



MEF Standard MEF 116

LSO Cantata and LSO Sonata Product Inventory API – Developer Guide

May 2022

Disclaimer

© MEF Forum 2022. All Rights Reserved.

The information in this publication is freely available for reproduction and use by any recipient and is believed to be accurate as of its publication date. Such information is subject to change without notice and MEF Forum (MEF) is not responsible for any errors. MEF does not assume responsibility to update or correct any information in this publication. No representation or warranty, expressed or implied, is made by MEF concerning the completeness, accuracy, or applicability of any information contained herein and no liability of any kind shall be assumed by MEF as a result of reliance upon such information.

The information contained herein is intended to be used without modification by the recipient or user of this document. MEF is not responsible or liable for any modifications to this document made by any other party.

The receipt or any use of this document or its contents does not in any way create, by implication or otherwise:

- a) any express or implied license or right to or under any patent, copyright, trademark or trade secret rights held or claimed by any MEF member which are or may be associated with the ideas, techniques, concepts or expressions contained herein; nor
- b) any warranty or representation that any MEF members will announce any product(s) and/or service(s) related thereto, or if such announcements are made, that such announced product(s) and/or service(s) embody any or all of the ideas, technologies, or concepts contained herein; nor
- c) any form of relationship between any MEF member and the recipient or user of this document.

Implementation or use of specific MEF standards, specifications, or recommendations will be voluntary, and no Member shall be obliged to implement them by virtue of participation in MEF Forum. MEF is a non-profit international organization to enable the development and worldwide adoption of agile, assured and orchestrated network services. MEF does not, expressly or otherwise, endorse or promote any specific products or services.

Table of Contents

1	List of Contributing Members	1
2	Abstract.....	1
3	Terminology and Abbreviations.....	2
4	Compliance Levels	3
5	Introduction.....	4
5.1	Description.....	5
5.2	Conventions in this Document	5
5.3	Relation to Other Documents	6
5.4	Approach	6
5.5	High-Level Flow.....	7
6	API Description.....	8
6.1	High-level Use Cases.....	8
6.2	Resource/Endpoint Description	8
6.2.1	Seller Side Endpoints	8
6.3	Specifying the Buyer ID and the Seller ID.....	9
6.4	Integration of Product Specifications into Product Inventory API.....	10
6.5	Sample Product Specification.....	12
6.6	Model Structural Validation	14
6.7	Security Considerations	14
7	API Interaction and Flows	15
7.1	Use Case 1: Retrieve Product List.....	15
7.2	Use Case 2: Retrieve Product by Identifier	18
8	API Details.....	26
8.1	API Patterns.....	26
8.1.1	Indicating Errors.....	26
8.1.2	Response Pagination	29
8.2	Management API Data Model.....	29
8.2.1	Product	30
8.2.2	Common.....	32
9	References.....	39

List of Figures

Figure 1 – The LSO Reference Architecture	4
Figure 2 – Cantata and Sonata API Framework	6
Figure 3 – Cantata and Sonata End-to-End Function Flow	7
Figure 4 – Use Cases.....	8
Figure 5 – The Extension Pattern with Sample Product Specific Extensions	12
Figure 6 – A Simplified View on the Access E-Line Product Specification Data Model.....	13
Figure 7 – A Simplified View on the UNI Product Specification Data Model	14
Figure 8 – Use Case 1: Retrieve Product List Flow.....	16
Figure 9 – Use Case 1: Retrieve Product List Model	18
Figure 10 – Use Case 2: Retrieve Product by Identifier Flow	18
Figure 11 – Use Case 2: Retrieve Product Model.....	19
Figure 12 – Product State Machine.....	23
Figure 13 – Data Model Types to Represent an Erroneous Response.....	26
Figure 14 – Product Inventory Data Model	29

List of Tables

Table 1 – Terminology and Abbreviations	2
Table 2 – Seller Side Endpoints.....	9
Table 3 – Use Case Descriptions	15
Table 4 – Product States	24
Table 5 – Required Related Contact Information role	24
Table 6 – Type Error	26
Table 7 – Type Error400	27
Table 8 – Type Error401	27
Table 9 – Type Error403	27
Table 10 – Type Error404	28
Table 11 – Type Error422	28
Table 12 – Type Error500	29
Table 13 – Type MEFProduct.....	31
Table 14 – Type MEFProduct_Find	31
Table 15 – enum MEFProductStatusType	32
Table 16 – Type MEFProductStatusChange.....	32
Table 17 – Type ProductPrice.....	32
Table 18 – Type Duration	32
Table 19 – Type FieldedAddress	33
Table 20 – Type GeographicSubAddress	33
Table 21 – Type MEFSUBUnit.....	34
Table 22 – Type MEFBillingAccountRef.....	34
Table 23 – enum MEFChargePeriod	34
Table 24 – Type MEFItemTerm	34
Table 25 – enum MEFEndOfTermAction	35
Table 26 – enum MEFPriceType	35
Table 27 – Type MEFProductConfiguration	35
Table 28 – Type MEFProductOrderItemRef	35
Table 29 – Type Price	36
Table 30 – Type Money	36
Table 31 – Type ProductOfferingRef	36
Table 32 – Type ProductRelationship.....	37
Table 33 – Type ProductSpecificationRef	37
Table 34 – Type RelatedContactInformation	37
Table 35 – Type ProductSpecificationRef	38
Table 36 – Type TimeUnit.....	38

1 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.

- Amartus
- Colt
- Lumen Technologies
- NEC/Netcracker
- Orange
- Proximus
- Spirent Communications

2 Abstract

This standard assists the implementation of the Product Inventory functionality defined for the LSO Cantata and LSO Sonata Interface Reference Points (IRPs), for which requirements and use cases are defined in MEF 81 *Product Inventory Management Requirements and Use Cases* [11] and MEF 81.0.1 *Amendment to MEF 81: Product Inventory Management* [12]. This standard consists of this document and complementary API definition. 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>

commit id: `f3c91e572b9bdecce6198fd15141d4f67e92e5f8`

- [productApi/inventory/productInventoryManagement.api.yaml](#)

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

commit id: `2e18fa505952a8bc1d3be1ff78306d406ed47b6d`

- [productApi/inventory/productInventoryManagement.api.yaml](#)

3 Terminology and Abbreviations

This section defines the terms used in this document. In many cases, the normative definitions to terms are found in other documents. In these cases, the third column is used to provide the reference that is controlling, in other MEF or external documents.

Term	Definition	Reference
Application Program Interface (API)	In the context of LSO, API describes one of the Management Interface Reference Points based on the requirements specified in an Interface Profile, along with a data model, the protocol that defines operations on the data and the encoding format used to encode data according to the data model. In this document, API is used synonymously with REST API.	MEF 55.1 [8]
Buyer	In the context of this document, denotes the organization or individual acting as the customer in a transaction over a Cantata (Customer ↔ Service Provider) or Sonata (Service Provider ↔ Partner) Interface.	This document, adapted from MEF 80 [10]
Requesting Entity	The business organization that is acting on behalf of one or more Buyers. In the most common case, the Requesting Entity represents only one Buyer and these terms are then synonymous.	MEF 81 [11]
Responding Entity	The business organization that is acting on behalf of one or more Sellers. In the most common case, the Responding Entity represents only one Seller and these terms are then synonymous.	MEF 81 [11]
Representational State Transfer Application Program Interface	REST provides a set of architectural constraints that, when applied as a whole, emphasizes scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems.	Fielding [7]
REST API	Representational State Transfer Application Program Interface	Fielding [7]
Seller	In the context of this document, denotes the organization acting as the supplier in a transaction over a Cantata (Customer ↔ Service Provider) or Sonata (Service Provider ↔ Partner) Interface.	This document; adapted from MEF 80 [11]

Table 1 – Terminology and Abbreviations

4 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 [2], RFC 8174 [5]) when, and only when, they appear in all capitals, as shown here. All key words 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 **MAY** or **OPTIONAL**) are labeled as [**Ox**] for optional.

5 Introduction

This standard specification document describes the Application Programming Interface (API) for Product Inventory Management functionality of the LSO Cantata Interface Reference Point (IRP) and LSO Sonata IRP as defined in the MEF 55.1 *Lifecycle Service Orchestration (LSO): Reference Architecture and Framework* [8]. The LSO Reference Architecture is shown in Figure 1 with both IRPs highlighted.

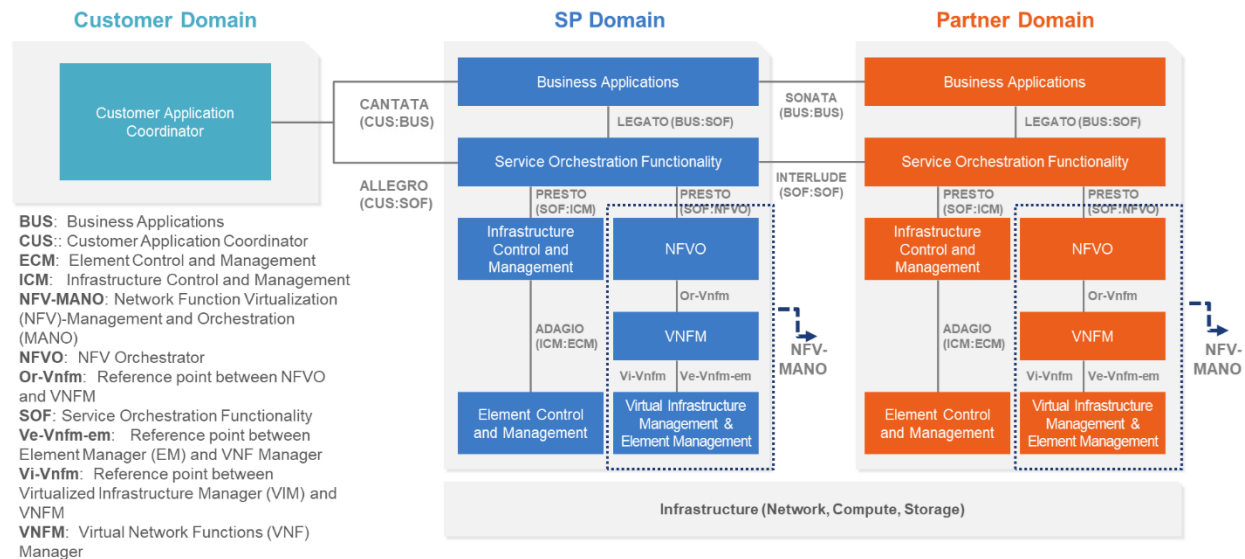


Figure 1 – The LSO Reference Architecture

Cantata and Sonata IRPs define pre-ordering and ordering functionalities that allow an automated exchange of information between business applications of the Buyer (Customer or Service Provider) and Seller (Partner) Domains. Those are:

- Address Validation
- Site Retrieval
- Product Offering Qualification
- Product Quote
- Product Ordering
- Product Inventory
- Trouble Ticketing
- Billing

The business requirements and use cases for Product Inventory are defined in MEF 81 *Quote Management Requirements and Use Cases* [11] and MEF 81.0.1 *Amendment to MEF 81: Product Inventory Management* [12].

This document focuses on implementation aspects and is structured as follows:

- Section 5 provides an introduction to Product Inventory and its description in a broader context of Cantata and Sonata and their corresponding SDKs.
- Section 6 gives an overview of endpoints, resource model, and design patterns.
- Use cases and flows are presented in Section 7.
- And finally, Section 8 complements previous sections with a detailed API description.

5.1 Description

The Product Inventory API allows the Buyer to retrieve information about existing (previously ordered) Products from the Seller's Inventory. The Seller's Product Inventory is a set of instances of Products that have been ordered by a Buyer. It is assumed, for a Product to exist in the Seller's Product Inventory, that the Seller has passed the `Product.id` to the Buyer per MEF 57.2.

The API payloads exchanged between the Buyer and the Seller consist of product-independent and product-specific parts. The product-independent part is technically defined in this standard. The product-specific part is defined in the product specification standard of the concerned product. Both standards must be used in combination to validate the correctness of the payloads. Section 6.4 explains how to use product specifications as the Quote API payloads.

This document uses samples of Access E-Line Product specification definitions to construct API payload examples in Section 7.

Note: The Access E-Line product is valid only in the Sonata context. It is used only for the explanation of the rules of combining the product-agnostic (envelope) and product-specific (payload) parts of the APIs. The examples are not normative and are not updated to reflect new versions of the product specification (MEF 106). It is out of the scope of this document to explain the details of any product.

Product specifications are defined using JSON Schema (draft 7) standard [1], whereas Product Inventory API is defined using OpenAPI 3.0 [13]. The payloads exchanged through Inventory endpoints must comply with the Product specification schema as well as with MEF 81 [11] and MEF 81.0.1 [12] requirements for Product Inventory.

5.2 Conventions in this 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. `GeographicAddress`).
- 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.

5.3 Relation to Other Documents

The requirements and use cases for Product Inventory Management are defined in MEF 81 [11] and MEF 81.0.1 [12]. The API definition builds on *TMF637 Product Inventory Management API REST Specification R19.0.0* [14]. Product Inventory Use Cases must support the use of any of MEF product specifications.

5.4 Approach

As presented in Figure 2, both Cantata and Sonata API frameworks consists 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)

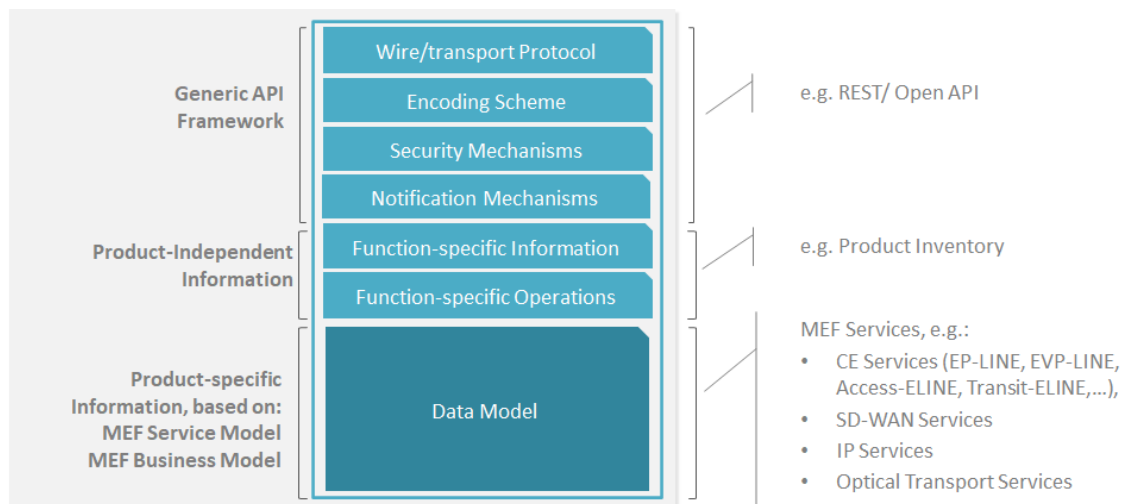


Figure 2 – Cantata and Sonata API Framework

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 API suites throughout all LSO Interface Reference Points' 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. For example, this standard is describing the Product Inventory model and operations that allow retrieval of detailed Product information from the Seller's system. 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.

This Developer Guide is not defining MEF product specifications but can be used in combination with any product specifications defined by or compliant with MEF.

5.5 High-Level Flow

Product Inventory 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 and Product Inventory's position within it.

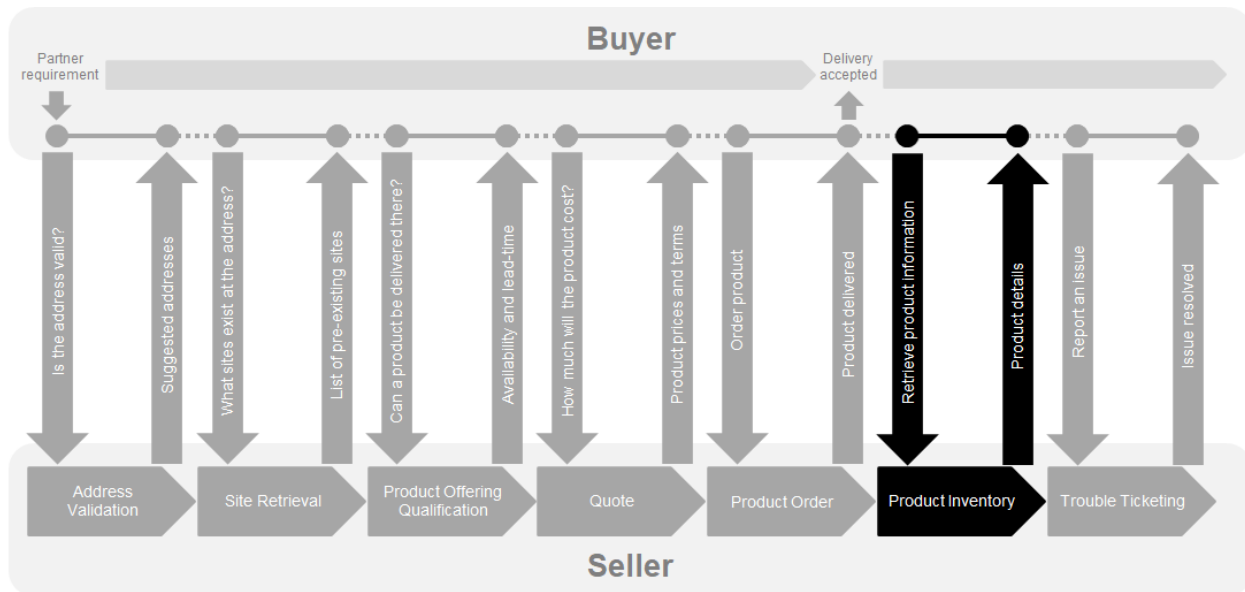


Figure 3 – Cantata and Sonata End-to-End Function Flow

- **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 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 the information about existing Product instances from Seller's Product Inventory.

- 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 managing issues and situations that are not part of normal operations of the Product provided by the Seller.

6 API Description

This section presents the API structure and design patterns. It starts with the high-level use cases diagram. Then it describes the REST endpoints with use case mapping. Next, it gives an overview of the API resource model and an explanation of the design pattern that is used to combine product-agnostic and product-specific parts of API payloads. Finally, payload validation and API security aspects are discussed.

6.1 High-level Use Cases

Figure 4 presents a high-level use case diagram as specified in MEF 81 [11] in Section 7.1. This picture aims to help understand the endpoint mapping. Use cases are described extensively in Section 7.

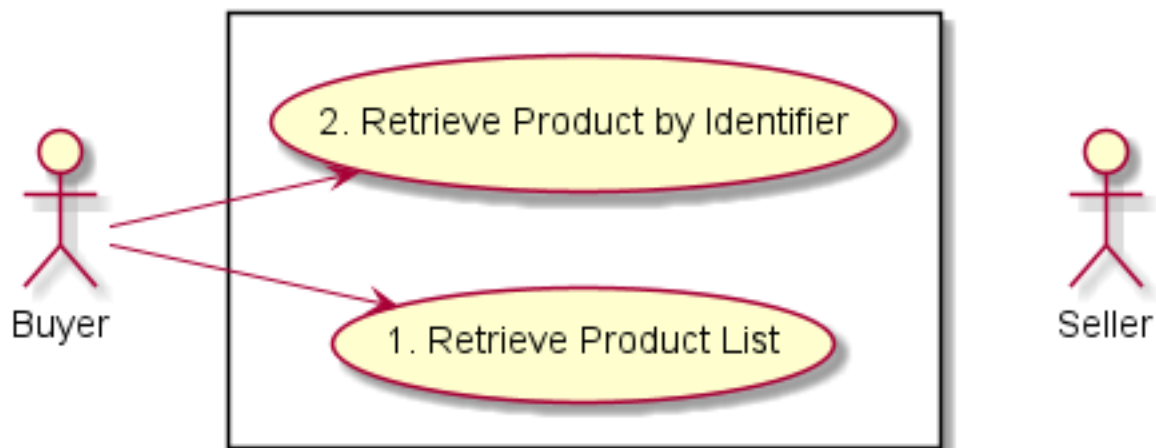


Figure 4 – Use Cases

6.2 Resource/Endpoint Description

6.2.1 Seller Side Endpoints

Base URL for Cantata:

https://{{serverBase}}:{{port}}{?/seller_prefix}}/mefApi/cantata/quoteManagement/v2/

Base URL for Sonata:

https://{{serverBase}}:{{port}}{?/seller_prefix}}/mefApi/sonata/quoteManagement/v8/

The following API endpoints are implemented by the Seller and allow the Buyer to retrieve existing Product details or a list of Products. The endpoints and corresponding data model are defined in `productApi/inventory/productInventoryManagement.api.yaml`.

API Endpoint	Description	MEF 81 Use Case Mapping
<code>GET /product</code>	A request initiated by the Buyer to retrieve a list of Products (in any state) from the Seller based on a set of filter criteria.	Use Case 1: Retrieve Product List
<code>GET /product/{{id}}</code>	A request initiated by the Buyer to retrieve full details of a single Product based on a Product identifier.	Use Case 2: Retrieve Product by Identifier

Table 2 – Seller Side Endpoints

- [R1] The Buyer implementation **MUST** be able to use all REST methods listed in the table above MEF 81 R3, R4, R5, and R6 [11].

6.3 Specifying the Buyer ID and the Seller ID

A business entity willing to represent multiple Buyers or multiple Sellers must follow requirements of MEF 81 [11] Section 8.3, 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:

- the list of Buyers the Requesting Entity represents when interacting with this Responding Entity; and
- the list of Sellers that this Responding Entity represents to this Requesting Entity.

In the API the `buyerId` and `sellerId` are represented as query parameters in each operation defined in `productInventoryManagement.api.yaml`.

- [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. MEF 81 R12 [11].
- [R3] If the Requesting Entity represents precisely one Buyer with the Responding Entity, the request **MUST NOT** specify the `buyerId`. MEF 81 R13 [11].

- [R4] If the Responding Entity represents more than one Seller to this Buyer the request **MUST** include the `sellerId` query parameter that identifies the Seller with whom this request is associated. MEF 81 R14 [11].
- [R5] If the Responding Entity represents precisely one Seller to this Buyer, the request **MUST NOT** specify the `sellerId`. MEF 81 R15 [11].

6.4 Integration of Product Specifications into Product Inventory API

Product specifications are defined using JSON Schema (draft 7) [1] format and are integrated into the `Product` payload using the TMF 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 between the Buyer and the Seller.

The example below shows a header of a Product Specification schema, where `"$id":urn:mef:lso:spec:sonata:access-eline:v1.0.0:quote` is the abovementioned URN:

```
'$schema': http://json-schema.org/draft-07/schema#
'$id': urn:mef:lso:spec:sonata:access-eline:v1.0.0:inventory
title: MEF LSO Sonata - Access Eline OVC (Inventory) Product Specification
```

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

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

Implementations might choose to integrate selected product specifications to 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 [13]. 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 snippets below present an example of a static binding of the envelope API with a number of MEF product specifications, from both the `MEFProductConfiguration` and the product specification point of view:

`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:AccessElineOvc:v1.0.0:inventory:
'#/components/schemas/AccessElineOvcInventory_v1.0.0'
urn:mef:lso:spec:cantata-sonata:SubscriberUni:v1.0.0:inventory:
'#/components/schemas/SubscriberUniInventory_v1.0.0'
urn:mef:lso:spec:cantata-sonata:EplEvc:v1.0.0:inventory:
'#/components/schemas/EplEvcInventory_v1.0.0'
urn:mef:lso:spec:sonata:OperatorUNI:v1.0.0:inventory:
'#/components/schemas/OperatorUNIInventory_v1.0.0'
  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

AccessElineOvcInventory_v1.0.0:
  allOf:
    - $ref: '#/components/schemas/MEFProductConfiguration'
    - description:
        OVC Service Attributes control the behavior observable at and between
        External Interfaces to the Carrier Ethernet Network (CEN). The
        behaviors are achieved by the Network Operator and the Operator's
        client (the Service Provider in this case) agreeing on the value for
        each of the Service Attributes.
```

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 is able to 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 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.

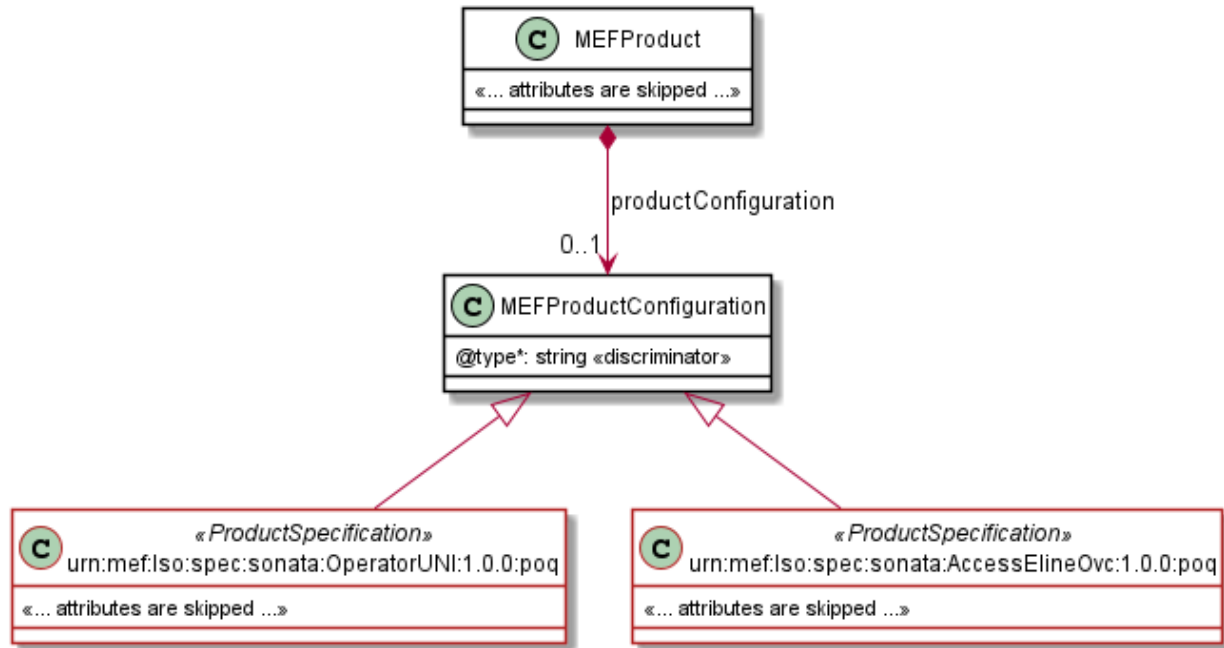


Figure 5 – The Extension Pattern with Sample Product Specific Extensions

Figure 5 presents two MEF `<<ProductSpecifications>>` that represent Access E-Line Operator UNI and OVC products. When these products are used as a Product Inventory payload the `@type` of `MEFProductConfiguration` takes the `"urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:inventory"` or `"urn:mef:lso:spec:sonata:OperatorUNI:1.0.0:inventory"` value to indicate which product specification should be used to interpret a set of product-specific attributes included in the payload. An example of a product configuration is presented in Section 7.1.

The *inventory* suffix after the product type name in the URN comes from the approach that the product schemas may differ depending on the Interface Reference Point function they are used with.

6.5 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. In the Celine release they are located in the SDK package at:

```

\productSchema\carrierEthernet\accessEline\inventory\accessElineOvc.yaml
\productSchema\carrierEthernet\carrierEthernetOperatorUni\inventory\carrierEthernetOperatorUni.yaml

```

The product specification data model definitions are available as JSON Schema (draft 7) [1] documents. Figure 6 and 7 depict simplified UML views on these data models in which:

- the mandatory attributes are denoted with `*`,
- the mandatory relations have a cardinality of `1` or `1..*`,
- some relations and attributes that are not essential to the understanding of the product specification model are omitted.

The red color in Figure 6 and 7 below highlights the data model of Access E-Line.

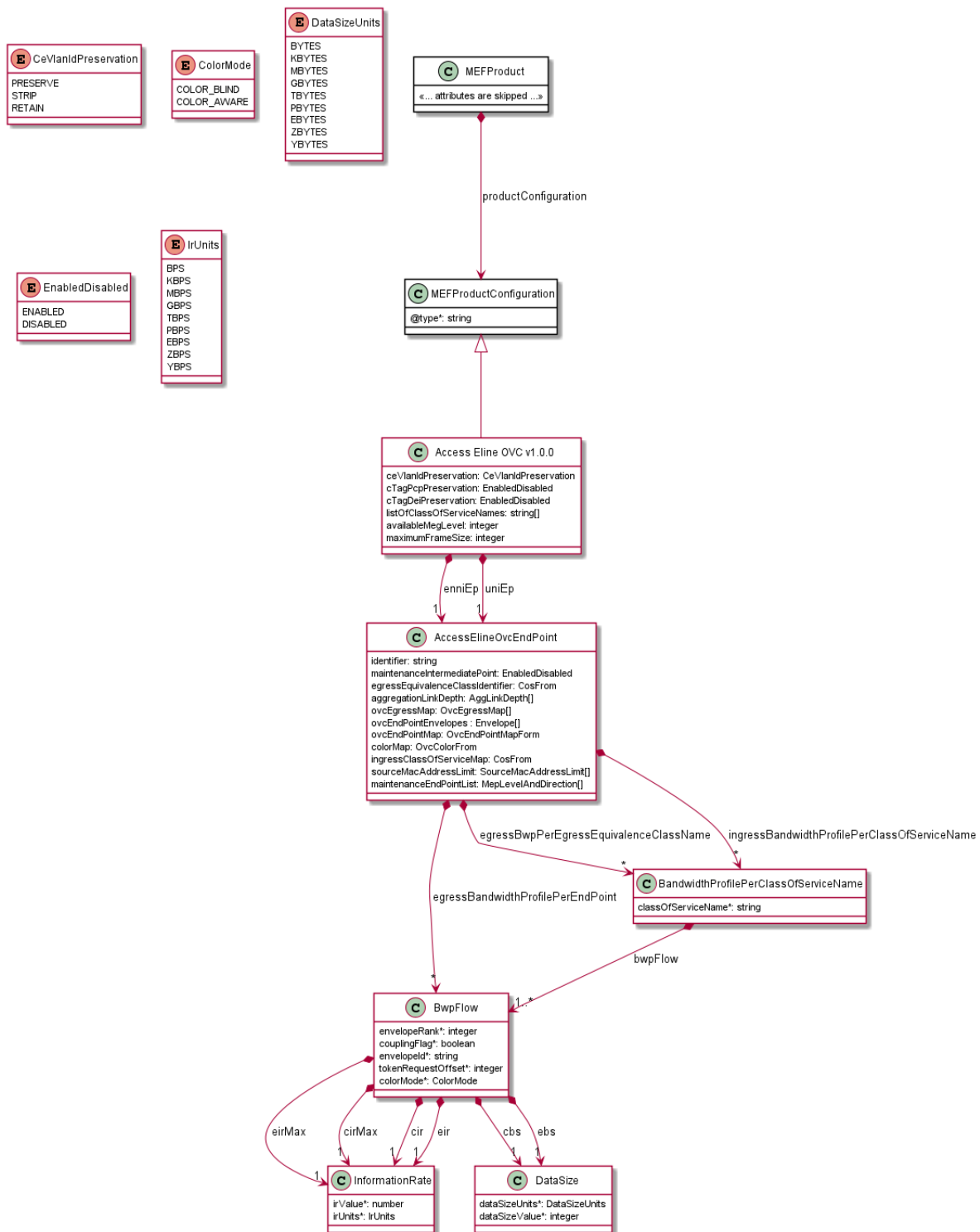


Figure 6 – A Simplified View on the Access E-Line Product Specification Data Model

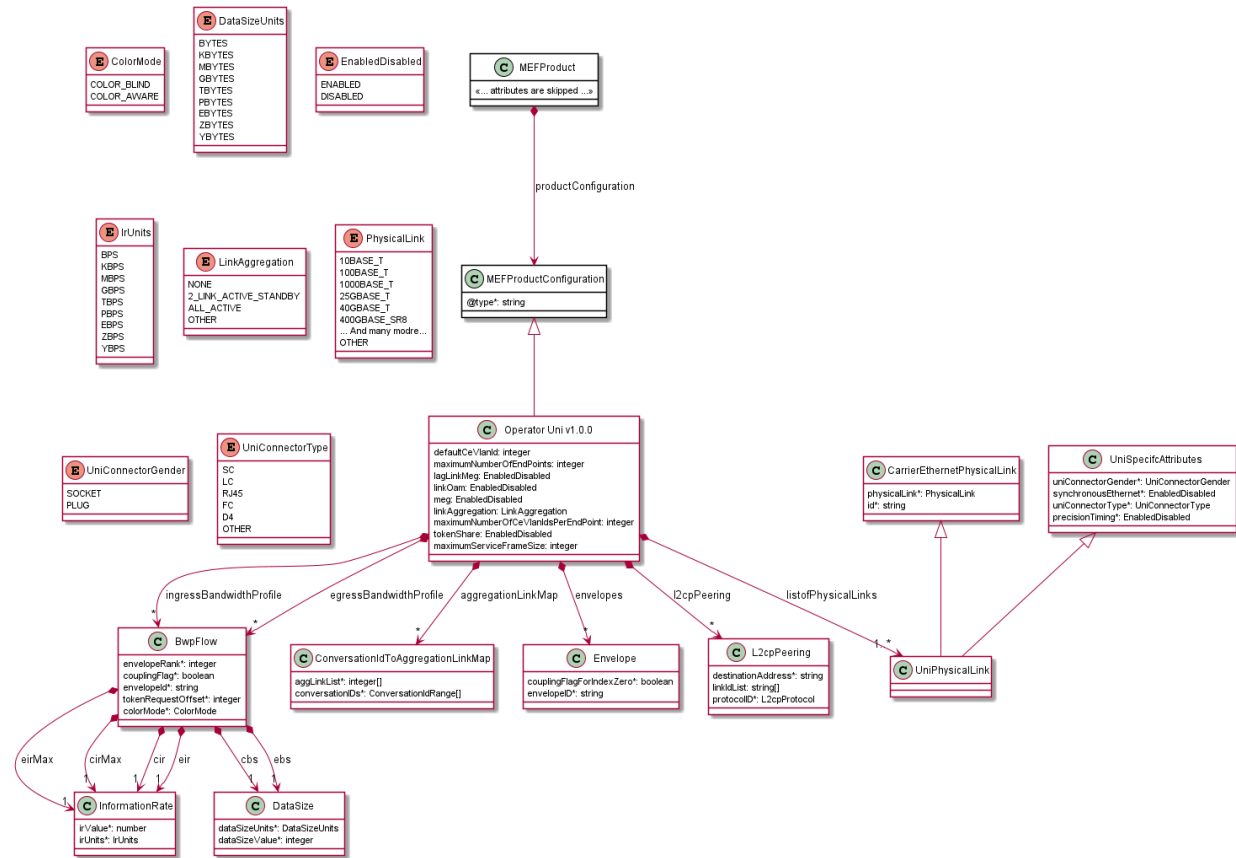


Figure 7 – A Simplified View on the UNI Product Specification Data Model

6.6 Model Structural Validation

The structure of the HTTP payloads exchanged via Product Inventory API endpoints is defined using:

- OpenAPI version 3.0 [13] for the product-agnostic part of the payload
- JSON Schema (draft 7) [1] for the product-specific part of the payload

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

[R10] A product specification **MAY** define additional consistency rules and requirements that **MUST** be respected by implementations. These are defined for:

- required relation type, multiplicity to other products in the Seller’s product inventory
- related contact information roles
- relations to places (**Sites**) and their roles

6.7 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 being worked on by a separate MEF Project (MEF 128).

7 API Interaction and Flows

This section provides a detailed insight into the API functionality, use cases, and flows. It starts with Table 3 presenting a list and short description of all business use cases then presents the variants of end-to-end interaction flows, and in following subsections describes the API usage flow and examples for each of the use cases.

Use Case #	Use Case Name	Use Case Description
1	Retrieve Product List	The Buyer requests a list of Products from the Seller based on filter criteria.
2	Retrieve Product by Identifier	The Buyer retrieves the details associated with the Product that matches the specified Identifier.

Table 3 – Use Case Descriptions

7.1 Use Case 1: Retrieve Product List

The Buyer can retrieve a list of **Products** by using a **GET /product** operation with the desired filtering criteria. The attributes that are available to be used are:

MEF 80 [11] specifies the possible filtering criteria, in MEF 80 O17:

- **state**
- **productSpecificationId**
- **productOfferingId**
- **externalId**
- **geographicalSiteId**
- **relatedProductId**
- **billingAccountId**
- **productOrderId**
- **startDate.gt**
- **endDate.lt**
- **lastUpdateDate.gt**
- **lastUpdateDate.lt**

The flow is a simple request – response pattern, as presented in Figure 8:

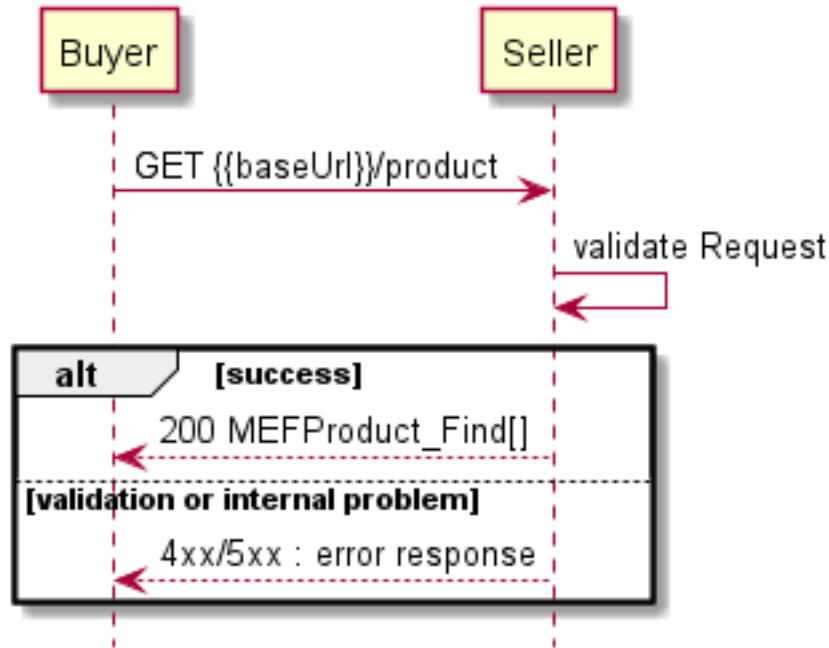


Figure 8 – Use Case 1: Retrieve Product List Flow

The part of the model taking part in this use case is presented in Figure 9 below.

<https://serverRoot/mefApi/sonata/productInventory/v7/product?status=pendingTerminate>

The example above shows a Buyer's request to get all **Products** that are in the **pendingTerminate** status. The correct response (HTTP code **200**) in the response body contains a list of **MEFProduct_Find** objects matching the criteria. To get more details (e.g. the item level information), the Buyer has to query a specific **Product** by **id**.

The snippet below shows an example of a response with 1 product matched:

```

[
  {
    "id": "01494079-6c79-4a25-83f7-48284196d44d",
    "href": "{{baseUrl}}/product/01494079-6c79-4a25-83f7-48284196d44d",
    "status": "pendingTerminate",
    "externalId": "BuyerProduct-001",
    "lastUpdateDate": "2021-06-01T08:55:54.155Z",
    "startDate": "2021-05-01T08:55:54.155Z",
    "billingAccount": {
      "id": "00000000-1111-0000-0000-000000000001"
    },
    "productOffering": {
      "id": "00000000-5555-0000-0000-000000000001"
    },
    "productOrderItem": [
      {
        "productOrderItemId": "item-001",

```

```
"productOrderHref": "{baseUri}/productOrder/00000000-1111-2222-3333-000000000123",
"productOrderId": "00000000-1111-2222-3333-000000000123"
}
],
"productRelationship": [
{
  "id": "00000000-6666-0000-0000-000000000001",
  "relationshipType": "ENNI_REFERENCE"
}
]
}
]
```

- [R11] The Buyer **MUST** be able to perform Buyer Inventory Query without any filter criteria (MEF 81 R7 [11]).
- [O1] The Seller **MAY** place a limit on the length of the list returned (MEF 81 O2 [11]).
- [O2] If the Buyer Inventory Query exceeds that length, the Seller **MAY** return an error ([Error422](#)) indicating that the list is too long (MEF 81 O3 [11]).

The Buyer may also ask for pagination with the use of the `offset` and `limit` parameters. The filtering and pagination attributes must be specified in URI query format [3]. Section 8.1.2 provides details about the implementation of pagination mechanism.

- [R12] In case no items matching the criteria are found, the Seller **MUST** return a valid response with an empty list.
- [R13] The Seller **MUST** put the following attributes (if set) into the `MEFProduct_Find` object in the response (MEF 81 R8 [11]):
- `id`
 - `status`
 - `externalId`
 - `lastUpdateDate`
 - `startDate`
 - `billingAccount`
 - `productOffering`
 - `productOrderItem`
 - `productRelationship`
 - `productSpecification`
 - `relatestSite`

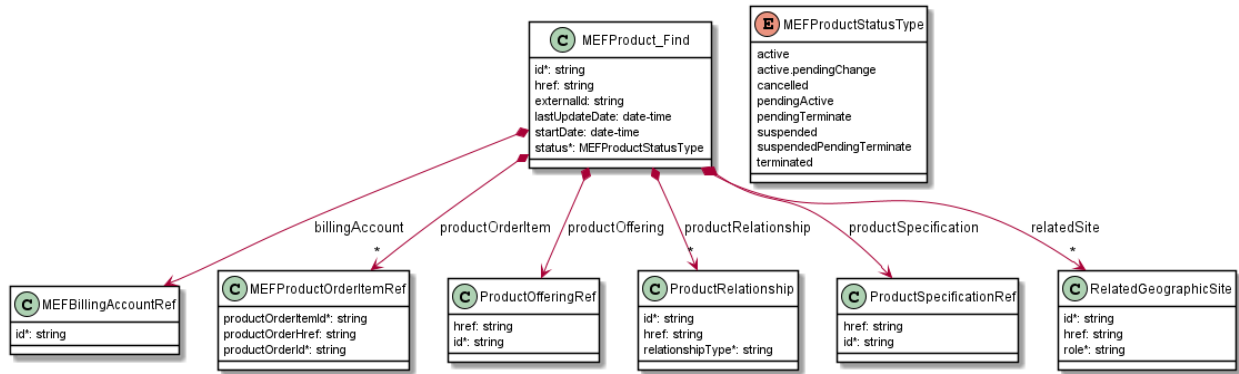


Figure 9 – Use Case 1: Retrieve Product List Model

7.2 Use Case 2: Retrieve Product by Identifier

To get detailed up to date information about the Product, the Buyer sends a Retrieve Product by Identifier request using a **GET** `/product/{id}` operation.

The flow is a simple request – response pattern, as presented in Figure 10:

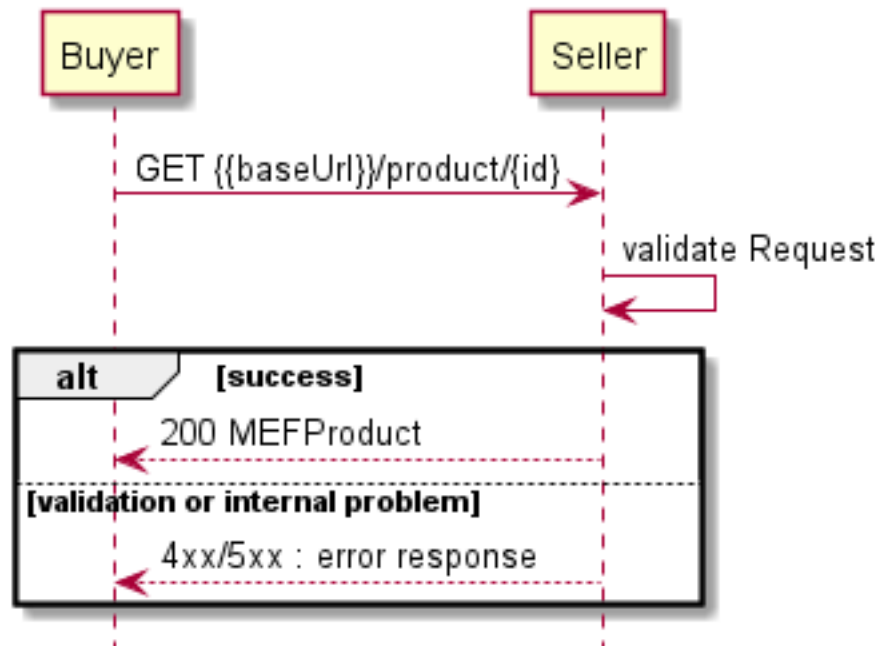


Figure 10 – Use Case 2: Retrieve Product by Identifier Flow

The part of the model taking part in this use case is presented in Figure 11.

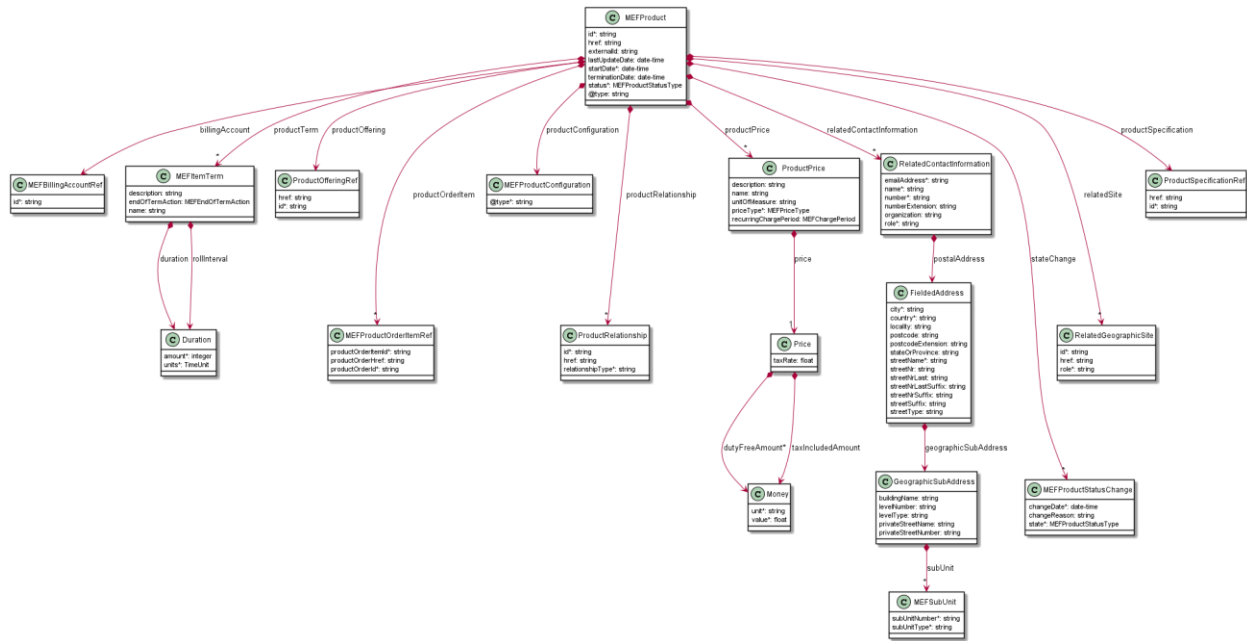


Figure 11 – Use Case 2: Retrieve Product Model

Example request and response:

GET /mefApi/sonata/productInventory/v7/product/01494079-6c79-4a25-83f7-48284196d44d

```
{
  "id": "01494079-6c79-4a25-83f7-48284196d44d",
  "href": "{{baseUrl}}/product/01494079-6c79-4a25-83f7-48284196d44d",
  "externalId": "BuyerProduct-001",
  "lastUpdateDate": "2021-06-01T08:55:54.155Z",
  "startDate": "2021-05-01T08:55:54.155Z",
  "status": "pendingTerminate",
  "@type": "MEFProduct",
  "productConfiguration": {
    "@type": "urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:inventory",
    "enniEp": {
      "ingressBandwidthProfilePerClassOfServiceName": [
        {
          "classOfServiceName": "silver",
          "bwpFlow": [
            {
              "envelopeRank": 1,
              "couplingFlag": false,
              "envelopeName": "defaultENNI",
              "tokenRequestedOffset": 0,
              "colorMode": "COLOR_BLIND",
              "cir": {
                "irValue": 20,
                "irUnits": "MBPS"
              }
            }
          ]
        }
      ]
    }
  }
}
```



```
"cbs": {
  "dataSizeValue": 50,
  "dataSizeUnits": "KBYTES"
},
"eir": {
  "irValue": 0,
  "irUnits": "BPS"
},
"ebs": {
  "dataSizeValue": 0,
  "dataSizeUnits": "BYTES"
},
"cirMax": {
  "irValue": 20,
  "irUnits": "MBPS"
},
"eirMax": {
  "irValue": 0,
  "irUnits": "BPS"
}
}
]
}
]
},
"maximumFrameSize": 1522,
"uniEp": {
  "ingressBandwidthProfilePerClassOfServiceName": [
    {
      "classOfServiceName": "silver",
      "bwpFlow": [
        {
          "envelopeRank": 1,
          "couplingFlag": false,
          "envelopeName": "defaultUNI",
          "tokenRequestedOffset": 0,
          "colorMode": "COLOR_BLIND",
          "cir": {
            "irValue": 20,
            "irUnits": "MBPS"
          },
          "cbs": {
            "dataSizeValue": 50,
            "dataSizeUnits": "KBYTES"
          },
          "eir": {
            "irValue": 0,
            "irUnits": "BPS"
          }
        }
      ]
    }
  ]
}
```

```
    },
    "ebs": {
      "dataSizeValue": 0,
      "dataSizeUnits": "BYTES"
    },
    "cirMax": {
      "irValue": 20,
      "irUnits": "MBPS"
    },
    "eirMax": {
      "irValue": 0,
      "irUnits": "BPS"
    }
  }
]
}
}
},
"billingAccount": {
  "id": "00000000-1111-0000-0000-000000000001"
},
"productOffering": {
  "id": "00000000-5555-0000-0000-000000000001"
},
"productOrderItem": [
  {
    "productOrderItemId": "item-001",
    "productOrderHref": "{baseUrl}/productOrder/00000000-1111-2222-3333-000000000123",
    "productOrderId": "00000000-1111-2222-3333-000000000123"
  }
],
"price": {
  "taxRate": 8,
  "dutyFreeAmount": {
    "unit": "USD",
    "value": 50
  },
  "taxIncludedAmount": {
    "unit": "USD",
    "value": 54
  }
},
"productRelationship": [
  {
    "id": "00000000-6666-0000-0000-000000000001",
    "relationshipType": "ENNI_REFERENCE"
  }
]
```

```
],
"productTerm": [
  {
    "duration": {
      "amount": 12,
      "units": "calendarMonths"
    },
    "endOfTermAction": "autoRenew",
    "name": "Yearly Subscription"
  }
],
"relatedContactInformation": [
  {
    "emailAddress": "Seller.AssuranceTechnicalContact@example.com",
    "name": "Seller Assurance Technical Contact",
    "number": "+98-765-432-10",
    "role": "sellerAssuranceTechnicalContact "
  },
  {
    "emailAddress": "Seller.CommercialContact@example.com",
    "name": "Seller Commercial Contact",
    "number": "+98-765-432-11",
    "role": "sellerCommercialContact"
  },
  {
    "emailAddress": "Seller.SLAManagementContact@example.com",
    "name": "Seller SLA Management Contact",
    "number": "+98-765-432-12",
    "role": "sellerSlaManagementContact"
  },
  {
    "emailAddress": "Buyer.AssuranceTechnicalContact@example.com",
    "name": "Buyer Assurance Technical Contact",
    "number": "+12-345-678-90",
    "role": "buyerAssuranceTechnicalContact "
  },
  {
    "emailAddress": "Buyer.CommercialContact@example.com",
    "name": "Buyer Commercial Contact",
    "number": "+12-345-678-91",
    "role": "buyerCommercialContact"
  },
  {
    "emailAddress": "Buyer.SLAManagementContact@example.com",
    "name": "Buyer SLA Management Contact",
    "number": "+12-345-678-92",
    "role": "buyerSlaManagementContact"
  }
]
```

```

    ],
    "statusChange": [
      {
        "changeDate": "2021-05-01T10:01:14.571Z",
        "status": "pendingActive"
      },
      {
        "changeDate": "2021-05-02T10:01:14.571Z",
        "status": "active"
      },
      {
        "changeDate": "2021-06-01T10:01:14.571Z",
        "status": "pendingTerminate"
      }
    ]
  }
}

```

Figure 12 below presents the Product's lifecycle.

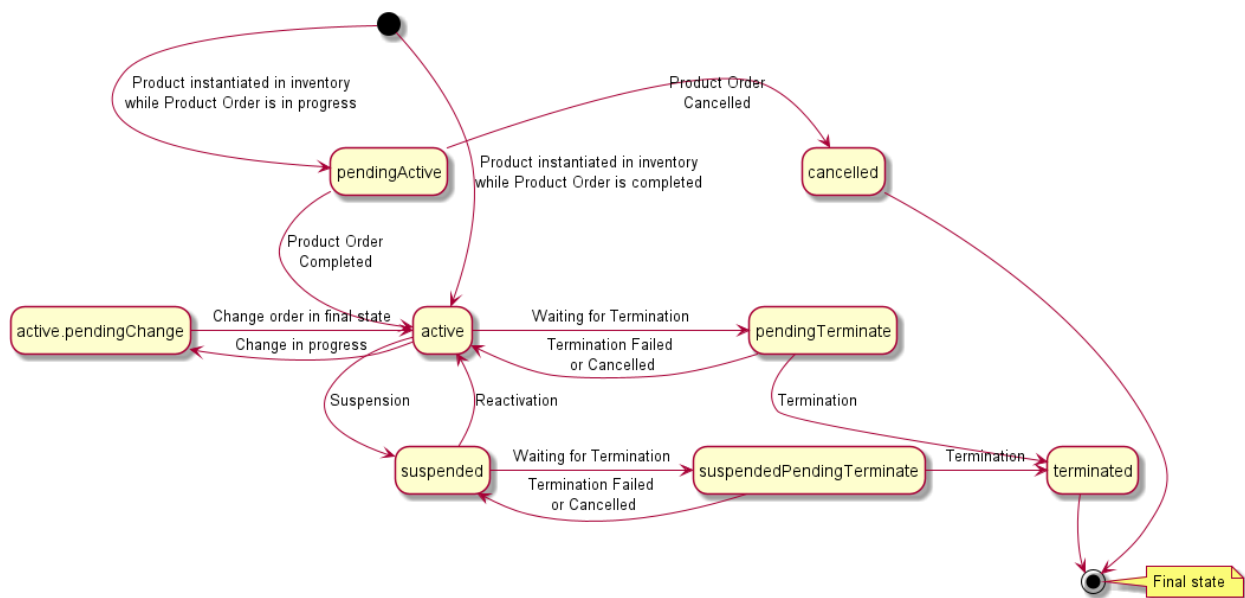


Figure 12 – Product State Machine

A detailed description of each state can be found in the table below.

Name	MEF 81 Name	Description
active	ACTIVE	The Product Order has been successfully completed and the Product Order and associated Product Order Items are in the Inventory.
active.pendingChange	ACTIVE_PENDING_CHANGE	The Product is active and has a Product Order to change the Product that is in progress. The status returns to active when the order is completed or if the Product Order is cancelled.

active.pendingChange	ACTIVE_PENDING_TERMINATE	The Product is active and has a disconnect Order submitted by the Buyer that is in progress. The status changes to terminated if the disconnect is successful. The status returns to active if the Product Order fails to be completed or the Product Order is cancelled.
cancelled	CANCELLED	The Product is cancelled when the Product Order has moved to the cancelled state.
pendingActive	PENDING	The Product Order has moved to the acknowledged state as defined in MEF 57.2 and the Product ID for one or more Product Items have been passed from the Seller to the Buyer. The Product Order is not completed.
suspended	SUSPENDED	A Product has been successfully suspended. Products are placed into suspended state for some reason (e.g. nonpayment of bill) and removed from suspended state for some reason (e.g. after payment).
suspendedPendingTerminate	SUSPENDED_PENDING_TERMINATE	The Product is in the process of being terminated by the Seller for some reason (e.g. non-payment). The status changes to terminated if the termination is successful. The status returns to suspended if the termination is not successful or cancelled.
terminated	TERMINATED	The Product has been successfully terminated via a disconnect Product order initiated by the Buyer or by the Seller for some reason (e.g. non-payment).

Table 4 – Product States

Products that are terminated might be removed from the Seller's inventory system or shown in the `terminated` state at the Seller's discretion.

[R14] The Seller **MUST** provide the following contact information (MEF 81 R11 [11]):

Contact Role	role Value	Description
Assurance Technical Contact	buyerAssuranceTechnicalContact, sellerAssuranceTechnicalContact	Operational contact such as Network Operations Center (NOC) for each party.
Commercial Contact	buyerCommercialContact, sellerCommercialContact	Contact for commercial issues like billing, contract extensions, etc. for each party.
SLA Management Contact	buyerSlaManagementContact, sellerSlaManagementContact	Contact for SLA related issues, lifecycle reports, etc. for each party.

Table 5 – Required Related Contact Information **role**

Note: The method used to update these contacts in the Seller's Inventory system is assumed to be agreed to between the Buyer and the Seller and is outside the scope of this document.

There is no step of Buyer's approval before moving a Product to **active** status. This might be part of a bilateral agreement or procedure that takes place outside of the Product Inventory API.

Additions and changes to Products in the Product Inventory can be performed on with use of Product Orders and the Product Order Management API, or by the request of the Seller.

8 API Details

8.1 API Patterns

8.1.1 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 depicted and described below.

Implementations can use HTTP error codes not specified in this standard in compliance with rules defined in RFC 7231 [4]. In such a case the error message body structure might be aligned with the **Error**.

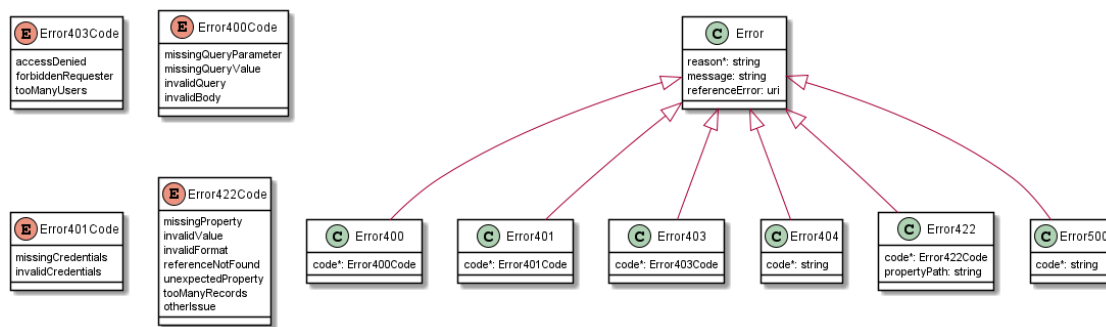


Figure 13 – Data Model Types to Represent an Erroneous Response

8.1.1.1 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
message	string	Text that provides more details and corrective actions related to the error. This can be shown to a client user.
reason*	string	Text that explains the reason for the error. This can be shown to a client user.
referenceError	string	URL pointing to documentation describing the error.

Table 6 – Type Error

8.1.1.2 Type Error400

Description: Bad Request. (<https://datatracker.ietf.org/doc/html/rfc7231#section-6.5.1>)

Inherits from:

- **Error**

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

Table 7 – Type Error400

8.1.1.3 Type Error401

Description: Unauthorized. (<https://datatracker.ietf.org/doc/html/rfc7235#section-3.1>)

Inherits from:

- [Error](#)

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

Table 8 – Type Error401

8.1.1.4 Type Error403

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

Inherits from:

- [Error](#)

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

Table 9 – Type Error403

8.1.1.5 Type Error404

Description: Resource for the requested path not found. (<https://datatracker.ietf.org/doc/html/rfc7231#section-6.5.4>)

Inherits from:

- [Error](#)

Name	Type	Description
<code>code*</code>	string	The following error code: - <code>notFound</code> : A current representation for the target resource not found.

Table 10 – Type Error404

8.1.1.6 Type Error422

The response for HTTP status `422` is a list of elements that are structured using the `Error422` data type. Each list item describes a business validation problem. This type introduces the `propertyPath` attribute which points to the erroneous property of the request, so that the Buyer may fix it easier. It is highly recommended that this property should be used, yet remains optional because it might be hard to implement.

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

Inherits from:

- [Error](#)

Name	Type	Description
<code>code*</code>	string	One of the following error codes: - <code>missingProperty</code> : The property the Seller has expected is not present in the payload. - <code>invalidValue</code> : The property has an incorrect value. - <code>invalidFormat</code> : The property value does not comply with the expected value format. - <code>referenceNotFound</code> : The object referenced by the property cannot be identified in the Seller system. - <code>unexpectedProperty</code> : Additional property, not expected by the Seller has been provided. - <code>tooManyRecords</code> : the number of records to be provided in the response exceeds the Seller's threshold. - <code>otherIssue</code> : Other problem was identified (detailed information provided in a reason).
<code>propertyPath</code>	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 [1].

Table 11 – Type Error422

8.1.1.7 Type Error500

Description: Internal Server Error. (<https://datatracker.ietf.org/doc/html/rfc7231#section-6.6.1>)

Inherits from:

- [Error](#)

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

Table 12 – Type Error500

8.1.2 Response Pagination

A response to retrieve a list of results (e.g. `GET /product`) can be paginated. The Buyer can specify following query attributes related to pagination:

- `limit` – number of expected list items
- `offset` – offset of the first element in the result list

The Seller returns a list of elements that comply with the requested `limit`. If the requested `limit` is higher than the supported list size the smaller list result is returned. In that case, the size of the result is returned in the header attribute `X-Result-Count`. The Seller can indicate that there are additional results available using:

- `X-Total-Count` header attribute with the total number of available results
- `X-Pagination-Throttled` header set to `true`

[R15] Seller **MUST** use either **X-Total-Count** or **X-Pagination-Throttled** to indicate that the page was truncated and additional results are available.

8.2 Management API Data Model

Figure 14 presents the whole Product Inventory data model. The data types, requirements related to them, and mapping to MEF 81 [11] specifications are discussed later in this section.

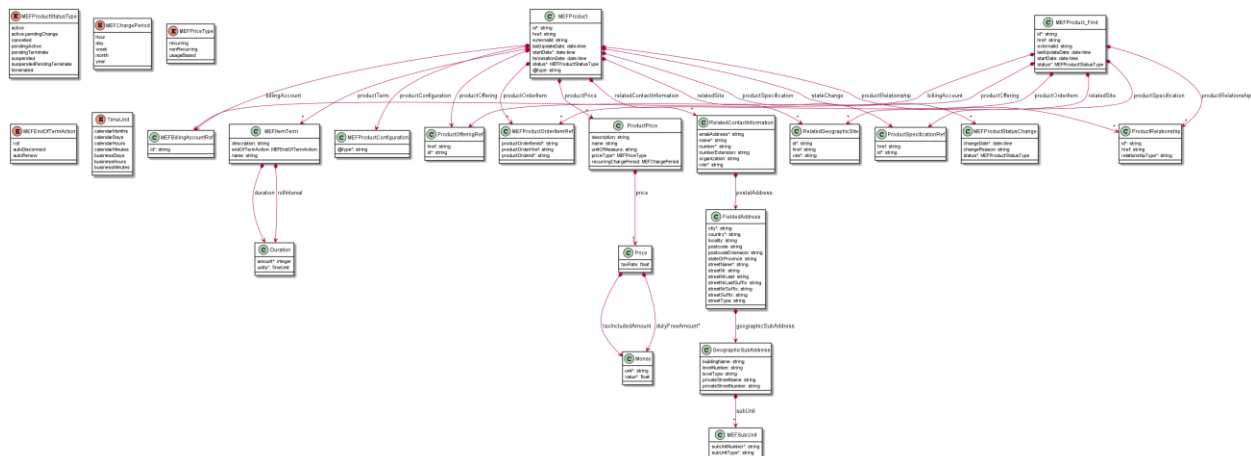


Figure 14 – Product Inventory Data Model



8.2.1 Product

8.2.1.1 Type MEFProduct

Description: A product is realized as one or more service(s) and/or resource(s).

Name	Type	Description	MEF 81
id*	string	Unique identifier of the product.	Seller Product Identifier
href	string	Reference of the product.	Not represented in MEF 81
externalId	string	Buyer identifier of the product.	Buyer Product Identifier
lastUpdateDate	date-time	Latest date when the product has been updated.	Last Updated Date
startDate*	date-time	This is the date from which the product starts. MEF: Start date is when the product is active for the first time (when the install in the product order has been processed).	Initial Order Completion Date
terminationDate	date-time	This is the date when the product was terminated. MEF: Termination date (commercial) is when the product has been terminated (when the disconnect in the product order has been processed).	Termination Date
productConfiguration	MEFProductConfiguration	MEFProductConfiguration is used to specify the MEF specific product payload.	Product
billingAccount	MEFBillingAccount	The Billing Account associated with the Product.	Billing Account Identifier
productOffering	ProductOfferingRef	A particular Product Offering defines the technical and commercial attributes and behaviors of a Product.	Product Offering ID
productOrderItem	MEFProductOrderItemRef[]	The Product Order Item of the associated Product order that resulted in the creation of this Product.	Product Order Identifier, Product Order Item Identifier
productPrice	ProductPrice[]	A list of Prices associated with the Product.	Product Price
productRelationship	ProductRelationship[]	A list of references to existing products that are related to the Product.	Product Relationship
productSpecification	ProductSpecificationRef	A reference to a Product Specification of the Product.	Product Specification ID
productTerm	MEFItemTerm[]	Term of the Product.	Product Order Item Term, Product Order Item Term End Date
relatedContactInformation	RelatedContactInformation[]	Party playing a role for this Product.	Buyer Assurance Technical Contact, Buyer Commercial Contact, Buyer SLA Management Contact, Seller Assurance Technical Contact, Seller Commercial Contact, Seller SLA Management Contact
relatedSite	RelatedGeographicSite[]	Reference to a Site where the Product is provided.	Service Site Identifier

statusChange	MEFProductStatusChange[]	State change for the Product.	Not represented in MEF 81
status*	MEFProductStatusType	The lifecycle status of the product.	Status
@type	string	When sub-classing, this defines the sub-class entity name.	Not represented in MEF 81

Table 13 – Type MEFProduct

8.2.1.2 Type MEFProduct_Find

Description: Class used to provide product overview retrieved in GET (by list) operation.

Name	Type	Description	MEF 81
id*	string	Unique identifier of the product.	Seller Product Identifier
href	string	Reference of the product.	Not represented in MEF 81
externalId	string	This identifier is optionally provided during the product ordering and stored for informative purpose in the Seller inventory.	Buyer Product Identifier
lastUpdateDate	date-time	Latest date when the product has been updated.	Last Updated Date
startDate	date-time	The date from which the product starts.	Initial Order Completion Date
billingAccount	MEFBillingAccountRef	The Billing Account associated with the Product.	Billing Account Identifier
productOffering	ProductOfferingRef	A particular Product Offering defines the technical and commercial attributes and behaviors of a Product.	Product Offering ID
productOrderItem	MEFProductOrderItemRef[]	The Product Order Item of the associated Product order that resulted in the creation of this Product.	Product Order Identifier, Product Order Item Identifier
productRelationship	ProductRelationship[]	A list of references to existing products that are related to the Product.	Product Relationship
productSpecification	ProductSpecificationRef	A reference to a Product Specification of the Product.	Product Specification ID
relatedSite	RelatedGeographicSite[]	Reference to a Site where the Product is provided.	Service Site Identifier
status*	MEFProductStatusType	The lifecycle status of the product.	Status

Table 14 – Type MEFProduct_Find

8.2.1.3 enum MEFProductStatusType

Description: Possible values for the status of a MEF product.:

Value	MEF 81
active	ACTIVE
active.pendingChange	ACTIVE_PENDING_CHANGE
pendingTerminate	ACTIVE_PENDING_TERMINATE
cancelled	CANCELLED
pendingActive	PENDING
suspended	SUSPENDED

suspendedPendingTerminate	SUSPENDED_PENDING_TERMINATE
terminated	TERMINATED

Table 15 – enum MEFProductStatusType

8.2.1.4 Type MEFProductStatusChange

Description: Holds the reached state, reasons, and associated date the Product Order status changed, populated by the Seller.

Name	Type	Description	MEF 81
changeDate*	date-time	The date and time the Status changed.	Not represented in MEF 81
changeReason	string	The reason why the Status changed.	Not represented in MEF 81
status*	MEFProductStatusType	Status of the product.	Not represented in MEF 81

Table 16 – Type MEFProductStatusChange

8.2.1.5 Type ProductPrice

Description: An amount, usually of money, that represents the actual price paid by a Customer for a purchase, a rent, or a lease of a Product. The price is valid for a defined period of time.

Name	Type	Description	MEF 81
description	string	A narrative that explains in detail the semantics of this product price.	Price Description
name	string	A short descriptive name such as ‘Subscription price’.	Price Name
unitOfMeasure	string	Unit of Measure if price depending on it (Gb, SMS volume, etc.).	Not represented in MEF 81
price*	Price	Value of the Price.	Price
priceType*	MEFPriceType	A category that describes the price, such as recurring, nonRecurring, usage-Based	Price Type
recurringChargePeriod	MEFChargePeriod	Charge period for recurring charge.	Price Recurring Charge Period

Table 17 – Type ProductPrice

8.2.2 Common

Types described in this subsection are shared among two or more Cantata and Sonata APIs.

8.2.2.1 Type Duration

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

Name	Type	Description	MEF 81
amount*	integer	Duration (number of seconds, minutes, hours, etc.)	Duration Value
units*	TimeUnit	Time unit type.	Duration Unit

Table 18 – Type Duration

8.2.2.2 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. Reference: MEF 79 Section 8.9.2 [9].

Name	Type	Description	MEF 81
city*	string	The city that the address is in.	Not represented in MEF 81
country*	string	The country that the address is in.	Not represented in MEF 81
geographicSubAddress	Geographic SubAddress	Additional fields used to specify an address, as detailed as possible.	Not represented in MEF 81
locality	string	The locality that the address is in.	Not represented in MEF 81
postcode	string	A descriptor for a postal delivery area, used to speed and simplify the delivery of mail (also known as zip code).	Not represented in MEF 81
postcodeExtension	string	An extension of a postal code. E.g. the part following the dash in a US urban property address.	Not represented in MEF 81
stateOrProvince	string	The State or Province that the address is in.	Not represented in MEF 81
streetName*	string	Name of the street or other street type.	Not represented in MEF 81
streetNr	string	Number identifying a specific property on a public street. It may be combined with streetNrLast for ranged addresses.	Not represented in MEF 81
streetNrLast	string	Last number in a range of street numbers allocated to a property.	Not represented in MEF 81
streetNrLastSuffix	string	Last street number suffix for a ranged address.	Not represented in MEF 81
streetNrSuffix	string	The first street number suffix.	Not represented in MEF 81
streetSuffix	string	A modifier denoting a relative direction.	Not represented in MEF 81
streetType	string	The type of street (e.g., alley, avenue, boulevard, brae, crescent, drive, highway, lane, terrace, parade, place, tarn, way, wharf).	Not represented in MEF 81

Table 19 – Type FieldedAddress

8.2.2.3 Type GeographicSubAddress

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

Name	Type	Description	MEF 81
buildingName	string	Allows for identification of places that require building name as part of addressing information.	Not represented in MEF 81
levelNumber	string	Used where a level type may be repeated e.g. BASEMENT 1, BASEMENT 2.	Not represented in MEF 81
levelType	string	Describes level types within a building.	Not represented in MEF 81
privateStreetName	string	Private streets internal to a property (e.g. a university) may have internal names that are not recorded by the land title office.	Not represented in MEF 81
privateStreetNumber	string	Private streets numbers internal to a private street.	Not represented in MEF 81
subUnit	MEFSubUnit[]	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 81

Table 20 – Type GeographicSubAddress

8.2.2.4 Type MEFSubUnit

Description: Allows for subunit identification.

Name	Type	Description	MEF 81
subUnitNumber*	string	The discriminator used for the subunit, often just a simple number but may also be a range.	Not represented in MEF 81
subUnitType*	string	The type of subunit e.g. BERTH, FLAT, PIER, SUITE, SHOP, TOWER, UNIT, WHARF.	Not represented in MEF 81

Table 21 – Type MEFSubUnit

8.2.2.5 Type MEFBillingAccountRef

Description: References the billing arrangement that a Buyer has with a Seller that provides products to the customer.

Name	Type	Description	MEF 81
id*	string	Identifies the buyer's billing account to which the recurring and non-recurring charges for this order or order item will be billed. Required if the Buyer has more than one Billing Account with the Seller and for all new Product Orders.	Billing Account Identifier

Table 22 – Type MEFBillingAccountRef

8.2.2.6 enum MEFChargePeriod

Description: Used for a recurring charge to indicate a period.

Value	MEF 81
hour	HOUR
day	DAY
week	WEEK
month	MONTH
year	YEAR

Table 23 – enum MEFChargePeriod

8.2.2.7 Type MEFItemTerm

Description: The term of the Item.

Name	Type	Description	MEF 81
description	string	Description of the term.	Not represented in MEF 81
duration*	Duration	Duration of the term.	Not represented in MEF 81
endOfTermAction*	MEFEndOfTermAction	The action that needs to be taken by the Seller once the term expires.	Not represented in MEF 81
name	string	Name of the term.	Not represented in MEF 81
rollInterval	Duration	The recurring period that the Buyer is willing to pay to the end of upon disconnecting the Product after the original term has expired.	Not represented in MEF 81

Table 24 – Type MEFItemTerm

8.2.2.8 enum *MEFEndOfTermAction*

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

Value	MEF 81
roll	ROLL
autoDisconnect	AUTO_DISCONNECT
value	AUTO_RENEW

Table 25 – enum *MEFEndOfTermAction*

8.2.2.9 enum *MEFPriceType*

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

Value	MEF 81
recurring	RECURRING
nonRecurring	NON_RECURRING
usageBased	Not represented in MEF 81

Table 26 – enum *MEFPriceType*

8.2.2.10 Type *MEFProductConfiguration*

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

Name	Type	Description	MEF 81
<i>@type*</i>	string	The name of the type, defined in the JSON schema specified above, for the product that is the subject of the POQ Request. The named type must be a subclass of <i>MEFProductConfiguration</i> .	Not represented in MEF 81

Table 27 – Type *MEFProductConfiguration*

8.2.2.11 Type *MEFProductOrderItemRef*

Description: A reference to a *ProductOrder* item.

Name	Type	Description	MEF 81
<i>productOrderItemId*</i>	string	ID of an Item within the <i>ProductOrder</i> .	Product Order Item Identifier
<i>productOrderHref</i>	string	Reference of the related <i>ProductOrder</i> .	Not represented in MEF 81
<i>productOrderId*</i>	string	Unique identifier of a <i>ProductOrder</i> .	Product Order Identifier

Table 28 – Type *MEFProductOrderItemRef*

8.2.2.12 Type *Price*

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

Name	Type	Description	MEF 81
taxRate	float	Price Tax Rate. Unit: [%]. E.g. value 16 stands for 16% tax.	Price Tax Rate
dutyFreeAmount*	Money	All taxes excluded amount (expressed in the given currency).	Price Duty Free Amount
taxIncludedAmount	Money	All taxes included amount (expressed in the given currency).	Price Tax Included Amount

Table 29 – Type Price

8.2.2.13 Type Money

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

Name	Type	Description	MEF 80
unit	string	Currency (ISO 4217 [6] uses 3 letters to define the currency)	Currency
value	float	A positive floating-point number	Value

Table 30 – Type Money

8.2.2.14 Type ProductOfferingRef

Description: A reference to a Product Offering offered by the Seller to the Buyer. A Product Offering contains the commercial and technical details of a Product sold by a particular Seller. A Product Offering defines all of the commercial terms and, through association with a particular Product Specification, defines all the technical attributes and behaviors of the Product. A Product Offering may constrain the allowable set of configurable technical attributes and/or behaviors specified in the associated Product Specification.

Name	Type	Description	MEF 81
id*	string	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 ID
href	string	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.	Not represented in MEF 81

Table 31 – Type ProductOfferingRef

8.2.2.15 Type ProductRelationship

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

Name	Type	Description	MEF 81
id*	string	Unique identifier.	Seller Product Identifier
href	string	Hyperlink of the referenced product.	Not represented in MEF 81

relationshipType	string	Specifies the type (nature) of the relationship to the related Product. The nature of required relationships vary 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.	Relationship Nature
-------------------------	--------	--	---------------------

Table 32 – Type ProductRelationship

8.2.2.16 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	Description	MEF 81
href	string	Hyperlink to a Product Specification in Seller’s 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.	Not represented in MEF 81
id*	string	Unique identifier of the product specification.	Product Specification ID

Table 33 – Type ProductSpecificationRef

8.2.2.17 Type RelatedContactInformation

Description: Contact information of an individual or organization playing a role for this Entity. The rule for mapping a represented attribute value to a **role** is to use the *lowerCamelCase* pattern e.g.

- Buyer Order Item Contact: **role=buyerOrderItemContact**
- Buyer Implementation Contact: **role=buyerImplementationContact**
- Buyer Technical Contact: **role=buyerTechnicalContact**

Name	Type	Description	MEF 81
emailAddress*	string	Email address.	Contact Email Address
name*	string	Name of the contact.	Contact Name
number*	string	Phone number.	Contact Phone Number
numberExtension	string	Phone number extension.	Contact Phone Number Extension
organization	string	The organization or company that the contact belongs to.	Not represented in MEF 81
role*	string	A role of the particular contact in a given context.	Contact Role
postalAddress	FieldedAddress	Identifies the postal address of the person or office to be contacted.	Not represented in MEF 81

Table 34 – Type RelatedContactInformation

The **role** attribute is used to provide a reason the particular party information is used. It can result from MEF 57.2 requirements (e.g. Seller Contact Information) or from the Product Specification requirements.

The rule for mapping a represented attribute value to a **role** is to use the *lowerCamelCase* pattern e.g.

- Seller Contact: **role=sellerContact**
- Buyer Contact Information: **role=buyerContactInformation**

8.2.2.18 Type RelatedGeographicSite

Description: A Geographic Site and an associated role as installation address, delivery address, etc.

Name	Type	Description	MEF 81
id*	string	Unique identifier of the geographic site.	Service Site Identifier
href	string	Hyperlink to the referenced geographic site.	Not represented in MEF 81
role*	string	Role of the geographic site, such as: [home delivery], [shop retrieval]) MEF: The role that the Site plays, e.g. Billing Address, UNI Site, or ENNI Site.	Not represented in MEF 81

Table 35 – Type ProductSpecificationRef

8.2.2.19 enum TimeUnit

Description: Represents a unit of time.

Value	MEF 81
calendarMonths	CALENDAR_MONTHS
calendarDays	CALENDAR_DAYS
calendarHours	CALENDAR_HOURS
calendarMinutes	CALENDAR_MINUTES
businessDays	BUSINESS_DAYS
businessHours	BUSINESS_HOURS
businessMinutes	BUSINESS_MINUTES

Table 36 – Type TimeUnit

- [R16]** The clarification of what Business days, hours, and minutes mean **MUST** be done between the Buyer and the Seller during the onboarding process.

9 References

- [1] IETF JSON Schema draft 7, *JSON Schema: A Media Type for Describing JSON Documents* and associated documents, by Austin Wright and Henry Andrews, March 2018. Copyright © 2018 IETF Trust and the persons identified as the document authors. All rights reserved.
- [2] IETF RFC 2119, *Key words for use in RFCs to Indicate Requirement Levels*, March 1997
- [3] IETF RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax*, by Tim Berners-Lee and Roy T. Fielding and Larry M Masinter, January 2005. Copyright © The Internet Society (2005).
- [4] IETF RFC 7231, *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*, by Roy T. Fielding and Julian Reschke, June 2014. Copyright © 2014 IETF Trust and the persons identified as the document authors. All rights reserved.
- [5] IETF RFC 8174, *Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words*, May 2017
- [6] ISO 4217:2015, *Currency Codes*, August 2018
- [7] Fielding, Roy Thomas, *Architectural Styles and the Design of Network-based Software Architectures (Ph.D)*, 2000
- [8] MEF 55.1, *Lifecycle Service Orchestration (LSO): Reference Architecture and Framework*, February 2021
- [9] MEF 79, *Address, Service Site, and Product Ordering Qualification Management – Requirements and Use Cases*, November 2019
- [10] MEF 80, *Quote Management Requirements and Use Cases*, July 2021
- [11] MEF 81, *Product Inventory Management Requirements and Use Cases*, November 2019
- [12] MEF 81.0.1, *Amendment to MEF 81: Product Inventory Management*, February 2020
- [13] OpenAPI Initiative, *OpenAPI Specification (OAS) v3.0.3*, February 2020
- [14] TMF637 TM Forum, *TMF637 Product Inventory Management API Rest Specification 19.0.0*, June 2019