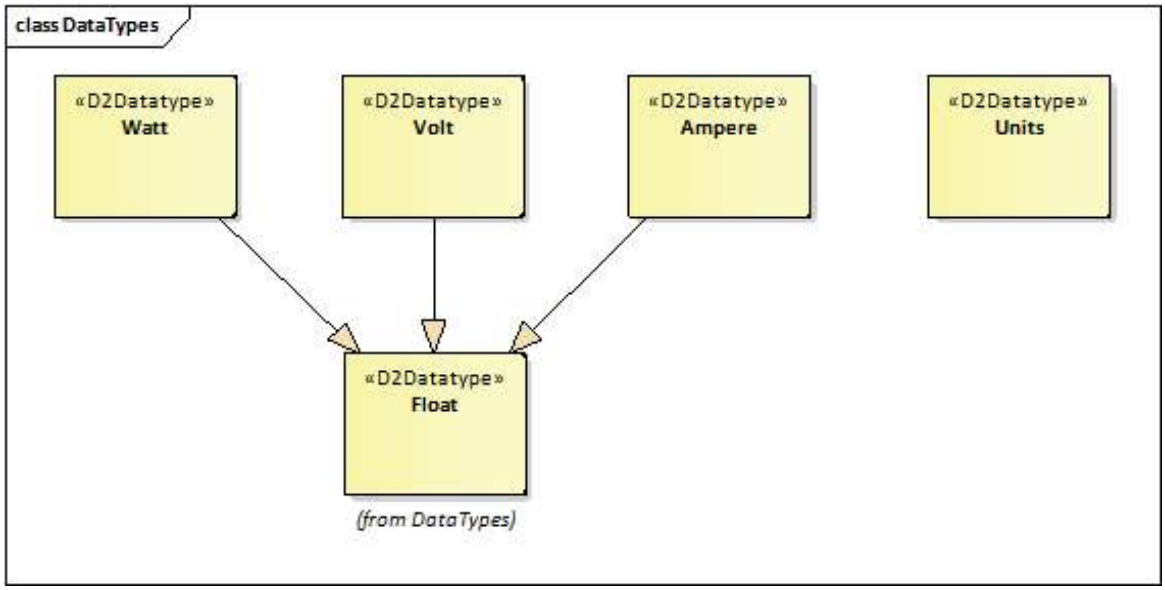

Figure 63 - Data types in the Energy Infrastructure domain

Enumerations (for Energy Infrastructure)

The following enumerations are defined in the <Energy Infrastructure> domain, see Figure 64.

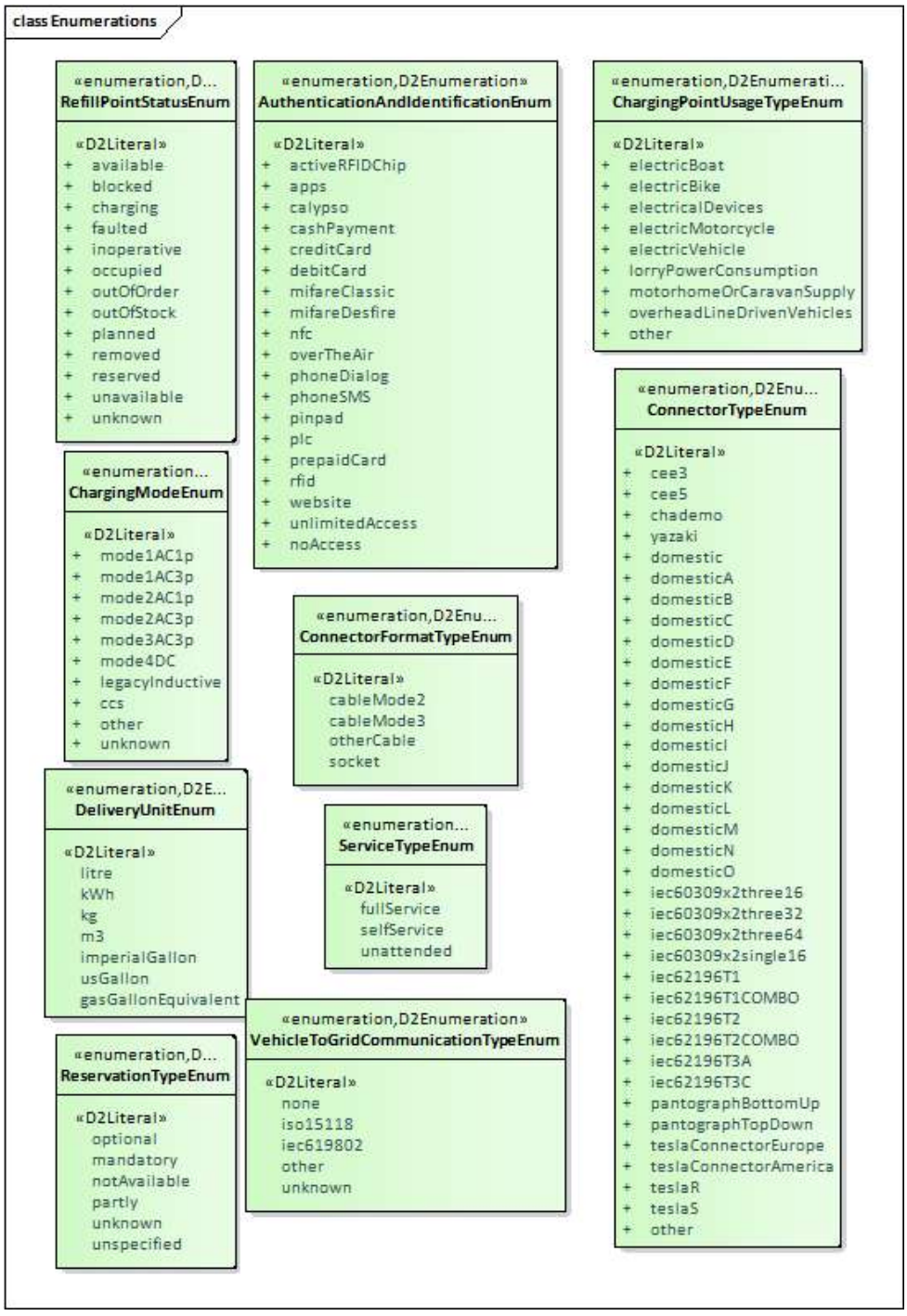

Figure 64 - Enumerations in the EnergyInfrastructure domain

External Codelists (for Energy Infrastructure)

No external codelist are defined for <Energy Infrastructure>. For general external code lists, see Section 6.5 – The Common domain.

Relationship to CEN DATEX II Data Modelling Concept and Framework

This data model defined in this document draws inputs from a number of sources including the CEN DATEX II series of standards.

By choice, the Alliance for Parking Data Standards has substantively adopted the DATEX II modelling approach, as this may facilitate simpler integration, at a later date, to data concepts and standards defining road traffic management and information concepts.

This model corresponds to the Level C model as defined in EN 16157-1 – in as much as DATEX II modelling concepts have been largely adhered to but this model is not purely an extension of the DATEX II platform independent data model.

There are some instances of divergence between this model and DATEX II modelling concepts, but these exist primarily at a data definition rather than modelling level. Some examples include:

- This model supports the use of a wider range of country codes, language codes and currency codes than supported in DATEX II (which is European-focused). In this model these refer to external codelists.
- This model supports non-SI measurement units that are particularly prevalent in the US (e.g., feet, pounds weight).

Conformance with this document shall require platform independent models from which platform specific models are generated to comply with the UML modelling rules defined in EN 16157-1 and with the requirements of this model, as defined in this document:

- comply with all stipulated minimum and maximum multiplicity requirements for UML elements and relationships.
- comply with all definitions, types and ordering.
- employ optional elements as specified.
- comply with all expressed constraints.

It should be noted that conformance with all the structural requirements stated above does not necessarily ensure that the informational content of that service will be semantically comprehensible (e.g., the information may be correctly structured, but make no sense).

Extending the APDS Model

APDS adopts the same approach to permitting extensions to the standardized data model as is defined by DATEX II in EN 16157-1.

NOTE: The rules stated below are adapted from but are consistent with the rules stated for DATEX II extensions given in EN 16157-1.

General

APDS, like the DATEX II models, enable application specific extensions. These extensions may implement innovative concepts, and while they may happily reuse data types, enumerations, components and even identifiable entities from an existing model, they cannot seek any type of system level interoperability with systems being implemented without being cognizant of this particular extension. Such extensions are denoted as Level C extensions.

In other scenarios, extensions seek to add some limited amount of application specific business logic whilst at the same time requiring backward compatibility with an existing model. "Compatibility" here means system level interoperability, e.g. for systems exchanging XML or JSON messages a valid instance of an extended model shall always be also a valid instance for the core model. This level of interoperability is denoted as Level B extension.

Requirements

1. A model that is conforming to the APDS specification may be extended. Extensions may either seek compatibility to an existing model (denoted 'core model' in this clause), or they may create a new model not compatible to any previous model, but nevertheless using the methodology provided within this specification and – potentially – reusing classes taken from other, existing models. A compatible extension is denoted within the APDS specification as a Level B extension. Non compatible extensions are denoted as Level C extensions.

2. All extensions shall fully conform with all other rules presented so far in this document.

3. An extended model shall provide extension name and version number in two UML Properties called "extensionName" and "extensionVersion" on the "D2ModelRoot" UML Stereotype of a root level element that shall be usable in conjunction with extended elements.

4. Extensions that add new "D2ModelRoot" classes are always Level C extensions. Extensions that do not add new "D2ModelRoot" classes may be Level B or Level C extensions, depending on whether they fulfil the Level B rules provided in this clause.

5. APDS classes belonging to a Level B extension shall not become superclasses of classes in the core model, i.e. UML generalizations from an APDS class from the core model to an APDS class in the extension shall not be added to the model.

6. APDS classes belonging to the core model may become superclasses of APDS classes in a Level B extension, i.e. UML generalizations from an APDS class from a Level B extension to an APDS class in the core model may be added to the model. Such generalizations shall have a "D2LevelBExtension" stereotype from the "UML Profile for DATEX II" assigned to it.

   NOTE Multiple Level B extensions of one core model class can exist.

7. "D2Relation" compositions between APDS classes in an extension and APDS classes in the core model may be added to a Level B extended model if they have a core model class on the

'element' side (i.e. the side without the diamond) and an extension class on their other end. Thus, existing classes from the core model may become components of containers in the Level B extension model (class reuse), but classes from the Level B extensions shall not become components of existing containers in the core model.

8. APDS enumerations belonging to a Level B extension shall not become superclasses of enumerations in the core model, i.e. UML generalizations from an APDS enumeration from the core model to an APDS enumerations in the extension shall not be added to the model.

9. APDS enumerations belonging to the core model may become superclasses of APDS enumerations in a Level B extension, i.e. UML Generalizations from an APDS enumeration from a Level B extension to an APDS enumerations in the core model may be added to the model. Such generalizations shall have a "D2LevelBExtension" stereotype from the "UML Profile for DATEX II" assigned to it.

   NOTE Multiple Level B extensions of one core model enumeration can exist.

10. The "order" properties of all literals of an enumeration and all its extensions shall be unique.

11. APDS datatypes having the "D2Datatype" stereotype assigned and belonging to a Level B extension shall not become superclasses of datatypes in the core model, i.e. UML generalizations from an APDS datatype from the core model to an APDS datatype in the extension shall not be added to the model.

12. APDS datatypes having the "D2Datatype" stereotype assigned and belonging to the core model may become superclasses of APDS datatypes in a Level B extension, i.e. UML Generalizations from an APDS datatype from a Level B extension to an APDS datatype in the core model may be added to the model.

13. UML generalizations with a "D2LevelBExtension" stereotype from the "UML Profile for DATEX II" assigned to them shall not be used in any situation other than those specified in 6, 9 or 12 above.

14. UML Datatypes and UML Enumerations of the core model may be reused in Level B extensions.

Glossary

Term	Definition
Alliance Parking Data Model	The Parking Data Model as specified by the Alliance for Parking Data Standards
Data Distributing Party	The entity holding data with permission to distribute, that is issuing data using the specification to other entities
Data Receiving Party	The entity that is requesting or receiving data using the specification from a reliable data source
IdentifiedArea	A collection of information relating to IdentifiedArea, which are an identifiable discrete bounded geographic zone that shares common characteristics and that may be used for parking or other mobility related purposes.
Place Hierarchy	A collection of information relating to a hierarchy of Place structures and facilities.
Place	Where a vehicle may park, stand, rest, or briefly transit to allow a person to change modes of transport (i.e., taxi drop-off/pickup, ride share drop-off/pickup, valet stand, etc.).
Space	A collection of information relating to a parking space, i.e., a location normally for parking a single vehicle.

References

Normative References

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document.

ISO 639-1:2002 – Codes for the representation of names of languages — Part 1: Alpha-2 code

ISO 639-2: 1998 – Codes for the Representation of Names of Languages – Part 2: alpha-3 codes

ISO 3166 – Country Codes

ISO 4217 – Currency codes

ISO 8601 – Data elements and interchange formats – Information interchange – Representation of dates and times

ISO/IEC 10646:2012 – Information technology -- Universal Coded Character Set (UCS)

ISO 14825:2014 – Intelligent transport systems — Geographic Data Files (GDF)

ISO 14819-3:2013 – Traffic and Travel Information (TTI) -- TTI messages via traffic message coding -- Part 3: Location referencing for ALERT-C

EN 16157-1:2018 – Intelligent transport systems — DATEX II data exchange specifications for traffic Management and information – Part 1: Context and framework

EN 16157-2:2019 – Intelligent transport systems — DATEX II data exchange specifications for traffic Management and information – Part 2: Location Referencing[1]

EN 16157-7:2018 – Intelligent transport systems — DATEX II data exchange specifications for traffic Management and information – Part 7: Common Data Elements[1]

ISO 19136:2007 – Geographic information -- Geography Markup Language (GML)
ISO 19148:2012 – Geographic information -- Linear referencing

ISO 19160-4:2017 - Addressing -- Part 4: International postal address components and template language

ISO/IEC 19505-1:2012 – Information technology - Object Management Group Unified Modeling Language (OMG UML), Part 1: Infrastructure

RFC3339. Date and Time on the Internet: Timestamps https://tools.ietf.org/html/rfc3339

Other References

IPI-DataEx – Parking Location Standard Working Papers, June 2017

CEN/TS 16157-6:2015 – Intelligent transport systems — DATEX II data exchange specifications for traffic Management and information – Part 6: Parking Publications

UML Notation

The UML notation used in this document shall be as described in ISO/IEC 19505-1:2012.

Note: The use of UML notation follows the DATEX II methodology which is specified in Part 1 of EN 16157:2018.