Working Draft
MEF W160 v0.1

LSO Cantata and LSO Sonata Product Offering Availability and Pricing Discovery API - Developer Guide

This draft represents MEF work in progress and is subject to change.

June 2023

Disclaimer

Copyright

Table of Contents

List of Contributing Members

The following members of the MEF participated in the development of this document and have requested to be included in this list.

Member

Table 1: Contributing Members

1. Abstract

This standard is intended to assist implementation of the Product Offering Availability and Pricing Discovery functionality defined for the LSO Cantata and LSO Sonata Interface Reference Points (IRPs), for which requirements and use cases are defined in MEF 110 *Product Offering Availability and Pricing Discovery - Business Requirements and Use Cases* [MEF110]. This standard consists of this document and complementary API definitions.

This standard normatively incorporates the following files by reference as if they were part of this document, from the GitHub repository:

https://github.com/MEF-GIT/MEF-LSO-Sonata-SDK

- `productApi/availabilityAndPricingDiscovery/productOfferingAvailabilityAndPricingDiscovery.api.yaml`

https://github.com/MEF-GIT/MEF-LSO-Cantata-SDK

- `productApi/availabilityAndPricingDiscovery/productOfferingAvailabilityAndPricingDiscovery.api.yaml`

2. Terminology and Abbreviations

This document does not define any new terms or definitions. All of them are defined in the standards referenced below and are included in this document by reference:

- MEF 110 - Product Offering Availability and Pricing Discovery - Business Requirements and Use Cases [MEF110]
- MEF 55.1 Lifecycle Service Orchestration (LSO): Reference Architecture and Framework [MEF55.1]

3. Compliance Levels

The key words **"MUST"**, **"MUST NOT"**, **"REQUIRED"**, **"SHALL"**, **"SHALL NOT"**, **"SHOULD"**, **"SHOULD NOT"**, **"RECOMMENDED"**, **"NOT RECOMMENDED"**, **"MAY"**, and **"OPTIONAL"** in this document are to be interpreted as described in BCP 14 (RFC 2119 [rfc2119], RFC 8174 [rfc8174]) when, and only when, they appear in all capitals, as shown here. All keywords must be in bold text.

Items that are **REQUIRED** (contain the words **MUST** or **MUST NOT**) are labeled as **[Rx]** for required. Items that are **RECOMMENDED** (contain the words **SHOULD** or **SHOULD NOT**) are labeled as **[Dx]** for desirable. Items that are **OPTIONAL** (contain the words **OPTIONAL** (contain the words MAY or OPTIONAL) are labeled as **[Ox]** for optional.

4. Introduction

This standard specification document describes the Application Programming Interface (API) for Product Offering Availability and Pricing Discovery functionality of the LSO Cantata Interface Reference Point (IRP) and Sonata IRP as defined in the MEF 55.1 *Lifecycle Service Orchestration (LSO): Reference Architecture and Framework* [MEF55.1]. The LSO Reference Architecture is shown in Figure 1 with both IRPs highlighted.
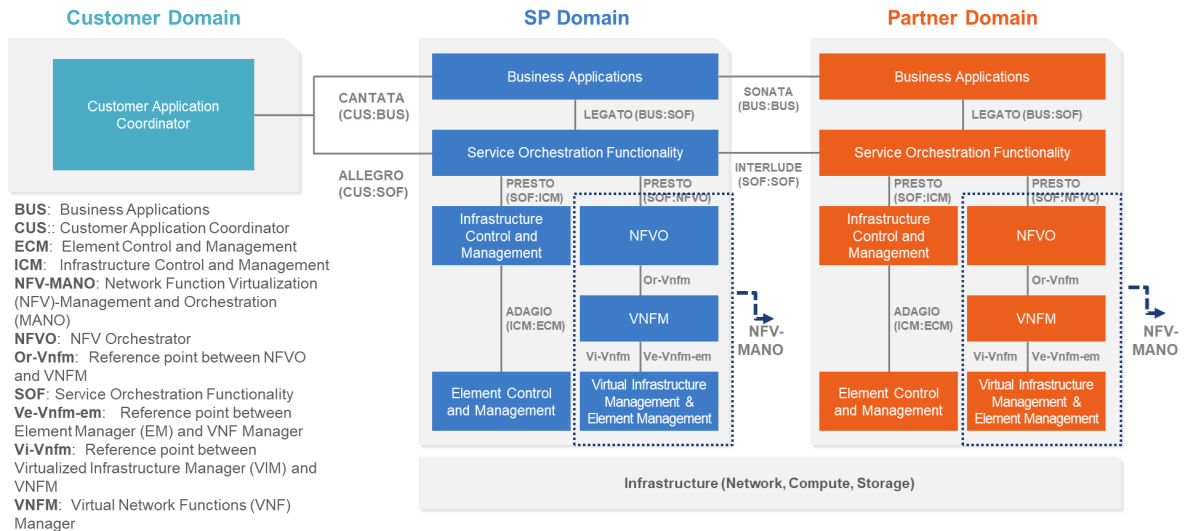
Figure 1. The LSO Reference Architecture

This document is structured as follows:

- Chapter 4 provides an introduction to Product Offering Availability and Pricing Discovery and its description in a broader context of Cantata and Sonata
- Chapter 5 gives an overview of endpoints, resource model and design patterns.
- Use cases and flows are presented in Chapter 6.
- And finally, Chapter 7 complements previous sections with a detailed resource model description.

4.1. Description

Product Offering Availability Discovery is a way for the Buyer to specify a Product Specification to the Seller and to receive from the Seller a list of zero or more Product Offering Configurations.

Pricing Discovery allows the Buyer to receive from the Seller a list of one or more terms and pricing information for the Product Configuration specified by the Buyer.

4.2. Conventions in the Document

- Code samples are formatted using code blocks. When notation `<< some text >>` is used in the payload sample it indicates that a comment is provided instead of an example value and it might not comply with the OpenAPI definition.
- Model definitions are formatted as in-line code (e.g. `ProductOfferingAvailability`).
- In UML diagrams the default cardinality of associations is `0..1`. Other cardinality markers are compliant with the UML standard.
- In the API details tables and UML diagrams required attributes are marked with a `*` next to their names.
- In UML sequence diagrams `{{variable}}` notation is used to indicate a variable to be substituted with a correct value.

4.3. Relation to Other Documents

The requirements and use cases for Product Offering Availability and Pricing Discovery functionality are defined in MEF 110 [MEF110].

4.4. Approach

As presented in Figure 2. both LSO Cantata and LSO Sonata API frameworks consist of three structural components:

- Generic API framework
- Product-independent information (Function-specific information and Function-specific operations)
- Product-specific information (MEF product specification data model)
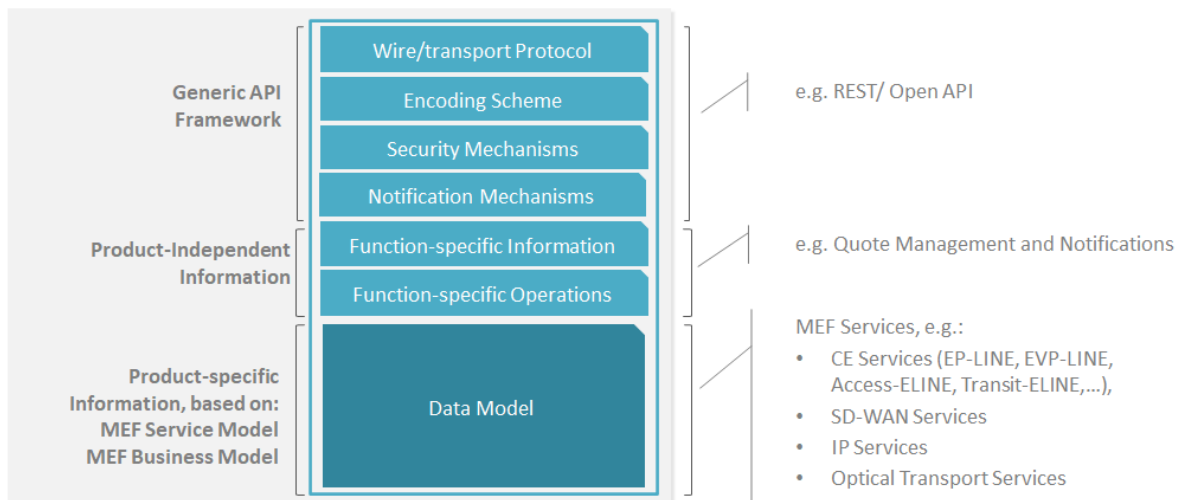
Figure 2. Cantata and Sonata API framework

The essential concept behind the framework is to decouple the common structure, information, and operations from the specific product information content.
Firstly, the Generic API Framework defines a set of design rules and patterns that are applied across all Cantata or Sonata APIs.

Secondly, the product-independent information of the framework focuses on a model of a particular Cantata or Sonata functionality and is agnostic to any of the product specifications.

Finally, the product-specific information part of the framework focuses on MEF product specifications that define business-relevant attributes and requirements for trading MEF subscriber and MEF operator services.

4.5. High-Level Flow

Product Offering Availability and Pricing Discovery is part of a broader Cantata and Sonata End-to-End flow. Figure 3. below shows a high-level diagram to get a good understanding of the whole process.
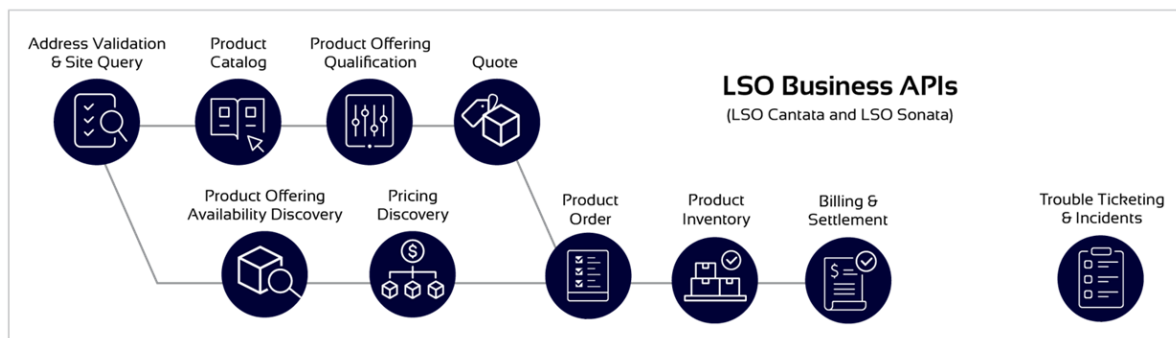
Figure 3. Cantata and Sonata End-to-End Function Flow

- Address Validation:
    - Allows the Buyer to retrieve address information from the Seller, including exact formats, for addresses known to the Seller.
- Site Retrieval:
    - Allows the Buyer to retrieve Service Site information including exact formats for Service Sites known to the Seller.
- Product Catalog:
    - Allows the Buyer to retrieve Product Categories and Product Offering information, including specification details from a Seller's Product Catalog.
- Product Offering Qualification (POQ):
    - Allows the Buyer to check whether the Seller can deliver a product or set of products from among their product offerings at the geographic address or a service site specified by the Buyer; or modify a previously purchased product.
- Quote:
    - Allows the Buyer to submit a request to find out how much the installation of an instance of a Product Offering, an update to an existing Product, or a disconnect of an existing Product will cost.
- Product Order:
    - Allows the Buyer to request the Seller to initiate and complete the fulfillment process of an installation of a Product Offering, an update to an existing Product, or

a disconnect of an existing Product at the address defined by the Buyer.
- Product Inventory:
  - Allows the Buyer to retrieve information about existing Product instances from Seller's Product Inventory.
- Billing:
  - Allows the Seller to generate the document to the Buyer relating to charges associated with Products provided by the Seller to the Buyer.
- Trouble Ticketing:
  - Allows the Buyer to create, retrieve, and update Trouble Tickets as well as receive notifications about Incidents' and Trouble Tickets' updates. This allows for managing issues and situations that are not part of the normal operations of the Product provided by the Seller.

Product Offering Availability and Pricing Discovery APIs are drawn as a fork to standard POQ and Quote as they implement the same functionality but in a slightly different approach. The discrepancies will be explained in detail in further sections.

5. API Description

This section discusses the API structure and design patterns. It starts with the high-level use cases diagram and then it describes the REST endpoints with use case mapping.

5.1. High-level use cases

Figure 4 presents a high-level use case diagram as specified in MEF 110 [MEF110]. This picture aims to help understand endpoint mapping. Use cases are described extensively in chapter 6
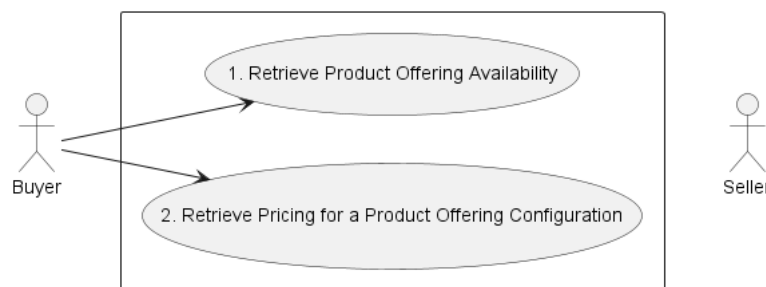
Figure 4. High-level use cases

5.2. API Endpoint and Operation Description

Base URL for Cantata: `https://{{serverBase}}:{{port}}`

`{{?/seller_prefix}}/mefApi/cantata/productOfferingAvailabilityAndPricingDiscovery/v1/`

Base URL for Sonata: `https://{{serverBase}}:{{port}}`

`{{?/seller_prefix}}/mefApi/sonata/productOfferingAvailabilityAndPricingDiscovery/v1/`

Note: All examples will include only the Sonata version of the Base Path.

Table 2 lists the API endpoints with mapping to business use cases:

API endpoint	Description	MEF 110 Use case mapping
`POST` `/productOfferingAvailability`	A request initiated by the Buyer to receive a list of zero or more Product Offering Configurations	UC 1: Retrieve Product Offering Availability
`POST /pricingDiscovery`	A request initiated by the Buyer to receive a list of one or more terms and pricing information for the Product Offering Configuration	UC 2: Retrieve Pricing for a Product Offering Configuration

Table 2. Seller side endpoints.

[R1] The Buyer implementation **MUST** be able to use all REST methods that are listed in Table 2. [MEF110 R1], [MEF110 R2], [MEF110 R29], [MEF110 R30]

5.3. Specifying the Buyer ID and the Seller ID

A business entity willing to represent multiple Buyers or multiple Sellers must follow requirements of MEF 79 [MEF79] chapter 8.8, which states:

> For requests of all types, there is a business entity that is initiating an Operation (called a Requesting Entity) and a business entity that is responding to this request (called the Responding Entity). In the simplest case, the Requesting Entity is the Buyer and the Responding Entity is the Seller. However, in some cases, the Requesting Entity may represent more than one Buyer, and similarly, the Responding Entity may represent more than one Seller.
>
> While it is outside the scope of this specification, it is assumed that the Requesting Entity and the Responding Entity are aware of each other and can authenticate requests initiated by the other party. It is further assumed that both the Buying Entity and the Requesting Entity know:
>
> a) the list of Buyers the Requesting Entity represents when interacting with this Responding Entity; and
> b) the list of Sellers that this Responding Entity represents to this Requesting Entity.

In the API the `buyerId` and `sellerId` are represented as an optional query parameters in each defined operation.

[R2] If the Requesting Entity has the authority to represent more than one Buyer the request **MUST** include the `buyerId` query parameter that identifies the Buyer being represented [MEF79 R80]

[R3] If the Requesting Entity represents precisely one Buyer with the Responding Entity, the request **MUST NOT** specify the `buyerId` [MEF79 R81]

[R4] If the Responding Entity represents more than one Seller to this Buyer the request **MUST** include the `sellerId` query parameter that identifies the The seller with whom this request is associated [MEF79 R82]

[R5] If the Responding Entity represents precisely one Seller to this Buyer, the request **MUST NOT** specify the `sellerId` [MEF79 R83]

5.4. Integration of Product Specifications into the API

Product specification schemas are defined and provided using JsonSchema (draft 7) format and are integrated into the API using the extension pattern.

The extension hosting type in the API data model is `MEFProductConfiguration`. The `@type` attribute of that type must be set to a value that uniquely identifies the product specification. A unique identifier for MEF standard product specifications is in URN format and is assigned by MEF. This identifier is provided as root schema `$id` and in product specification documentation. Use of non-MEF standard product definitions is allowed. In such a case the schema identifier must be agreed upon between the Buyer and the Seller.
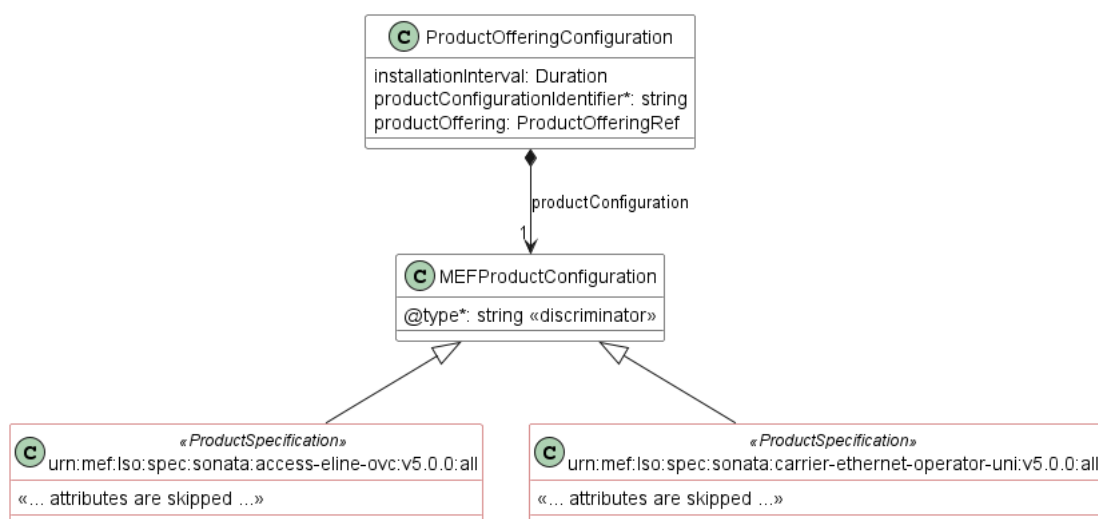
Figure 5. The Extension Pattern

Figure 5 presents two MEF `<<ProductSpecifications>>` that represent Access E-Line OVC and Carrier Ethernet Operator UNI products. When these products are used as a Product

Offering Availability or Pricing Discovery payload the `@type` of `MEFProductConfiguration` takes `"urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all"` or `"urn:mef:lso:spec:sonata:carrier-ethernet-operator-uni:v5.0.0:all"` value to indicate which product specification schema must be used to interpret a set of product-specific attributes included in the payload. An example of a product definition inside the `ProductOrderItem` is presented in Section 6.1.6.

The *all* suffix after the product type name in the URN indicates that the schema can be used in the context of all APIs (POQ, Quote, Order, or Inventory).

The example below shows a header of a Product Specification schema, which is referring to the Access E-Line OVC, where `"$id": urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all` is the abovementioned URN:

```
'$schema': http://json-schema.org/draft-07/schema#
'$id': urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all
title: MEF LSO Sonata - Access Eline OVC Product Schema
```

Product specifications are provided as Json schemas without the `MEFProductConfiguration` context.

Product-specific attributes are introduced via the `ProductOfferingConfiguration`. This entity has the `productConfiguration` attribute of type `MEFProductConfiguration` which is used as an extension point for product-specific attributes.

Implementations might choose to integrate selected product specifications into the data model during development. In such a case an integrated data model is built and product specifications are in an inheritance relationship with `MEFProductConfiguration` as described in the OAS specification. This pattern is called **Static Binding**. The SDK is additionally shipped with a set of API definitions that statically bind all product-related APIs (POQ, Quote, Order, Inventory) with all corresponding product specifications available in the release. The snippet below presents an example of a static binding of the envelope API with two product specifications. This is a subset of one yaml file describing the API. Some attributes and their descriptions are truncated for readability.

```
MEFProductConfiguration:
  description:
    MEFProductConfiguration is used as an extension point for MEF-specific
    product/service payload. The `@type` attribute is used as a discriminator
  discriminator:
    mapping:
      urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all: '#/components/schemas/AccessElineOvc'
      urn:mef:lso:spec:sonata:carrier-ethernet-operator-uni:v5.0.0:all:
'#/components/schemas/CarrierEthernetOperatorUni'
    propertyName: '@type'
  properties:
    '@type':
      description:
        The name of the type, defined in the JSON schema specified above, for
        the product that is the subject of the Request. The named type must be
        a subclass of MEFProductConfiguration.
      type: string
AccessElineOvc:
  allOf:
    - $ref: '#/components/schemas/MEFProductConfiguration'
    - $ref: '#/components/schemas/AccessElineOvcCommon'
```

```yaml
          - type: object
            properties:
              uniEp:
                $ref: '#/components/schemas/AccessElineOvcEndPoint'
                description:
                  MEF 26.2 sec. 16 - The OVC EP object for the OVC EP at the UNI. The
                  UNI OVC End Point must be included in the Access E-Line Product.
              enniEp:
                $ref: '#/components/schemas/AccessElineOvcEndPoint'
                description:
                  MEF 26.2 sec. 16 - The OVC EP object for the OVC EP at the ENNI.
                  The ENNI OVC End Point must be included in the Access E-Line
                  Product.
    CarrierEthernetOperatorUni:
        allOf:
          - $ref: '#/components/schemas/MEFProductConfiguration'
          - properties:
              listOfPhysicalLinks:
                type: array
                items:
                  $ref: '#/components/schemas/UniPhysicalLink'
                minItems: 1
                uniqueItems: true
              linkAggregation:
                $ref: '#/components/schemas/LinkAggType'
              aggregationLinkMap:
                type: array
                items:
                  $ref: '#/components/schemas/ConversationIdToAggregationLinkMap'
                minItems: 0
                uniqueItems: true
              maximumServiceFrameSize:
                type: integer
                minimum: 1522
                ...
```

Alternatively, implementations might choose not to build an integrated model and choose a different mechanism allowing runtime validation of product-specific fragments of the payload. The system can validate a given product against a new schema without redeployment. This pattern is called **Dynamic Binding.**

Regardless of chosen implementation pattern, the HTTP payload is exactly the same. Both implementation approaches must conform to the requirements specified below.

[R6] `MEFProductConfiguration` type is an extension point that **MUST** be used to integrate product specifications' properties into a request/response payload.

[R7] The `@type` property of `MEFProductConfiguration` **MUST** be used to specify the type of the extending entity.

[R8] Product attributes specified in the payload must conform to the product specification specified in the `@type` property.

5.5. Sample Product Specification

The SDK contains product specification definitions, from which UNI and Access E-Line (OVC) are used in the payload samples in this section. They are located in the SDK package at:

\productSchema\carrierEthernet\operatorEthernet\accessEline\accessElineOvc.yaml

\productSchema\carrierEthernet\operatorEthernet\carrierEthernetOperatorUni\carrierEthernetOperatorUni.yaml

The product specification data model definitions are available as JsonSchema (version `draft 7`) documents. Figures 6 and 7 present an instance diagram of OVC and UNI products with simplified configuration. This document aims to explain the pattern of exchanging product-specific attributes, not to explain the particular product itself.
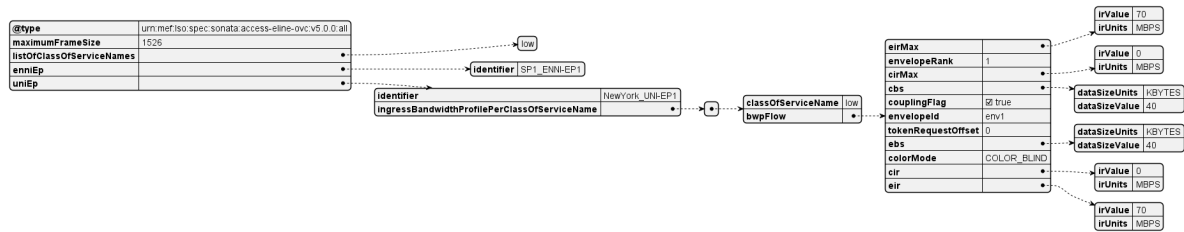
Figure 6. Access E-Line OVC Product Shortened Configuration Example
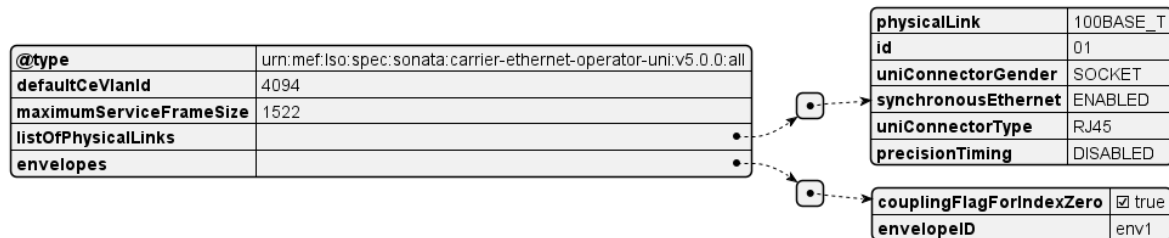
Figure 7. Operator Carrier Ethernet UNI Product Shortened Configuration Example

Product specifications define several product-related and envelope-related requirements. For example:

- for an Access E-Line OVC product two mandatory relationship roles must be specified, one with the Operator ENNI (`ENNI_REFERENCE`) and a second with the operator UNI (`UNI_REFERENCE`). First must be realized as a product relationship (relation to product existing in Seller's Inventory), second might be realized as an order item (being part of the same order) or as a product relationship
- in the case of a `modify` action, product relationships must have the same value as in the `add` action. They must not be changed
- for an Operator UNI product a place relationship (`INSTALL_LOCATION`) must be specified
- in the case of a `modify` action, place relationships must have the same value as in the `add` action. They must not be changed

Figure 8 presents the Access E-Line product required relations. The Access E-Line OVC has two product relationships:

- towards ENNI - `ENNI_REFERENCE`
- towards UNI - `UNI_REFERENCE`

The UNI product has one place relationship pointing to `INSTALL_LOCATION`.
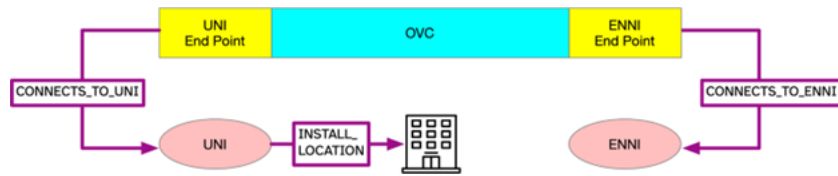
Figure 8. Operator Carrier Ethernet UNI Product Shortened Configuration Example

In case, some of these requirements are violated the Seller returns an error response to the Buyer that indicates specific functional errors. These errors are listed in the response body (a list of `Error422` entries) for HTTP `422` response.

5.6. Model Structural Validation

The structure of the HTTP payloads exchanged via Address Validation API endpoints is defined using OpenAPI version 3.0.

[R9] Implementations **MUST** use payloads that conform to these definitions.

5.7. Security Considerations

There must be an authentication mechanism whereby a Seller can be assured who a Buyer is and vice-versa. There must also be authorization mechanisms in place to control what a particular Buyer or Seller is allowed to do and what information may be obtained. However, the definition of the exact security mechanism and configuration is outside the scope of this document. It is specified by a separate MEF Project (MEF 128).

6. API Interaction & Flows

This section provides a detailed insight into the API functionality, use cases, and flows. First, it presents a list of business use cases and then provides examples with a comprehensive explanation of all usage aspects.

Use Case #	Use Case Name	Use Case Description
1	Retrieve Product Offering Availability	The Buyer requests a list of available Product Offering Configurations from the Seller for a specific Product Specification within the specified Delivery Context. The Seller responds to the Buyer with a list of Product Offering Configurations meeting the Buyer's criteria and the Installation Interval for each for the specified Delivery Context of these. Each Product Offering Configuration has a unique identifier that is passed to the Buyer by the Seller. This identifier is used for Use Case 2.

Use Case #	Use Case Name	Use Case Description
2	Retrieve Pricing for a Product Offering Configuration	The Buyer requests a list of Commercial and Contractual Options from the Seller for a specific Product Offering Configuration Identifier. The Seller synchronously responds to the Buyer with a list of Commercial and Contractual Options meeting the Buyer's Product Offering Configuration.

Table 3. Use cases description

The detailed business requirements of each of the use cases are described in section 8 of MEF 110 [MEF110].

6.1. Use case 1: Retrieve Product Offering Availability

To send a Retrieve Product Offering Availability request the Buyer uses the `POST` `/productOfferingAvailability`. The retrieve operation is performed with a `POST` operation to allow passing complex data structure of the Delivery Context as a request payload.

The flow is a simple request-response pattern, as presented in Figure 9:
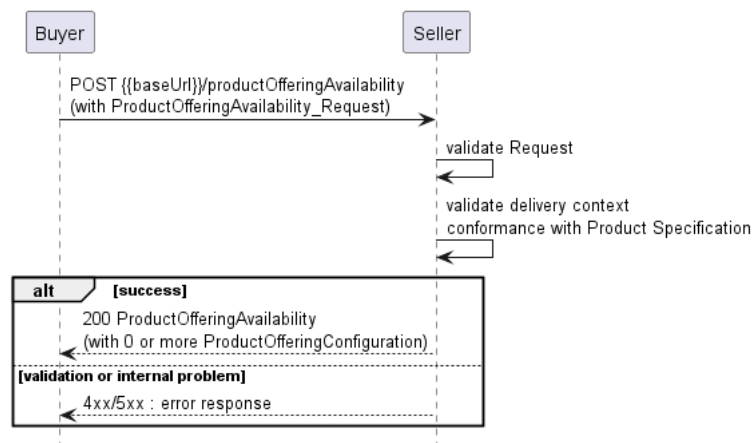
Figure 9. Use Case 1 Flow

Figure 10 presents the model of Use Case 1. The request uses `ProductOfferingAvailability_Request` as the root object and the response is provided with the use of `ProductOfferingAvailability`.
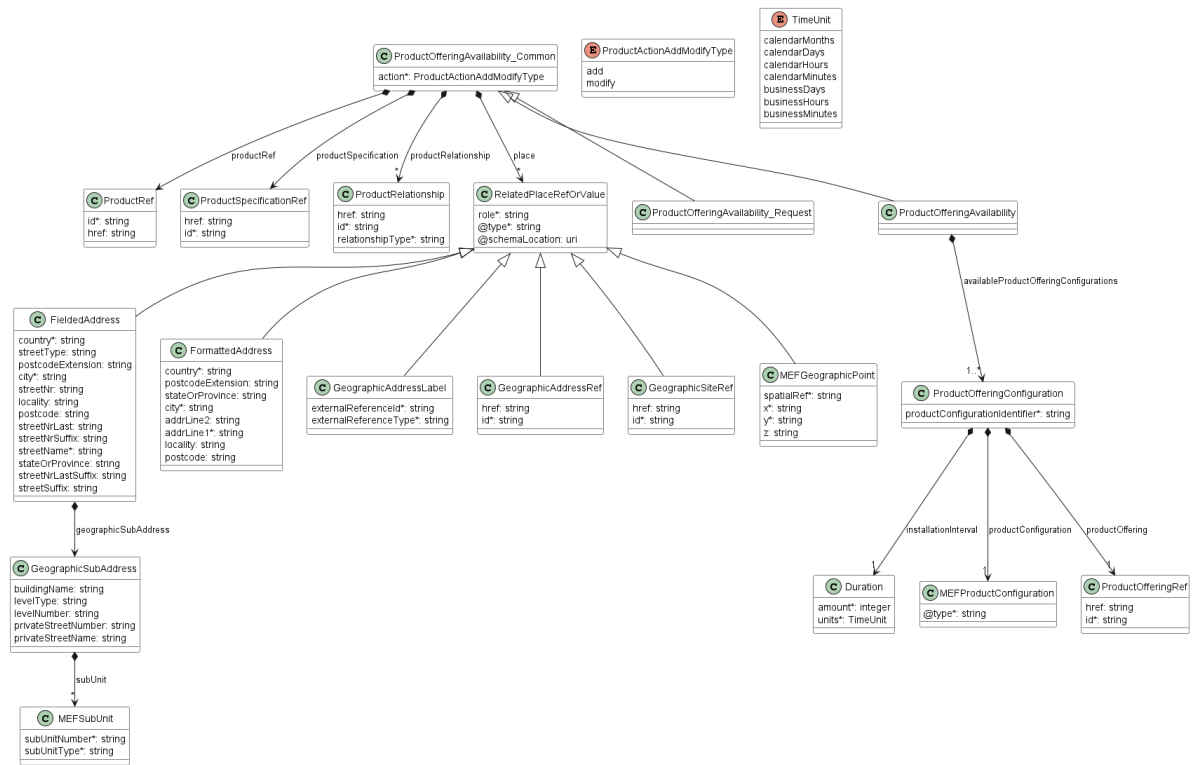
Figure 10. Use Case 1 Product Offering Availability Model

6.1.1 Request

The question that the Buyer asks with this request is "What `availableProductOfferingConfigurations` of product of given `productSpecification` can you provide me with given delivery context?".

The context can be provided by:

- `place` - a reference to a Geographic Address or Site where the product is to be installed. This applies to location-oriented products, e.g. Operator UNI.
- `productRelationship` - relation to an existing product to which product being asked for references, e.g. Access E-Line OVC pointing to UNI and ENNI products.
- `productRef` - in case of modification request a reference to an existing product that is to be updated. In this case, there is no need to provide additional context.

Note: The `place` can be provided by a reference or by value. Please refer to MEF 79 [MEF79] and MEF 79.0.2 [MEF79.0.2] for details on Address and Site modeling and example. This API specification contains a model of Address and Site but does not define it. Any further changes of these types will update the API specification, but will not be reflected in this document.

The following snippet shows the body of a Product Offering Availability check request:

`ProductOfferingAvailability_Request:`

```
  {
    "action": "add",
    "productSpecification": {
```

```
      "id": "urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all"
    },
    "productRelationship": [
      {
        "id": "UNI-ID-0001",
        "relationshipType": "CONNECTS_TO_UNI"
      },
      {
        "id": "ENNI-ID-0001",
        "relationshipType": "CONNECTS_TO_ENNI"
      }
    ]
  }
```

The request is very simple and has only three parameters:

- `action` - to state if the Buyer is interested in the creation or modification of a product.
- `productSpecification`: to point to product specification that is of Buyer's interest. For the sake of example, the `id` in the example is the `urn` as defined by MEF standard describing the Access E-Line. In practise, this is an identifier as presented by the Seller in Product Catalog.
- `productRelationship` - to provide the delivery context. As shown in Figure 8 Access E-Line requires 2 product relationships. Other products may require providing place relationship (e.g. UNI)

[R10] If `action=add`, the request **MUST** provide `productSpecification`. [MEF110 R3]

[R11] If `action=add`, the request **MUST NOT** provide `productRef`. [MEF110 R7]

[R12] If `action=add` the request **MUST** only provide `place` that conform to the requirements of Product Specification (if any). [MEF110 R5], [MEF110 R9]

[R13] If `action=add` the request **MUST** only provide `productRelationship` that conforms to requirements of Product Specification (if any). [MEF110 R6], [MEF110 R11]

[R14] If `action=modify`, the request **MUST** provide `productRef`. [MEF110 R4]

[R15] If `action=modify`, the request **MUST NOT** provide following attributes: [MEF110 R8], [MEF110 R10]

- `place`
- `productRelationship`
- `productSpecification`

6.1.2 Response

The Seller performs necessary checks for the request's correctness and compliance with provided Product Specifications and provides a response.

An example of a response to an example of request is presented in the following snippet:

```json
{
  "action": "add",
  "productSpecification": {
    "id": "urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all"
  },
  "productRelationship": [
    {
      "id": "UNI-ID-0001",
      "relationshipType": "CONNECTS_TO_UNI"
    },
    {
      "id": "ENNI-ID-0001",
      "relationshipType": "CONNECTS_TO_ENNI"
    }
  ],
  "availableProductOfferingConfigurations": [
    {
      "productOffering": {
        "id": "Access E-Line OVC - Low Class of Service"
      },
      "productConfigurationIdentifier": "PC-ID-0001",
      "productConfiguration": {
        "@type": "urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all",
        "maximumFrameSize": 1526,
        "listOfClassOfServiceNames": ["low"],
        "enniEp": {
          "identifier": "ENNI-ID-0001-EndPoint-0001"
        },
        "uniEp": {
          "identifier": "UNI-ID-0001-EndPoint-0001",
          "ingressBandwidthProfilePerClassOfServiceName": [
            {
              "classOfServiceName": "low",
              "bwpFlow": {
                "cir": {
                  "irValue": 0,
                  "irUnits": "MBPS"
                },
                "cirMax": {
                  "irValue": 0,
                  "irUnits": "MBPS"
                },
                "eir": {
                  "irValue": 70,
                  "irUnits": "MBPS"
                },
                "eirMax": {
                  "irValue": 70,
                  "irUnits": "MBPS"
                }
              }
            }
          ]
        }
      },
      "installationInterval": {
        "amount": 3,
        "units": "businessMinutes"
      }
    },
    {
      "productOffering": {
        "id": "Access E-Line OVC - High Class of Service"
      },
      "productConfigurationIdentifier": "PC-ID-0002",
      "productConfiguration": {
        "@type": "urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all",
        "maximumFrameSize": 1526,
        "listOfClassOfServiceNames": ["high"],
        "enniEp": {
          "identifier": "ENNI-ID-0001-EndPoint-0001"
        },
        "uniEp": {
          "identifier": "UNI-ID-0001-EndPoint-0001",
          "ingressBandwidthProfilePerClassOfServiceName": [
            {
              "classOfServiceName": "high",
              "bwpFlow": {
                "cir": {
                  "irValue": 200,
```

```
                "irUnits": "MBPS"
              },
              "cirMax": {
                "irValue": 200,
                "irUnits": "MBPS"
              },
              "eir": {
                "irValue": 0,
                "irUnits": "MBPS"
              },
              "eirMax": {
                "irValue": 0,
                "irUnits": "MBPS"
              }
            }
          }
        ]
      }
    },
    "installationInterval": {
      "amount": 3,
      "units": "businessMinutes"
    }
  }
 ]
}
```

Figure 11 presents the general structure of the response. It echoes back the Buyer's request and provides a list of `availableProductOfferingConfigurations`. For better readability the details of `productConfigurations` are `"<<skipped>>"` and will be presented on further figures.
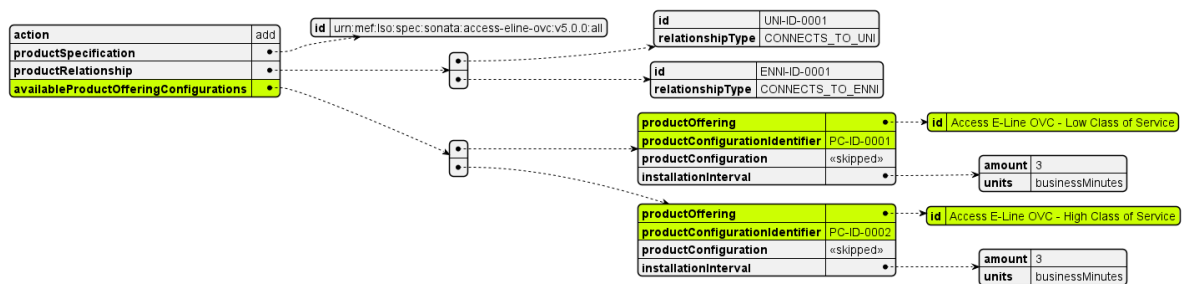
Figure 11. UC1 Response structure

In this example in the given Delivery Context and Product Specification the Seller has two possible Product Configurations. Each has its own distinct `productConfigurationIdentifier` and is from a different Product Offering of Access E-Line OVC. Different available Product Configurations can potentially be from the same Product Offering as well.

Note: The `productConfigurationIdentifier` identifies only the Product Specific Attributes values as provided in the `productConfiguration` attribute. It does not identify the action or the delivery context. This allows to ask for Pricing Discovery of the same `productConfiguration` in different delivery contexts.
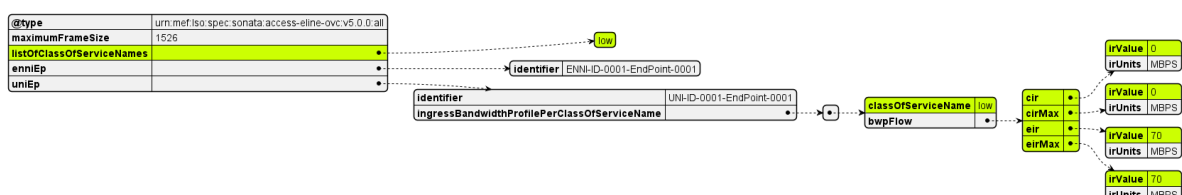
Figure 12. UC1 Response Product Configuration 1
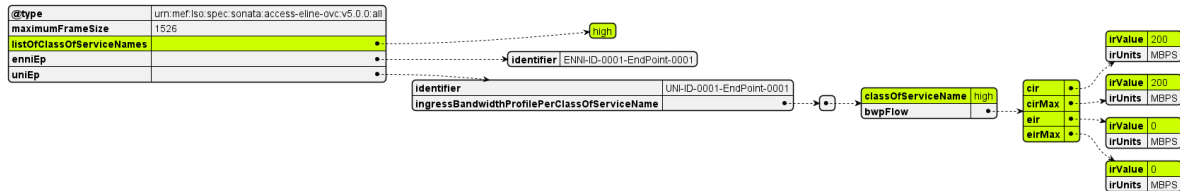
Figure 13. UC1 Response Product Configuration 2

Figures 12 and 13 present details of two product configurations and highlight discrepancies between them. The first one has a `low` class of service and has `eir` and `eirMax` attributes set to `70 MBPS`. This means it offers a best-effort bandwidth of a maximum 70 MBPS. The second one is of a `high` class of service and has the `cir` and `cirMax` set to `200 MBPS` thus offering a guaranteed bandwidth of `200 MBPS`.

[R16] The Seller **MUST** echo back the attributes of the Buyer's Product Offering Availability request. [MEF110 R17]

[R17] The Seller **MUST NOT** return a response that contains `productConfigurations` that differ only by `installationInterval` (in which all other attributes are the same). [MEF110 R18]

[R18] If the request is successful, the response given by the Seller **MUST** include a list of zero or more `availableProductOfferingConfigurations`. [MEF110 R19]

[R19] If the request is unsuccessful or fails validation, the Seller **MUST** return an `Error` response with the appropriate Error Code. [MEF110 R20]

[R20] For each returned `ProductOfferingConfiguration` the Seller **MUST** include following attributes: [MEF110 R21], [MEF110 R22], [MEF110 R24], [MEF110 R26]

- `installationInterval`
- `productOffering`
- `productConfigurationIdentifier`
- `productConfiguration`

[R21] Every returned `productConfigurationIdentifier` **MUST** uniquely identify a `productConfiguration` within the Seller. [MEF110 R21]

[R22] Every returned `productConfiguration` **MUST** contain only attributes specified by Product Specification. [MEF110 R25]

[R23] `productConfigurationIdentifier` **MUST** be valid for Pricing Discovery request for a period of at least 15 minutes. [MEF110 R28]

6.2. Use case 2: Retrieve Pricing for a Product Offering Configuration

This Use Case allows the Buyer to perform a Pricing Discovery (ask for Commercial and Contractual Options) for a specific Product Configuration identified by `productConfigurationIdentifier` (most probably) obtained in previous use case.

This can be done with the use of the `POST /pricingDiscovery` operation. The retrieve operation is performed with a `POST` operation to allow passing the complex data structure of the Delivery Context as a request payload.

The flow is a simple request-response pattern, as presented in Figure 14:
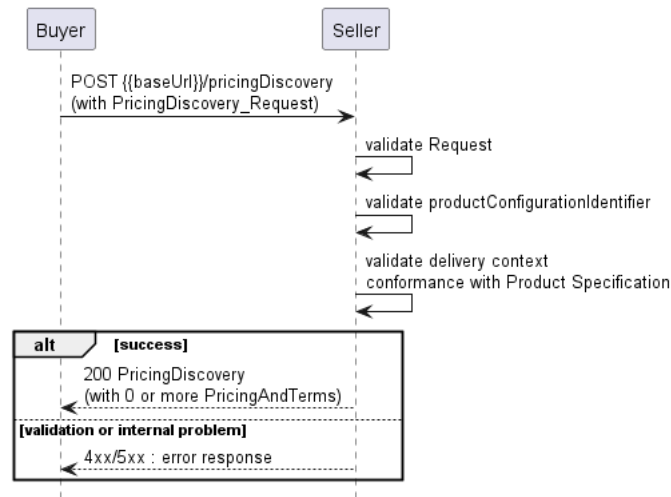
Figure 14. Retrieve Address by Identifier Flow

Figure 15 presents the model of Use Case 1. The request uses `PricingDiscovery_Request` as the root object and the response is provided with use of `PricingDiscovery`.
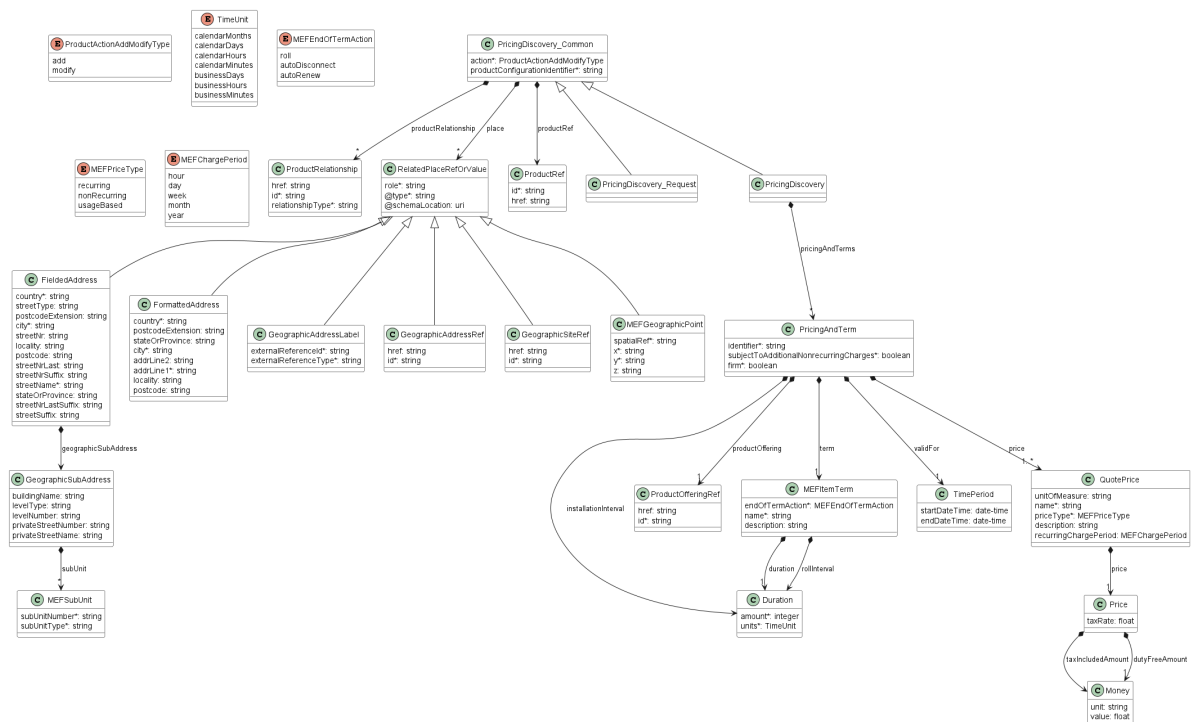
Figure 15. Use Case 2 Pricing Discovery Model

6.2.1 Request

The question that the Buyer asks with this request is "What `PricingAndTerms` for a `productConfigurationIdentifier` can you provide me with given delivery context?".

The rules of providing the delivery context are the same as described in Use Case 1.

The following snippet shows the body of a Price Discovery request:

`PriceDiscovery_Request:`

```
{
  "action": "add",
  "productRelationship": [
    {
      "id": "UNI-ID-0001",
      "relationshipType": "CONNECTS_TO_UNI"
    },
    {
      "id": "ENNI-ID-0001",
      "relationshipType": "CONNECTS_TO_ENNI"
    }
  ],
  "productConfigurationIdentifier": "PC-ID-0002"
}
```

The request is very simple and has only three parameters:

- `action` - to state if the Buyer is interested in the creation or modification of a product.
- `productConfigurationIdentifier`: to refer to Product Configuration that the Buyer is willing to price
- `productRelationship` - to provide the delivery context. As shown on Figure 8 Access E-Line requires 2 product relationships. Other products may require providing place relationship (e.g. UNI)

[R24] The request **MUST** provide `productConfigurationIdentifier`. [MEF110 R31], [MEF110 R33]

[R25] If `action=add` and the Product Specification defines mandatory place relations, the request **MUST** provide them accordingly in the `place` attribute. [MEF110 R31]

[R26] If `action=add` and the Product Specification (derived from `productConfigurationIdentifier`) defines mandatory product relations, the request **MUST** provide them accordingly in the `productRelationship` attribute. [MEF110 R6], [MEF110 R132]

[R27] If `action=add`, the request **MUST NOT** provide `productRef`. [MEF110 R32]

[R28] If `action=modify`, the request **MUST** provide `productRef`. [MEF110 R33]

[R29] If `action=add` the request **MUST** only provide `productRelationship` that conforms to requirements of Product Specification (if any). [MEF110 R35], [MEF110 R36], [MEF110 R40]

[R30] If `action=add` the request **MUST** only provide a `place` that conform to the requirements of Product Specification (if any). [MEF110 R37], [MEF110 R38], [MEF110 R41], [MEF110 R42]

6.2.1 Response

The Seller performs necessary checks for the request's correctness, compliance with provided Product Specification, and `productConfigurationIdentifier` validity and then provides a response.

An example of a response to an example of request is presented in the following snippet:

```
{
  "action": "add",
  "productRelationship": [
    {
      "id": "UNI-ID-0001",
      "relationshipType": "CONNECTS_TO_UNI"
    },
    {
      "id": "ENNI-ID-0001",
      "relationshipType": "CONNECTS_TO_ENNI"
    }
  ],
  "productConfigurationIdentifier": "PC-ID-0002",
  "pricingAndTerms": [
    {
      "term": {
        "duration": {
          "amount": 12,
          "units": "calendarMonths"
        },
        "endOfTermAction": "roll",
        "name": "1-year term",
        "rollInterval": {
          "amount": 1,
          "units": "calendarMonths"
        }
      },
      "validFor": {
        "startDateTime": "2023-06-02T12:24:48.687Z",
        "endDateTime": "2023-06-09T12:24:48.687Z"
      },
      "subjectToAdditionalNonrecurringCharges": true,
      "price": [
        {
          "price": {
            "taxRate": 10,
            "taxIncludedAmount": {
              "unit": "EUR",
              "value": 110
            },
            "dutyFreeAmount": {
              "unit": "EUR",
              "value": 100
            }
          },
          "name": "Monthly price for a 1-year subscription",
          "priceType": "recurring",
          "recurringChargePeriod": "month"
        }
      ],
      "firm": true,
      "installationInterval": {
        "amount": 3,
        "units": "businessMinutes"
      },
      "productOffering": {
        "id": "Access E-Line OVC - High Class of Service"
      }
```

```
    },
    {
      "term": {
        "duration": {
          "amount": 36,
          "units": "calendarMonths"
        },
        "endOfTermAction": "roll",
        "name": "3-year term",
        "rollInterval": {
          "amount": 1,
          "units": "calendarMonths"
        }
      },
      "validFor": {
        "startDateTime": "2023-06-02T12:24:48.687Z",
        "endDateTime": "2023-06-09T12:24:48.687Z"
      },
      "subjectToAdditionalNonrecurringCharges": true,
      "price": [
        {
          "price": {
            "taxRate": 10,
            "taxIncludedAmount": {
              "unit": "EUR",
              "value": 88
            },
            "dutyFreeAmount": {
              "unit": "EUR",
              "value": 80
            }
          },
          "name": "Monthly price for a 3-year subscription",
          "priceType": "recurring",
          "recurringChargePeriod": "month"
        }
      ],
      "firm": true,
      "installationInterval": {
        "amount": 3,
        "units": "businessMinutes"
      },
      "productOffering": {
        "id": "Access E-Line OVC - High Class of Service"
      }
    }
  }
 ]
}
```

Figure 16 presents the Seller's response. It echoes back the Buyer's request and provides a list of `pricingAndTerms`. In this example in the given Delivery Context and `productConfigurationIdentifier` the Seller has two possible `PricingAndTerms`. Discrepancies between them are highlighted. The first one is for a 1-year contract with a 110 EUR monthly rate, the other is for a 3-year contract with a 88 EUR monthly rate. Each has its own distinct `identifier`.
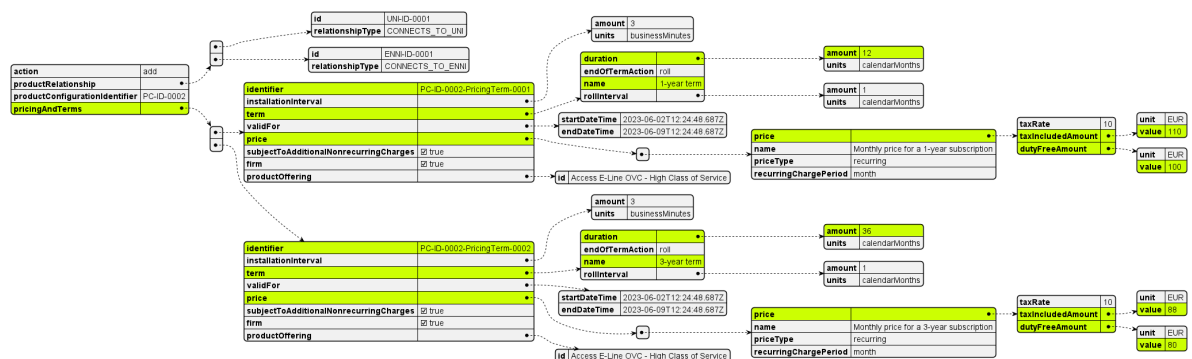
Figure 16. UC2 Response

[R31] The Seller **MUST** echo back the attributes of the Buyer's Pricing Discovery request. [MEF110 R48] Sample request:

[R32] If the request is successful, the response given by the Seller **MUST** include a list of zero or more `pricingAndTerms`. [MEF110 R44]

[R33] For each returned `PricingAndTerm` the Seller **MUST** include following attributes: [MEF110 R49], [MEF110 R50], [MEF110 R51], [MEF110 R55], [MEF110 R56], [MEF110 R57]

- `identifier`
- `term`
- `term.endOfTermAction`
- `validFor`
- `price`
- `subjectToAdditionalNonrecurringCharges`

[D1] The period of time after `autoRenew` occurs and in which the Buyer can disconnect the Product without penalty **SHOULD** be agreed to by the Buyer and Seller. [MEF110 D1]

[R34] If the `endOfTermAction` is set to `roll` the Seller **MUST** provide the `rollInterval` attribute. [MEF110 R53]

[R35] If the `endOfTermAction` is set to `autoDisconnect` or `autoRenew` the Seller **MUST NOT** provide the `rollInterval` attribute. [MEF110 R54]

[R36] If a Seller has returned multiple `productConfigurationIdentifiers` to the Buyer in different Product Offering Availability responses that refer to the same `productConfiguration` values, then the Seller **MUST** return the same response to a Pricing Discovery request for a given delivery context that contains any of those `productConfigurationIdentifiers` that are within their validity period. [MEF110 R47]

[D2] For every unique combination of `productConfigurationIdentifier`, `action`, `place`, `productAELRelationships`, `term`, `endOfTermAction`, `rollInterval` and `price`, the Seller **SHOULD** give the `PriceAndTerm` a consistent `identifier` no matter how many times the Buyer performs the request. [MEF110 D2]

The following requirements apply to every `QuotePrice` object returned in the `proce` attribute of the response.

[R37] The `recurringChargePeriod` **MUST** be provided if the `priceType` is `recurring`. [MEF110 R63]

[R38] The `recurringChargePeriod` **MUST NOT** be provided if the `priceType` is `nonRecurring` or `usageBased`. [MEF110 R64]

[R39] The `unitOfMeasure` **MUST** be provided if the `priceType` is `usageBased`. [MEF110 R65]

[R40] The `unitOfMeasure` **MUST NOT** be provided if the `priceType` is `recurring` or `nonRecurring`. [MEF110 R66]

[R41] The `priceAndTerms.firm` **MUST** be provided if the `priceType` is `nonRecurring`. [MEF110 R67]

[R42] The `priceAndTerms.firm` attribute **MUST** have the value of `true` if the `priceType` is `recurring` or `usageBased`. [MEF110 R68]

7. API Details

7.1. API patterns

7.1.1. Indicating errors

Erroneous situations are indicated by appropriate HTTP responses. An error response is indicated by HTTP status 4xx (for client errors) or 5xx (for server errors) and appropriate response payload. The Product Order API uses the error responses as depicted and described below.

Implementations can use HTTP error codes not specified in this standard in compliance with rules defined in RFC 7231 [RFC7231]. In such a case, the error message body structure might be aligned with the `Error`.
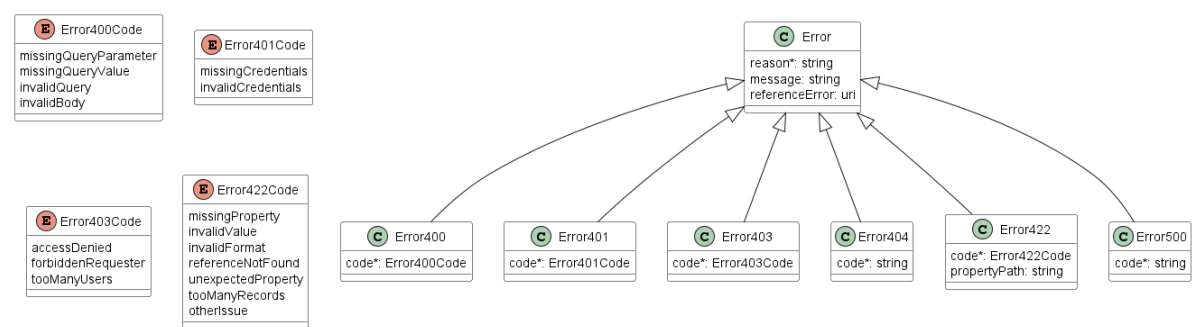
Figure 43. Data model types to represent an erroneous response

7.1.1.1. Type Error

Description: Standard Class used to describe API response error Not intended to be used directly. The `code` in the HTTP header is used as a discriminator for the type of error returned in runtime.

Name	Type	Description
reason*	string *maxLength = 255*	Text that explains the reason for the error. This can be shown to a client user.

Name	Type	Description
message	string	Text that provides mode details and corrective actions related to the error. This can be shown to a client user.
referenceError	uri *format = uri*	URL pointing to documentation describing the error

7.1.1.2. Type Error400

Description: Bad Request. (https://tools.ietf.org/html/rfc7231#section-6.5.1)

Inherits from:

- Error

Name	Type	Description
code*	Error400Code	One of the following error codes: - missingQueryParameter: The URI is missing a required query-string parameter - missingQueryValue: The URI is missing a required query-string parameter value - invalidQuery: The query section of the URI is invalid. - invalidBody: The request has an invalid body

7.1.1.3. enum Error400Code

Description: One of the following error codes:

- missingQueryParameter: The URI is missing a required query-string parameter
- missingQueryValue: The URI is missing a required query-string parameter value
- invalidQuery: The query section of the URI is invalid.
- invalidBody: The request has an invalid body

7.1.1.4. Type Error401

Description: Unauthorized. (https://tools.ietf.org/html/rfc7235#section-3.1)

Inherits from:

- Error

Name	Type	Description

Name	Type	Description
code*	Error401Code	One of the following error codes: - missingCredentials: No credentials provided. - invalidCredentials: Provided credentials are invalid or expired

7.1.1.5. enum Error401Code

Description: One of the following error codes:

- missingCredentials: No credentials provided.
- invalidCredentials: Provided credentials are invalid or expired

7.1.1.6. Type Error403

Description: Forbidden. This code indicates that the server understood the request but refuses to authorize it. (https://tools.ietf.org/html/rfc7231#section-6.5.3)

Inherits from:

- Error

Name	Type	Description
code*	Error403Code	This code indicates that the server understood the request but refuses to authorize it because of one of the following error codes: - accessDenied: Access denied - forbiddenRequester: Forbidden requester - tooManyUsers: Too many users

7.1.1.7. enum Error403Code

Description: This code indicates that the server understood the request but refuses to authorize it because of one of the following error codes:

- accessDenied: Access denied
- forbiddenRequester: Forbidden requester
- tooManyUsers: Too many users

7.1.1.8. Type Error422

Description: Unprocessable entity due to a business validation problem. (https://tools.ietf.org/html/rfc4918#section-11.2)

Inherits from:

- Error

Name	Type	Description
code*	Error422Code	One of the following error codes: - missingProperty: The property the Seller has expected is not present in the payload - invalidValue: The property has an incorrect value - invalidFormat: The property value does not comply with the expected value format - referenceNotFound: The object referenced by the property cannot be identified in the Seller system - unexpectedProperty: Additional property, not expected by the Seller has been provided - tooManyRecords: the number of records to be provided in the response exceeds the Seller's threshold. - otherIssue: Other problem was identified (detailed information provided in a reason)
propertyPath	string	A pointer to a particular property of the payload that caused the validation issue. It is highly recommended that this property should be used. Defined using JavaScript Object Notation (JSON) Pointer (https://tools.ietf.org/html/rfc6901).

7.1.1.9. enum Error422Code

Description: One of the following error codes:

- missingProperty: The property the Seller has expected is not present in the payload
- invalidValue: The property has an incorrect value
- invalidFormat: The property value does not comply with the expected value format
- referenceNotFound: The object referenced by the property cannot be identified in the Seller system
- unexpectedProperty: Additional property, not expected by the Seller has been provided
- tooManyRecords: the number of records to be provided in the response exceeds the Seller's threshold.
- otherIssue: Other problem was identified (detailed information provided in a reason)

7.1.1.10. Type Error500

Description: Internal Server Error. (https://tools.ietf.org/html/rfc7231#section-6.6.1)

Inherits from:

- Error

Name	Type	Description
code*	string	The following error code: - internalError: Internal server error - the server encountered an unexpected condition that prevented it from fulfilling the request.

7.2. API Data model

7.2.1. Product Offering Availability

7.2.1.1 Type ProductOfferingAvailability_Common

Description: Common attributes for ProductOfferingAvailability_Request and ProductOfferingAvailability

Name	Type	M/O	Description	MEF 110
action	ProductActionAddModifyType	M	The action to be performed by the Seller to fulfil any Order that results from this interaction.	Action
productRef	ProductRef	O	A reference to a Product in the Product Inventory. It MUST be provided when the `action` = `modify`. It MUST NOT be provided when the `action` = `add`	Product Identifier

Name	Type	M/O	Description	MEF 110
productSpecification	ProductSpecificationRef	O	A reference to a Product Specification used to describe the Product. This MUST be provided when the `action` is `add`. It MUST NOT be provided when the `action` = `modify`	Product Specification Identifier
productRelationship	ProductRelationship[]	O	A list of Product Relationships as defined by the Product Specification.	Product Relationships
place	RelatedPlaceRefOrValue[]	O	A list of Addresses or Sites and their roles with relation to the Product.	Locations

7.2.1.2. Type ProductOfferingAvailability_Request

Description: Product Offering Availability Request allows the Buyer to provide the action, delivery context and Product Specification to ask the Seller to provide a list of available Product Offering Configurations.

Inherits from:

- ProductOfferingAvailability_Common

7.2.1.3. Type ProductOfferingAvailability

Description: Product Offering Availability allows the Seller to provide the list of Product Offering Configurations matching action, delivery context and Product Specification provided by the Buyer

Inherits from:

- ProductOfferingAvailability_Common

Name	Type	M/O	Description	MEF
availableProductOfferingConfigurations	ProductOfferingConfiguration[] *minItems = 0*	M		Not repres in ME 110

7.2.1.4. `enum` ProductActionAddModifyType

Description: Action to be performed on the Product.

The following mapping has been used between `ProductActionType` and MEF 110:

ProductActionType	MEF 110
add	INSTALL
modify	CHANGE

7.2.1.5. Type ProductRelationship

Description: A relationship to an existing Product. The requirements for usage for given Product are described in the Product Specification.

Name	Type	M/O	Description	MEF 110
href	string	O	Hyperlink to the product in Seller's inventory that is referenced Hyperlink MAY be used when providing a response by the Seller Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request	Not represented in MEF 110
id	string	M	Unique identifier of the related Product	Related Product Identifier

Name	Type	M/O	Description	MEF 110
relationshipType	string	M	Specifies the type (nature) of the relationship to the related Product. The nature of required relationships varies for Products of different types. For example, a UNI or ENNI Product may not have any relationships, but an Access E-Line may have two mandatory relationships (related to the UNI on one end and the ENNI on the other). More complex Products such as multipoint IP or Firewall Products may have more complex relationships. As a result, the allowed and mandatory `relationshipType` values are defined in the Product Specification.	Product Relationship Nature

7.2.1.6. Type ProductOfferingConfiguration

Description: Allows the Seller to provide a detailed information of Product Configuration matching Buyer's Product Offering Availability Request.

Name	Type	M/O	Description	MEF
productOffering	ProductOfferingRef	M	A reference for the Product Offering that this configuration must be ordered as.	Produ Offeri Identi
productConfigurationIdentifier	string	M	An identifier of the returned `productConfiguration` that can be used in next step for Pricing Discovery	Produ Offeri Config Identi

Name	Type	M/O	Description	MEF
productConfiguration	MEFProductConfiguration	M	MEFProductConfiguration is used to specify the product specific attributes. The @type is used as a discriminator. It holds the detailed configuration of the product attributes.	Produ Specif Attrib
installationInterval	Duration	M	The indicative duration >0 for the delivery of the configuration in the specified Delivery Context when manual work is required. The shortest interval is specified by the Seller. It is not considered a commitment by the Seller.	Install Interv

7.2.1.7. Type ProductOfferingRef

Description: A reference to a Product Offering offered by the Seller to the Buyer.

Name	Type	M/O	Description	MEF 110
href	string	O	Hyperlink to a Product Offering in Sellers catalog. In case Seller is not providing a catalog API this field is not used. The catalog is provided by the Seller to the Buyer during onboarding. Hyperlink MAY be used by the Seller in responses Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in MEF 110
id	string	M	id of a Product Offering. It is assigned by the Seller. The Buyer and the Seller exchange information about offerings' ids during the onboarding process.	Product Offering Identifier

7.2.1.8. Type MEFProductConfiguration

Description: MEFProductConfiguration is used as an extension point for MEF specific product/service payload. The @type attribute is used as a discriminator

Name	Type	M/O	Description	MEF 110
@type	string	M	The name of the type that uniquely identifies the type of the product that is the subject of the POQ Request. In case of MEF product this is the URN provided in the Product Specification.	Not represented in MEF 110

7.2.1.9. Type ProductRef

Description: A reference to a Product in the Product Inventory

Name	Type	M/O	Description	MEF 110
id	string	M	Unique identifier of a Product	Product Identifier
href	string	O	Hyperlink to a Product instance in Sellers Product Inventory. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in MEF 110

7.2.1.10. Type ProductSpecificationRef

Description: A reference to a structured set of well-defined technical attributes and/or behaviors that are used to construct a Product Offering for sale to a market.

Name	Type	M/O	Description	MEF 110
href	string	O	Hyperlink to a Product Specification in Sellers catalog. In case Seller is not providing a catalog API this field is not used. The catalog is provided by the Seller to the Buyer during onboarding. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in MEF 110
id	string	M	Unique identifier of the Product Specification	Product Specification Identifier

7.2.1.11. Type MEFItemTerm

Description: Describes a term (also known as commitment)

Name	Type	M/O	Description	MEF 110
duration	Duration	M	Duration of the term	Duration
endOfTermAction	MEFEndOfTermAction	M	The action the Seller will take once the term expires.	End of Term Action
name	string	M	Name of the term	Not represented in MEF 110
description	string	O	Description of the term	Description
rollInterval	Duration	O	The recurring period that the Buyer is required to pay at the end of the term to extend the term after the original term has expired. If `endOfTermAction` is equal to `roll` then `rollInterval` MUST be specified. If `endOfTermAction` is equal to `autoRenew` or `autoDisconnect`, then `rollInterval` MUST NOT be specified.	Roll Interval

7.2.1.12. Type Duration

Description: A Duration in a given unit of time e.g. 3 hours, or 5 days.

Name	Type	M/O	Description	MEF 110
amount	integer *minimum = 0*	M	Duration (number of seconds, minutes, hours, etc.)	Amount
units	TimeUnit	M	Time unit enumerated	Units

7.2.1.13. Type TimePeriod

Description: A period of time, either as a deadline (endDateTime only) a startDateTime only, or both.

Name	Type	M/O	Description	MEF 110
startDateTime	date-time *format = date-time*	O	Start of the time period, using IETC-RFC-3339 format. If you define a start, you must also define an end	Not represented in MEF 110
endDateTime	date-time *format = date-time*	O	End of the time period, using IETC-RFC-3339 format	Valid Until

7.2.1.14. enum TimeUnit

Description: Represents a unit of time.

Value	MEF 110
calendarMonths	CALENDAR_MONTHS
calendarDays	CALENDAR_DAYS
calendarHours	CALENDAR_HOURS
calendarMinutes	MINUTES
businessDays	BUSINESS_DAYS
businessHours	BUSINESS_HOURS
businessMinutes	MINUTES

7.2.2. Pricing Discovery

7.2.2.1 Type PricingDiscovery_Common

Description: A set of attribute common to PricingDiscovery_Request and PricingDiscovery

Name	Type	M/O	Description	MEF 110
action	ProductActionAddModifyType	M	The action to be performed by the Seller to fulfill any Order that results from this interaction.	Action

Name	Type	M/O	Description	MEF 110
productConfigurationIdentifier	string	M	The identifier for the Product Configuration Pricing Discovery is being requested for.	Product Offering Configuration Identifier
productRef	ProductRef	O	A reference to a Product in the Product Inventory	Product Identifier
productRelationship	ProductRelationship[]	O	A list of Product Relationships as defined by the Product Specification.	Product Relationships
place	RelatedPlaceRefOrValue[]	O	A list of Addresses or Sites and their roles with relation to the Product.	Locations

7.2.2.2. Type PricingDiscovery_Request

Description: Pricing Discovery Request allows the Buyer to provide the action, delivery context and Product Configuration Identifier to ask the Seller to provide a list of available Pricing and Terms

Inherits from:

- PricingDiscovery_Common

7.2.2.3. Type PricingDiscovery

Description: PricingDiscovery allows the Seller to provide the list of Pricing and Terms matching action, delivery context and Product Configuration Identifier provided by the Buyer

Inherits from:

- PricingDiscovery_Common

Name	Type	M/O	Description	MEF 110
pricingAndTerms	PricingAndTerm[]	M	Pricing and terms matching Buyer's criteria	Product Offering Configuration Commercial and Contractual Options

7.2.2.4. Type PricingAndTerm

Description: Pricing and term specification for an available Product Configuration

Name	Type	M/O	Description
identifier	string	M	The identifier of the Pricing and Term.
installationInterval	Duration	M	The indicative duration >0 for the delivery of the configuration in the specified Delivery Context when manual work is required. The shortest interval is specified by the Seller. It is not considered a commitment by the Seller.
term	MEFItemTerm	M	The minimum term for this PricingAndTerm for the given productConfigurationIdentifier within the requested Delivery Context.

Name	Type	M/O	Description
validFor	TimePeriod	M	The date that the PricingAndTerm is valid until. For use in the context of this attribute, only the endDateTime attribute must be used.
price	QuotePrice[] *minItems = 1*	M	The prices that apply to this PricingAndTerm for the given Product Configuration within the requested Delivery Context.
subjectToAdditionalNonrecurringCharges	boolean	M	An indicator to inform the Buyer that additional non-recurring charges may be added during fulfillment.
firm	boolean	M	An indicator specifying whether or not this charge may change during fulfillment.
productOffering	ProductOfferingRef	M	A reference for the Product Offering that this configuration must be ordered as.

7.2.2.5. Type QuotePrice

Description: Description of price and discount awarded

Name	Type	M/O	Description	MEF 110
name	string	M	Name of the price	Name
description	string	O	Description of the price	Description
priceType	MEFPriceType	M	Indicates if the price is for recurring, non-recurring, or usage based charges	Price Type

Name	Type	M/O	Description	MEF 110
unitOfMeasure	string	O	Unit of Measure if price depending on it (Gb, SMS volume, etc..) MSU be specified when `priceType` is `usageBased`	Unit Of Measure
price	Price	M	The associated price	Price Tax Rate Price Tax Included Amount Price Duty Free Amount
recurringChargePeriod	MEFChargePeriod	O	The recurring duration for which this charge will be applied. MUST be provided if `priceType is `recurring`	Recurring Charge Period

7.2.2.6. enum MEFChargePeriod

Description: Used for a recurring charge to indicate period.

Value	MEF 110
hour	HOUR
day	DAY
week	WEEK
month	MONTH
year	YEAR

7.2.2.7. enum MEFEndOfTermAction

Description: The action that needs to be taken by the Seller once the term expires.

Value	Description

Value	Description
roll	The Product's contract will continue on a rolling basis once the contract's initial term ex-pires,
autoDisconnect	The Product will automatically be disconnected (and contract terminated) by the Seller once the contract term expires,
autoRenew	The Product's contract will be renewed for another term equivalent to the original contract term.

7.2.2.8. enum **MEFPriceType**

Description: Indicates if the price is for recurring or non-recurring charges.

Value	MEF 110
recurring	RECURRING
nonRecurring	NON_RECURRING
usageBased	USAGE_BASED

7.2.2.9. Type Money

Description: A base/value business entity used to represent money

Name	Type	M/O	Description	MEF 110
unit	string	M	Currency (ISO4217 norm uses 3 letters to define the currency)	Currency
value	float *format = float*	M	A positive floating point number	Value

7.2.2.10. Type Price

Description: Provides all amounts (tax included, duty-free, tax rate), used currency and percentage to apply for Price Alteration.

Name	Type	M/O	Description	MEF 110
taxRate	float *format = float*	O	Price Tax Rate. Unit: [%]. E.g. value 16 stand for 16% tax.	Price Tax Rate
taxIncludedAmount	Money	O	All taxes included amount (expressed in the given currency)	Price Tax Included Amount

Name	Type	M/O	Description	MEF 110
dutyFreeAmount	Money	M	All taxes excluded amount (expressed in the given currency)	Price Duty Free Amount

7.2.3. Place representation

There are several formats in which place information can be represented. The model is presented in Figure 43.
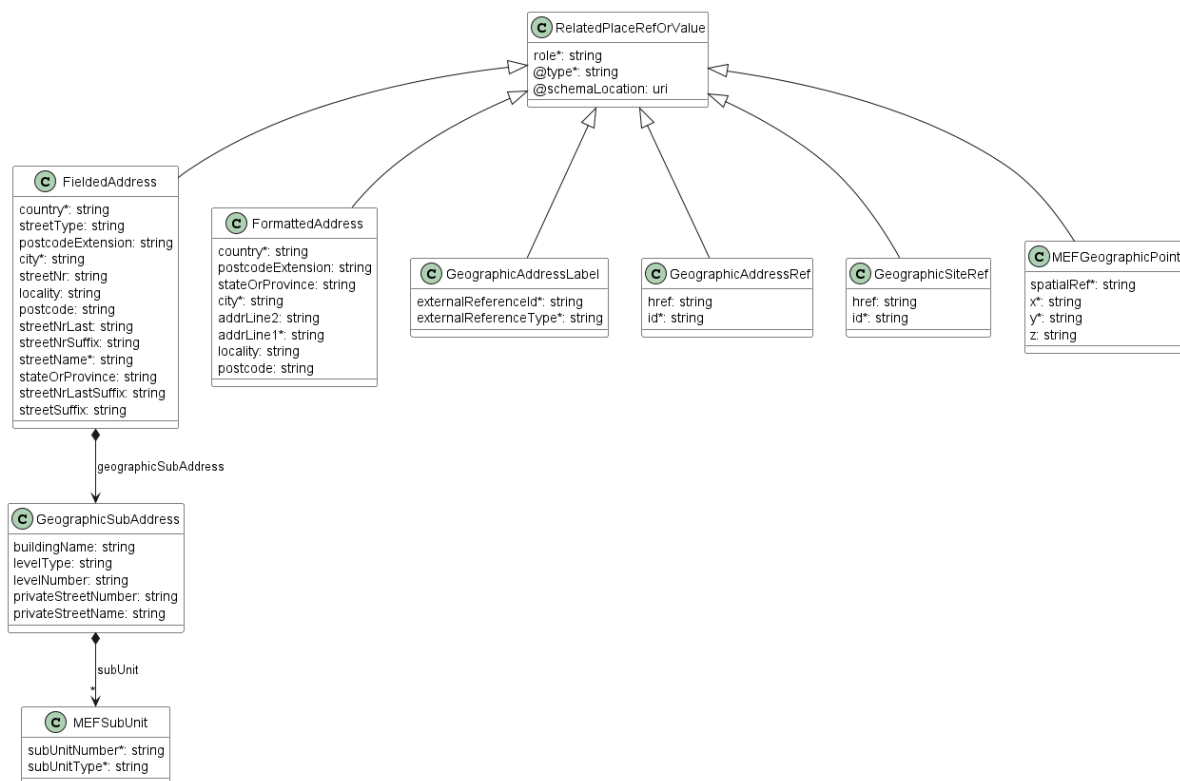
Figure 43. Data model types representing a place

7.2.3.1. Type RelatedPlaceRefOrValue

Description: Place defines the places (locations) where the products being subject of this qualification are to be provided.

Name	Type	M/O	Description	MEF 110
role	string	M	Role of this place. The values that can be specified here are described by Product Specification (e.g. "INSTALL_LOCATION").	Location Role

Name	Type	M/O	Description	MEF 110
@type	string	M	This field is used as discriminator. The value is the name of one of the types that inherit from it using 'allOf', i.e. one of FieldedAddress, FormattedAddress, GeographicAddressLabel, MEFGeographicPoint, GeographicAddressRef, GeographicSiteRef. Moreover, it might discriminate for an additional related place as defined in '@schemaLocation'.	Not represented in MEF 110
@schemaLocation	uri *format = uri*	O	A URL to a JSON-Schema file that defines additional attributes and relationships. May be used to define additional related place types. Usage of this attribute must be agreed between Buyer and Seller.	Not represented in MEF 110

7.2.3.2. Type FieldedAddress

Description: A type of Address that has a discrete field and value for each type of boundary or identifier down to the lowest level of detail. For example "street number" is one field, "street name" is another field, etc.

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	M/O	Description	MEF 110
country	string	M	Country that the address is in	Country

Name	Type	M/O	Description	MEF 110
streetType	string	O	The type of street (e.g., alley, avenue, boulevard, brae, crescent, drive, highway, lane, terrace, parade, place, tarn, way, wharf)	Street Type
postcodeExtension	string	O	An extension of a postal code. E.g. the part following the dash in a US urban property address	Postal Code Extension
city	string	M	The city that the address is in	City
streetNr	string	O	Number identifying a specific property on a public street. It may be combined with streetNrLast for ranged addresses. MEF 79 defines it as required for the Seller response, however in certain countries it is not used, so it's optional in API.	Street Number
locality	string	O	The locality that the address is in	Locality

Name	Type	M/O	Description	MEF 110
postcode	string	O	A descriptor for a postal delivery area, used to speed and simplify the delivery of mail (also known as zip code) MEF 79 defines it as required however as in certain countries it is not used we make it optional in API.	Postal Code
streetNrLast	string	O	Last number in a range of street numbers allocated to a property	Street Number Last
streetNrSuffix	string	O	The first street number suffix	Street Number Suffix
streetName	string	M	Name of the street or other street type	Street Name
stateOrProvince	string	O	The State or Province that the address is in	State Or Province
streetNrLastSuffix	string	O	Last street number suffix for a ranged address	Street Number Suffix Last
geographicSubAddress	GeographicSubAddress	O	Additional fields used to specify an address, as detailed as possible.	Not represented in MEF 110
streetSuffix	string	O	A modifier denoting a relative direction	Street Suffix

7.2.3.3. Type FormattedAddress

Description: A type of Address that has discrete fields for each type of boundary or identifier with the exception of the street and more specific location details, which are combined into a maximum of two strings based on local postal addressing conventions.

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	M/O	Description	MEF 110
country	string	M	Country that the address is in	Country
postcodeExtension	string	O	An extension of a postal code. E.g. the part following the dash in a US urban property address	Postal Code Extension
stateOrProvince	string	O	The State or Province that the address is in	State Or Province
city	string	M	The city that the address is in	City
addrLine2	string	O	The second address line in a formatted address	Address Line 2
addrLine1	string	M	The first address line in a formatted address	Address Line 1
locality	string	O	An area of defined or undefined boundaries within a local authority or other legislatively defined area, usually rural or semi-rural in nature	Locality
postcode	string	O	A descriptor for a postal delivery area used to speed and simplify the delivery of mail (also known as ZIP code)	Postal Code

7.2.3.4. Type **MEFGeographicPoint**

Description: A MEFGeographicPoint defines a geographic point through coordinates.

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	M/O	Description	MEF 110

Name	Type	M/O	Description	MEF 110
spatialRef	string	M	The spatial reference system used to determine the coordinates (e.g. "WGS84"). The system used and the value of this field are to be agreed during the onboarding process.	Spatial Reference
x	string	M	The latitude expressed in the format specified by the `spacialRef`	Latitude
y	string	M	The longitude expressed in the format specified by the `spacialRef`	Longitude
z	string	O	The elevation expressed in the format specified by the `spacialRef`	Elevation

7.2.3.5. Type GeographicSubAddress

Description: Additional fields used to specify an address, as detailed as possible.

Name	Type	M/O	Description	MEF 110
buildingName	string	O	Allows for identification of places that require building name as part of addressing information	Building Name
subUnit	MEFSubUnit[]	O	Representation of a MEFSubUnit It is used for describing subunit within a subAddress e.g. BERTH, FLAT, PIER, SUITE, SHOP, TOWER, UNIT, WHARF.	Not represented in MEF 110
levelType	string	O	Describes level types within a building	Level Type
levelNumber	string	O	Used where a level type may be repeated e.g. BASEMENT 1, BASEMENT 2	Level Number
privateStreetNumber	string	O	Private streets numbers internal to a private street	Private Street Number

Name	Type	M/O	Description	MEF 110
privateStreetName	string	O	Private streets internal to a property (e.g. a university) may have internal names that are not recorded by the land title office	Private Street Name

7.2.3.6. Type GeographicAddressRef

Description: A reference to a Geographic Address resource available through Address Validation API.

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	M/O	Description	MEF 110
href	string	O	Hyperlink to the referenced Address. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in MEF 110
id	string	M	Identifier of the referenced Geographic Address. This identifier is assigned during a successful address validation request (Geographic Address Management API)	Fielded | Formatted | Geographic Address Label | Geographic Point Identifier

7.2.3.7. Type GeographicSiteRef

Description: A reference to a Geographic Site resource available through Service Site API

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	M/O	Description	MEF 110
href	string	O	Hyperlink to the referenced Site. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in MEF 110

Name	Type	M/O	Description	MEF 110
id	string	M	Identifier of the referenced Geographic Site.	Site Identifier

7.2.3.8. Type GeographicAddressLabel

Description: A unique identifier controlled by a generally accepted independent administrative authority that specifies a fixed geographical location.

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	M/O	Description	MEF 110
externalReferenceId	string	M	The unique reference to an Address as provided by the Administrative Authority.	Administrative Authority Address Label
externalReferenceType	string	M	The organization or standard from the organization that administers this Geographic Address Label ensuring it is unique within the Administrative Authority. The value(s) to be used are to be agreed upon during the onboarding. For North American providers this would normally be CLLI (Common Language Location Identifier) code.	Administrative Authority

7.2.3.9. Type MEFSubUnit

Description: Allows for sub unit identification

Name	Type	M/O	Description	MEF 110
subUnitNumber	string	M	The discriminator used for the subunit, often just a simple number but may also be a range.	Sub Unit Name

Name	Type	M/O	Description	**MEF 110**
subUnitType	string	M	The type of subunit e.g.BERTH, FLAT, PIER, SUITE, SHOP, TOWER, UNIT, WHARF.	Sub Unit Type

8. References

- MEF55.1, Lifecycle Service Orchestration (LSO): Reference Architecture and Framework, February 2021
- MEF79, Address, Service Site, and Product Offering Qualification Management, Requirements and Use Cases, November 2019
- MEF79.0.2, Amendment to MEF 79: Address Validation, July 2021
- MEF110 Product Offering Availability and Pricing Discovery - Business Requirements and Use Cases, Draft Standard 1, June 2023
- [MEF139] Internet Access Product Schemas and Developer Guide, July 2023
- RFC2119, Key words for use in RFCs to Indicate Requirement Levels, March 1997
- RFC7231, Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content, June 2014 https://tools.ietf.org/html/rfc7231
- RFC8174, Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words, May 2017