Working Draft

MEF W123, v0.1

LSO Cantata and LSO Sonata Product Order Management API - Developer Guide

June 2021

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

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 the implementation of the Product Order functionality defined for the LSO Cantata and LSO Sonata Interface Reference Points (IRPs), for which requirements and use cases are defined in MEF 57.2 *Product Order Management Requirements and Use Cases* [MEF57.2]. This standard consists of this document and complementary API definitions for Product Order Management and Product Order Notification.

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/order/productOrderManagement.api.yaml`
- `productApi/order/productOrderNotification.api.yaml`

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

- `productApi/order/productOrderManagement.api.yaml`
- `productApi/order/productOrderNotification.api.yaml`

2. Terminology and Abbreviations

This section defines the terms used in this document. In many cases, the normative definitions of 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	Description	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.	[MEF55.1]
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 [MEF80]

Connection Charge	A one-off charge set by the Seller to connect a Product Order Item to the Seller's network.	[MEF57.2]
Construction Charge	A one-off charge set by the Seller resulting from special construction required to provide a Buyer requested Product Order Item.	[MEF57.2]
Disconnect Charge	A one-off charge set by the Seller that results from a request by the Buyer to disconnect a Product.	[MEF57.2]
Expedite Charge	A one-off charge set by the Seller resulting from a request by the Buyer to expedite the Product Order Item.	[MEF57.2]
In-Flight	A Product Order Item that the Seller is still actively working on or intends to start or continue to work on. A Product Order is considered In-Flight when at least one of the Product Order Item it contains is In-Flight. An In-Flight Product Order may be updated or cancelled.	[MEF57.2]
Point of No Return	A point in the fulfillment of aa Product Order Item past which a Seller is unable or unwilling to accept a cancellation request on it. A Product Order is considered past the Point of No Return when all of its Product Order Items have reached their Point of No Return.	[MEF57.2]
Telecommunication Service Priority	A US centric term used to assign a priority for restoration of a Product in the event of a natural or other disaster impacting multiple Products.	[MEF57.2]
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.	[MEF79]
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.	[MEF79]
REST API	Representational State Transfer. 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.	[REST]
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 [MEF80]

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 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.

4. Introduction

This standard specification document describes the Application Programming Interface (API) for Product Order 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* [MEF55.1]. The LSO Reference Architecture is shown in Figure 1 with both IRPs highlighted.

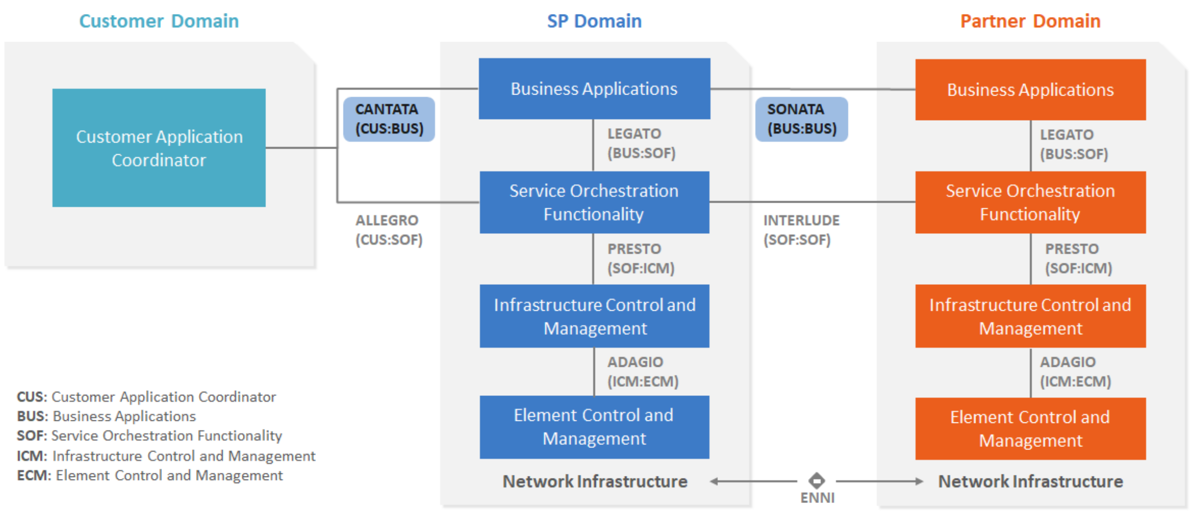

Customer Domain
SP Domain
Partner Domain
Customer Application Coordinator
CANTATA (CUS:BUS)
ALLEGRO (CUS:SOF)
Business Applications
LEGATO (BUS:SOF)
Service Orchestration Functionality
PRESTO (SOF:ICM)
Infrastructure Control and Management
ADAGIO (ICM:ECM)
Element Control and Management
Network Infrastructure
SONATA (BUS:BUS)
INTERLUDE (SOF:SOF)
ENNI
CUS: Customer Application Coordinator
BUS: Business Applications
SOF: Service Orchestration Functionality
ICM: Infrastructure Control and Management
ECM: Element Control and Management

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 (Service Provider or Partner) Domains. Those are:

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

The business requirements and use cases for Product Order Management are defined in MEF 57.2 *Product Order Management Requirements and Use Cases* [MEF57.2]. This document refers to the Draft Standard (R3) version, May 2021.

This document is structured as follows:

- Chapter 4 provides an introduction to Product Order Management and its description in a broader context of Cantata and Sonata and their corresponding SDKs.
- 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 API description.

4.1. Description

The Product Order Management API allows the Buyer to submit a Product Order request containing one or more Product Order items. The Buyer may place a Product Order for an installation (`add`) of a new service, Change (`modify`) to an existing service, or a Disconnect (`delete`) of an existing service.

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 5.4 explains how to use product specifications as the Product Order API payloads.

This document contains examples of Access E-Line Product as specified by *LSO Sonata Product Specification - Access E-Line Product Schema Guide* [MEF106]. These sample product specification definitions are used to construct API payload examples that illustrate API usage.

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. 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 [JS], whereas Product Order API is defined using OpenAPI 3.0 [OAS-V3]. The payloads exchanged through Product Order endpoints must comply with the Product specification schema as well as with MEF 57.2 [MEF57.2] requirements for Product Order Management.

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. `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 indicates a variable to be substituted with a correct value.

4.3. Relation to Other Documents

The requirements and use cases for Product Order Management are defined in MEF 57.2 [MEF57.2]. The API definition builds on *TMF622 Product Order Management API REST Specification R19.0.1* [TMF622]. Product Order Use Cases must support the use of any of MEF product specifications, in particular, the ones defined for the *LSO Sonata Product Specification - Access E-Line Product Schema Guide* in MEF W106 [MEF106].

4.4. Approach

As presented in Figure 2. both Cantata and 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)

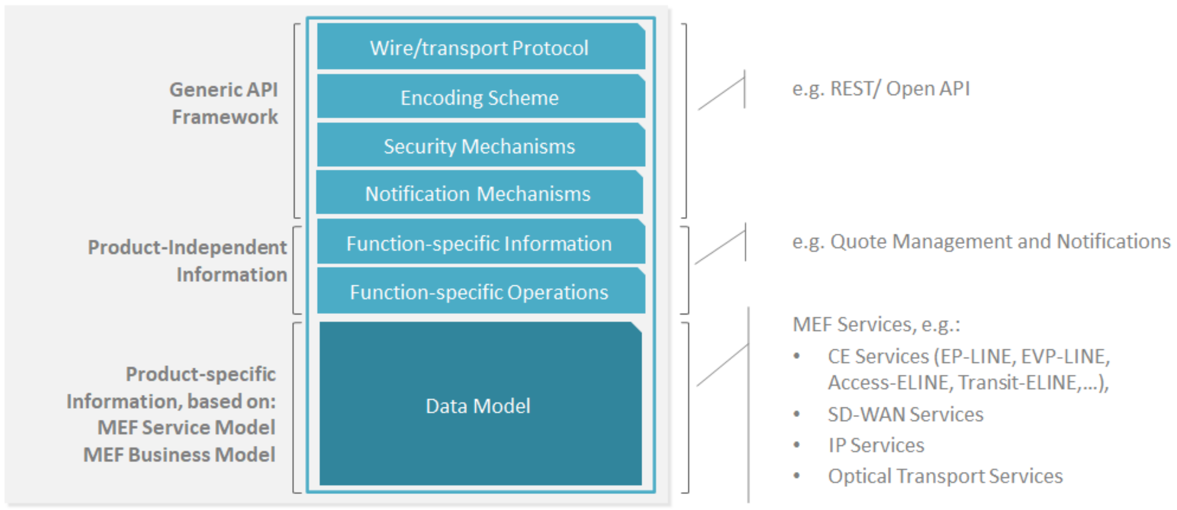

Generic API Framework
Product-Independent Information
Product-specific Information, based on: MEF Service Model MEF Business Model
Wire/transport Protocol
Encoding Scheme
Security Mechanisms
Notification Mechanisms
Function-specific Information
Function-specific Operations
Data Model
e.g. REST/ Open API
e.g. Quote Management and Notifications
MEF Services, e.g.:
CE Services (EP-LINE, EVP-LINE, Access-ELINE, Transit-ELINE,...),
SD-WAN Services
IP Services
Optical Transport Services

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. For example, this standard is describing the Product Order model and operations that allow performing quoting of any product that is aligned with either MEF or custom 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.

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.

4.5. High-Level Flow

Product Order Management 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 Order Management's position within it.

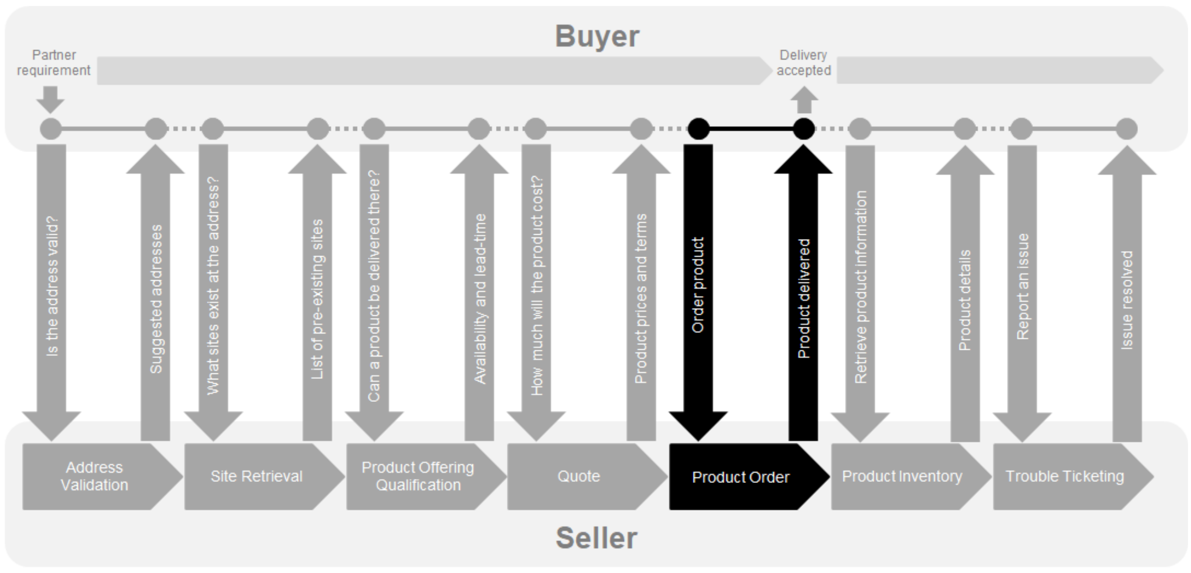


Figure 3. Cantata and Sonata End-to-End Function Flow

- 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.

5. 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.

5.1. High-level use cases

Figure 4 presents a high-level use case diagram as specified in MEF 57.2 [MEF57.2] in section 8.1. This picture aims to help understand the endpoint mapping. Use cases are described extensively in chapter 6

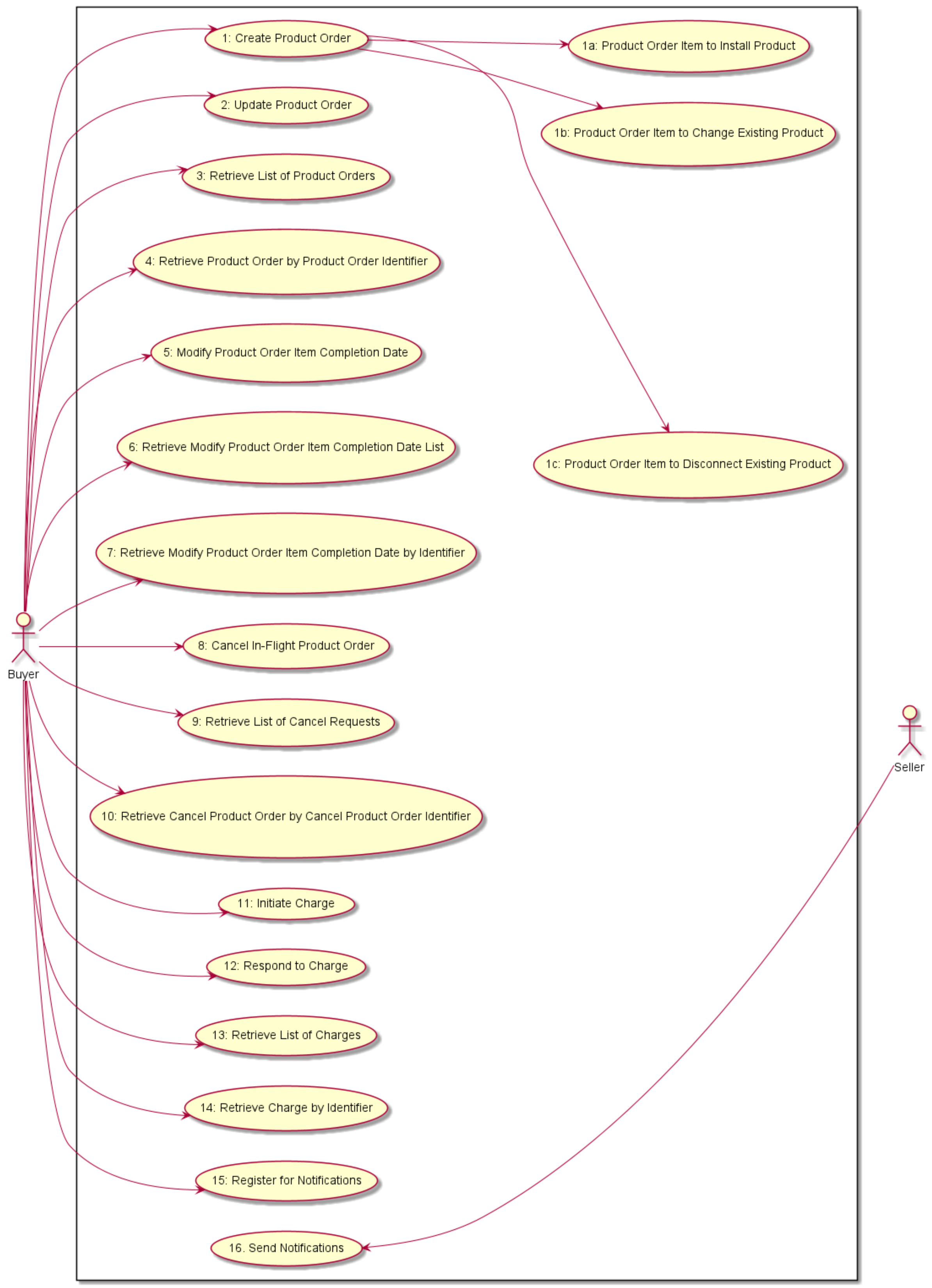

1: Create Product Order
1a: Product Order Item to Install Product
2: Update Product Order
1b: Product Order Item to Change Existing Product
3: Retrieve List of Product Orders
4: Retrieve Product Order by Product Order Identifier
5: Modify Product Order Item Completion Date
6: Retrieve Modify Product Order Item Completion Date List
1c: Product Order Item to Disconnect Existing Product
7: Retrieve Modify Product Order Item Completion Date by Identifier
Buyer
8: Cancel In-Flight Product Order
9: Retrieve List of Cancel Requests
Seller
10: Retrieve Cancel Product Order by Cancel Product Order Identifier
11: Initiate Charge
12: Respond to Charge
13: Retrieve List of Charges
14: Retrieve Charge by Identifier
15: Register for Notifications
16. Send Notifications

Figure 4: Use cases

5.2. API Endpoint and Operation Description

5.2.1. Seller side API Endpoints

BasePath for Cantata: `https://{{server}}:{{port}}{{?/seller_prefix}}/mefApi/cantata/productOrderManagement/v2/`

BasePath for Sonata: `https://{{server}}:{{port}}{{?/seller_prefix}}/mefApi/sonata/productOrderManagement/v7/`

The following API endpoints are implemented by the Seller and allow the Buyer to send Product Order requests, retrieve existing Product Orders or Product Order details, manage Charges and notification registrations. The endpoints and corresponding data model are defined in `productApi/order/productOrderManagement.api.yaml`.

API endpoint	Description	MEF 57.2 Use Case mapping
`POST /productOrder`	A request initiated by the Buyer to order a new product component(s).	UC 1: Create Product Order UC 1a: Product Order Item to Install Product UC 1b: Product Order Item to Change Existing Product UC 1c: Product Order Item to Disconnect Existing Product
`GET /productOrder`	A request initiated by the Buyer to retrieve a list of Product Orders that match the provided filter criteria	UC 3: Retrieve List of Product Orders
`GET /productOrder/{{id}}`	A request initiated by the Buyer to retrieve the details associated with a specific Product Order with the given Product Order Identifier.	UC 4: Retrieve Product Order by Product Order Identifier
`PATCH /productOrder/{{id}}`	Allows the Buyer to update some Product Order and Product Order Item Attributes	UC 2: Update Product Order
`POST /cancelProductOrder`	A request initiated by the Buyer to cancel an In-Flight Product Order.	UC 8: Cancel In-Flight Product Order
`GET /cancelProductOrder`	A request initiated by the Buyer to retrieve a list of Cancel Requests that match the provided filter criteria	UC 9: Retrieve List of Cancel Requests
`GET /cancelProductOrder/{{id}}`	A request initiated by the Buyer to retrieve the details associated with a specific Cancel Request with the given Cancel Request Identifier.	UC 10: Retrieve Cancel Product Order by Cancel Product Order Identifier
`GET /charge`	A request initiated by the Buyer to retrieve a list of Charges that match the provided filter criteria	UC 13: Retrieve List of Charges
`GET /charge/{{id}}`	A request initiated by the Buyer to retrieve the details associated with a specific Charge with the given Charge Identifier.	UC 14: Retrieve Charge by Charge Identifier

API endpoint	Description	MEF 57.2 Use Case mapping
`PATCH /charge/{{id}}`	Process to communicate if the Buyer accepts or rejects the charges.	UC 12: Respond to Charge
`POST /modifyProductOrderItemCompletionDate`	A request initiated by the Buyer to modify either the Expedite Indicator or the Requested Completion Date of a Product Order Item.	UC 5: Modify Product Order Item Completion Date
`GET /modifyProductOrderItemCompletionDate`	A request initiated by the Buyer to retrieve a list of Modify Product Order Item Completion Date that match the provided filter criteria	UC 6: Retrieve Modify Product Order Item Completion Date List
`GET /modifyProductOrderItemCompletionDate/{{id}}`	A request initiated by the Buyer to retrieve the details associated with a specific Modify Product Order Item Date with the given Modify Product Order Item Completion Date Identifier.	UC 7: Retrieve Modify Product Order Item Completion Date by Identifier
`POST /hub`	The Buyer requests to subscribe to notifications.	UC 15: Register for Notifications
`GET /hub/{{id}}`	A request initiated by the Buyer to retrieve the details of the notification subscription.	UC 15: Register for Notifications
`DELETE /hub/{{id}}`	A request initiated by the Buyer to instruct the Seller to stop sending notifications.	UC 15: Register for Notifications

Table 2. Seller side API endpoints

[RXXX] The Seller **MUST** Support:

- `POST /productOrder`
- `GET /productOrder`
- `GET /productOrder/{{id}}`

5.2.2. Buyer side API Endpoints

BasePath for Cantata: `https://{{server}}:{{port}}{{?/seller_prefix}}/mefApi/cantata/productOrderNotification/v2/`

BasePath for Sonata: `https://{{server}}:{{port}}{{?/seller_prefix}}/mefApi/sonata/productOrderNotification/v7/`

The following API Endpoints are used by the Seller to post notifications to registered listeners. The endpoints and corresponding data model are defined in `productApi/order/productOrderNotification.api.yaml`

API Endpoint	Description	MEF 57.2 Use Case Mapping
`POST /listener/productOrderStateChangeEvent`	A request initiated by the Seller to notify the Buyer on `ProductOrder` state change.	UC 15: Send Notification

API Endpoint	Description	MEF 57.2 Use Case Mapping
`POST /listener/productOrderItemStateChangeEvent`	A request initiated by the Seller to notify the Buyer on `ProductOrderItem` state change.	UC 15: Send Notification
`POST /listener/productOrderExpectedCompletionDateSet`	A request initiated by the Seller to notify the Buyer on `productOrder.expectedCompletionDate` value change.	UC 15: Send Notification
`POST /listener/productOrderItemExpectedCompletionDateSet`	A request initiated by the Seller to notify the Buyer on `productOrder.productOrderItem.expectedCompletionDate` value change.	UC 15: Send Notification
`POST /listener/productSpecificProductOrderMilestoneEvent`	A request initiated by the Seller to notify the Buyer on Product Specific Product Order Milestone reached event.	UC 15: Send Notification
`POST /listener/productSpecificProductOrderItemMilestoneEvent`	A request initiated by the Seller to notify the Buyer on Product Specific Product Order Item Milestone reached event.	UC 15: Send Notification
`POST /listener/cancelProductOrderStateChangeEvent`	A request initiated by the Seller to notify the Buyer on `CancelProductOrder` state change.	UC 15: Send Notification
`POST /listener/chargeCreateEvent`	A request initiated by the Seller to notify the Buyer on `Charge` create event to initiate the charge process.	UC 11: Initiate Charge UC 15: Send Notification
`POST /listener/chargeStateChangeEvent`	A request initiated by the Seller to notify the Buyer on `Charge` state change.	UC 15: Send Notification
`POST /listener/chargeTimeoutEvent`	A request initiated by the Seller to notify the Buyer on `Charge` timeout event.	UC 15: Send Notification

Table 3. Buyer side API endpoints

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 query parameters in each operation defined in `productOrderManagement.api.yaml` and as attributes of events as described in `productOrderNotification.api.yaml`.

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

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

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

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

[RXXX] If `buyerId` or `sellerId` attributes were specified in the request same attributes **MUST** be used in the notification payload.

5.4. Integration of Product Specifications into Product Order Management API

Product specifications are defined using JsonSchema (draft 7) format and are integrated into the `ProductOrder` 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 upon 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:order` is the abovementioned URN:

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

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

Product-specific attributes are introduced via the `MEFProductRefOrValue` (defined by the Buyer). 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 to data model during development. In such a case an integrated data model is built and product specifications are in 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 snippets below present an example of a static binding of the envelope API with several MEF product specifications, from both `MEFProductConfiguration` and 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:order: '#/components/schemas/AccessElineOvcOrder_v1.0.0'
      urn:mef:lso:spec:cantata-sonata:SubscriberUni:v1.0.0:order:
'#/components/schemas/SubscriberUniOrder_v1.0.0'
      urn:mef:lso:spec:cantata-sonata:EplEvc:v1.0.0:order: '#/components/schemas/EplEvcOrder_v1.0.0'
      urn:mef:lso:spec:sonata:OperatorUNI:v1.0.0:order: '#/components/schemas/OperatorUNIOrder_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
```

```
AccessElineOvcOrder_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 the requirements specified below.

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

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

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

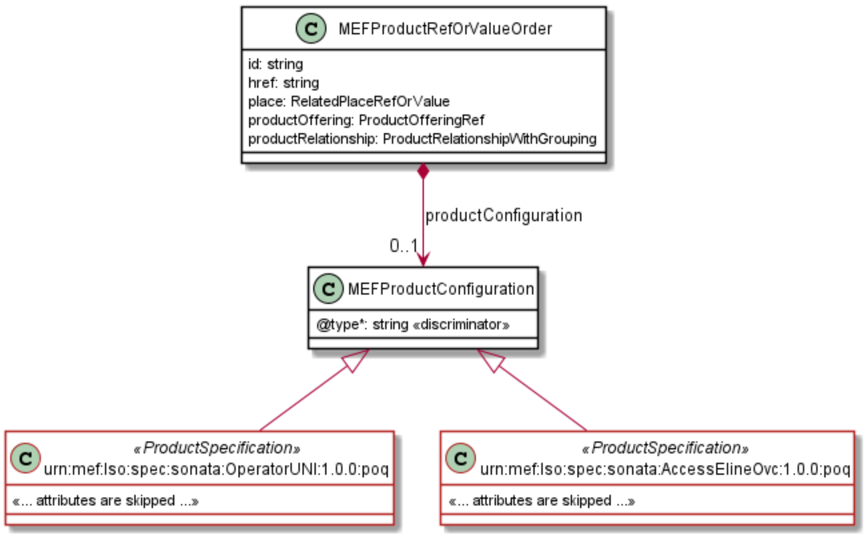

MEFProductRefOrValueOrder
id: string
href: string
place: RelatedPlaceRefOrValue
productOffering: ProductOfferingRef
productRelationship: ProductRelationshipWithGrouping
productConfiguration
0..1
MEFProductConfiguration
@type*: string «discriminator»
«ProductSpecification»
urn:mef:lso:spec:sonata:OperatorUNI:1.0.0:poq
«... attributes are skipped ...»
«ProductSpecification»
urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:poq
«... attributes are skipped ...»

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 Order payload the `@type` of `MEFProductConfiguration` takes `"urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:order"` or `"urn:mef:lso:spec:sonata:OperatorUNI:1.0.0:order"` value to indicate which product specification should be used to interpret a set of product-specific properties included in the payload. An example of a product definition inside the `ProductOrderItem` is presented in Section 6.1.6.

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

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.

The product specification data model definitions are available as JsonSchema (version `draft 7`) documents. Figure 6. and 7 depict simplified UML views on these data models in which:

- the mandatory attributes are marked with `*`,
- the mandatory relations have a cardinality of `1` or `1..*`,
- some relations and attributes are omitted.

The detailed Access E-line product specification description is provided in MEF W106 [MEF106].

The red color on the figures below highlights the model of Access E-Line.

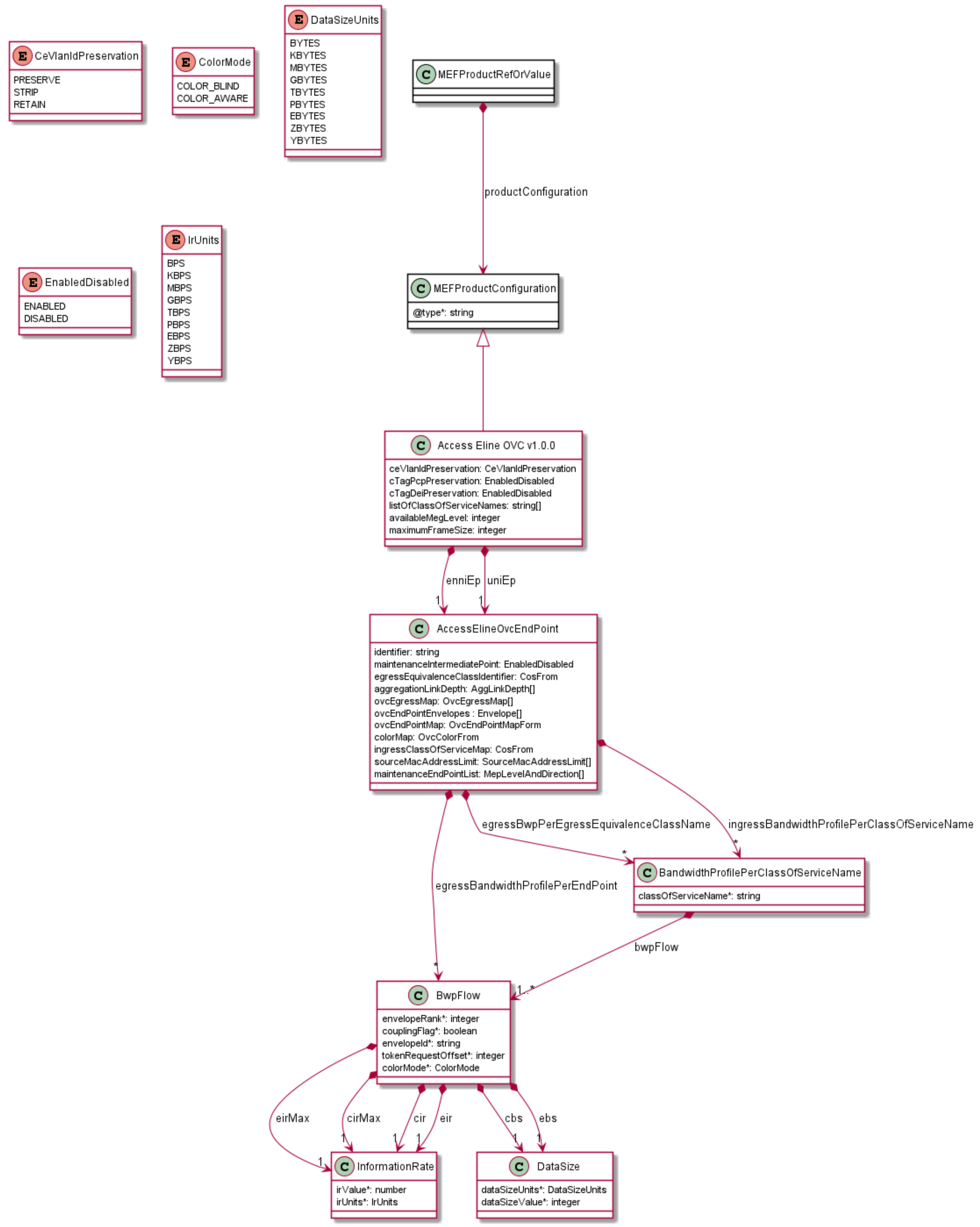

CeVlanIdPreservation
PRESERVE
STRIP
RETAIN
ColorMode
COLOR_BLIND
COLOR_AWARE
DataSizeUnits
BYTES
KBYTES
MBYTES
GBYTES
TBYTES
PBYTES
EBYTES
ZBYTES
YBYTES
MEFProductRefOrValue
productConfiguration
EnabledDisabled
ENABLED
DISABLED
IrUnits
BPS
KBPS
MBPS
GBPS
TBPS
PBPS
EBPS
ZBPS
YBPS
MEFProductConfiguration
@type*: string
Access Eline OVC v1.0.0
ceVlanIdPreservation: CeVlanIdPreservation
cTagPcpPreservation: EnabledDisabled
cTagDeiPreservation: EnabledDisabled
listOfClassOfServiceNames: string[]
availableMegLevel: integer
maximumFrameSize: integer
enniEp
uniEp
1
1
AccessElineOvcEndPoint
identifier: string
maintenanceIntermediatePoint: EnabledDisabled
egressEquivalenceClassIdentifier: CosFrom
aggregationLinkDepth: AggLinkDepth[]
ovcEgressMap: OvcEgressMap[]
ovcEndPointEnvelopes : Envelope[]
ovcEndPointMap: OvcEndPointMapForm
colorMap: OvcColorFrom
ingressClassOfServiceMap: CosFrom
sourceMacAddressLimit: SourceMacAddressLimit[]
maintenanceEndPointList: MepLevelAndDirection[]
egressBwpPerEgressEquivalenceClassName
ingressBandwidthProfilePerClassOfServiceName
*
*
egressBandwidthProfilePerEndPoint
BandwidthProfilePerClassOfServiceName
classOfServiceName*: string
bwpFlow
*
1..*
BwpFlow
envelopeRank*: integer
couplingFlag*: boolean
envelopeId*: string
tokenRequestOffset*: integer
colorMode*: ColorMode
eirMax
cirMax
cir
eir
cbs
ebs
1
1
1
1
1
1
InformationRate
irValue*: number
irUnits*: IrUnits
DataSize
dataSizeUnits*: DataSizeUnits
dataSizeValue*: integer

Figure 6. A simplified view on Access E-Line product specification data model

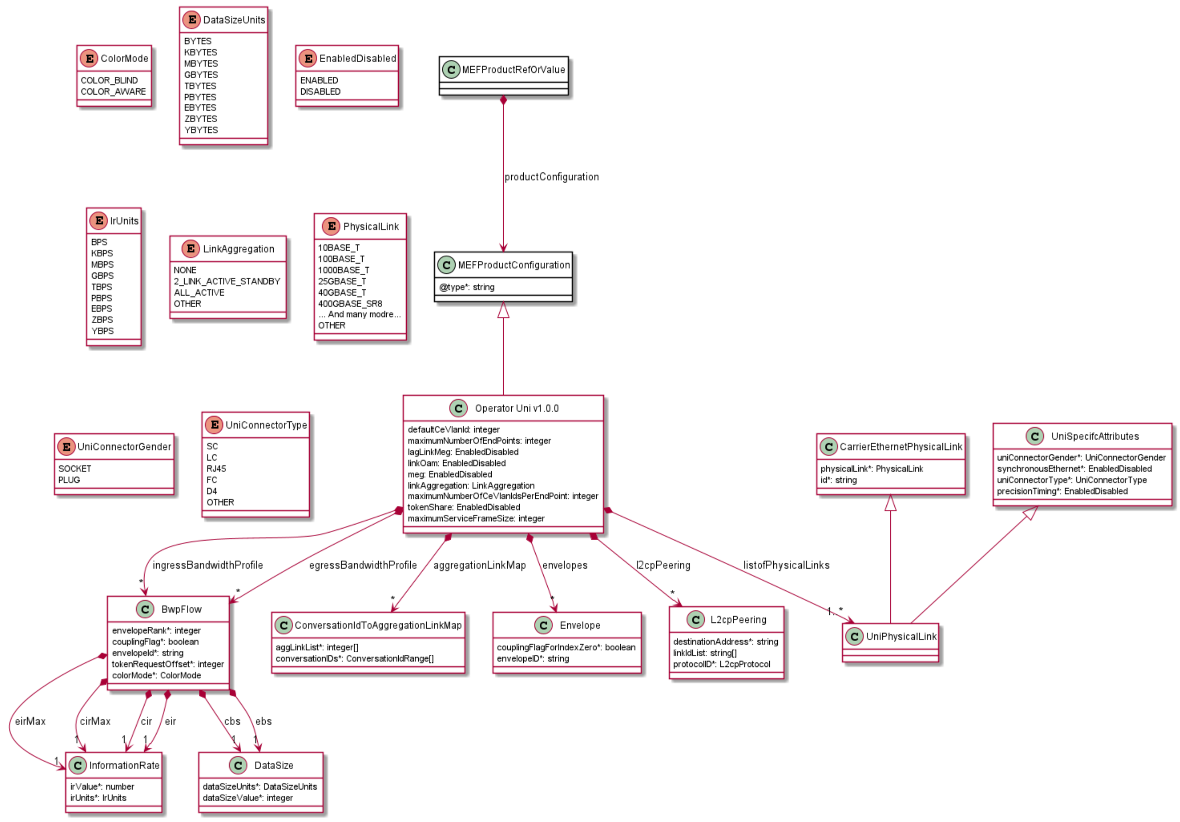


Figure 7. A simplified view on UNI product specification data model

Product specifications define several product-related and envelope-related requirements. For example (as of MEF 106):

- an Access E-Line product defines two mandatory product relationship roles, one with the operator ENNI (`ENNI_REFERENCE`) and a second with the operator UNI (`UNI_REFERENCE`) for `add` action. 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 order) or product relationship
- product relationships cannot be specified in the case of `modify` or `delete` actions
- an operator UNI product defines a place relationship (`INSTALL_LOCATION`) that must be specified for `add` action
- place relationships cannot be specified in the case of `modify` or `delete` actions

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 Product Order API endpoints is defined using:

- OpenAPI version 3.0 for product-agnostic part of the payload
- JsonSchema (draft 7) for product-specific part of the payload

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

[RXXX] 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 items in the same Product Order request
- required relation type, multiplicity to entities in the Seller's product inventory

- related contact information roles that are to be defined at the item level
- relations to places (locations) and their roles that are to be defined at the item level [MEF57.2 R23]

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 is outside the scope of this document. It is being worked on by a separate MEF Project and will be applied to the APIs once provided as a standard.

6. API Interactions and Flows

This section provides a detailed insight into the API functionality, use cases, and flows. It starts with Table 4 presenting a list and short description of all business use cases then presents the variants of end-to-end interaction flows, and in following subchapters describes the API usage flow and examples for each of the use cases.

Use Case #	**Use Case Name**	**Use Case Description**
1	Create Product Order	A request initiated by the Buyer to Product Order a new product or service component(s). A Product Order must contain at least one Product Order Item (Use Case # 1-a, 1-b, or 1-c) as shown below. A Product Order may contain more than one Product Order Item and Product Order Items within a Product Order are not required to have relationships between them.
1-a	Product Order Item to Install Product	Product Order Item installs a new Product.
1-b	Product Order Item to Change Existing Product	Product Order Item changes attributes of a specific active Product.
1-c	Product Order Item to Disconnect Existing Product	Product Order Item disconnects an active Product.
2	Update Product Order	Allows the Buyer to update some Product Order and Product Order Item Attributes
3	Retrieve List of Product Orders	A request initiated by the Buyer to retrieve a list of Product Orders that match the provided filter criteria

Use Case #	Use Case Name	Use Case Description
4	Retrieve Product Order by Product Order Identifier	A request initiated by the Buyer to retrieve the details associated with a specific Product Order with the given Product Order Identifier.
5	Modify Product Order Item Completion Date	A request initiated by the Buyer to modify either the Expedite Indicator or the Requested Completion Date of a Product Order Item.
6	Retrieve Modify Product Order Item Completion Date List	A request initiated by the Buyer to retrieve a list of Modify Product Order Item Completion Date that match the provided filter criteria
7	Retrieve Modify Product Order Item Completion Date by Identifier	A request initiated by the Buyer to retrieve the details associated with a specific Modify Product Order Item Date with the given Modify Product Order Item Completion Date Identifier.
8	Cancel In-Flight Product Order	A request initiated by the Buyer to cancel an In-Flight Product Order.
9	Retrieve List of Cancel Requests	A request initiated by the Buyer to retrieve a list of Cancel Requests that match the provided filter criteria
10	Retrieve Cancel Product Order by Cancel Product Order Identifier	A request initiated by the Buyer to retrieve the details associated with a specific Cancel Product Order Request with the given Cancel Product Order Request Identifier.
11	Initiate Charge	Process to communicate charges from the Seller to Buyer
12	Respond to Charge	Process to communicate if the Buyer accepts or rejects the charges.

Use Case #	Use Case Name	Use Case Description
13	Retrieve List of Charges	A request initiated by the Buyer to retrieve a list of Charges that match the provided filter criteria
14	Retrieve Charge by Identifier	A request initiated by the Buyer to retrieve the details associated with a specific Charge with the given ChargeIdentifier.
15	Register for Notifications	The Buyer requests to subscribe to notifications.
16	Send Notification	A notification initiated by the Seller to the Buyer providing subsequent status information on Product OrderCancel Requests, and ChargesCharge.

Table 4. Use cases description

The detailed business requirements of each of the use cases are described in sections 8 and 10 of MEF 57.2 [MEF57.2].

6.1. Use case 1: Create Product Order

This is the initial step for Product Order processing.

6.1.1. Interaction flow

The flow of this use case is very simple and is described in Figure 8.

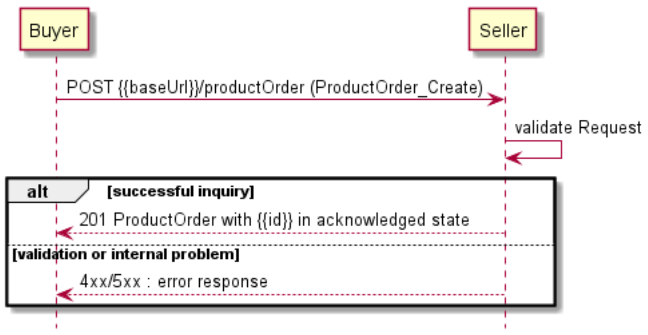

Buyer
Seller
POST {{baseUrl}}/productOrder (ProductOrder_Create)
validate Request
alt
[successful inquiry]
201 ProductOrder with {{id}} in acknowledged state
[validation or internal problem]
4xx/5xx : error response

Figure 8: Use Case 1 - Product Order create request flow

The Buyer sends a request with a `ProductOrder_Create` type in the body. The Seller performs request validation, assigns an `id`, and returns `ProductOrder` type in the response body, with a `state` set to `acknowledged`. From this point, the Product Order is ready for further processing. The Buyer can track the progress of the process either by subscribing for notifications or by periodically polling the `ProductOrder` status. The two patterns are presented in the following two diagrams.

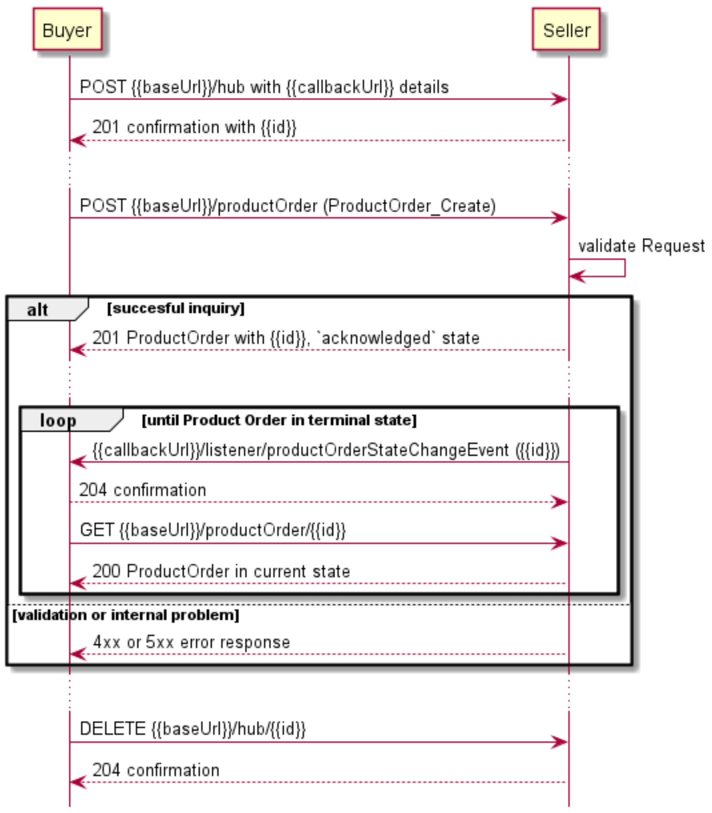

Buyer
Seller
POST {{baseUrl}}/hub with {{callbackUrl}} details
201 confirmation with {{id}}
POST {{baseUrl}}/productOrder (ProductOrder_Create)
validate Request
alt
[succesful inquiry]
201 ProductOrder with {{id}}, `acknowledged` state
loop
[until Product Order in terminal state]
{{callbackUrl}}/listener/productOrderStateChangeEvent ({{id}})
204 confirmation
GET {{baseUrl}}/productOrder/{{id}}
200 ProductOrder in current state
[validation or internal problem]
4xx or 5xx error response
DELETE {{baseUrl}}/hub/{{id}}
204 confirmation

Figure 9: Product Order progress tracking - Notifications

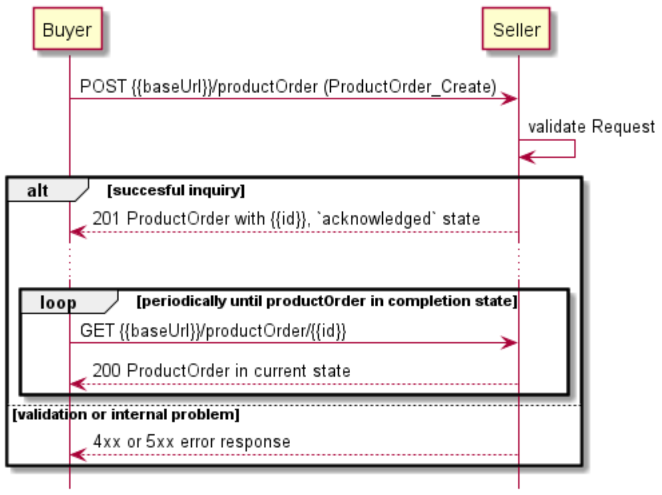

Buyer
Seller
POST {{baseUrl}}/productOrder (ProductOrder_Create)
validate Request
alt
[succesful inquiry]
201 ProductOrder with {{id}}, `acknowledged` state
loop
[periodically until productOrder in completion state]
GET {{baseUrl}}/productOrder/{{id}}
200 ProductOrder in current state
[validation or internal problem]
4xx or 5xx error response

Figure 10: Product Order progress tracking - Polling

Note: The context of notifications is not a part of the considered use case itself. It is presented to show the big picture of end-to-end flow. This applies also to all further use case flow diagrams with notifications.

6.1.2. Key Entities - Request

Figure 11 presents the most important parts of the data model used during the Product Order request (`POST /productOrder`) that is sent by a Buyer (see Section 5.2.1 for details). The model of the request message is a subset of the `ProductOrder` model and contains only attributes that can (or must) be set by the Buyer. The Seller then enriches the entity in the response with additional information.

[RXXX] `ProductOrder_Create` is the root entity of a Product Order request. It **MUST** contain one or more items of type `ProductOrderItem_Create`.

Note: `ProductOrder_Create` and `ProductOrderItem_Create` are entities used by the Buyer to make a request. `ProductOrder` and `ProductOrderItem` are entities used by the Seller to provide a response. The request entities have a subset of attributes of the response entities. Thus for visibility of these shared attributes `ProductOrder_Common` and `ProductOrderItem_Common` have been introduced. Though, these are not to be used directly in the exchange.

A `ProductOrderItem_Create` defines details of the product(s) being subject of the ordering (in `MEFProductRefOrValueOrder` structure) and allows for the definition of additional information like related parties (`RelatedContactInformation`) or relations to other items (`ProductOrderItemRelationship`).

`MEFProductRefOrValueOrder` allows for the introduction of MEF product-specific properties to the Product Order payload. The extension mechanism is described in details in Section 5.4. `MEFProductRefOrValueOrder` may be also used to specify relations to places (using specializations of `RelatedPlaceOrValue`) and/or to a product that exists in the Seller's inventory (using `ProductRelationship`).

The full list of attributes is available in Section 7 and in the API specification which is an integral part of this standard.

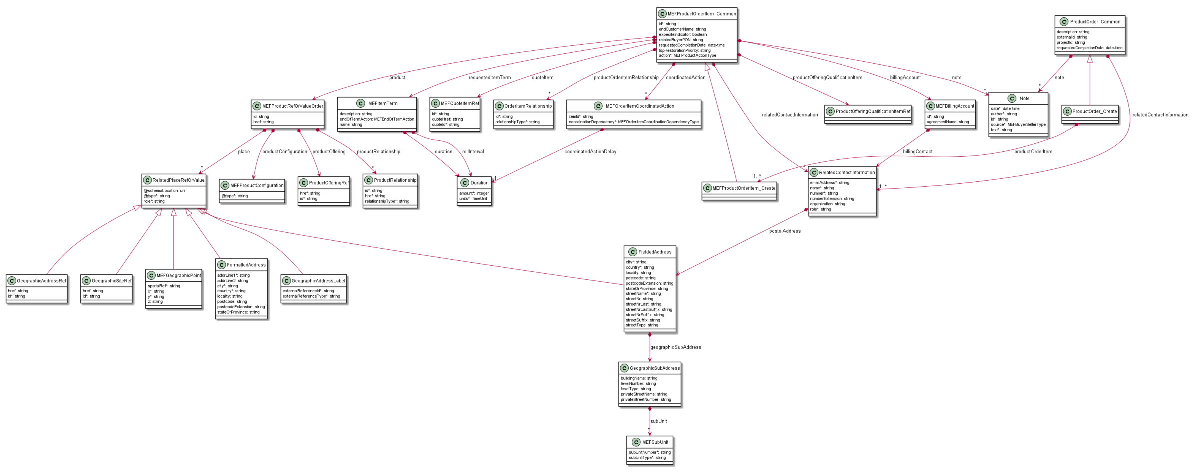


Figure 11: Key Entities - Create Request

6.1.3. Request Example

To send a Product Order request the Buyer uses the `createProductOrder` operation from the API: `POST /productOrder`. For clarity, some of the Product Order payload's attributes might be omitted to improve examples' readability. The `ProductOrder_Create` is a simple structure that is common for all types of requests (`add`, `modify`, `delete`), most of the information is in the `ProductOrderItem_Create`.

`Product Order` Create

```
{
  "description": "A free text.",
  "externalId": "buyerOrder-001",
  "projectId": "buyerProject-001",
  "requestedCompletionDate": "2021-06-19T20:59:28.299Z",
  "relatedContactInformation": [
    {
      "emailAddress": "john.example@example.com",
      "name": "John Example",
      "number": "12-345-6789",
      "numberExtension": "1234",
      "organization": "Example Co.",
      "role": "productOrderContact"
    }
  ],
  "productOrderItem": [
    {
      "id": "item-001",
      "action": "add",
      "endCustomerName": "End Customer Name",
      "expediteIndicator": false,
      "relatedBuyerPON": "PON-12-2021",
      "requestedCompletionDate": "2021-06-19T20:59:28.299Z",
      "billingAccount": {
        "id": "00000000-1111-0000-0000-000000000001",
        "billingContact": {
          "emailAddress": "bill.contact@example.com",
          "name": "Bill Contact",
          "number": "+12-345-678-90",
          "numberExtension": "string",
          "organization": "string",
          "role": "billingContact"
        },
        "agreementName": "Buyer-Seller General Agreement 03/2021"
      },
      "coordinatedAction": [
        {
          "itemId": "item-002",
          "coordinatedActionDelay": {
            "amount": 1,
            "units": "calendarWeeks"
          },
          "coordinationDependency": "startToStart"
        }
      ],
      "product": {
        "productConfiguration": { << product specific attributes and configuration, see 6.1.6 >>
        },
        "productOffering": {
          "id": "00000000-5555-0000-0000-000000000001"
        },
        "productRelationship": [
          {
            "id": "00000000-6666-0000-0000-000000000001",
            "relationshipType": "ENNI_REFERENCE"
          }
        ]
      },
      "productOfferingQualificationItem": {
        "id": "poqItem-001",
        "productOfferingQualificationId": "00000000-2222-0000-0000-000000000001"
      },
      "productOrderItemRelationship": [
        {
          "id": "item-002",
          "relationshipType": "UNI_REFERENCE"
        }
      ],
      "quoteItem": {
        "id": "quoteItem-001",
        "quoteId": "00000000-4444-0000-0000-000000000001"
      },
      "relatedContactInformation": [
        {
          "emailAddress": "Buyer.ProductOrderItemContact@example.com",
          "name": "Buyer Product Order Item Contact",
          "number": "+12-345-678-90",
          "role": "buyerProductOrderItemContact"
        },
        {
          "emailAddress": "Buyer.ImplementationContact@example.com",
          "name": "Buyer Implementation Contact",
          "number": "+12-345-678-90",
          "role": "buyerImplementationContact"
        },
        {
          "emailAddress": "Buyer.TechnicalContact@example.com",
          "name": "Buyer Technical Contact ",
          "number": "+12-345-678-90",
          "role": "buyerTechnicalContact "
        }
      ],
      "requestedItemTerm": {
        "duration": {
          "amount": 12,
          "units": "calendarMonths"
        },
        "endOfTermAction": "autoRenew",
        "name": "Yearly Subscription"
      }
```

```
      },
      {
        "id": "item-002",
        "action": "add"
        ...
        << attributes skipped for readability >>
      }
    ]
  }
```

[RXXX] The Buyer's request **MUST** contain at least one `productOrderItem`. [MEF57.2 R3]

[RXXX] The Buyer's request **MUST** specify a `relatedContactInformation` item with a `role` set to `productOrderContact`. [MEF57.2 R3]

For each `productOrderItem`:

[RXXX] The Buyer's Create Product Order request **MUST** contain: [MEF57.2 R14]

- `id`,
- `action`,
- `requestedCompletionDate`,
- `relatedContactInformation` items with following values of `role` set:
  - `buyerProductOrderItemContact`,
  - `buyerImplementationContact`,
  - `buyerTechnicalContact`.

[OXXX] The Seller **MAY** require that the `billingAccount` attributes be the same for all Product Order Items in a Product Order. [MEF57.2 O7]

[OXXX] The Seller **MAY** require the Buyer to perform a POQ prior to submitting the Product Order. [MEF57.2 O3]

[RXXX < OXXX] The Buyer's request **MUST** provide the `productOfferingQualificationItem` if required by the Seller. [MEF57.2 CR1<O3]

[OXXX] The Seller **MAY** require the Buyer to perform a Quote prior to submitting the Product Order. [MEF57.2 O4]

[RXXX < OXXX] The Buyer's request **MUST** provide the `quoteItem` if required by the Seller.[MEF57.2 CR2<O4]

[RXXX] If the Buyer requires the `tspRestorationPriority` to be specified for the Product Order Item, the Buyer's Create Product Order request **MUST** provide it. [MEF57.2 R16]

6.1.4. Key Entities - Response

Figure 12 presents the most important data model parts used to provide a response to a Buyer's Create Product Order (`POST /productOrder`) or to retrieve a `ProductOrder` by identifier (`GET /productOrder/{{id}}`) request. Please note that the model differs only with the number of attributes for `ProductOrder` and `ProductOrderItem` entities.

`ProductOrder` is the root entity of a response and it contains the same number of `ProductOrderItems` as in the request.

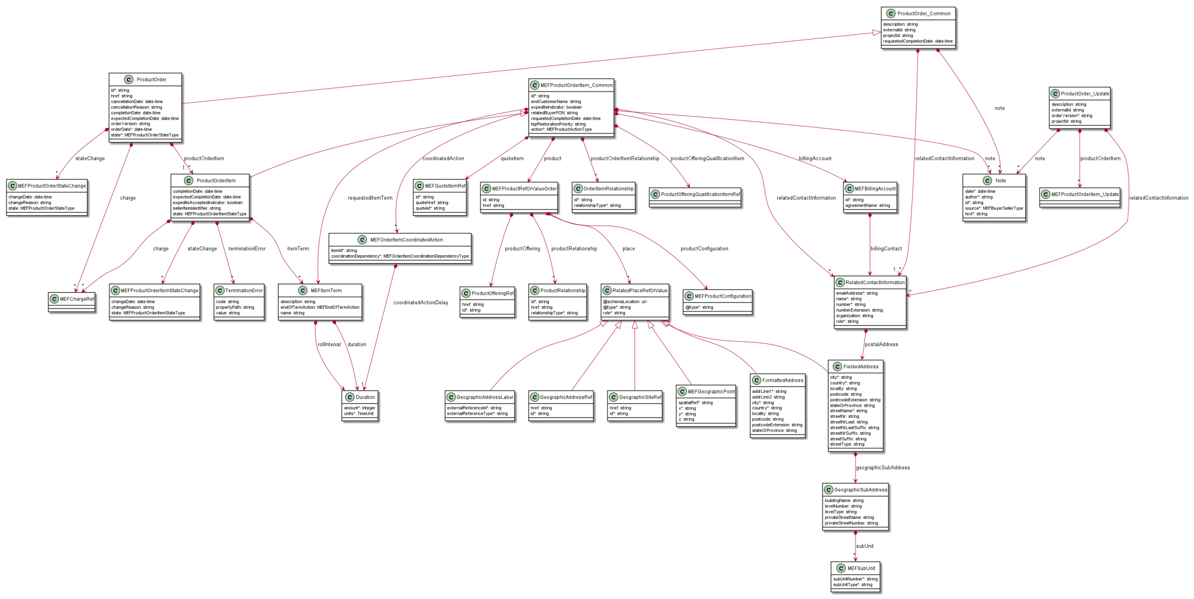


Figure 12: Key Entities - Response

Note: The term "Seller Response Code" used in the Business Requirements maps to HTTP response code, where `2xx` indicates *Success* and `4xx` or `5xx` indicate *Failure*.

6.1.5. Response Example

The following snippet presents the Seller's response. It has the same structure as in the retrieve by identifier operation.

```
{
  "id": "00000000-1111-2222-3333-000000000123",
  "href": "{{baseUrl}}/productOrder/00000000-1111-2222-3333-000000000123",
  "expectedCompletionDate": "2021-05-31T00:00:00.000Z",
  "orderVersion": "1",
  "orderDate": "2021-05-19T07:01:02.983Z",
  "state": "acknowledged",
  "description": "A free text.", << as provided by the Buyer >>
  "externalId": "buyerOrder-001", << as provided by the Buyer >>
  "projectId": "buyerProject-001", << as provided by the Buyer >>
  "requestedCompletionDate": "2021-06-19T20:59:28.299Z", << as provided by the Buyer >>
  "relatedContactInformation": [
    { << as provided by the Buyer >>
      "emailAddress": "john.example@example.com",
      "name": "John Example",
      "number": "12-345-6789",
      "numberExtension": "1234",
      "organization": "Buyer Example Co.",
      "role": "productOrderContact",
    },
    {
      "emailAddress": "kate.example@example.com",
      "name": "Kate Example",
      "number": "12-345-67890",
      "organization": "Seller Example Co.",
      "role": "sellerContact"
    }
  ],
  "productOrderItem": [
    {
      "id": "item-001", << as provided by the Buyer >>
      "action": "add", << as provided by the Buyer >>
      "endCustomerName": "End Customer Name", << as provided by the Buyer >>
      "expediteIndicator": false, << as provided by the Buyer >>
      "relatedBuyerPON": "PON-12-2021", << as provided by the Buyer >>
      "requestedCompletionDate": "2021-06-19T20:59:28.299Z", << as provided by the Buyer >>
      "expectedCompletionDate": "2021-05-31T00:00:00.000Z",
      "expediteAcceptedIndicator": false,
      "sellerItemIdentifier": "sellerItemId-001",
      "state": "acknowledged",
      "billingAccount": { << as provided by the Buyer >> },
      "coordinatedAction": [ << as provided by the Buyer >> ],
      "product": { << as provided by the Buyer >> },
      "productOfferingQualificationItem": { << as provided by the Buyer >> },
      "productOrderItemRelationship": [ << as provided by the Buyer >> ],
      "quoteItem": { << as provided by the Buyer >> },
      "relatedContactInformation": [
        {
          "emailAddress": "Buyer.ProductOrderItemContact@example.com",
          "name": "Buyer Product Order Item Contact",
          "number": "+12-345-678-90",
          "role": "buyerProductOrderItemContact"
        },
        {
          "emailAddress": "Buyer.ImplementationContact@example.com",
          "name": "Buyer Implementation Contact",
          "number": "+12-345-678-90",
          "role": "buyerImplementationContact"
        },
        {
          "emailAddress": "Buyer.TechnicalContact@example.com",
          "name": "Buyer Technical Contact ",
          "number": "+12-345-678-90",
          "role": "buyerTechnicalContact "
        },
        {
          "emailAddress": "Seller.Contact@example.com",
          "name": "Seller Contact",
          "number": "+12-345-678-90",
          "role": "sellerContact"
        }

      ],
      "requestedItemTerm": {
        "duration": {
          "amount": 12,
          "units": "calendarMonths"
        },
        "endOfTermAction": "autoRenew",
        "name": "Yearly Subscription",
      },
      "itemTerm": [
        {
          "duration": {
            "amount": 12,
            "units": "calendarMonths"
          },
          "endOfTermAction": "autoRenew",
          "name": "Yearly Subscription",
        }
      ],
      "stateChange": [
        {
          "changeDate": "2021-05-19T07:01:02.983Z",
          "state": "acknowledged"
        }
      ]
    },
    {
      "id": "item-002",
      "action": "add"
      ...
      << attributes skipped for readability >>
    }
  ],
  "stateChange": [
    {
    "changeDate" : "2021-05-19T07:01:02.983Z",
    "state" : "acknowledged"
    }
  ]
}
```

The response to the create request does not contain all possible attributes. Some of them are valid only in the future lifecycle of the `Product Order` (e.g. `cancellationDate`, `cancellationReason`, `completionDate`).

[RXXX] The Seller's response **MUST** include all and unchanged attributes' values provided in the request. [MEF57.2 R8], [MEF57.2 R25]

These attributes are indicated above with an appropriate comment: `<< as provided by the Buyer >>`.

The Seller might append related contact information if required, either at item or Product Order level but cannot modify related contact information provided by the Buyer.

[RXXX] The Seller **MUST** specify the following attributes in a response: [MEF57.2 R5]

- `id`,
- `productOrderItem`,
- `orderVersion`,
- `state`,
- `relatedContactInformation` item with a `role` set to `sellerContact`

[RXXX] Each item in `productOrderItem` list **MUST** correspond to one and only one Product Order Item in the Buyer's request.

[RXXX] The `stateChange` **MUST** contain a full history of the `ProductOrder.state`. [MEF57.2 R10], [MEF57.2 R37], [MEF57.2 R47], [MEF57.2 R51]

For each `productOrderItem`:

[RXXX] The response **MUST** have the `state` attribute set. [MEF57.2 R24]

[RXXX] The `stateChange` **MUST** contain a full history of the `state`.

[RXXX] If in the request the `expediteIndicator` is `false`, the Seller's response **MUST NOT** have the `expediteAcceptedIndicator` attribute set to `true`. [MEF57.2 R26]

[RXXX] The response **MUST NOT** include the `expediteAcceptedIndicator` attribute set to `true` until the `Charge` process for any charges associated with the expedite is complete. [MEF57.2 R27]

[RXXX] The Seller **MUST** set the `orderVersion` to `1` at the time that the Buyer Create Product Order is acknowledged.

6.1.6 Use Case 1a: Product Order Item to Install Product

When requesting a new product installation (`action` equal to `add`) the Buyer needs to provide all of its configuration information. The example below shows a request for Access E-Line product (type `urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:order`). Assuming this is an extension of a previous example, the Product Order and less important attributes are omitted.

```
{
  <<ProductOrder attributes...>>
  "productOrderItem": [
    {
      "id": "item-001",
      "action": "add",
      ...
      "product": {
        "@type": "MEFProductRefOrValueOrder",
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:order",
          "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"
                    },
                }
            ]
        }
    ]
  }
},
"productOffering": {
  "id": "00000000-5555-0000-0000-000000000001"
},
"productRelationship": [
  {
```

```
              "id": "00000000-6666-0000-0000-000000000001",
              "relationshipType": "ENNI_REFERENCE"
            }
          ]
        },
        "productOfferingQualificationItem": {
          "id": "poqItem-001",
          "productOfferingQualificationId": "00000000-2222-0000-0000-000000000001"
        },
        "productOrderItemRelationship": [
          {
            "id": "item-002",
            "relationshipType": "UNI_REFERENCE"
          }
        ],
        "quoteItem": {
          "id": "quoteItem-001",
          "quoteId": "00000000-4444-0000-0000-000000000001"
        },
        "relatedContactInformation": [
          {
            "emailAddress": "Buyer.ProductOrderItemContact@example.com",
            "name": "Buyer Product Order Item Contact",
            "number": "+12-345-678-90",
            "role": "buyerProductOrderItemContact"
          },
          {
            "emailAddress": "Buyer.ImplementationContact@example.com",
            "name": "Buyer Implementation Contact",
            "number": "+12-345-678-90",
            "role": "buyerImplementationContact"
          },
          {
            "emailAddress": "Buyer.TechnicalContact@example.com",
            "name": "Buyer Technical Contact ",
            "number": "+12-345-678-90",
            "role": "buyerTechnicalContact "
          }
        ],
        "requestedItemTerm": {
          "duration": {
            "amount": 12,
            "units": "calendarMonths"
          },
          "endOfTermAction": "autoRenew",
          "name": "Yearly Subscription",
        }
      },
      {
        "id": "item-002",
        "action": "add"
        <<Product Order Item Item with UNI Product configuration that the E-Line OVC refers to>>
      }
    ]
  }
```

The following requirements apply when `productOrderItem.action` is `add`:

[RXXX] The Buyer **MUST** provide the `productOrderItem.product`. [MEF57.2 R15]

[RXXX] If there is a relationship with another Product Order Item within the same Product Order, the `productOrderItem.product.productRelationship` **MUST** be specified. [MEF57.2 R31]

[RXXX] `productOrderItem.product.productOffering` **MUST** be provided. [MEF57.2 R32]

[RXXX] The Buyer **MUST** provide the `productOrderItem.billingAccount`. [MEF57.2 R33]

[RXXX] The Buyer **MUST NOT** specify the `productOrderItem.product.id` in the request. It is the Seller who assigns this id.

An Access E-Line product specification defines two mandatory relationship types that have to be specified in case of ordering an `add` action: `ENNI_REFERENCE` and `UNI_REFERENCE`.
The reference to an operator UNI product might use another Product Order item or an existing product from

the Seller's inventory. This example assumes that the UNI product is another item of the request with a unique identifier `item-002`. This Access E-Line product references an existing ENNI product which is uniquely identified with id `00000000-6666-0000-0000-000000000001` in the Seller's inventory.

The place is not provided as Access E-Line product specification does not allow for a place description to be part of the request. Values for some of the available product attributes are provided under `productConfiguration` node. This example uses only a tiny subset of available Access E-Line attributes. It aims to explain the Product definition and relation patterns, not to focus on the product configurations themselves.

This specification describes the structure and requirements defined for this product with which the payload should be validated. Product specification is a subject of MEF standardization. It is published as a dedicated MEF standard. It is build of:

- the JSON Schemas for technical specifications. Those can be found in the SDK in the `\productSchema\` directory.
- a document with a textual description of the product and a list of the requirements (not all of them can be technically included in the JSON schema). Such documents can be found in the `\documentation\productSchema\` directory of the SDK package.

The product offering is a business representation of a product specification version offered by the Seller for purchase. Product offering associates commercial attributes to a product specification. The product offering model is not part of the standardization and is up to the Seller to define their offering.

Until the Product Catalog API is available, both product specifications and product offerings are not negotiated and exchanged within Cantata and Sonata. They are agreed between the Buyer and the Seller during the onboarding process. After that, they are only referenced as in the example above.

6.1.7 Use case 1b: Product Order Item to Change Existing Product

The following example shows a request for a quotation of an existing Access E-Line Product modification (`action` equal to `modify`). In particular, changes to `cir` (Committed Information Rate) and `cbs` (Committed Burst Size) values for `ENNI` and `UNI` bandwidth profiles are introduced.
The Access E-Line product exists in Seller's inventory and is identified as `01494079-6c79-4a25-83f7-48284196d44d`.

The following requirements apply to `productOrderItem` when `action` is `modify`:

[RXXX] The modify request **MUST** specify a reference (provide `product.id`) to an existing product which is a subject of this order, provide `product.productConfiguration` and if set previously or to be set: `product.productRelationship` and `product.place`. [MEF57.2 R44]

[RXXX] If there is a relationship with another Product Order Item within the same Product Order, the `product.productRelationship` **MUST** be specified. [MEF57.2 R43]

[OXXX] The Buyer **MAY** include the `billingAccount`. [MEF57.2 O11]

[OXXX] The Seller **MAY** require that the `billingAccount` attributes be the same for all Product Order Items in a Product Order. [MEF57.2 O12]

There is no possibility to send an update to single attributes. The Buyer must send a full product description (the whole `product.productConfiguration` section and if set previously or to be set: `product.productRelationship` and `product.place`), that means all attributes that represent the desired state, even if some of them do not change.
If Seller does not allow for some of the attributes to change an appropriate error response (`422`) must be returned to the Buyer.

Please also note, that in the `add` case, a reference to the UNI product used the `productOrderItemRelationship` pointing to another `productOrderItem` in the same Product Order Request. This is because the UNI was not existing at that moment and was also a part of the order. In the case of ordering the update of an existing Access E-Line, the UNI is also existing and it must be referenced with the use of `productRelationship`. This example assumes that the UNI product is available in Seller's Inventory with the `id` equals `"00000000-0000-000a-0000-000000000098"`.

The references to `quoteItem` and `productOfferingQualificationItem`, if provided, would point to a different Quote and POQ than the ones provided in the `add` request, as for the `modify` case also the POQ and Quote have to be performed explicitly for the `modify` action.

```
{
  <<ProductOrder attributes...>>
  "productOrderItem": [
    {
      "id": "item-001",
      "action": "modify",
      ...
      "product": {
        "id" : "01494079-6c79-4a25-83f7-48284196d44d",
        "@type" : "MEFProductRefOrValueOrder",
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:order",
          "enniEp": {
            "ingressBandwidthProfilePerClassOfServiceName": [
                {
                    "classOfServiceName": "silver",
                    "bwpFlow": [
                        {
                            "envelopeRank": 1,
                            "couplingFlag": false,
                            "envelopeName": "defaultENNI",
                            "tokenRequestedOffset": 0,
                            "colorMode": "COLOR_BLIND",
                            "cir": {
                                "irValue": 40, << this value to be updated >>
                                "irUnits": "MBPS"
                            },
                            "cbs": {
                                "dataSizeValue": 100, << this value to be updated >>
                                "dataSizeUnits": "KBYTES"

                            },
                            "eir": {
                                "irValue": 0,
                                "irUnits": "BPS"
                            },
                            "ebs": {
                                "dataSizeValue": 0,
                                "dataSizeUnits": "BYTES"

                            },
                            "cirMax": {
                                "irValue": 40,  << this value to be updated >>
                                "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": 40, << this value to be updated >>
                                "irUnits": "MBPS"
                            },
                            "cbs": {
                                "dataSizeValue": 100, << this value to be updated >>
                                "dataSizeUnits": "KBYTES"

                            },
                            "eir": {
                                "irValue": 0,
                                "irUnits": "BPS"
                            },
                            "ebs": {
                                "dataSizeValue": 0,
                                "dataSizeUnits": "BYTES"

                            },
                            "cirMax": {
                                "irValue": 40, << this value to be updated >>
                                "irUnits": "MBPS"
                            },
                            "eirMax": {
                                "irValue": 0,
                                "irUnits": "BPS"
                            },
                        }
                    ]
                }
            ]
          }
        }, << lack of productOffering >>
        "productRelationship": [
          {
            "id": "00000000-6666-0000-0000-000000000001",
            "relationshipType": "ENNI_REFERENCE"
          },
          { << UNI referenced as existing product >>
            "relationshipType": "UNI_REFERENCE",
            "id": "00000000-0000-000a-0000-000000000098"
          }
        ]
      }, << lack of productOrderItemRelationship for UNI >>
      "productOfferingQualificationItem": { << POQ id different than in the add case >>
        "id": "poqItem-001",
        "productOfferingQualificationId": "00000000-2222-0000-0000-000000000002"
      },
      "quoteItem": { << Quote id different than in the add case >>
        "id": "quoteItem-001",
        "quoteId": "00000000-4444-0000-0000-000000000002"
      },
      "relatedContactInformation": [
        {
          "emailAddress": "Buyer.ProductOrderItemContact@example.com",
          "name": "Buyer Product Order Item Contact",
          "number": "+12-345-678-90",
          "role": "buyerProductOrderItemContact"
        },
        {
          "emailAddress": "Buyer.ImplementationContact@example.com",
          "name": "Buyer Implementation Contact",
          "number": "+12-345-678-90",
          "role": "buyerImplementationContact"
        },
        {
          "emailAddress": "Buyer.TechnicalContact@example.com",
          "name": "Buyer Technical Contact ",
          "number": "+12-345-678-90",
          "role": "buyerTechnicalContact "
        }
      ],
    }
  ]
}
```

6.1.8 Use case 1c: Product Order Item to Disconnect Existing Product

The example below represents a single Product Order request for deletion (`action` equals `delete`) of an existing Access E-Line product (type `urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:order`).

```
{
  <<ProductOrder attributes...>>
  "productOrderItem": [
    {
      "id": "item-001",
      "action": "delete",
      "product": {
        "id" : "01494079-6c79-4a25-83f7-48284196d44d"
      }
    }
  ]
}
```

The following requirements apply to `productOrderItem` when `action` is `delete`:

[RXXX] `product.id` **MUST** be provided. [MEF57.2 R49]

[OXXX] The Buyer **MAY** include the `billingAccount`. [MEF57.2 O13]

6.1.8 Product Order State Machine

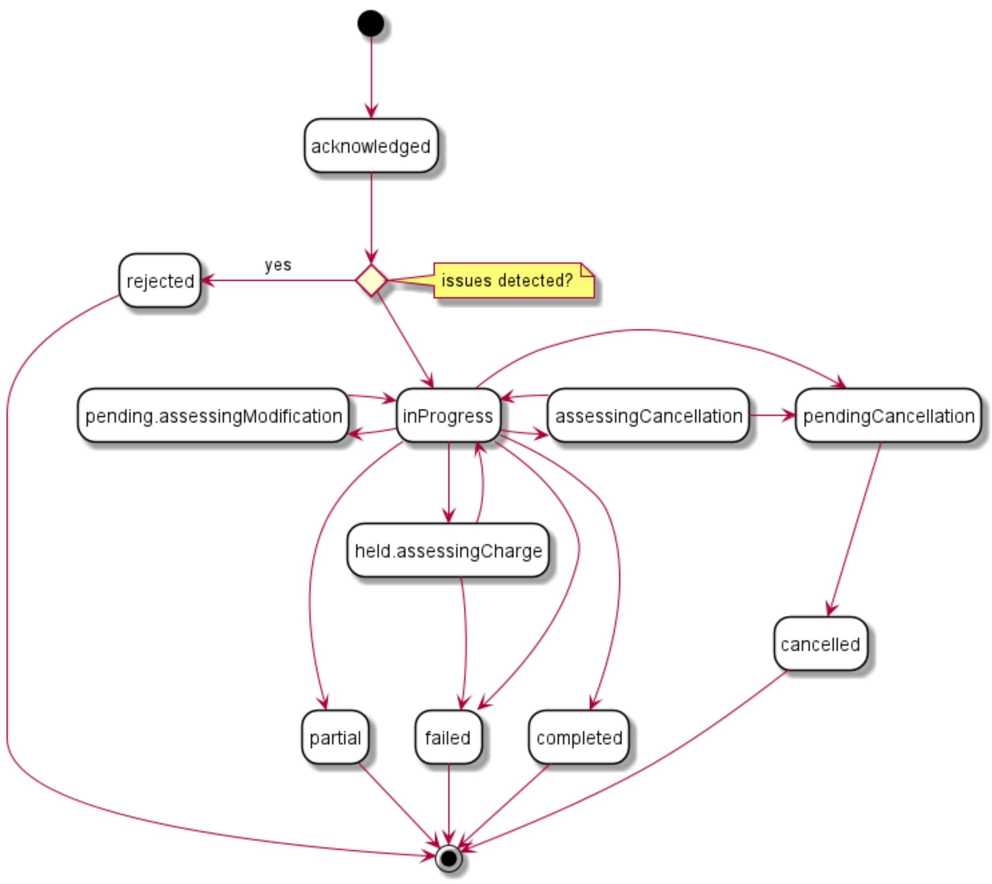


Figure 13: Product Order State Machine

Figure 13 presents the state machine for the Product Order. After receiving the request, the Seller performs basic checks of the message. If any problem is found an Error response is provided. If the validation passes a response is provided with `ProductOrder` and all `ProductOrderItems` in `acknowledged` state. Before moving the order to the `inProgress` state, the Buyer performs all the remaining business and time-consuming validations. At this point, an Error response cannot be provided anymore so the order moves to a `rejected` state if some issues are found. The `productOrderItem.terminationError` acts as a placeholder to provide a detailed description of what caused the problem.

Table 5 presents the mapping between the API `state` names (aligned with TMF) and the MEF 57.2 naming, together with states' description.

state	MEF 57.2 name	Description
`acknowledged`	ACKNOWLEDGED	A Product Order has been received by the Seller and has passed basic validation. A `productOrder.id` is assigned in the `acknowledged` state and a response is returned to the Buyer. The Product Order remains in the `acknowledged` state while validations of Product Order and Product Order Item(s) attributes as applicable is completed. If the Product Order and Product Order Item attributes are validated the Product Order moves to the `inProgress` state. If not validated, the Product Order moves to the `rejected` state.
`assessingCancellation`	ASSESSING_CANCELLATION	A request has been made by the Buyer to cancel the Product Order and the Product Order is currently being assessed to determine whether it can be cancelled. If there are any charges associated with the Buyer's Cancel Request, the Seller initiates a Charge which communicates the related charges to the Buyer, the Product Order remains in the `assessingCancellation` state until the Charge is completed or withdrawn by the Seller. Once the cancellation assessment is complete, the Product Order moves to the `pendingCancellation` state.

state	MEF 57.2 name	Description
`held.assessingCharge`	ASSESSING_CHARGE	A Charge has been initiated by the Seller that is not the result of a Modify Product Order Item or Cancel Product Order request and the Seller is awaiting a Buyer response to the Charge. If a blocking or non-blocking charge is accepted by the Buyer, the Product Order moves to `inProgress`. If a non-blocking charge is declined by the Buyer, the Product Order moves to `inProgress`. If a blocking charge is declined by the Buyer and there are no unrelated Product Order Items in the Product Order, the Product Order moves to the FAILED state. If a blocking charge is declined by the Buyer and there are unrelated Product Order Items in the Product Order, the Product Order moves to the `inProgress` state.
`pending.assessingModification`	ASSESSING_MODIFICATION	A request has been made by the Buyer to modify either the `expediteIndicator` or the `requestedCompletionDate` of a Product Order Item. The Product Order Item is currently being assessed to determine whether the Modify Product Order Item Completion Date is valid. If there is a charge associated with the Modify Product Order Item Completion Date, the Product Order remains in the `pending.assessingModification` state until the Charge is completed or withdrawn by the Seller. Once the Buyer's request has been validated and any associated Charges completed, the Product Order returns to the `inProgress` state.
`cancelled`	CANCELLED	The In-Flight Product Order has been successfully cancelled. This is a terminal state.
`pendingCancellation`	CANCELLING	The Buyer's Cancel Request has been assessed and it has been determined that it is feasible to proceed with the cancellation. This state can also result from a Seller cancelling the Product Order within their systems without a request from the Buyer.
`completed`	COMPLETED	The Product Order has completed fulfillment and the Product is now active. This is a terminal state

state	MEF 57.2 name	Description
`failed`	FAILED	All Product Order Items have failed which results in the entire Product Order failing. This is a terminal state.
`inProgress`	IN_PROGRESS	The Product Order has been successfully validated, and fulfillment has started.
`partial`	PARTIAL	Fulfillment of at least one Product Order Item has failed, and fulfillment of at least one Product Order Item has been successful. This is a terminal state.
`rejected`	REJECTED	A Product Order was submitted, and it has failed at least one of the validation checks the Seller performs after it reached the `acknowledged` state

Table 5: Product Order states

6.1.9 Product Order Item State Machine

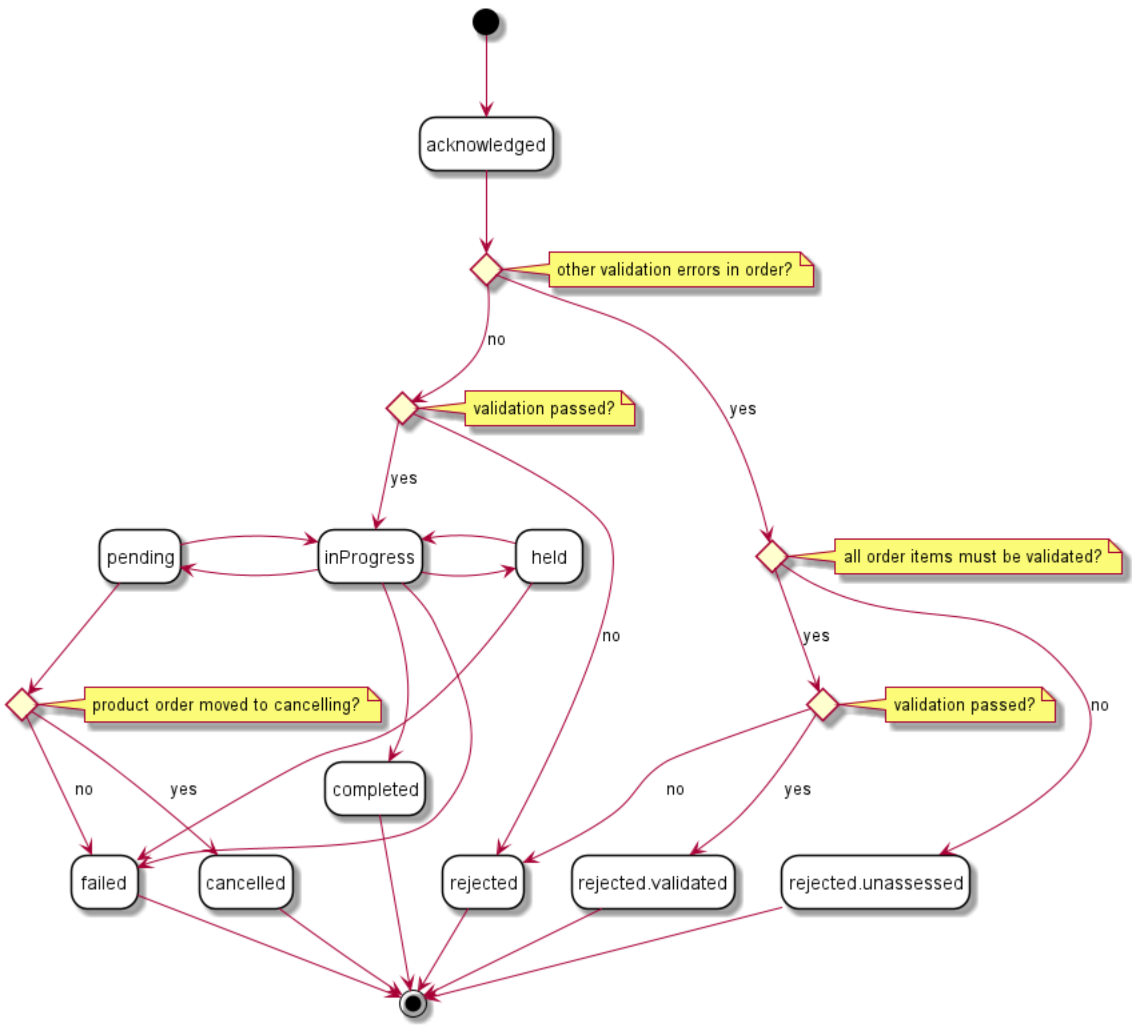


Figure 14: Product Order Item State Machine

Table 6 presents the mapping between the API `state` names (aligned with TMF) and the MEF 57.2 naming, together with the corresponding descriptions.

state	MEF 57.2 name	Description
`acknowledged`	ACKNOWLEDGED	A Product Order Item has been received and has passed basic business validations. From the `acknowledged` state the Product Order Item is further validated and depending on the results of the validation and if other Product Order Items in the Product Order are also validated the Product Order Item moves to `inProgress`, `rejected.validated`, or `rejected.unassessed`.
`cancelled`	CANCELLED	The In-Flight Product Order has moved to the `pendingCancellation` state. All Product Order Items move to `cancelled`.
`completed`	COMPLETED	The Product Order Item has completed provisioning. This is an end state
`failed`	FAILED	The fulfillment of a Product Order Item has failed. A Product Order Item may fail because the Buyer declined a Blocking charge identified via the Charge, the Buyer failed to respond to a Charge Item included in a Charge, or the Seller is unable to fulfill the Product Order Item. A Product Order Item moving to `failed` state results in the Product Order State being `failed` or `partial`. This is a terminal state.
`held`	HELD	The Product Order Item cannot be progressed due to Charge the Seller awaiting a response from the Buyer on a Charge. The Seller stops work on the Product Order Item until the Charge has completed. Upon acceptance by the Buyer of all Blocking charges, the Product Order Item returns to `inProgress` state If the Buyer rejects a Blocking charge, the Product Order Item moves to the `failed` state.
`inProgress`	IN_PROGRESS	The Product Order Item has been successfully validated and fulfillment has started. If the Seller's system links validation between Product Order Items in a Product Order, a Product Order Item in this state also indicates that the other Product Order Items passed validation.
`pending`	PENDING	The Product Order Item cannot be progressed due to Charge the Seller assessing a Cancel Product Order or Modify Product Order Item Completion Date request. The Seller stops work on the Product Order Item until either the Cancel Product Order has been accepted and the Product Order state moves to `pendingCancellation` and the Product Order Item state moves to `cancelled`, the Cancel Product Order has been rejected and the Product Order Item State moves to `inProgress`, the Modify Product Order Item Completion Date has been accepted and the Product Order Item State moves to `inProgress`, or the Modify Product Order Item Completion Date moves to `done.declined` and the Product Order Item state moves to `failed`. Charge
`rejected`	REJECTED	A Product Order Item was submitted, and it has failed at least one validation checks the Seller performs during the `acknowledged` state.

state	MEF 57.2 name	Description
`rejected.unassessed`	UNASSESSED	A Product Order was submitted and all validation checks the Seller performs during the `acknowledged` state have not been completed, but another Product Order Item in the Product Order has moved to the `rejected` state.
`rejected.validated`	VALIDATED	A Product Order was submitted, and it has passed all validation checks the Seller performs during the `acknowledged` state, but another Product Order Item in the Product Order has moved to the `rejected` state

Table 6: Product Order Item states

6.1.10 Requirements for Product Order and Product Order Item Lifecycle

Requirements below are applied to a Product Order processing lifecycle - after providing an initial response where the Product Order was `acknowledged`. It assumes a Seller's response to a GET by `id` request.

[RXXX] If the Product Order `state` in the Seller's response is `cancelled`, the `expectedCompletionDate` attribute **MUST NOT** be provided. [MEF57.2 R11]

[RXXX] If the Product Order `state` in the Seller's response is `completed`, the response **MUST** contain the `completionDate` attribute. [MEF57.2 R12]

[RXXX] The Seller **MUST** increment the `orderVersion` by 1 (one) each time a PATCH Request is accepted for this `ProductOrder`. [MEF57.2 R13]

[OXXX] The Seller **MAY** add a Note to any Product Order. [MEF57.2 R8]

[RXXX] If the Product Order Item `state` in the Seller's response is `inProgress`, the `expectedCompletionDate` attribute **MUST** be provided. [MEF57.2 R38], [MEF57.2 R48], [MEF57.2 R52]

[RXXX] If the Product Order Item `state` in the Seller's response is `cancelled`, the `expectedCompletionDate` attribute **MUST NOT** be provided. [MEF57.2 R39], [MEF57.2 R52]

[RXXX] If the Product Order Item `state` in the Seller's response is `completed`, the response **MUST** contain the `completionDate` attribute. [MEF57.2 R40], [MEF57.2 R54]

[RXXX] If the Product Order Item `state` in the Seller's response is not `completed`, the response **MUST NOT** contain the `completionDate` attribute. [MEF57.2 R41], [MEF57.2 R55]

[RXXX] If the Seller revises the `expectedCompletionDate` for any Product Order Item, they **MUST** include a `note` that indicates that the date has been revised and the reason for the revision. [MEF57.2 R42]

6.1.11. Specifying Place Details

Some product specifications may define requirements concerning place definition in case `add` or `modify` action is used. For example, an Operator UNI product specification requires an `INSTALL_LOCATION` place definition in the case of the `add` action.

There are different formats in which place information may be provided: geographic point (`MEFGeographicPoint`), fielded (`FieldedAddress`), formatted (`FormattedAddress`), geographic address identifier (`GeographicAddressLabel`), geographic site reference (`GeographicSiteRef`), and a geographic address reference (`GeographicAddressRef`). The first four of them can be used to provide a full place description by value. The site and address reference allow specifying the place information as a reference to previously validated address or site available through Seller's Addressing and Site API endpoints, which definition is provided in the SDK:

- `productApi/serviceability/address/geographicAddressManagement.api.yaml`
- `productApi/serviceability/site/geographicSiteManagement.api.yaml`

The master class for all address types is the `RelatedPlaceRefOrValue` which adds the `role` to add more context to the specified address. To distinguish between place types the `@type` discriminator is used.

Note: The *RefOrValue* stands for a pattern where an address can be provided either by `id` (using `GeographicSiteRef` or `GeographicAddressRef`) OR by value (with use of `MEFGeographicPoint`, `FieldedAddress`, `FormattedAddress`, `GeographicAddressLabel`). There is no way to specify an address with use both ref AND value at the same time.

Examples of different place specification formats are provided below.

6.1.11.1. Fielded Address

```
{
  "@type": "FieldedAddress",
  "streetType": "ul.",
  "streetName": "Edmunda Wasilewskiego",
  "streetNr": "20",
  "streetNrSuffix": "14",
  "city": "Kraków",
  "stateOrProvince": "Lesser Poland",
  "postcode": "30-305",
  "country": "Poland",
  "geographicSubAddress": {
    "levelType": "floor",
    "levelNumber": "4"
  },
  "role": "INSTALL_LOCATION"
}
```

Fielded address example of a place specification. The type discriminator has the value `FieldedAddress`. A subset of available attributes is used to describe the place. The fielded address has an optional `geographicSubAddress` structure that defines several attributes that can be used in case precise address information has to be provided. In the example above, a floor in the building at the given address is specified using this structure. The role of the place is assigned according to the requirements of the Operator UNI product specification.

6.1.11.2. Formatted Address

```
{
  "@type": "FormattedAddress",
  "addrLine1": "ul. Edmunda Wasilewskiego 20/14",
  "addrLine2": "Floor 4",
  "city": "Kraków",
  "stateOrProvince": "Lesser Poland",
  "postcode": "30-305",
  "country": "Poland",
  "role": "INSTALL_LOCATION"
}
```

Place information in a form of a formatted address. The type discriminator has the value `FormattedAddress`. This example contains the same information as the previous `FieldedAddress` example.

6.1.11.3. Geographic Point

```
{
  "@type": "MEFGeographicPoint",
  "spatialRef": "EPSG:4326 WGS 84",
  "x": "50.048868",
  "y": "19.929523",
  "role": "INSTALL_LOCATION"
}
```

Place information in a form of geographic point. `spatialRef` determines the standard that has to be used to interpret coordinates provided in the required `x` (latitude), `y` (longitude), and optional `z` (elevation) values.

This type allows only providing a point. It cannot carry more detailed information like the floor number from previous examples.

[RXXX] The `spatialRef` value that can be used **MUST** be agreed between Buyer and Seller.

6.1.11.4. Geographic Address Label

```
{
  "@type": "GeographicAddressLabel",
  "externalReferenceType": "CLLI",
  "externalReferenceId": "PLTXCL01",
  "role": "INSTALL_LOCATION"
}
```

The Geographic Address Label represents a unique identifier controlled by a generally accepted independent administrative authority that specifies a fixed geographical location. The example above is a place that represents a CLLI (Common Language Location Identifier) identifier which is commonly used to refer locations in North America for network equipment installations.

6.1.11.5. Geographic Site Reference

```
{
  "@type": "GeographicSiteRef",
  "id": "18d3bb74-997a-4a62-8198-84250766765a",
  "role": "INSTALL_LOCATION"
}
```

`GeographicSiteRef` type is used to specify a `GeographicSite` by reference in the request. In the above example, a `GeographicSite` identified as `18d3bb74-997a-4a62-8198-84250766765a` in the Sellers Service Site API is used.

6.1.11.6. Geographic Address Reference

```
{
  "@type": "GeographicAddressRef",
  "id": "8198bb74-18d3-9ef0-4913-66765a842507",
  "role": "INSTALL_LOCATION"
}
```

`GeographicAddressRef` type is used to specify a `GeographicAddress` by reference in the request. In the above example a `GeographicAddress` identified as `8198bb74-18d3-9ef0-4913-66765a842507` in the Sellers Service Site API is used.

6.2. Use Case 2: Update Product Order

The update operation is realized with the use of the REST PATCH operation. For that purpose a specialized types `ProductOrder_Update` and `ProductOrderItem_Update` are provided. Their lists of attributes are limited to a subset that includes only the Buyer settable and not Product Order processing affecting attributes.

The PATCH usage recommendation follows TMF 622 json/merge (https://tools.ietf.org/html/rfc7386).

Figure 15 presents the model used in the PATCH request. The Seller responds with a `ProductOrder` type.

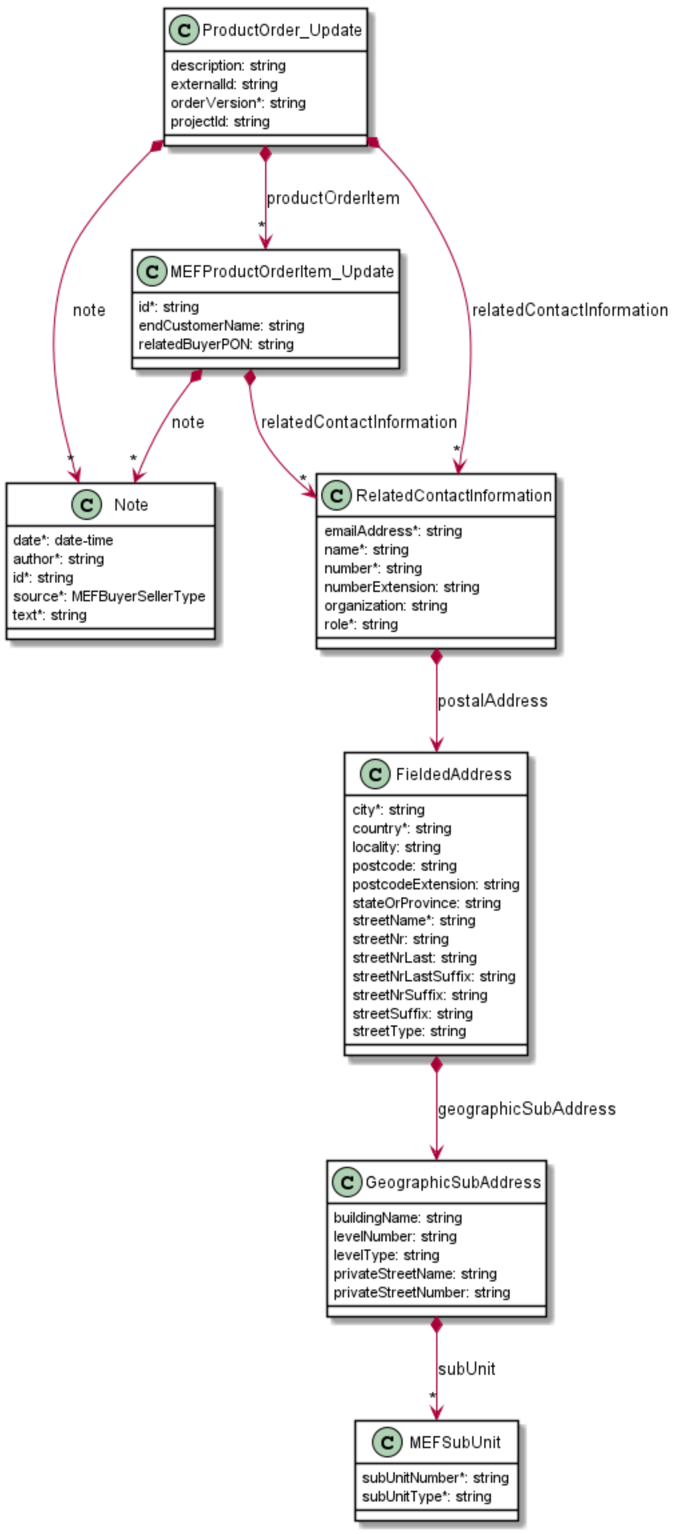


Figure 15: Patch request Model

The example below shows a request to change the `description`, the Product Order Contact (`relatedContactInformation` with `role` set to `productOrderContact`), and the `endCustomerName` of the first Product Order item.

```
{
  "orderVersion": "1", << version must match the current Product Order version >>
  "description": "An updated description",
  "relatedContactInformation": [
    { << updated contact >>
      "emailAddress": "Richard.example@example.com",
      "name": "Richard Example",
      "number": "98-765-4321",
      "organization": "Buyer Example Co.",
      "role": "productOrderContact",
    },
    { << not changed >>
      "emailAddress": "kate.example@example.com",
      "name": "Kate Example",
      "number": "12-345-67890",
      "organization": "Seller Example Co.",
      "role": "sellerContact"
    }
  ],
  "productOrderItem": [
    {
      "id": "item-001",
      "endCustomerName": "Updated End Customer Name"
    },
    {
      "id": "item-002"
    }
  ]
}
```

Note: The `productOrderItem.id` attribute cannot be updated. It is used only to refer to identify and items to be updated.

Note: The `orderVersion` attribute cannot be updated. It is used only to identify the version of the Product Order that the Buyer wants to update. If there is a mismatch with the Seller's system, the Seller will reject the request with an error response.

[RXXX] A Buyer's PATCH request **MUST** contain one or more of the `ProductOrder` updateable attributes (apart from `orderVersion`). [MEF57.2 R57]

[RXXX] If a Buyer's PATCH request contains a Product Order Item, it **MUST** provide one or more of the Product Order Item's updateable attributes (apart from `id`). [MEF57.2 R61]

The Buyer can update a Buyer-related contact by providing a full list of existing `relatedContactInformation` items, and updating the value of the one with given `role`. The `role` acts as a key in the contacts list.

The Buyer can update a Buyer-related note by providing a full list of existing `note` items, and updating the value of the one with the given `id`. The `id` acts as a key in the notes list.

6.3. Use Case 3: Retrieve List of Product Orders

The Buyer can retrieve a list of `ProductOrders` by using a `GET /productOrder` operation with desired filtering criteria. The attributes that are available to be used are:

- `state`
- `externalId`
- `projectId`
- `orderDate.gt`
- `orderDate.lt`
- `completionDate.gt`
- `completionDate.lt`
- `requestedCompletionDate.gt`

- `requestedCompletionDate.lt`
- `expectedCompletionDate.gt`
- `expectedCompletionDate.lt`
- `orderCancellationDate.gt`
- `orderCancellationDate.lt`

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 RFC3986. Section 7.1.2. provides details about the implementation of pagination mechanism.

```
https://serverRoot/mefApi/sonata/productOrderManagement/v7/productOrder?state=completed&projectId=myProject
```

The example above shows a Buyer's request to get all `ProductOrders` that are in the `completed` state and are part of `myProject`. The correct response (HTTP code `200`) in the response body contains a list of `ProductOrder_Find` objects matching the criteria. To get more details (e.g. the item level information), the Buyer has to query a specific `ProductOrder` by `id`.

[RXXX] The Seller **MUST** put the following attributes into the `ProductOrder_Find` object in the response: [MEF57.2 R83]:

- `id`
- `cancellationDate`
- `completionDate`
- `requestedCompletionDate`
- `externalId`
- `orderDate`
- `orderVersion`
- `projectId`
- `state`

[RXXX] In case no items matching the criteria are found, the Seller **MUST** return a valid response with an empty list.

6.4. Use Case 4: Retrieve Product Order by Product Order Identifier

The Buyer can get detailed information about the Product Order from the Seller by using a `GET /productOrder/{{id}}` operation. In case `id` does not allow to find a `ProductOrder` in Seller's system, an error response `Error404` must be returned. The payload returned in the response includes all attributes the Buyer has provided while sending a Product Order create request. The attributes provided by the Seller depend on the status of the `ProductOrder` and may require some time to be set.

[RXXX] The Seller's response **MUST** comply with the states and attributes detailed in Table 7 and Table 8. [MEF57.2 R86]

Please note that for readability purposes following tables do not show attributes specified by the Buyer that must be only echoed back ("E") by the Seller without any change. Attributes required to be provided by the Seller are shown by an "R", Required if Populated by the Seller shown by a "PR", or Optional to be provided by the Seller or the Buyer shown by an "O".

	acknowledged	assessingCancellation	held.assessingCharge	cancelled	pendingCa
id	R	R	R	R	R

	acknowledged	assessingCancellation	held.assessingCharge	cancelled	pendingCa
orderVersion	R	R	R	R	R
orderDate	R	R	R	R	R
state	R	R	R	R	R
relatedContactInformation	R	R	R	R	R
cancellationReason				E - Buyer / R - Seller	
cancellationDate				R	
expectedCompletionDate		R	R	O	R
completionDate				R	
note	E - Buyer / PR - Seller	E - Buyer / PR - Seller	E - Buyer / PR - Seller	E - Buyer / PR - Seller	E - Buyer / Seller

Table 7. Seller Response Product Order Attributes Based on Product Order State

	acknowledged	cancelled	completed	failed	held	inProgress	pending	rejecte
sellerItemIdentifier	R	R	R	R	R	R	R	R
note	E - Buyer / PR - Seller	E - Buyer / PR - Seller	E - Buyer / PR - Seller	E - Buyer / PR - Seller	E - Buyer / PR - Seller	E - Buyer / PR - Seller	E - Buyer / PR - Seller	E - Buyer PR - Seller
expediteAcceptedIndicator	PR	PR	PR	PR	PR	PR	PR	PR
charge		PR	PR	PR	PR	PR	PR	
stateChange	R	R	R	R	R	R	R	R
expectedCompletionDate		R	R	R	R	R	R	
completionDate			R					
state	R	R	R	R	R	R	R	R
requestedItemTerm	E - Buyer	E - Buyer	E - Buyer	E - Buyer	E - Buyer	E - Buyer	E - Buyer	E
itemTerm	PR - Seller	PR - Seller	PR - Seller	PR - Seller	PR - Seller	PR - Seller	PR - Seller	PR - Seller
terminationError				R				R

Table 8. Seller Response Product Order Item Attributes Based on Product Order Item State

6.5. Use case 5: Modify Product Order Item Completion Date

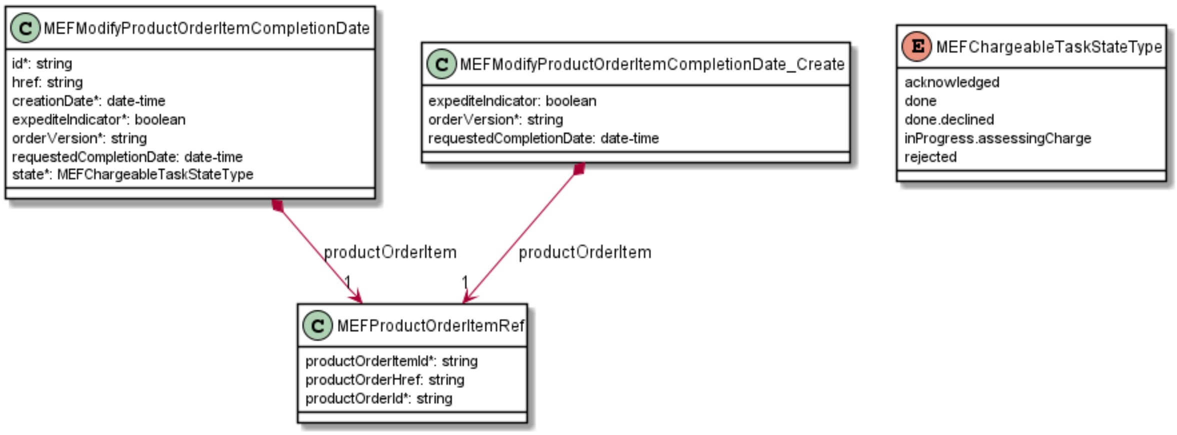
The Product Order PATCH operation is limited to a subset of attributes that includes only the Buyer settable and not Product Order processing affecting ones (Section 6.2). Modification of `requestedCompletionDate` or `expediteIndicator` may bring a significant processing and business impact hence it is extracted to a separate dedicated process.

The Buyer may issue the request by using a dedicated endpoint: `POST /modifyProductOrderItemCompletionDate` and providing a `MEFModifyProductOrderItemCompletionDate_Create` in the request body.

There are two functions supported by the Modify Product Order Item Completion Date request:

- changing the `expediteIndicator`
- changing the `requestedCompletionDate` of the Product Order Item.

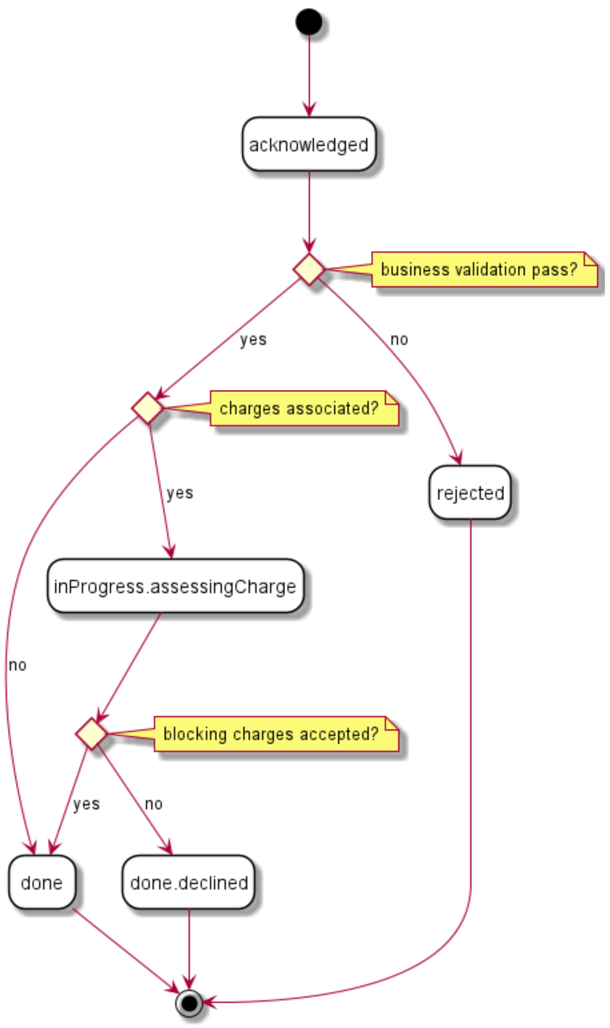
Figure 16 presents entity types that take part in the Modify Product Order Item Completion Date use cases:



MEFModifyProductOrderItemCompletionDate
id*: string
href: string
creationDate*: date-time
expediteIndicator*: boolean
orderVersion*: string
requestedCompletionDate: date-time
state*: MEFChargeableTaskStateType
MEFModifyProductOrderItemCompletionDate_Create
expediteIndicator: boolean
orderVersion*: string
requestedCompletionDate: date-time
MEFChargeableTaskStateType
acknowledged
done
done.declined
inProgress.assessingCharge
rejected
productOrderItem
1
productOrderItem
1
MEFProductOrderItemRef
productOrderItemId*: string
productOrderHref: string
productOrderId*: string

Figure 16: Modify Product Order Item Completion Date Model

The state transition and detailed description are presented in Figure 17 and Table 9:



acknowledged
business validation pass?
yes
no
charges associated?
yes
rejected
inProgress.assessingCharge
no
blocking charges accepted?
yes
no
done
done.declined

Figure 17: Modify Product Order Item Completion Date State Machine

Name	MEF 57.2 Name	Description
inProgress.assessingCharge	ACCESSING_CHARGE	The Modify Product Order Item Completion Date request results in a Charge being initiated by the Seller. The Modify Product Order Item Completion Date remains in this state until the Charge is completed or withdrawn by the Seller. All charges within a Charge that was initiated due to a Modify Product Order Item Completion Date are considered Blocking charges. If any charge is not accepted by the Buyer, the Modify Product Order Item Completion Date moves from the inProgress.assessingCharge state to the `done.declined` state.
acknowledged	ACKNOWLEDGED	A Modify Product Order Item Completion Date request has been received and has passed basic validation. The Modify Product Order Item Completion Date Identifier is assigned in the `acknowledged` state. Validation of Modify Product Order Item Completion Date attributes as applicable is completed in the `acknowledged` state.
done	COMPLETED	A Modify Product Order Item Completion Date request has been received, passed all validations, if a Charge is associated all Charge Items have been accepted by the Buyer, and the Product Order Item Completion Date has been updated as requested.
done.declined	DECLINED	Blocking charges associated with a Modify Product Order Item Completion Date have been declined by the Buyer. No updates are made to the Product Order Item.
rejected	REJECTED	A Modify Product Order Item Completion Date request was submitted by the Buyer, and it has failed any validation checks the Seller performs during the `acknowledged` state. No updates are made to the referenced Product Order Item.

Table 9. Modify Product Order Item Completion Date States

Example of a Buyer's request (`ModifyProductOrderItemCompletionDate_Create`):

```
{
  "expediteIndicator": true,
  "orderVersion": "2",
  "requestedCompletionDate": "2021-05-25T21:32:28.826Z",
  "productOrderItem": {
    "id": "00000000-1111-2222-3333-000000000123",
    "productOrderId": "item-001"
  }
}
```

Example of a Seller's response (`ModifyProductOrderItemCompletionDate`):

```
{
  "id": "00000000-8888-0000-0000-000000000001",
  "expediteIndicator": true,
  "orderVersion": "2",
  "requestedCompletionDate": "2021-05-25T21:32:28.826Z",
  "productOrderItem": {
    "id": "00000000-1111-2222-3333-000000000123",
    "productOrderId": "item-001"
  },
  "state": "acknowledged"
}
```

Below you can find a flow of this use case when there are no additional charges identified. A case with additional charges handling is presented in Section 6.11.2

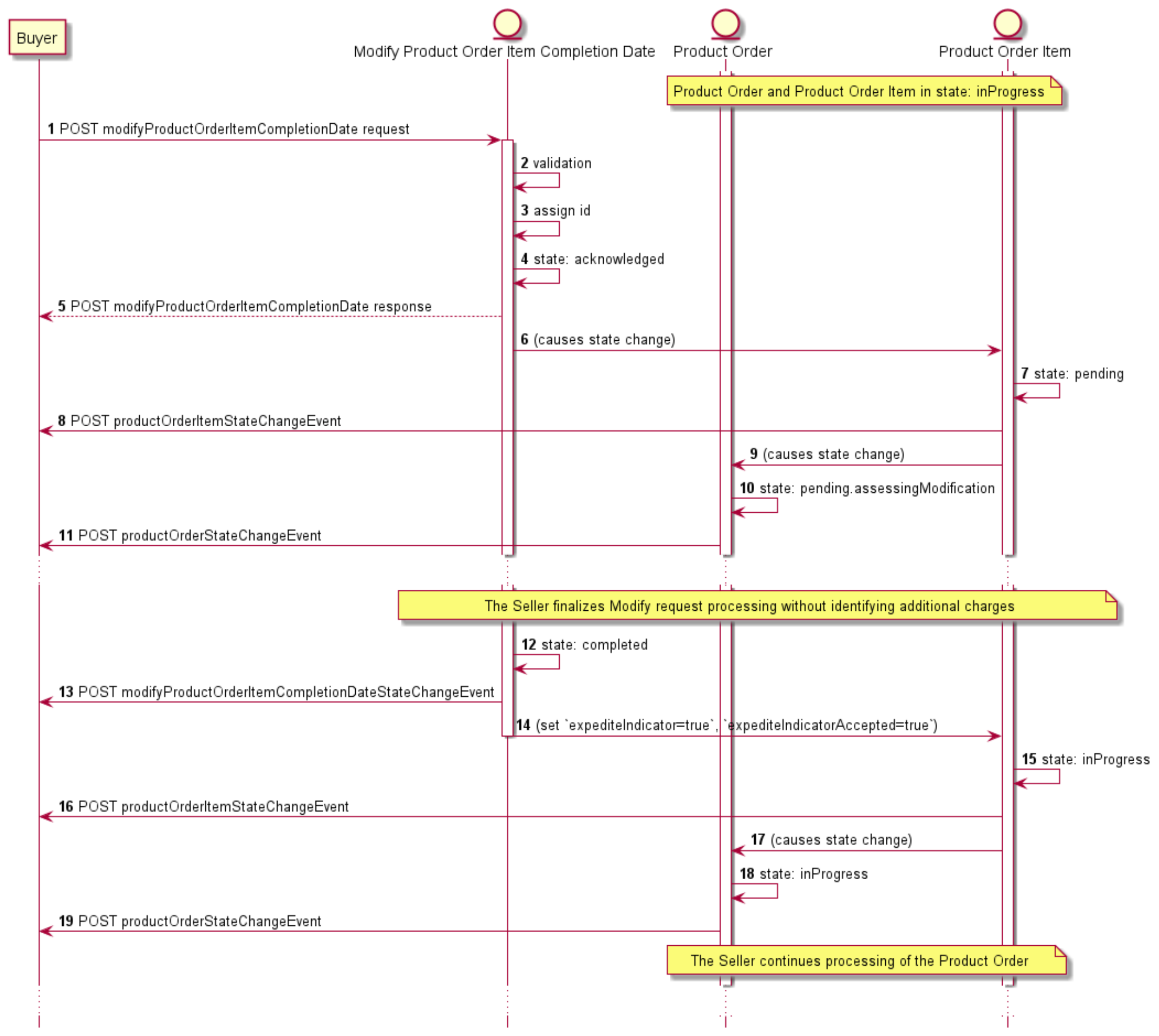

Buyer
Modify Product Order Item Completion Date
Product Order
Product Order Item
Product Order and Product Order Item in state: inProgress
1 POST modifyProductOrderItemCompletionDate request
2 validation
3 assign id
4 state: acknowledged
5 POST modifyProductOrderItemCompletionDate response
6 (causes state change)
7 state: pending
8 POST productOrderItemStateChangeEvent
9 (causes state change)
10 state: pending.assessingModification
11 POST productOrderStateChangeEvent
The Seller finalizes Modify request processing without identifying additional charges
12 state: completed
13 POST modifyProductOrderItemCompletionDateStateChangeEvent
14 (set `expediteIndicator=true`, `expediteIndicatorAccepted=true`)
15 state: inProgress
16 POST productOrderItemStateChangeEvent
17 (causes state change)
18 state: inProgress
19 POST productOrderStateChangeEvent
The Seller continues processing of the Product Order

Figure 18: Modify Product Order Item Completion Date Flow

- The Buyer sends a `modifyProductOrderItemCompletionDate` request with the `expediteIndicator` set to `true` (and/or `requestedCompletionDate`) set to new value.
- The Seller validates the request.
- The Seller initiates the Modify Date process, sets the `modifyProductOrderItemCompletionDate.state` to `acknowledged`, then assigns a unique `id` and changes the `state` of the referenced `ProductOrderItem` to `pending` and `ProductOrder` to `pending.assessingModification`.
- The Seller notifies the Buyer of any charges resulting from the request while the `ModifyProductOrderItemCompletionDate` is in the `acknowledged` state (see Section 6.11.2 for details).

- The Seller accepts the requested change. The `ModifyProductOrderItemCompletionDate` is set to `completed` and the Seller updates the `expediteIndicator` and the `expediteAcceptedIndicator` (and/or `requestedCompletionDate`).
- The Seller sets the referenced `ProductOrderItem.state` back to `inProgress` and `ProductOrder.state` to `inProgress`.
- The Seller continues their work to fulfill the Product Order.

6.5.1. Use case 5a: Modify Product Order Item Completion Date (Expedite Indicator) Request

In this case, the Buyer requests to expedite an in-flight Product Order. The

[RXXX] The Buyer's sent `ModifyProductOrderItemCompletionDate_Create` **MUST** contain the following attributes: [MEF57.2 R64]

- `expediteIndicator`
- `orderVersion`
- `productOrderItem`

Buyer sets the `expediteIndicator` to `true` if they want the Seller to fulfill the Product Order Item in a shorter period than the `installationInterval` (provided in product offering qualification and/or quote step).

[RXXX] The Buyer's sent `ModifyProductOrderItemCompletionDate_Create` **MAY** contain the `requestedCompletionDate`. [MEF57.2 O15]

If the Buyer sets the `expediteIndicator` to `true` and sets a `requestedCompletionDate` the they are requesting that the Product Order Item be fulfilled in a shorter time period than the `installationInterval` and have provided a date they would like it fulfilled by. The `requestedCompletionDate` must indicate a shorter time period than the `installationInterval`. The Seller may try to honor the date or may ignore it.

6.5.2. Use case 5b: Modify Product Order Item Completion Date (Requested Completion Date) Request

In this case, the Buyer requests to change the `expectedCompletionDate` of an in-flight Product Order.

[RXXX] The Buyer's sent `ModifyProductOrderItemCompletionDate_Create` **MUST** contain the following attributes: [MEF57.2 R68]

- `orderVersion`
- `productOrderItem`
- `requestedCompletionDate`

If the Buyer wants to push out or delay fulfillment of the Product Order Item, they set a new `requestedCompletionDate` and the `expediteIndicator` to `false` (or just not specify it all as the default value for `expediteIndicator` is `false`).

6.6. Use case 6: Retrieve Modify Product Order Item Completion Date List

The Buyer can retrieve a list of `ModifyProductOrderItemCompletionDate` by using a `GET /modifyProductOrderItemCompletionDate` operation with desired filtering criteria. The attributes that are avaible to be used are: [MEF57.2 O18]:

- `state`

- `expediteIndicator`
- `productOrderId`
- `requestedCompletionDate.gt`
- `requestedCompletionDate.lt`

The rules of using pagination and an example request are provided in section 6.3. Please refer to it as the rules also apply to this case.

[RXXX] The Seller must put the following attributes into the response: [MEF57.2 R87]:

- `id`
- `expediteIndicator`
- `orderVersion`
- `requestedCompletionDate`
- `orderDate`
- `state`
- `creationdate`

[RXXX] In case no items matching the criteria are found, the Seller **MUST** return a valid response with an empty list. [MEF57.2 R88]

6.7. Use case 7: Retrieve Modify Product Order Item Completion Date by Identifier

The Buyer can get detailed information about the Modify Product Order Item Completion Date from the Seller by using a `GET /modifyProductOrderItemCompletionDate/{{id}}` operation.

[RXXX] In case `id` does not allow to find a `ModifyProductOrderItemCompletionDate` in Seller's Inventory, an error response `404` must be returned. [MEF57.2 R92]

6.8. Use case 8: Cancel In-Flight Product Order

The Buyer may request to Cancel an in-flight Product Order by using `POST /cancelProductOrder` and providing a `CancelProductOrder_Create` in the request body.

The following Figures present the use case's model and flow diagrams.

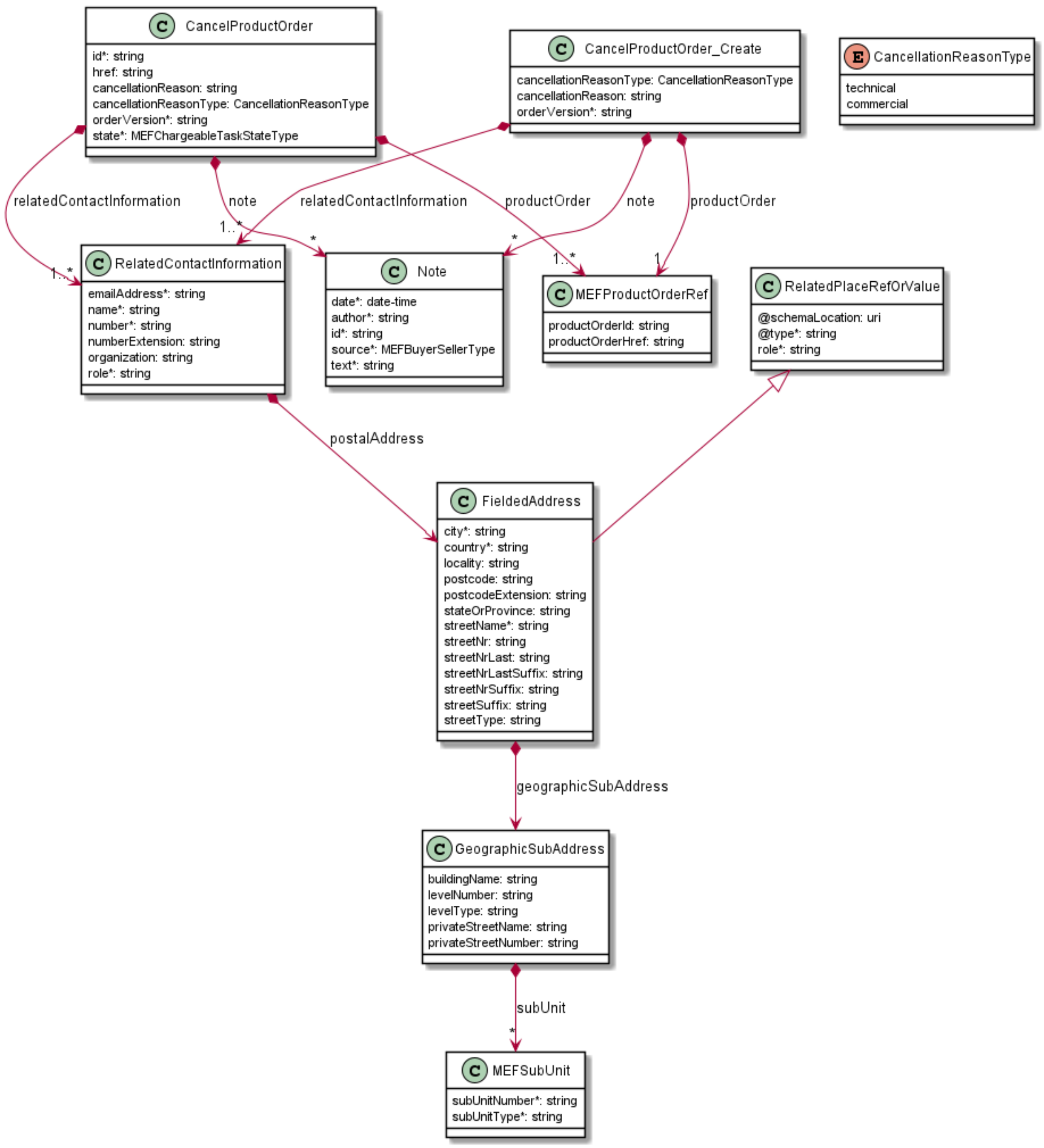


Figure 19: Cancel Product Order Model

The state transition and detailed description are presented in Figure 20 and Table 10:

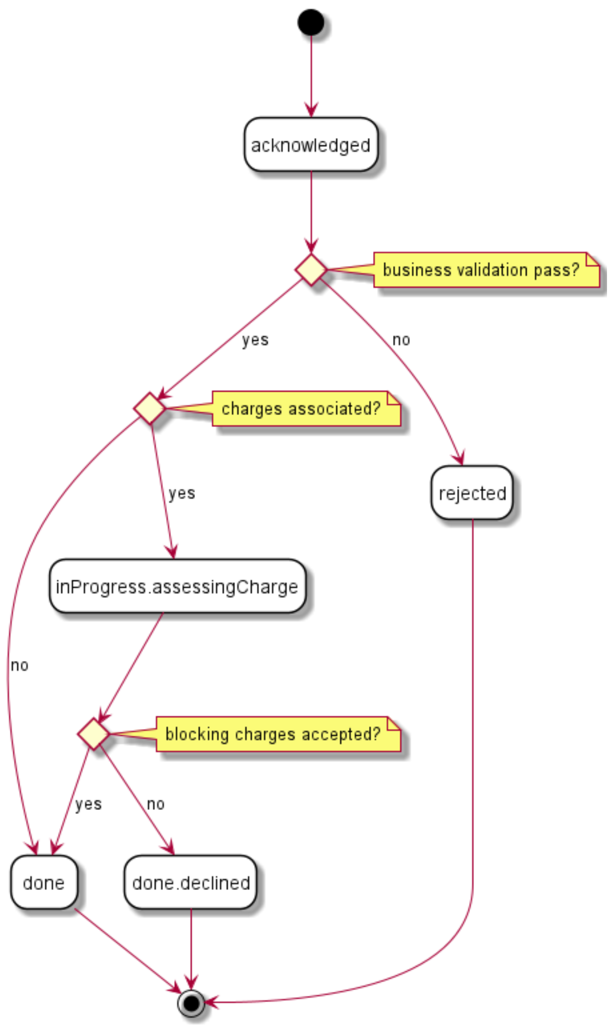

acknowledged
business validation pass?
yes
no
charges associated?
yes
rejected
inProgress.assessingCharge
no
blocking charges accepted?
yes
no
done
done.declined

Figure 20: Cancel Product OrderState Machine

Name	MEF 57.2 Name	Description
`acknowledged`	ACKNOWLEDGED	A Cancel Request has been received and has passed basic validation. Seller `id` is assigned in the `acknowledged` state. Validation of Cancel attributes as applicable is completed in the `acknowledged` state.
`inProgress.assessingCharge`	ACCESSING_CHARGE	The Cancel Request results in a Charge being initiated by the Seller. The Cancel Request remains in this state until the Charge is completed or withdrawn by the Seller.
`done`	COMPLETED	A Cancel Request has been received, passed all validations, if a Charge is associated all Charge Items have been accepted by the Buyer, and the Product Order has been cancelled as requested.
`done.declined`	DECLINED	Blocking charges associated with a Cancel Product Order have been declined by the Buyer. No updates are made to the Product Order.

Name	MEF 57.2 Name	Description
`rejected`	REJECTED	A Cancel Request was submitted, and it has failed any validation checks the Seller performs during the `acknowledged` state e.g. the Product Order being in an incorrect state. No updates are made to the referenced Product Order.

Table 10. Cancel Product Order States

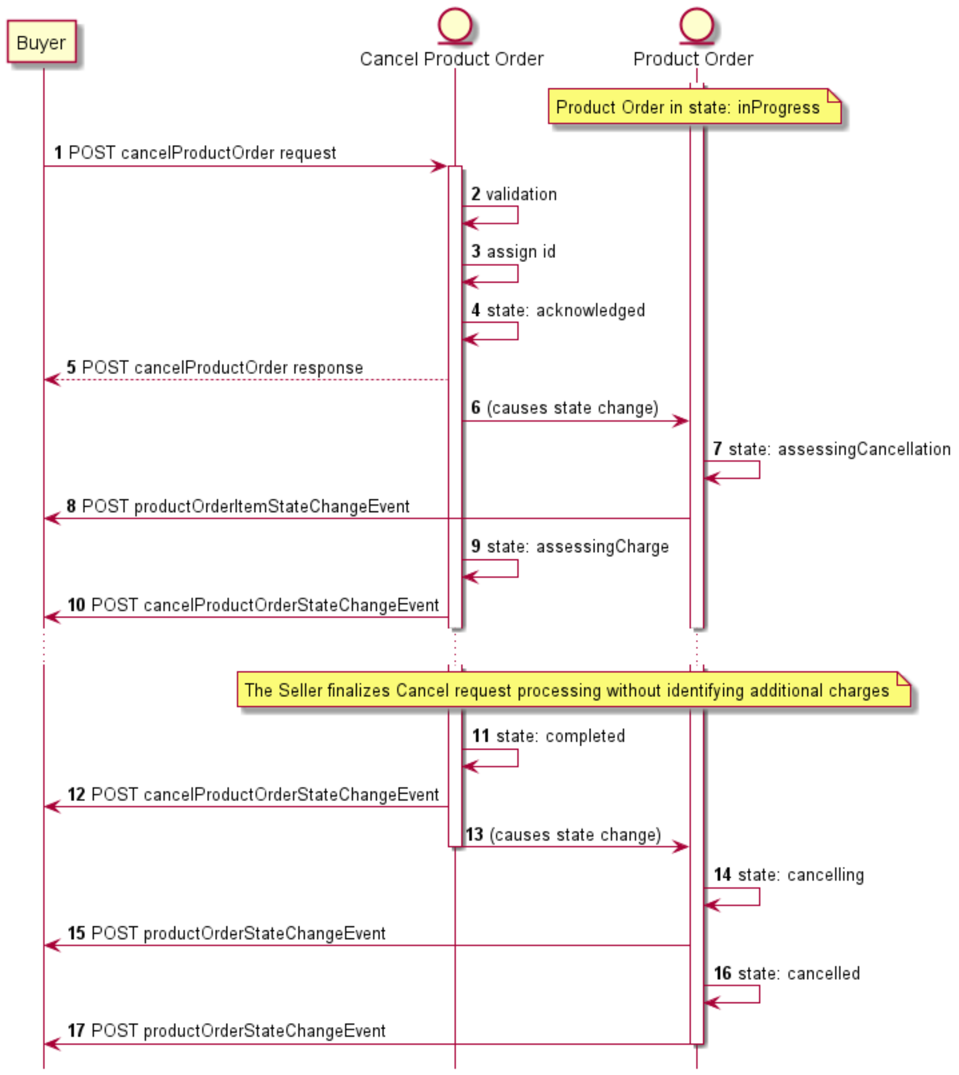
Example of a Buyer's request (`CancelProductOrder_Create`):

```
{
  "cancellationReasonType": "technical",
  "cancellationReason": "A technical reason for cancelling the ProductOrder",
  "orderVersion": "2",
  "note": [
    {
      "date": "2021-05-22T23:30:47.999Z",
      "author": "Cancel Product Order Contact",
      "id": "1",
      "source": "buyer",
      "text": "We have an equipment swap and the requirements will change. Will issue another Product Order
once done."
    }
  ],
  "relatedContactInformation": [
    {
      "emailAddress": "Cancel.ProductOrderContact@example.com",
      "name": "Cancel Product Order Contact",
      "number": "+12-345-678-90",
      "organization": "Buyer",
      "role": "cancelProductOrderContact"
    }
  ],
  "productOrder": {
    "id": "00000000-1111-2222-3333-000000000123"
  }
}
```

Example of a Seller's response (`CancelProductOrder`):

```
{
  "id": "00000000-9999-0000-0000-000000000003",
  "state": "acknowledged",
  "cancellationReasonType": "technical",
  "cancellationReason": "A technical reason for cancelling the ProductOrder",
  "orderVersion": "2",
  "note": [
    {
      "date": "2021-05-22T23:30:47.999Z",
      "author": "Cancel Product Order Contact",
      "id": "1",
      "source": "buyer",
      "text": "We have an equipment swap and the requirements will change. Will issue another Product Order
once done."
    }
  ],
  "relatedContactInformation": [
    {
      "emailAddress": "Cancel.ProductOrderContact@example.com",
      "name": "Cancel Product Order Contact",
      "number": "+12-345-678-90",
      "organization": "Buyer",
      "role": "cancelProductOrderContact"
    },
    {
      "emailAddress": "Seller.Contact@example.com",
      "name": "Seller Contact",
      "number": "+12-345-678-90",
      "organization": "Seller",
      "role": "sellerContact"
    }
  ],
  "productOrder": {
    "id": "00000000-1111-2222-3333-000000000123"
  }
}
```

Note: In the response, the `orderVersion` is only and always echoed value from the Buyer's request. There will be no increments even if in the background the Product ORder will change its state to `assessingCancellation` and `cancelled`.



Buyer
Cancel Product Order
Product Order
Product Order in state: inProgress
1 POST cancelProductOrder request
2 validation
3 assign id
4 state: acknowledged
5 POST cancelProductOrder response
6 (causes state change)
7 state: assessingCancellation
8 POST productOrderItemStateChangeEvent
9 state: assessingCharge
10 POST cancelProductOrderStateChangeEvent
The Seller finalizes Cancel request processing without identifying additional charges
11 state: completed
12 POST cancelProductOrderStateChangeEvent
13 (causes state change)
14 state: cancelling
15 POST productOrderStateChangeEvent
16 state: cancelled
17 POST productOrderStateChangeEvent

Figure 21: Cancel Product Order Flow

- The Buyer sends a Cancel Product Order request with `CancelProductOrder_Create`.
- The Seller validates the request.
- The Seller initiates the Cancel process, sets the `CancelProductOrder.state` to `acknowledged`, assigns a `CancelProductOrder.id`, and changes the referenced `ProductOrder.state` to `assessingCancellation`.
- The Seller notifies the Buyer of any charges resulting from cancelling the referenced Product Order while the Cancel Request is in the `acknowledged` state (see Section 6.11.3 for details).
- The Seller accepts the Cancel Request. The `CancelProductOrder.state` set to `done` and the referenced `ProductOrder.state` is set to `pendingCancellation`.
- Once the Seller has completed the cancellation process, the referenced `ProductOrder.state` is changed to `cancelled`.

[RXXX] A Buyer **MUST** have submitted the Product Order Request to be able to submit a Cancel Request on the Product Order. [MEF57.2 R77]

6.9. Use case 9: Retrieve List of Cancel Requests

The Buyer can retrieve a list of `CancelProductOrder` by using a `GET /cancelProductOrder` operation with desired filtering criteria. The attributes that are available to be used are: [MEF57.2 O19]

- `productOrderId`
- `state`
- `cancellationReasonType`

The rules of using pagination and an example request are provided in section 6.3. Please refer to it as the rules also apply to this case.

[RXXX] The Seller must put the following attributes into the response: [MEF57.2 R95]:

- `id`
- `cancellationReasonType`
- `productOrder`
- `relatedContactInformation` - item with `role=sellerContact`
- `state`

[RXXX] In case no items matching the criteria are found, the Seller **MUST** return a valid response with an empty list. [MEF57.2 R96]

6.10. Use case 10: Retrieve Cancel Product Order by Cancel Product Order Identifier

The Buyer can get detailed information about the Cancel Product Order request from the Seller by using a `GET /cancelProductOrder/{{id}}` operation.

[RXXX] The Seller's response **MUST** echo back all attributes provided by the Buyer in the request and provide the following attributes: [MEF57.2 R99]

- `id`
- `relatedContactInformation` - item with `role=sellerContact`
- `state`

6.11. Use case 11: Initiate Charge

When new or changes to existing charges are identified by the Seller during processing of a Product Order, the Seller must communicate them to the Buyer and the Buyer must respond if they accept or reject each charge.

Within the Charge, the Seller indicates for each Charge Item, if the Charge Item is Blocking or non-Blocking. If the Buyer rejects a Blocking Charge, the Seller will cancel the processing of the related entity (depending on the sub-case - as described below).

The seller may identify Charges during:

- standard processing Product Order Item,
- processing of Buyer's Cancel Product Order request,
- processing of buyer's Modify Product Order Item Completion Date request.

The variants are described as separate use cases and are explained in next sections.

Figure 22 presents the model taking part in the use case. It is common for all sub-use cases:

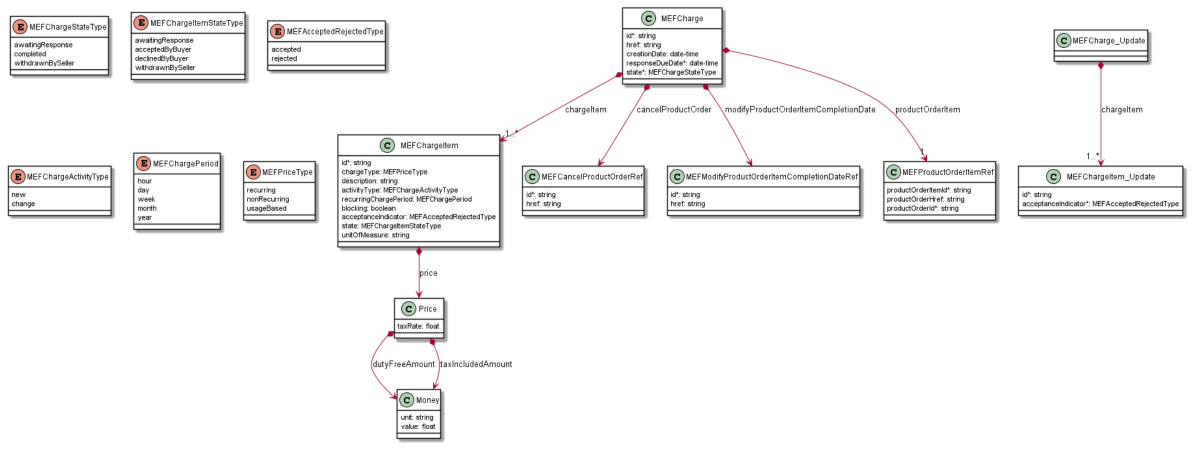


Figure 22: Charge Model

The Figures and Tables below present the Charge and Charge Item states.

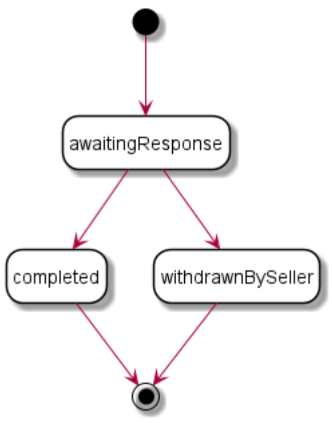


Figure 23: Charge State Machine

State	Description
`completed`	All Charge Items included in the Charge for a given Product Order Item have moved to either the `accepted` state or the `declined` state.
`awaitingResponse`	A Charge has been initiated by the Buyer. The Charge includes one or more charges.
`withdrawnBySeller`	The Seller determines that the Charge is incorrect. They withdraw the Charge and initiate a new Charge with the required correction(s).

Table 11. Charge States

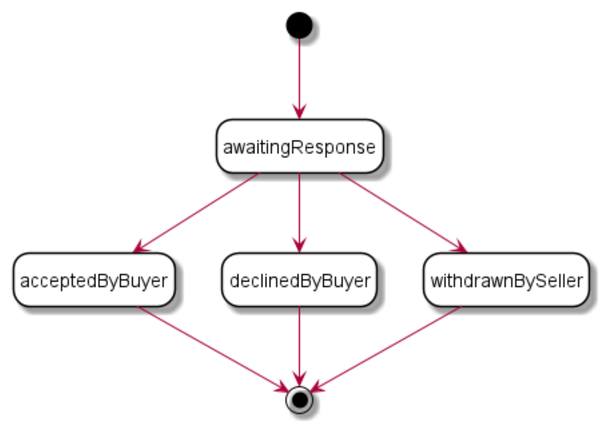


Figure 24: Charge Item State Machine

State	Description
`acceptedByBuyer`	A Charge Item identified in the Charge has been accepted by the Buyer.
`awaitingResponse`	A Charge Item has been identified by the Seller and awaits Buyer's acceptance.
`declinedByBuyer`	A Charge Item identified in the Charge has been declined by the Buyer. The referenced Product Order and Product Order Items are updated.
`withdrawnBySeller`	The Seller determines that the Charge Item is incorrect. They withdraw the Charge Item and initiate a new Charge with the required correction(s).

Table 12. Charge Item States

[RXXX] When the Seller creates a Charge, the following attributes **MUST** be set: [MEF57.2 R72]

- `id`
- `productOrderItemId`
- `chargeItem`
- `responseDueDate`
- `state`

[RXXX] When the Charge was identified as an effect of a Modify Product Order Item Completion Date request the Seller **MUST** provide the `modifyProductOrderItemCompletionDate`.

[RXXX] When the Charge was identified as an effect of a Cancel Product Order request the Seller **MUST** provide the `cancelProductOrder`.

[RXXX] For each Charge Item included in the Charge, the Seller **MUST** include the following attributes: [MEF57.2 R73]

- `id`
- `chargeType`
- `description`
- `blocking`
- `price`
- `state`

[RXXX] Table 13 shows the attributes that **MUST** be included in the Charge Item based on the `chargeType`: [MEF57.2 R74]

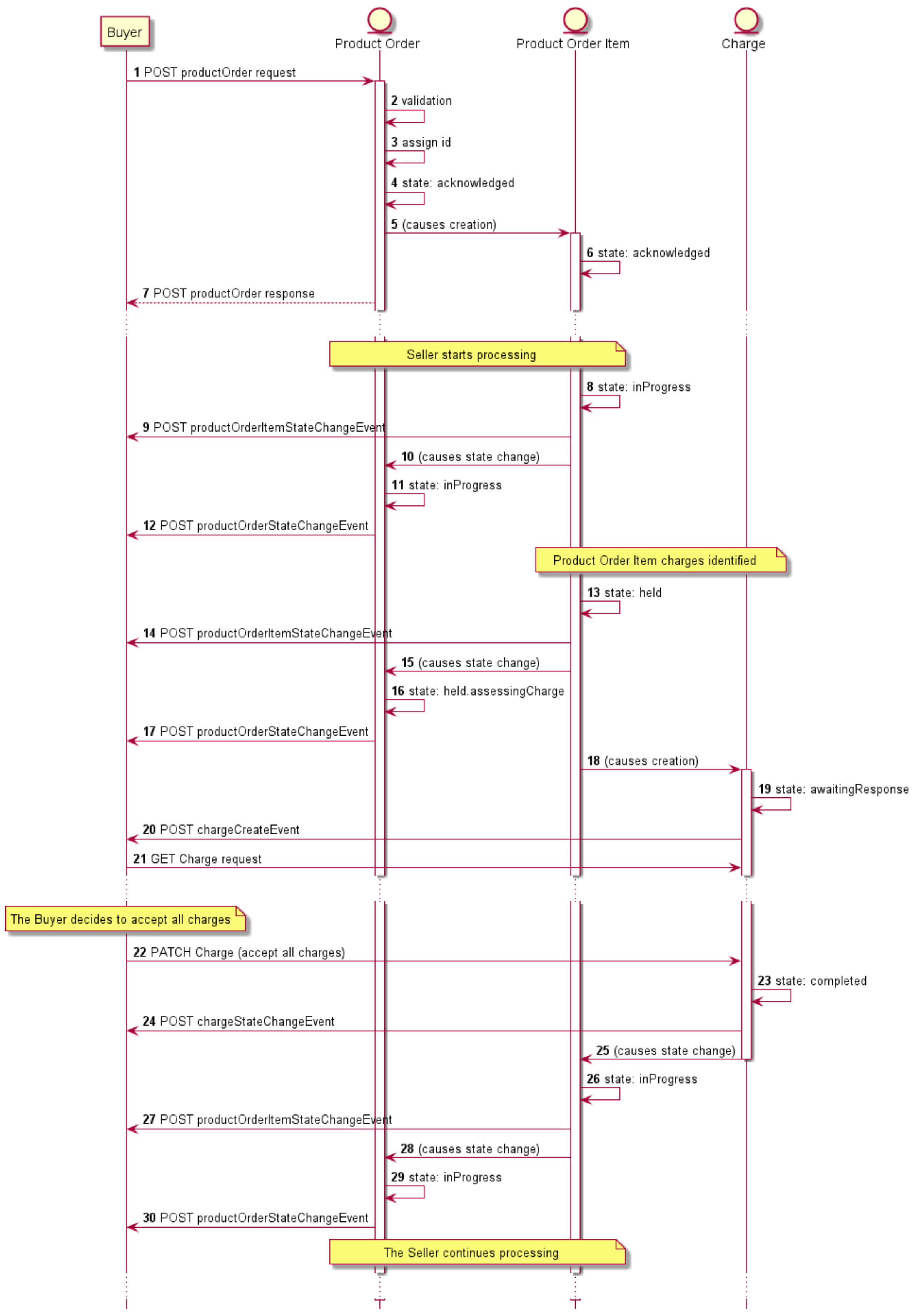
`chargeType`	`recurringChargePeriod`	`unitOfMeasure`	`price.dutyFreeAmount`	Comments

chargeType	recurringChargePeriod	unitOfMeasure	price.dutyFreeAmount	Comments
recurring	X		X	
nonRecurring			X	
usageBased		X	X	price.dutyFreeAmount is the charge per unitOfMeasure

Table 13. Price Type Required Information

6.11.1 Use case 11a: Initiate Charge Associated to Product Order Item

In this case, the Seller identifies new non-recurring or changes on recurring charges during standard processing of a Product Order Item. The model and states have been described earlier. The sequence diagram below presents a Charge use case together with a context of the Use Case 1.



Buyer
Product Order
Product Order Item
Charge
1 POST productOrder request
2 validation
3 assign id
4 state: acknowledged
5 (causes creation)
6 state: acknowledged
7 POST productOrder response
Seller starts processing
8 state: inProgress
9 POST productOrderItemStateChangeEvent
10 (causes state change)
11 state: inProgress
12 POST productOrderStateChangeEvent
Product Order Item charges identified
13 state: held
14 POST productOrderItemStateChangeEvent
15 (causes state change)
16 state: held.assessingCharge
17 POST productOrderStateChangeEvent
18 (causes creation)
19 state: awaitingResponse
20 POST chargeCreateEvent
21 GET Charge request
The Buyer decides to accept all charges
22 PATCH Charge (accept all charges)
23 state: completed
24 POST chargeStateChangeEvent
25 (causes state change)
26 state: inProgress
27 POST productOrderItemStateChangeEvent
28 (causes state change)
29 state: inProgress
30 POST productOrderStateChangeEvent
The Seller continues processing

Figure 25: Use case 11a: Initiate Charge Associated to Product Order Item Flow

The snippet below presents how a Charge related to this use case may look like. This exact part will be a body of a response to a Buyer's GET by is request (point 21).

```
{
  "id": "00000000-0000-1111-0000-000000000001",
  "href": "{{baseUrl}}/charge/00000000-0000-1111-0000-00000000000",
  "creationDate": "2021-05-25T22:05:48.319Z",
  "productOrderId": "00000000-1111-2222-3333-000000000123",
  "productOrderItemId": "item-001",
  "chargeItem": [
    {
      "id": "item-001",
      "chargeType": "nonRecurring",
      "description": "Because of COVID sanitary restrictions there is an additional for the on-site
installation visit",
      "activityType": "new",
      "blocking": true,
      "price": {
        "taxRate": 8,
        "dutyFreeAmount": {
          "unit": "USD",
          "value": 50
        },
        "taxIncludedAmount": {
          "unit": "USD",
          "value": 54
        }
      },
      "state": "awaitingResponse",
    }
  ],
  "cancelProductOrder": { }, << set only if Charge is a result of a Cancel Request >>
  "modifyProductOrderItemCompletionDate": { }, << set only if Charge is a result of a Modify Request >>
  "responseDueDate": "2021-05-25T22:05:48.319Z",
  "state": "awaitingResponse"
}
```

6.11.2 Use case 11b: Initiate Charge Associated to Modify Product Order Item Completion Date

In this case, the Charges are identified as a result of a Modify Product Order Item Completion Date request. The model and states have been described earlier. The sequence diagram below presents a Charge use case together with a context of the Use Case 5a: Modify Product Order Item Completion Date (Expedite Indicator) Request - setting the `expediteIndicator` to `true`.

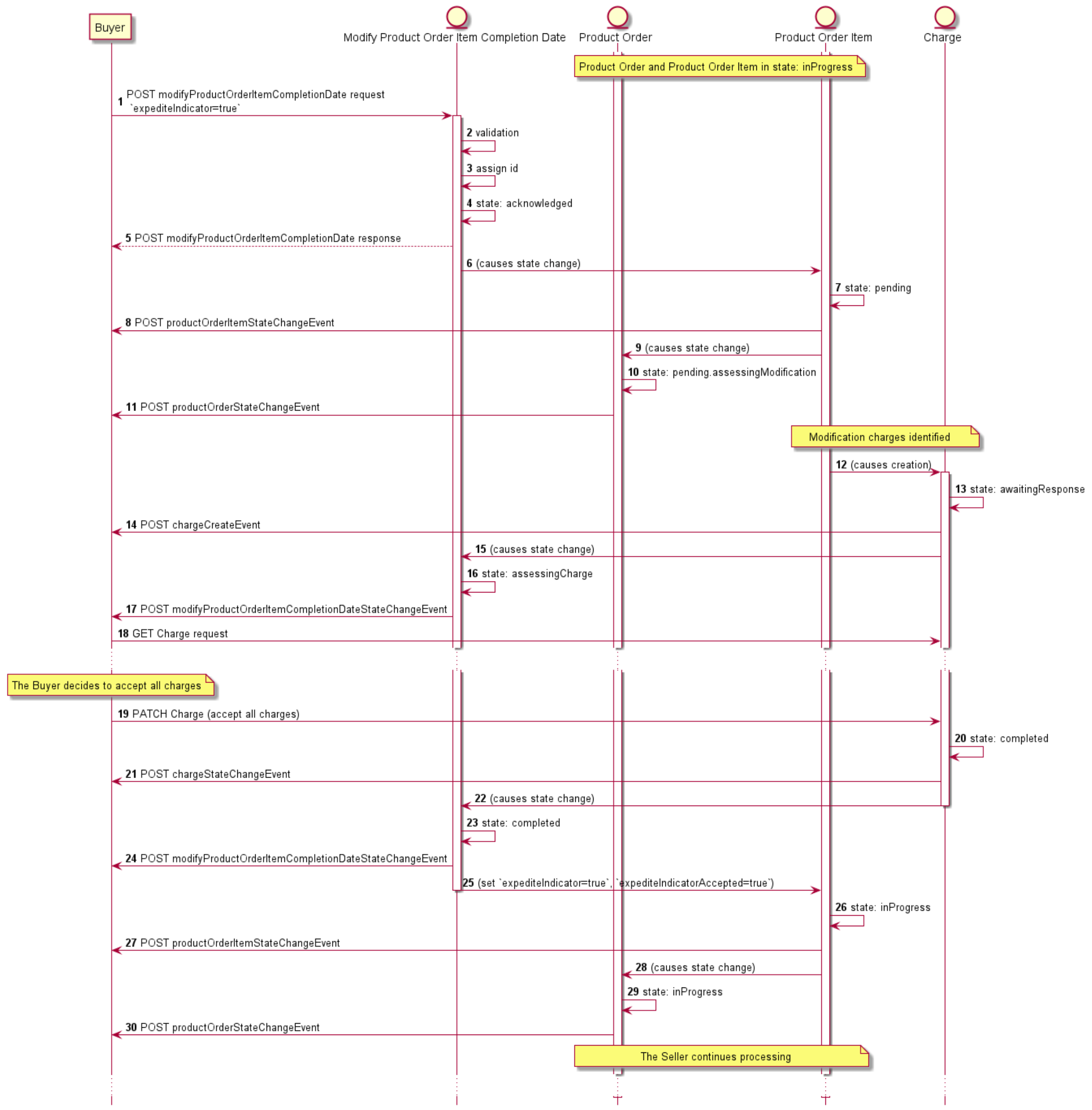

Buyer
Modify Product Order Item Completion Date
Product Order
Product Order Item
Charge
Product Order and Product Order Item in state: inProgress
1 POST modifyProductOrderItemCompletionDate request `expediteIndicator=true`
2 validation
3 assign id
4 state: acknowledged
5 POST modifyProductOrderItemCompletionDate response
6 (causes state change)
7 state: pending
8 POST productOrderItemStateChangeEvent
9 (causes state change)
10 state: pending.assessingModification
11 POST productOrderStateChangeEvent
Modification charges identified
12 (causes creation)
13 state: awaitingResponse
14 POST chargeCreateEvent
15 (causes state change)
16 state: assessingCharge
17 POST modifyProductOrderItemCompletionDateStateChangeEvent
18 GET Charge request
The Buyer decides to accept all charges
19 PATCH Charge (accept all charges)
20 state: completed
21 POST chargeStateChangeEvent
22 (causes state change)
23 state: completed
24 POST modifyProductOrderItemCompletionDateStateChangeEvent
25 (set `expediteIndicator=true`, `expediteIndicatorAccepted=true`)
26 state: inProgress
27 POST productOrderItemStateChangeEvent
28 (causes state change)
29 state: inProgress
30 POST productOrderStateChangeEvent
The Seller continues processing

Figure 26: Use case 11b: Initiate Charge Associated to Modify Product Order Item Completion Date Flow

6.11.3 Use case 11c: Initiate Charge Associated to Cancel Product Order

In this case, the Charges are identified as a result of a Cancel Product Order request. The model and states have been described earlier. The sequence diagram below presents a Charge use case together with a context of the Use Case 8: Cancel In-Flight Product Order Request

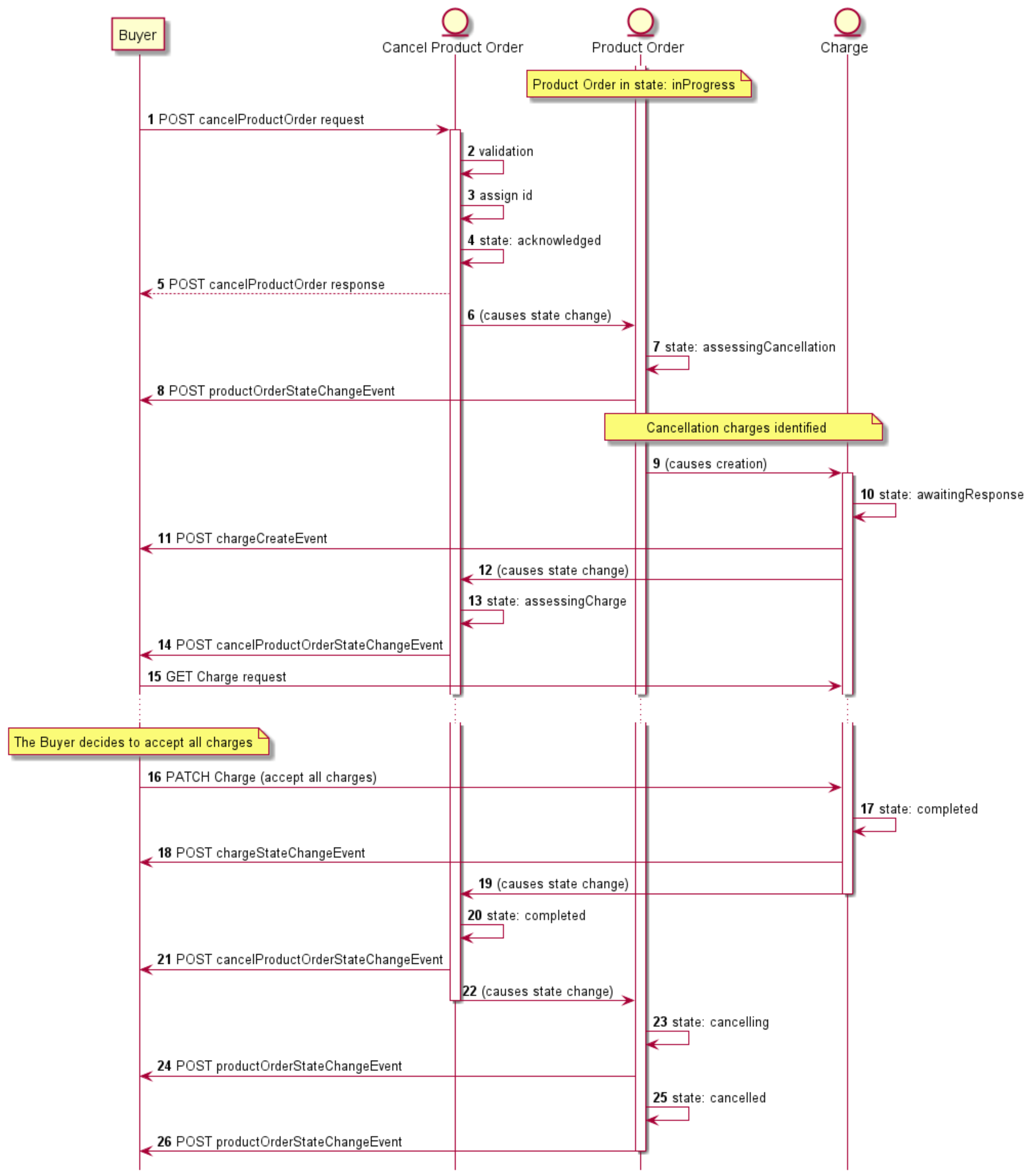

Buyer
Cancel Product Order
Product Order
Charge
Product Order in state: inProgress
1 POST cancelProductOrder request
2 validation
3 assign id
4 state: acknowledged
5 POST cancelProductOrder response
6 (causes state change)
7 state: assessingCancellation
8 POST productOrderStateChangeEvent
Cancellation charges identified
9 (causes creation)
10 state: awaitingResponse
11 POST chargeCreateEvent
12 (causes state change)
13 state: assessingCharge
14 POST cancelProductOrderStateChangeEvent
15 GET Charge request
The Buyer decides to accept all charges
16 PATCH Charge (accept all charges)
17 state: completed
18 POST chargeStateChangeEvent
19 (causes state change)
20 state: completed
21 POST cancelProductOrderStateChangeEvent
22 (causes state change)
23 state: cancelling
24 POST productOrderStateChangeEvent
25 state: cancelled
26 POST productOrderStateChangeEvent

Figure 27: Use case 11c: Initiate Charge Associated to Cancel Product Order Flow

6.12. Use case 12: Respond to Charge

The Buyer may respond to a Charge initiated by the Seller with the use of a `PACTH /charge` operation. The model for this case is in Figure 22 section 6.11.

The PATCH usage recommendation follows TMF 622 json/merge (https://tools.ietf.org/html/rfc7386).

Below is an example of such Charge response - PATCH request:

```
{
  "chargeItem": [
    {
      "id": "item-001",
      "acceptanceIndicator": "accepted"
    }
  ]
}
```

[RXXX] The Buyer's response to the Charge **MUST** update the `acceptanceIndicator` for each and every Charge Item included in the Charge. [MEF57.2 R76]

[RXXX] The Buyer **MUST** update all Charge Items included in a Charge at once. [MEF57.2 R75]

Note: If a `responseDueDate` is passed the Seller **MUST** treat all Charge Items as declined.

If in Use Case 11a the Buyer rejects a Charge Item that is identified as Blocking, the Seller changes the state of the Charge to `completed`, changes the referenced Product Order Item state to `failed`, and changes any Product Order Items related to the referenced Product Order Item to `failed`.

If in Use Case 11b the Buyer rejects a Blocking Charge Item, the Seller changes the state of the Charge to `complete` and changes the referenced Modify Product Order Item Completion Date state to `declined`. No modification to the Product Order Item is Performed.

If in Use Case 11c the Buyer rejects a Blocking Charge Item, the Seller changes the state of the Charge to `complete` and changes the referenced Cancel Product Order state to `declined`, and returns the Product Order to `inProgress`. The Product Order is not cancelled.

6.13. Use case 13: Retrieve List of Charges

The Buyer can retrieve a list of `Charges` by using a `GET /charge` operation with desired filtering criteria. The attributes that are avaible to be used are: [MEF57.2 R100]:

- `productOrderId`
- `productOrderItemId`
- `responseDueDate.gt`
- `responseDueDate.lt`
- `state`

The rules of using pagination and an example request are provided in section 6.3. Please refer to it as the rules also apply to this case.

[RXXX] The Seller must put the following attributes into the response: [MEF57.2 R87]:

- `id`
- `productOrderItem`
- `orderVersion`
- `state`

[RXXX] In case no items matching the criteria are found, the Seller **MUST** return a valid response with an empty list. [MEF57.2 R102]

6.14. Use case 14: Retrieve Charge by Identifier

The Buyer can get detailed information about the Charge communicated by the Seller by using a `GET /charge/{{id}}` operation.

[RXXX] The Seller's response **MUST** provide the following attributes: [MEF57.2 R106]

- `id`
- `relatedContactInformation` - item with `role=sellerContact`
- `state`

6.15. Use case 15: Register for Notifications

The Seller communicates with the Buyer with Notifications provided that:

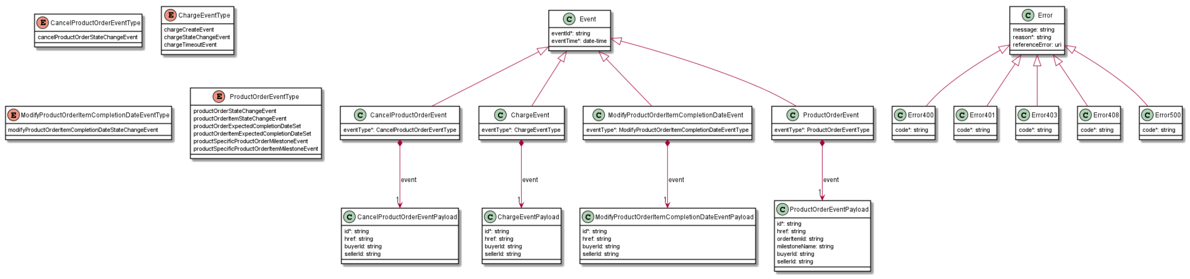
- both Seller and Buyer support notification mechanism
- Buyer has registered to receive notifications from the Seller

To register for notifications the Buyer uses the `registerListener` operation from the API: `POST /hub`. The request model contains only 2 attributes:

- `callback` - mandatory, to provide the callback address the events will be notified to,
- `query` - optional, to provide the required types of event.

The usage of a combination of these attributes fulfills the [MEF57.2 CR3<O20], and [MEF57.2 CR4<O20] requirements.

The figure below shows all entities involved in the Notification use cases.



CancelProductOrderEventType
cancelProductOrderStateChangeEvent
ChargeEventType
chargeCreateEvent
chargeStateChangeEvent
chargeTimeoutEvent
ModifyProductOrderItemCompletionDateEventType
modifyProductOrderItemCompletionDateStateChangeEvent
ProductOrderEventType
productOrderStateChangeEvent
productOrderItemStateChangeEvent
productOrderExpectedCompletionDateSet
productOrderItemExpectedCompletionDateSet
productSpecificProductOrderMilestoneEvent
productSpecificProductOrderItemMilestoneEvent
Event
eventId*: string
eventTime*: date-time
CancelProductOrderEvent
eventType*: CancelProductOrderEventType
ChargeEvent
eventType*: ChargeEventType
ModifyProductOrderItemCompletionDateEvent
eventType*: ModifyProductOrderItemCompletionDateEventType
ProductOrderEvent
eventType*: ProductOrderEventType
event
1
CancelProductOrderEventPayload
id*: string
href: string
buyerId: string
sellerId: string
ChargeEventPayload
id*: string
href: string
buyerId: string
sellerId: string
ModifyProductOrderItemCompletionDateEventPayload
id*: string
href: string
buyerId: string
sellerId: string
ProductOrderEventPayload
id*: string
href: string
orderItemId: string
milestoneName: string
buyerId: string
sellerId: string
Error
message: string
reason*: string
referenceError: uri
Error400
code*: string
Error401
code*: string
Error403
code*: string
Error408
code*: string
Error500
code*: string

Figure 28. Product Order Management Notification Data Model

By using a simple request:

```
{
  "callback": "https://buyer.com/listenerEndpoint"
}
```

The Buyer subscribes for notification of all types of events. Those are:

- `productOrderStateChangeEvent`
- `productOrderItemStateChangeEvent`
- `productSpecificProductOrderMilestoneEvent`
- `productSpecificProductOrderItemMilestoneEvent`
- `productOrderExpectedCompletionDateSetEvent`
- `productOrderItemExpectedCompletionDateSetEvent`
- `cancelProductOrderStateChangeEvent`
- `chargeCreateEvent`
- `chargeStateChangeEvent`
- `chargeTimeoutEvent`
- `modifyProductOrderItemCompletionDateStateChangeEvent`

If the Buyer wishes to receive only notification of a certain type, a `query` must be added:

```
{
  "callback": "https://buyer.com/listenerEndpoint",
  "query": "eventType=productOrderStateChangeEvent"
}
```

If the Buyer wishes to subscribe to 2 different types of events, there are 2 possible syntax variants [TMF630]:

```
eventType=productOrderStateChangeEvent,chargeCreateEvent
```

or

```
eventType=productOrderStateChangeEvent&eventType=chargeCreateEvent
```

The `query` formatting complies to RCF3986 RFC3986. According to it, every attribute defined in the Event model (from notification API) can be used in the `query`. However, this standard requires only `eventType` attribute to be supported.

[RXXX] The Seller **MAY** support Notifications. [MEF57.2 O20]

[RXXX] `eventType` is the only attribute that the Seller **MUST** support in the query.

The Seller responds to the subscription request by adding the `id` of the subscription to the message that must be further used for unsubscribing.

```
{
  "id": "00000000-0000-0000-0000-000000000678",
  "callback": "https://buyer.com/listenerEndpoint",
  "query": "eventType=productOrderStateChangeEvent"
}
```

Example of a final address that the Notifications will be sent to (for Sonata, `productOrderStateChangeEvent`):

- `https://buyer.com/listenerEndpoint/mefApi/sonata/productOrderNotification/v7/listener/productOrderStateChangeEvent`

6.16. Use case 16: Send Notification

Notifications are used to asynchronously inform the Buyer about the respective objects and attributes changes. The Seller's synchronous response to a Product Order, Cancel Product Order, and Modify Product Order Item Completion Date create requests are considered to act as a Create Notification so there is no explicit respective Create Notification type. The next notification must be sent when the `state` changes compared to the previously sent one.

For sake of readability, all previous flow diagrams presented only cases of using only the `productOrderStateChangeEvent`. Figure 29 presents the end-to-end sequence of communication in Use Case 1 - Create Product Order with Buyer's subscription to both `productOrderStateChangeEvent` and `productOrderItemStateChangeEvent` event types.

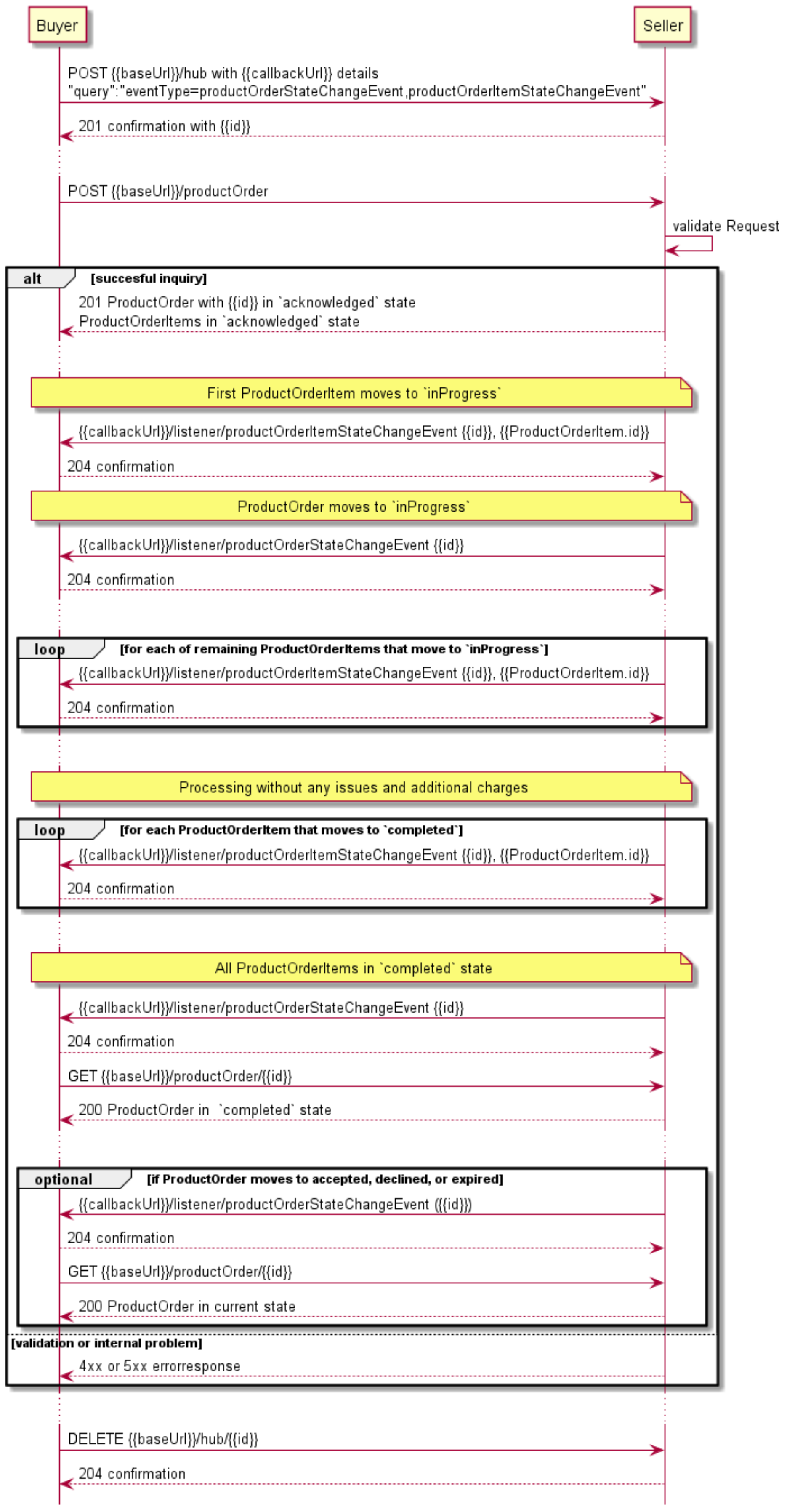

Buyer
Seller
POST {{baseUrl}}/hub with {{callbackUrl}} details
"query":"eventType=productOrderStateChangeEvent,productOrderItemStateChangeEvent"
201 confirmation with {{id}}
POST {{baseUrl}}/productOrder
validate Request
alt
[succesful inquiry]
201 ProductOrder with {{id}} in `acknowledged` state
ProductOrderItems in `acknowledged` state
First ProductOrderItem moves to `inProgress`
{{callbackUrl}}/listener/productOrderItemStateChangeEvent {{id}}, {{ProductOrderItem.id}}
204 confirmation
ProductOrder moves to `inProgress`
{{callbackUrl}}/listener/productOrderStateChangeEvent {{id}}
204 confirmation
loop
[for each of remaining ProductOrderItems that move to `inProgress`]
{{callbackUrl}}/listener/productOrderItemStateChangeEvent {{id}}, {{ProductOrderItem.id}}
204 confirmation
Processing without any issues and additional charges
loop
[for each ProductOrderItem that moves to `completed`]
{{callbackUrl}}/listener/productOrderItemStateChangeEvent {{id}}, {{ProductOrderItem.id}}
204 confirmation
All ProductOrderItems in `completed` state
{{callbackUrl}}/listener/productOrderStateChangeEvent {{id}}
204 confirmation
GET {{baseUrl}}/productOrder/{{id}}
200 ProductOrder in `completed` state
optional
[if ProductOrder moves to accepted, declined, or expired]
{{callbackUrl}}/listener/productOrderStateChangeEvent ({{id}})
204 confirmation
GET {{baseUrl}}/productOrder/{{id}}
200 ProductOrder in current state
[validation or internal problem]
4xx or 5xx errorresponse
DELETE {{baseUrl}}/hub/{{id}}
204 confirmation

Figure 29. Use Case 1 - Create Product Order with Product Order Item Notifications

After a successful Notification subscription, the Buyer sends a Product Order create request. The Seller responds with Product Order and all items in `acknowledged` state. When the first Product Order Item moves to `inProgress`, a `productOrderItemStateChangeEvent` is sent. Immediately the Product Order also changes its state to `inProgress` and the `productOrderStateChangeEvent` is sent. Then the rest (if any) of the Product Order Items are processed. Let's assume that no additional charges were found and the process ends smoothly. When particular items are done processing they reach the `completed` state. Once all are successfully done, the Product Order also changes state to `completed`. The Buyer will likely now ask for the Product Order details.

The events are sent only after a synchronous response to the Product Order create request was provided. Thus there must be no state change notifications set for Product Order and Product Order Items reaching the `acknowledged` state.

[RXXX] The Seller **MUST NOT** send Notifications to Buyers who have not registered for them. [MEF57.2 R107]

[RXXX] The Seller **MUST** send Notifications to Buyers who have registered for them. [MEF57.2 R108].

[OXXX] The Seller **MAY** support sending Notifications. [MEF57.2 O20]

Following snippets present example of `productOrderStateChangeEvent` and `productOrderItemStateChangeEvent`:

```
{
  "eventId": "event-001",
  "eventType": "productOrderStateChangeEvent",
  "eventTime": "2021-06-02T00:00:00.000Z",
  "event": {
    "id": "00000000-1111-2222-3333-000000000123"
  }
}
```

[RXXX] An event triggered by the state change of the Product Order Item **MUST** additionally contain the relative `orderItemId`.

```
{
  "eventId": "event-002",
  "eventType": "productOrderItemStateChangeEvent",
  "eventTime": "2021-06-02T00:00:00.000Z",
  "event": {
    "id": "00000000-1111-2222-3333-000000000123",
    "orderItemId": "item-001"
  }
}
```

Note: the body of the event carries only source object's `id`. The Buyer needs to query it later by `id` to get details.

To stop receiving events, the Buyer has to use the `unregisterListener` operation from the `DELETE /hub/{id}` endpoint. The `id` is the identifier received from the Seller during the listener registration.

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`.

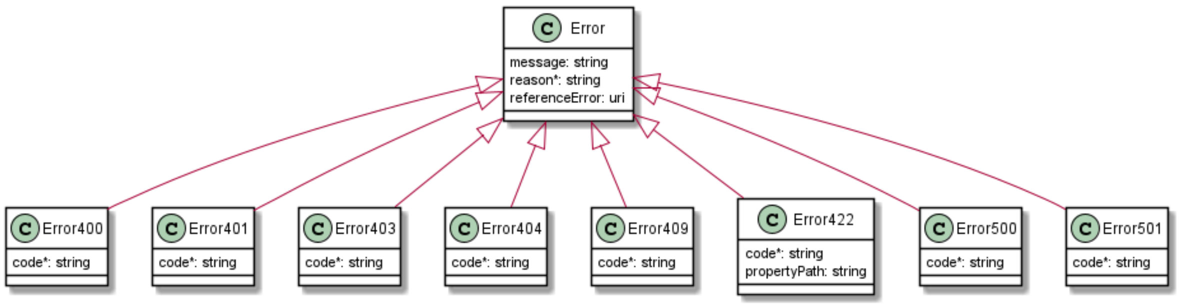


Figure 30. Data model types to represent an erroneous response

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
message	string	Text that provides mode 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	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*	string	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. Type Error401

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

Inherits from:

- Error

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

7.1.1.4. 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*	string	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

7.1.1.5. Type Error404

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

Inherits from:

- Error

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

7.1.1.6. Type Error409

Description: Conflict (https://datatracker.ietf.org/doc/html/rfc7231#section-6.5.8)

Inherits from:

- Error

Name	Type	Description
code*	string	The following error code: - conflict: The client has provided a value whose semantics are not appropriate for the property.

7.1.1.7. 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://tools.ietf.org/html/rfc4918#section-11.2)

Inherits from:

- Error

Name	Type	Description
code*	string	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).

7.1.1.8. 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.

7.1.1.9. Type Error501

Description: Not Implemented. (https://tools.ietf.org/html/rfc7231#section-6.6.2)

Inherits from:

- Error

Name	Type	Description
code*	string	The following error code: - notImplemented: Method not supported by the server

7.1.2. Response pagination

A response to retrieve a list of results (e.g. `GET /productOfferingQualification`) 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`

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

7.2. Management API Data model

Figure 31 presents the whole Product Order Management data model. The data types, requirements related to them and mapping to MEF 57.2 specification are discussed later in this section.

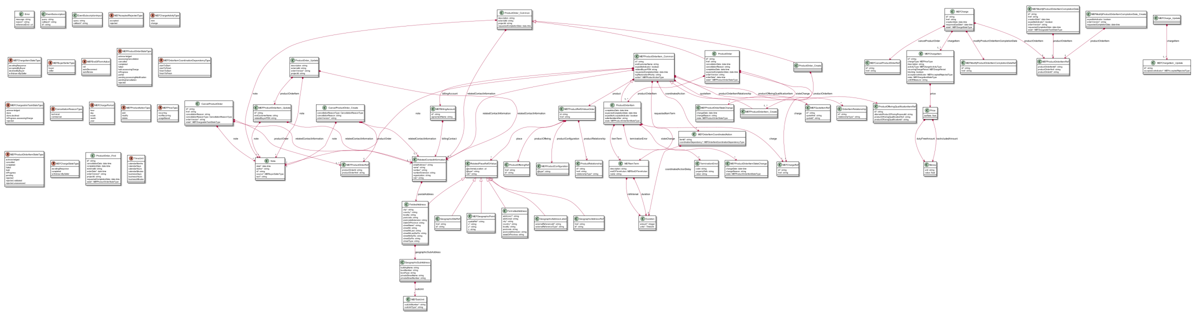


Figure 31. Product Order Management Data Model

7.2.1. ProductOrder

7.2.1.1 Type ProductOrder_Common

Description: A Product Order is a type of order which can be used to place an order between a customer and a service provider or between a service provider and a partner and vice versa,

Name	**Type**	**Description**	**MEF 57.2**
description	string	Description of the product order. It is a free text for Buyer purpose. The Seller is not obliged to read it.	Not represented in MEF 57.2
externalId	string	An identifier for this order within the Buyer's enterprise.	Buyer Product Order Identifier
projectId	string	An identifier that is used to group Orders that is important to the Buyer. A ProjectId can be used to relate multiple Orders together.	Project Identifier
requestedCompletionDate	date-time	This is requested date to get this Product Order completed	Not represented in MEF 57.2
note	Note[]		Note

Name	Type	Description	MEF 57.2
relatedContactInformation*	RelatedContactInformation[]	Contact information of an individual or organization playing a role in this context. (e.g. Product Order Contact: role=productOrderContact; Seller Contact: role=sellerContact) Providing the Product Order Contact in the request is mandatory.	Product Order Contact, Seller Contact

7.2.1.2. Type ProductOrder_Create

Description: A Product Order is a type of order which can be used to place an order between a customer and a service provider or between a service provider and a partner and vice versa, Skipped properties: id,href,completionDate,orderDate,state,stateChange,cancellationDate,cancellationReason

Inherits from:

- ProductOrder_Common

Name	Type	Description	MEF 57.2
productOrderItem*	MEFProductOrderItem_Create[]	Items contained in the Product Order.	Product Order Item

7.2.1.3. Type ProductOrder

Description: A Product Order is a type of order which can be used to place an order between a customer and a service provider or between a service provider and a partner and vice versa

Inherits from:

- ProductOrder_Common

Name	Type	Description	MEF 57.2
id*	string	Unique identifier for the Product Order that is generated by the Seller when the Product Order is initially accepted via an API.	Product Order Identifier
href	string	Hyperlink to access the order	Not represented in MEF 57.2
cancellationDate	date-time	Identifies the date the Seller cancelled the Order. Set by Seller when the Order is moved to the cancelled state.	Product Order Cancellation Date

Name	Type	Description	MEF 57.2
cancellationReason	string	An optional free-form text field for the Seller to provide additional information regarding the reason for the cancellation.	Cancellation Reason
completionDate	date-time	Identifies the date that all Product Order Items within the Order have reached a terminal state. No further action is permitted on the Product Order.	Product Order Final State Date
expectedCompletionDate	date-time	Expected delivery date amended by the provider. MEF: Identifies the date the Seller expects to complete the Product Order.	Not represented in MEF 57.2
orderVersion	string	The version of the Product Order, set by the Seller	Product Order Version
orderDate*	date-time	Date when the order was created in the Seller's system	Product Order Create Date
productOrderItem*	ProductOrderItem[]		Product Order Item
state*	MEFProductOrderStateType	Tracks the lifecycle status of the product order, such as Acknowledged, Rejected, InProgress, Pending and so on.	Product Order State
stateChange	MEFProductOrderStateChange[]	State change for the Product Order	Not represented in MEF 57.2

7.2.1.4. Type ProductOrder_Update

Description: A request initiated by the Buyer to update Product Order and/or Product

Name	Type	Description	MEF 57.2
description	string	Description of the product order. It is a free text for Buyer purpose. The Seller is not obliged to read it.	Not represented in MEF 57.2
externalId	string	An identifier for this order within the Buyer's enterprise.	Not represented in MEF 57.2

Name	Type	Description	MEF 57.2
orderVersion*	string	The version of the Product Order. The `orderVersion` attribute cannot be updated. It is used only to identify the version of the Product Order that the Buyer wants to update. If there is a mismatch with the Seller's system, the Seller will reject the request with an error response.	Buyer Product Order Identifier
projectId	string	An identifier that is used to group Orders that is important to the Buyer. A ProjectId can be used to relate multiple Orders together.	Project Identifier
note	Note[]		Note
relatedContactInformation	RelatedContactInformation[]	Contact information of an individual or organization playing a role in this context. The Buyer is allowed to update the Product Order Contact: role=productOrderContact;	Product Order Contact, Seller Contact
productOrderItem	MEFProductOrderItem_Update[]	Order Item attributes that may be updated	Product Order Item

7.2.1.5. Type ProductOrder_Find

Description: Structure to define GET without id response. A list of productOrder matching request criteria. Provides Product order summary view.

Name	Type	Description	MEF 57.2
id*	string	Unique identifier for the order that is generated by the Seller when the order is initially accepted via an API.	Product Order Identifier
cancellationDate	date-time	Identifies the date the Seller cancelled the Order. Set by Seller when the Order is moved to the cancelled state.	Not represented in MEF 57.2

Name	Type	Description	MEF 57.2
completionDate	date-time	Identifies the date that all Product Order Items within the Order have reached a terminal state. No further action is permitted on the Product Order after this notification.	Product Order Cancellation Date
externalId	string	ID given by the consumer and only understandable by him (to facilitate his searches afterwards). MEF: Buyer Purchase Order Number	Buyer Product Order Identifier
orderDate*	date-time	Date when the order was created	Product Order Create Date
orderVersion*	string	The version of the Product Order, assigned by the Seller	Product Order Version
projectId	string	An identifier that is used to group Orders that is important to the Buyer. A ProjectId can be used to relate multiple Orders together.	Project Identifier
requestedCompletionDate	date-time	This is requested date to get this Product Order completed	Not represented in MEF 57.2
state*	MEFProductOrderStateType	The states as defined by TMF622 and extended to meet MEF requirements. These states are used to convey the Product Order status during the lifecycle of the Product Order.	Product Order State

7.2.1.6. `enum` MEFProductOrderStateType

Description: Possible values for the state of the order The following mapping has been used between `MEFProductOrderStateType` and MEF 57.2:

MEFProductOrderStateType	MEF 57.2
acknowledged	ACKNOWLEDGED
assessingCancellation	ASSESSING_CANCELLATION
cancelled	CANCELLED
completed	COMPLETE
failed	FAILED
held.assessingCharge	ASSESSING_CHARGE
inProgress	IN_PROGRESS

MEFProductOrderStateType	MEF 57.2
partial	PARTIAL
pending.assessingModification	ASSESSING_MODIFICATION
pendingCancellation	CANCELLING
rejected	REJECTED

7.2.1.7. Type MEFProductOrderStateChange

Description: Holds the State notification reasons and associated date the State changed, populated by the server

Name	Type	Description	MEF 57.2
changeDate	date-time	The date on when the state was reached	Not represented in MEF 57.2
changeReason	string	Additional comment related to state change	Not represented in MEF 57.2
state	MEFProductOrderStateType	Reached state	Not represented in MEF 57.2

7.2.2. Product Order Item

7.2.2.1 Type MEFProductOrderItem_Common

Description: An identified part of the order. A product order is decomposed into one or more order items.

Name	Type	Description	ME
id*	string	Identifier of the item (generally it is a sequence number 01, 02, 03, ...) MEF: A Buyer provided identifier to identify Product Order Items and to be able to relate them to one another. This is set by the Buyer and is unique within the Product Order. Examples of Reference Identifier could be 1, 2, 3 or A, B, C. The Reference Identifier can be reused in multiple Product Orders to identify a Product Order Item within that Product Order.	Proc Refe
endCustomerName	string	The name of the End Customer, either a business name or an individual name depending on the end customer.	Proc End Nan
expediteIndicator	boolean	Indicates that expedited treatment is requested. Set by the Buyer. If this is set to TRUE, the Buyer sets the Requested Completion Date to the expedited date. See MEF 57.2 section 7.2 for a description of the interaction between the Buyer and the Seller.	Exp

Name	Type	Description	ME
relatedBuyerPON	string	Identifies the Buyer Purchase Product Order Number that is related to this Product Order.	Rela Purc Nun
requestedCompletionDate	date-time	Identifies the Buyer's desired due date (requested delivery date)	Proc Req Con
tspRestorationPriority	string	Within the United States, indicates the provisioning and restoration priority as defined under the TSP Service Vendor Handbook. The valid values are defined in ATIS OBF document: ATIS-0404001.	Proc Tele Serv Prio Prio
action*	MEFProductActionType	Action to be applied to this portion of the Product Order to the product referred by this Product Order Item	Proc Proc
billingAccount	MEFBillingAccount	References the billing arrangement that a buyer has with a seller that provides products to the customer.	Buy Info
coordinatedAction	MEFOrderItemCoordinatedAction[]	The interval after the completion of one or more related Product Order Items that this Product Order Item can be started or completed	Proc Coo
note	Note[]		Not
product	MEFProductRefOrValueOrder	The Buyer's existing Product for which the Product Order is being requested. Set by the Buyer if the Product Action is modify or delete.	Proc Proc
productOfferingQualificationItem	ProductOfferingQualificationItemRef	The POQ and POQ Item associated to this Product Order Item. The relation may be required by the Seller. In that case, this is a mandatory field. If the Seller does not require the POQ Item reference, then this is an optional attribute.	Proc POC
productOrderItemRelationship	OrderItemRelationship[]	The relationship between Product Order Items in the Product Order.	Proc Rela
quoteItem	MEFQuoteItemRef	The Quote Item associated to this Product Order Item. The Quote Item reference may be required by the Seller. In that case, this is a mandatory field. If the Seller does not require the Quote, then this is an optional attribute.	Proc Quo

Name	Type	Description	ME
relatedContactInformation	RelatedContactInformation[]	Contact information of an individual or organization playing a role for this Order Item. The rule for mapping a represented attribute value to a `role` is to use the _lowerCamelCase_ pattern e.g. - Buyer Product Order Item Contact: `role=buyerProductOrderItemContact` - Buyer Implementation Contact: `role=buyerImplementationContact` - Buyer Technical Contact : `role=buyerTechnicalContact`	Buy Ord Con Imp Con Tecl
requestedItemTerm	MEFItemTerm	Requested term of the Product Order Item	Not ME

7.2.2.2. Type MEFProductOrderItem_Create

Description: An identified part of the order. A product order is decomposed into one or more order items.

Inherits from:

- MEFProductOrderItem_Common

7.2.2.3. Type ProductOrderItem

Description: An identified part of the order. A product order is decomposed into one or more order items.

Inherits from:

- MEFProductOrderItem_Common

Name	Type	Description	MEF 57.2
completionDate	date-time	Identifies the date the Seller completed the Product Order Item.	Product Order Item Completion Date
expectedCompletionDate	date-time	Expected delivery date amended by the provider. MEF: Identifies the date the Seller expects to complete the Product Order Item.	Product Order Item Expected Completion Date

Name	Type	Description	MEF 57.2
expediteAcceptedIndicator	boolean	Indicates if the Seller has accepted the Buyer's Expedite request. See MEF 57.2 section 7.2 for a description of the interaction between the Buyer and Seller. If this is set to true, the Seller provides the costs to expedite the Product Order in the charge attribute	Product Order Item Expedite Accepted Indicator
sellerItemIdentifier	string	A Seller provided identifier to identify the Product Order Items within a Product Order. This is only used when required for discussion between the Buyer and Seller.	Seller Product Order Item Identifier
charge	MEFChargeRef[]	The Charges associated to this Product Order Item	Related Charges
itemTerm	MEFItemTerm[]		Product Order Item Term
state	MEFProductOrderItemStateType	State of the order item : described in the state machine diagram	Product Order Item State
stateChange	MEFProductOrderItemStateChange[]	State change for the Product Order Item	Not represented in MEF 57.2
terminationError	TerminationError[]	When the Seller cannot process the request, the Seller returns a text-based list of reasons here.	Not represented in MEF 57.2

7.2.2.4. Type MEFProductOrderItem_Update

Description: An updatable representation of the Product Order Item.

Name	Type	Description	MEF 57.2

Name	Type	Description	MEF 57.2
id*	string	Identifier of the Item. This is to address the Item to be updated within the Product Order. The id itself cannot be updated.	Product Order Item Reference Number
endCustomerName	string	The name of the End Customer, either a business name or an individual name depending on the end customer.	Product Order Item End Customer Name
relatedBuyerPON	string	Identifies the Buyer Purchase Product Order Number that is related to this Product Order.	Related Buyer Purchase Order Number
note	Note[]		Note
relatedContactInformation	RelatedContactInformation[]	Contact information of an individual or organization playing a role for this Order Item. The rule for mapping a represented attribute value to a \`role\` is to use the _lowerCamelCase_ pattern e.g. - Buyer Product Order Item Contact: \`role=buyerProductOrderItemContact\` - Buyer Implementation Contact: \`role=buyerImplementationContact\` - Buyer Technical Contact : \`role=buyerTechnicalContact\`	Buyer Product Order Item Contact</br>Buyer Implementation Contact</br>Buyer Technical Contact

7.2.2.5. `enum` MEFProductActionType

Description: Action to be performed on the Product that the Order Item refers to.

ProductActionType	MEF 57.2
add	INSTALL
modify	CHANGE
delete	DISCONNECT

7.2.2.6. `enum` MEFProductOrderItemStateType

Description: Possible values for the state of the Order Item

The following mapping has been used between `MEFProductOrderItemStateType` and MEF 57.2:

MEFProductOrderItemStateType	MEF 57.2
acknowledged	ACKNOWLEDGED
cancelled	CANCELLED
completed	COMPLETED
failed	FAILED
held	HELD

MEFProductOrderItemStateType	MEF 57.2
inProgress	IN_PROGRESS
pending	PENDING
rejected	REJECTED
rejected.validated	VALIDATED
rejected.unassessed	UNASSESSED

7.2.2.7. Type MEFProductOrderItemStateChange

Description: Holds the State notification reasons and associated date the State changed, populated by the server

Name	Type	Description	MEF 57.2
changeDate	date-time	The date on when the state was reached	Not represented in MEF 57.2
changeReason	string	Additional comment related to state change.	Not represented in MEF 57.2
state	MEFProductOrderItemStateType	Reached state	Not represented in MEF 57.2

7.2.2.8. Type ProductOfferingQualificationItemRef

Description: It's a productOfferingQualification item that has been executed previously.

Name	Type	Description	MEF 57.2
id*	string	Id of an item of a product offering qualification	POQ Item Identifier
alternateProductOfferingProposalId	string	A unique identifier for this Alternate Product Proposal assigned by the Seller.	Alternate Product Proposal Identifier
productOfferingQualificationHref	string	Reference of the related entity.	Not represented in MEF 57.2
productOfferingQualificationId*	string	Unique identifier of a related entity.	POQ Identifier

7.2.2.9. 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 57.2
href	string	Hyperlink to a Product Offering in Sellers catalog. In case Seller is not providing a catalog capabilities this field is not used. The catalog API definition 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 57.2

Name	Type	Description	MEF 57.2
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 Identifier

7.2.2.10. Type OrderItemRelationship

Description:

Name	Type	Description	MEF 57.2
id*	string	Id of the related Order Item (must be in the same Order).	Identifier Related Product Order Item Reference Identifier
relationshipType*	string	Specifies the nature of the relation-ship to the related Product Order Items. String that is one of the relationship types specified in the Product Specification.	Product Order Item Relationship Nature

7.2.2.11. Type MEFOrderItemCoordinatedAction

Description:

Name	Type	Description	MEF 57.2
itemId*	string	Specifies Product Order Item that is to be coordinated with this Product Order Item.	Product Order Item Reference Identifier
coordinatedActionDelay*	Duration	The period of time for which the coordinated action is delayed.	Coordinated Action Delay
coordinationDependency*	MEFOrderItemCoordinationDependencyType	A dependency between the Product Order Item and a related Product Order Item	Product Order Item Coordination Dependency

7.2.2.12. enum MEFOrderItemCoordinationDependencyType

Description: Possible values of the Order Item Coordination Dependency

OrderItemCoordinationDependencyType	MEF 57.2
startToStart	START_TO_START

OrderItemCoordinationDependencyType	MEF 57.2
startToFinish	START_TO_FINISH
finishToStart	FINISH_TO_START
finishToFinish	FINISH_TO_FINISH

7.2.2.13. Type MEFProductOrderItemRef

Description: It's a ProductOrder item

Name	Type	Description	MEF 57.2
productOrderItemId*	string	Id of an Item within the Product Order	Product Order Item Reference Identifier
productOrderHref	string	Reference of the related ProductOrder.	Not represented in MEF 57.2
productOrderId*	string	Unique identifier of a ProductOrder.	Product Order Identifier

7.2.2.14. Type MEFQuoteItemRef

Description: It's a Quote item that has been executed previously.

Name	Type	Description	MEF 57.2
id*	string	Id of an Item of a Quote	Quote Item Identifier
quoteHref	string	Reference of the related Quote.	Not represented in MEF 57.2
quoteId*	string	Unique identifier of a Quote.	Quote Identifier

7.2.2.15. Type MEFChargeRef

Description: a reference to a Charge instance

Name	Type	Description	MEF 57.2
id*	string	A unique identifier of the Charge	Charge Identifier
href	string	Hyperlink to access the Charge	Not represented in MEF 57.2

7.2.3. Product representation

7.2.3.1. Type MEFProductRefOrValueOrder

Description: Used by the Buyer to point to existing and/or describe the desired shape of the product. In case of `add` action - only `productConfiguration` MUST be specified. For `modify` action - both `id` and `productConfiguration` MUST be provided to point which product instance to update and to what state. In `delete` only the `id` must be provided.

Name	Type	Description	MEF 57.2

Name	Type	Description	MEF 57.2
id	string	The unique identifier of an in-service Product that is the ordering subject. This field MUST be populated if an item `action` is either `modify` or `delete`. This field MUST NOT be populated if an item `action` is `add`.	Product Identifier
href	string	Hyperlink to the referenced Product. 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 57.2
place	RelatedPlaceRefOrValue[]	The relationships between this Product Order Item and one or more Places as defined in the Product Specification.	Product Order Item Place Relationship
productConfiguration	MEFProductConfiguration	MEFProductConfiguration is used to specify the MEF specific product payload. This field MUST be populated if an item `action` is `add` or `modify`. It MUST NOT be populated when an item `action` is `delete`. The @type is used as a discriminator.	Product Specific Attributes
productOffering	ProductOfferingRef	A particular Product Offering defines the technical and commercial attributes and behaviors of a Product.	Product Order Item Product Offering Identifier
productRelationship	ProductRelationship[]	A list of references to existing products that are related to the orderedProduct.	Product Relationship

7.2.3.2. 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	Description	MEF 57.2
@type*	string	The name of the type, defined in the JSON schema specified above, for the product that is the subject of the Product Order Request. The named type must be a subclass of MEFProductConfiguration.	Not represented in MEF 57.2

7.2.3.3. Type ProductRelationship

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

Name	Type	Description	MEF 57.2

Name	Type	Description	MEF 57.2
id*	string	unique identifier	Related Product Identifier
href	string	Hyperlink to the product in Seller's inventory that is referenced Hyperlink MAY be used when providing 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 57.2
relationshipType*	string	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

7.2.4. Place representation

There are several formats in which place information can be introduced to the Product Order request.

[RXXX] `GeographicAddressRef` or `GeographicSiteRef` **MUST** be used to provide place information by reference. This method is referred to as "Known Address ID method" in MEF 79 Sn 8.9.3.1.

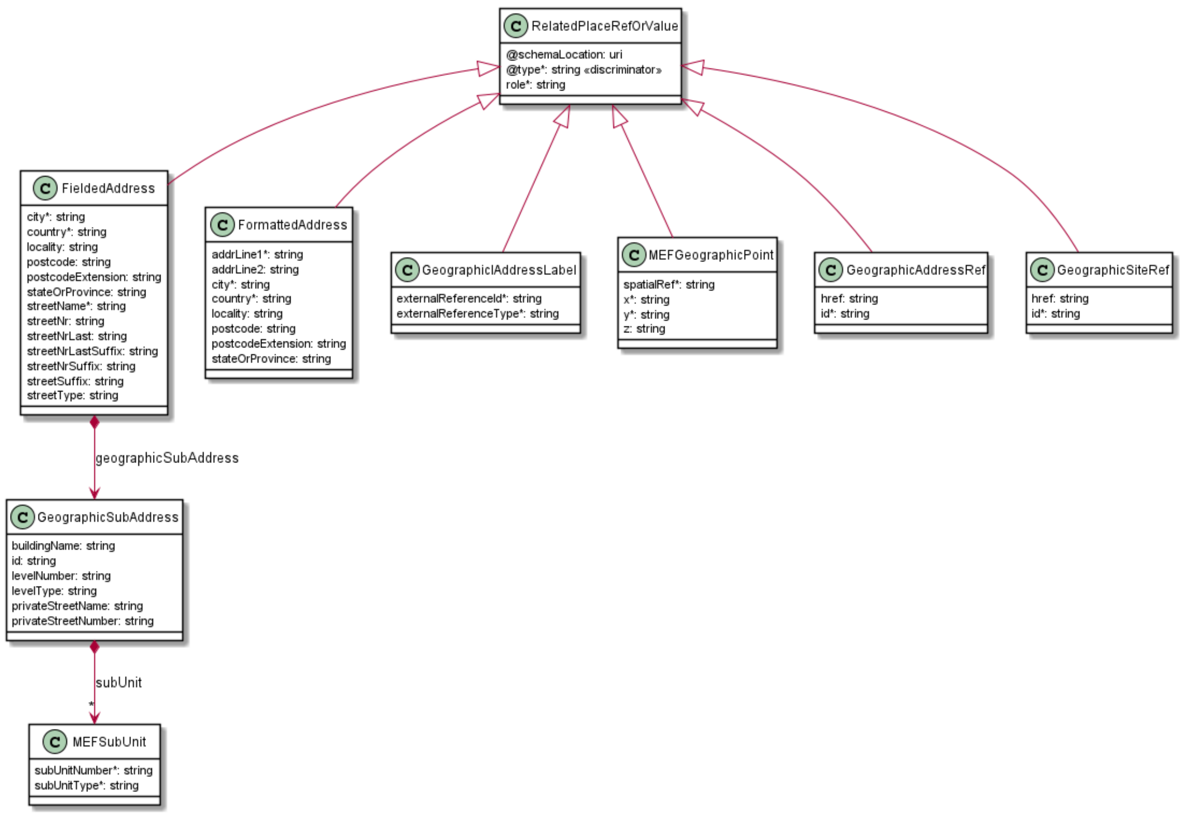


Figure 32. Data model types representing a place

7.2.4.1. Type RelatedPlaceRefOrValue

Description: Place defines the places where the products order must be done.

Name	Type	Description	MEF 57.2
@schemaLocation	uri	A URI 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 57.2
@type*	string	This field is used as discriminator and is used between different place representations. This type might discriminate for additional related place as defined in '@schemaLocation'.	Not represented in MEF 57.2
role*	string	Role of this place	RelatedPlaceRefOrValue

7.2.4.2. 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 (Sn 8.9.2)

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	Description	MEF 57.2
city*	string	The city that the address is in	City
country*	string	Country that the address is in	Country
geographicSubAddress	GeographicSubAddress		Not represented in MEF 57.2
locality	string	The locality that the address is in	Locality
postcode	string	Descriptor for a postal delivery area, used to speed and simplify the delivery of mail (also known as zip code)	Postal Code
postcodeExtension	string	An extension of a postal code. E.g. the part following the dash in a US urban property address	Postal Code Extension
stateOrProvince	string	The State or Province that the address is in	State Or Province
streetName*	string	Name of the street or other street type	Street Name
streetNr	string	Number identifying a specific property on a public street. It may be combined with streetNrLast for ranged addresses. MEF 79 defines it as required however as in certain countries it is not used we make it optional in API.	Street Number

Name	Type	Description	MEF 57.2
streetNrLast	string	Last number in a range of street numbers allocated to a property	Street Number Last
streetNrLastSuffix	string	Last street number suffix for a ranged address	Street Number Suffix Last
streetNrSuffix	string	The first street number suffix	Street Number Suffix
streetSuffix	string	A modifier denoting a relative direction	Street Suffix
streetType	string	The type of street (e.g., alley, avenue, boulevard, brae, crescent, drive, highway, lane, terrace, parade, place, tarn, way, wharf)	Street Type

7.2.4.3. 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	Description	MEF 57.2
addrLine1*	string	The first address line in a formatted address	Address Line 1
addrLine2	string	The second address line in a formatted address	Address Line 2
city*	string	The city that the address is in	City
country*	string	Country that the address is in	Country
locality	string	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	Descriptor for a postal delivery area, used to speed and simplify the delivery of mail (also known as ZIP code)	Postal Code
postcodeExtension	string	An extension of a postal code. E.g. the part following the dash in an US urban property address	Postal Code Extension
stateOrProvince	string	The State or Province that the address is in	State Or Province

7.2.4.4. Type MEFGeographicPoint

Description: A MEFGeographicPoint defines a geographic point through coordinates.

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	Description	MEF 57.2
spatialRef*	string	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	The latitude expressed in the format specified by the `spacialRef`	Latitude
y*	string	The longitude expressed in the format specified by the `spacialRef`	Longitude
z	string	The elevation expressed in the format specified by the `spacialRef`	Elevation

7.2.4.5. Type GeographicSubAddress

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

Name	Type	Description	MEF 57.2
buildingName	string	Allows for identification of places that require building name as part of addressing information	Building Name
levelNumber	string	Used where a level type may be repeated e.g. BASEMENT 1, BASEMENT 2	Level Number
levelType	string	Describes level types within a building	Level Type
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	Private Street Name
privateStreetNumber	string	Private streets numbers internal to a private street	Private Street Number
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 57.2

7.2.4.6. Type GeographicAddressRef

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

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	Description	MEF 57.2
href	string	Hyperlink to the referenced GeographicAddress. 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 57.2

Name	Type	Description	MEF 57.2
id*	string	Identifier of the referenced Geographic Address. This identifier is assigned during a successful address validation request (Geographic Address Validation API)	Fielded | Formatted | Geographic Address Label | Geographic Point Identifier

7.2.4.7. Type GeographicSiteRef

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

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	Description	MEF 57.2
href	string	Hyperlink to the referenced GeographicSite. 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 57.2
id*	string	Identifier of the referenced Geographic Site.	Site Identifier

7.2.4.8. 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	Description	MEF 57.2
externalReferenceId*	string	A reference to an address by id	Administrative Authority Address Label
externalReferenceType*	string	Uniquely identifies the authority that specifies the addresses reference and/or its type (if the authority specifies more than one type of address). The value(s) to be used are to be agreed during the onboarding. For North American providers this would normally be CLLI (Common Language Location Identifier) code.	Administrative Authority

7.2.4.9. Type MEFSubUnit

Description: Allows for sub unit identification

Name	Type	Description	MEF 57.2
subUnitNumber*	string	The discriminator used for the subunit, often just a simple number but may also be a range.	Sub Unit Name

Name	Type	Description	MEF 57.2
subUnitType*	string	The type of subunit e.g.BERTH, FLAT, PIER, SUITE, SHOP, TOWER, UNIT, WHARF.	Sub Unit Type

7.2.5. Cancel Product Order

7.2.5.1. Type CancelProductOrder_Create

Description: Request for cancellation an existing product order Skipped properties: id,href,state,effectiveCancellationDate

Name	Type	Description	MEF 57.2
cancellationReasonType	CancellationReasonType	Identifies the type of reason, Technical or Commercial, for the Cancellation Request	Cancellation Reason Type
cancellationReason	string	An optional attribute that allows the Buyer to provide additional detail to the Seller on their reason for cancelling the Prod-uct Order	Cancellation Reason
orderVersion*	string	The version of the Product Order. Set by the Buyer using Seller specified Product Order Version of the Product Order that is to be cancelled.	Product Order Version
note	Note[]		Note
relatedContactInformation*	RelatedContactInformation[]	Contact information of an individual or organization playing a role for this Cancel Product Order. The rule for mapping a represented attribute value to a \`role\` is to use the _lowerCamelCase_ pattern e.g. - Cancel Product Order Contact: \`role=cancelProductOrderContact\` - Seller Contact: \`role=sellerContact\`	Cancel Product Order Contact, Seller Contact
productOrder*	MEFProductOrderRef		Product Order Identifier

7.2.5.2. Type CancelProductOrder

Description: Request for cancellation an existing product order

Name	Type	Description	MEF 57.2

Name	Type	Description	MEF 57.2
id*	string	Unique identifier for the Cancel Product Order that is generated by the Seller when the Cancel Product Order request is accepted via an API.	Product Order Identifier
href	string	Hyperlink to the cancellation request. 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 57.2
cancellationReason	string	An optional attribute that allows the Buyer to provide additional detail to the Seller on their reason for cancelling the Product Order	Cancellation Reason
cancellationReasonType	CancellationReasonType	Identifies the type of reason, Technical or Commercial, for the Cancellation Request	Cancellation Reason Type
orderVersion*	string	The version of the Product Order. Set by the Buyer using Seller specified Product Order Version of the Product Order that is to be cancelled.	Product Order Version
note	Note[]		Note
productOrder*	MEFProductOrderRef		Product Order Identifier
relatedContactInformation*	RelatedContactInformation[]	Contact information of an individual or organization playing a role for this Cancel Product Order. The rule for mapping a represented attribute value to a \`role\` is to use the _lowerCamelCase_ pattern e.g. - Cancel Request Product Order Contact: \`role=cancelRequestProductOrderContact\` - Seller Contact: \`role=sellerContact\`	Cancel Product Order Contact, Seller Contact
state*	MEFChargeableTaskStateType	The states as defined by TMF622 and extended to meet MEF requirements. These states are used to convey the Cancel Product Order status during the lifecycle of the Product Order.	Cancel Product Order State

7.2.5.3. `enum` CancellationReasonType

Description: Identifies the type of reason, Technical or Commercial, for the Cancellation Request

Value	MEF 57.2	Description
technical	TECHNICAL	
commercial	COMMERCIAL	

7.2.5.4. Type MEFProductOrderRef

Description: Holds the MEF Product Order reference

Name	**Type**	**Description**	**MEF 57.2**
productOrderId	string	Unique (within the ordering domain) identifier for the order that is generated by the seller when the order is initially accepted.	Product Order Identifier
productOrderHref	string	Hyperlink to access the order	Not represented in MEF 57.2

7.2.6. Charge

7.2.6.1. Type MEFCharge

Description: When non-recurring or updated recurring charges are identified by the Seller during their processing of a Product Order, the Seller must communicate these charges to the Buyer and the Buyer must respond to the Seller informing the Seller if they accept or reject each charge. The Seller indicates for each charge if the charge is Blocking or non-Blocking. If the Buyer rejects a Blocking Charge, the Seller will cancel that Product Order Item and any related Product Order Items. If the Buyer rejects a non-blocking Charge, the Seller may proceed with fulfillment of the Product Order Item.

Name	**Type**	**Description**	**MEF 57.**
id*	string	A unique identifier of the Charge	Charge Identifier
href	string	Hyperlink to the Charge. 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 represent in MEF 57.2
creationDate	date-time	Date that the Charge was created by the Seller.	Charge Creation Date

Name	Type	Description	MEF 57.
responseDueDate*	date-time	The date that the Buyer must respond to the Seller's Charge. If there is no response received by the Due Date the Seller will treat all charges as declined.	Response Due Date
cancelProductOrder	MEFCancelProductOrderRef	A reference to the Cancel Product Order request that is cause of the Charge. Required if the Charge was caused by a Cancel Product Order.	
chargeItem*	MEFChargeItem[]		Charge Items

Name	Type	Description	MEF 57.
modifyProductOrderItemCompletionDate	MEFModifyProductOrderItemCompletionDateRef	A reference to the Modify Product Order Item Completion Date request that is cause of the Charge. Required if the Charge was caused by a Modify Product Order Item Completion Date request.	
productOrderItem*	MEFProductOrderItemRef	A reference to the Product Order Item that the Charge is related to.	
state*	MEFChargeStateType	The state of the Charge	Charge State

7.2.6.2. Type MEFCharge_Update

Description: A subset of MEFCharge

Name	Type	Description	MEF 57.2
chargeItem*	MEFChargeItem_Update[]		Charge Items

7.2.6.3. `enum` MEFChargeActivityType

Description: Possible values for the state of the Charge Activity Type

Value	MEF 57.2
new	NEW
change	CHANGE

7.2.6.4. `enum` MEFChargeStateType

Description: Possible values for the state of the Charge

Value	MEF 57.2
awaitingResponse	AWAITING_RESPONSE
completed	COMPLETED
withdrawnBySeller	WITHDRAWN_BY_SELLER

7.2.6.5. Type MEFChargeItem

Description: A single component part of the Charge

Name	Type	Description	MEF 57.2
id*	string	An identifier that is unique among Charge	Charge Item Identifier
chargeType	MEFPriceType	The state of the Charge	Charge Type
description	string	A description of the Charge	Charge Description
activityType	MEFChargeActivityType	Indicates if this is a new charge or a change to a charge provided in a Quote.	Charge Activity Type
recurringChargePeriod	MEFChargePeriod	Used for a recurring Charge Item with a chargeType = recurring to indicate the period	Recurring Charge Period
blocking	boolean	Indicates if rejecting the charge will cause the Seller to cancel the Product Order Item, or close the Cancel Product Order or Modify Product Order Item Completion Date without action.	Blocking
price	Price	The value of the Price associated with the charge	Charge Price
acceptanceIndicator	MEFAcceptedRejectedType	Indicates if this is a new charge or a change to a charge provided in a Quote.	Charge Acceptance Indicator
state	MEFChargeItemStateType	The state of the Charge Item	Charge Item State
unitOfMeasure	string	Unit of Measure if price depending on it is usage based (Gb, SMS volume, etc..)	Charge Item Price Unit Of Measure

7.2.6.6. Type MEFChargeItem_Update

Description: A type used to perform Buyer's response to a Charge Item - to accept or reject it.

Name	Type	Description	MEF 57.2

Name	Type	Description	MEF 57.2
id*	string	An identifier that is unique among Charge	Charge Item Identifier
acceptanceIndicator*	MEFAcceptedRejectedType	Indicates if this is a new charge or a change to a charge provided in a Quote.	Charge Acceptance Indicator

7.2.6.7. enum MEFChargeItemStateType

Description: Possible values for the state of the Charge Item

Value	MEF 57.2
awaitingResponse	AWAITING_RESPONSE
acceptedByBuyer	ACCEPTED_BY_BUYER
declinedByBuyer	DECLINED_BY_BUYER
withdrawnBySeller	WITHDRAWN_BY_SELLER

7.2.6.8. Type MEFCancelProductOrderRef

Description: A reference to a Cancel Product Order instance

Name	Type	Description	MEF 57.2
id*	string	A unique identifier of the Cancel Product Order	
href	string	Hyperlink to access the Cancel Product Order	

7.2.6.9. Type MEFModifyProductOrderItemCompletionDateRef

Description: a reference to Modify Product Order Item Completion Date

Name	Type	Description	MEF 57.2
id*	string	A unique identifier of the Modify Product Order Item Completion Date	
href	string	Hyperlink to access the Modify Product Order Item Completion Date	

7.2.7. Modify Product Order Item Completion Date

7.2.7.1. Type MEFModifyProductOrderItemCompletionDate_Create

Description: A request initiated by the Buyer to modify the Requested Completion Date or the Expedite Indicator of a Product Order Item.

Name	Type	Description	MEF 57.2
expediteIndicator	boolean	Indicates that expedited treatment is requested. Set by the Buyer. Default Value = FALSE. If this is set to TRUE, the Buyer sets the Requested Completion Date to the expedited date	Product Order Item Expedite Indicator

Name	Type	Description	MEF 57.2
orderVersion*	string	The version of the Product Order. Set by the Buyer using Seller specified Product Order Version of the Product Order that is to be modified.	Product Order Version
requestedCompletionDate	date-time	Identifies the Buyer's desired due date (requested delivery date)	Product Order Item Requested Completion Date
productOrderItem*	MEFProductOrderItemRef	A reference to the Product Order Item to be modified.	Product Order Identifier, Product Order Item Identifier

7.2.7.2. Type MEFModifyProductOrderItemCompletionDate

Description: A response to a request initiated by the Buyer to modify the Requested Completion Date or the Expedite Indicator of a Product Order Item.

Name	Type	Description	MEF 57.
id*	string	Unique identifier for the MEFModifyProductOrderItemCompletionDate that is generated by the Seller when the MEFModifyProductOrderItemCompletionDate request is moved to the acknowledged state.	Modify Product Order Iter Completic Date Identifier
href	string	Hyperlink to the modification request. 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 represente in MEF 57.2
creationDate*	date-time	Date that the Modify Product Order Item Completion Date was created in the Seller's system.	Not represente in MEF 57.2
expediteIndicator*	boolean	Indicates that expedited treatment is requested. Set by the Buyer. Default Value = FALSE. If this is set to TRUE, the Buyer sets the Requested Completion Date to the expedited date	Product Order Iter Expedite Indicator
orderVersion*	string	The version of the Product Order. Set by the Buyer using Seller specified Product Order Version of the Product Order that is to be modified.	Product Order Version

Name	Type	Description	MEF 57.2
requestedCompletionDate	date-time	Identifies the Buyer's desired due date (requested delivery date)	Product Order Iter Requestec Completic Date
productOrderItem*	MEFProductOrderItemRef	A reference to the Product Order Item to be modified.	Product Order Identifier, Product Order Iter Identifier
state*	MEFChargeableTaskStateType	The state of the Modify Product Order Item Completion Date request	Modify Product Order Iter Completic Date State

7.2.8. Notification registration

Notification registration and management are done through `/hub` API endpoint. The below sections describe data models related to this endpoint.

7.2.8.1. Type EventSubscriptionInput

The `query` attribute is used to constrain the notification types that the Buyer is willing to receive to the callback endpoint. The `query` formatting complies to RCF3986 rfc3986 and TMF630. Every attribute defined in the Event model (from notification API) can be used in the `query`. Example:

```
"query":"eventType=productOrderStateChangeEvent"
```

If the Buyer wishes to subscribe to 2 different types of events, there are 2 possible syntax variants:

- `eventType=productOrderStateChangeEvent,productOrderItemStateChangeEvent` or
- `eventType=productOrderStateChangeEvent&eventType=productOrderItemStateChangeEvent`

Description: This class is used to register for Notifications.

Name	Type	Description
query	string	This attribute is used to define to which type of events to register to. Example: "query":"eventType = productOrderStateChangeEvent". To subscribe for more than one event type, put the values separated `eventType=productOrderStateChangeEvent,productOrderItemStateChangeEvent`. The possible valu by 'ProductOrderEventType', `CancelProductOrderEventType` in productOrderNotification.api.yaml. is treated as specifying no filters - ending in subscription for all event types.
callback*	string	This callback value must be set to *host* property from Buyer Product Order Notification API (productOrderNotification.api.yaml). This property is appended with the base path and notification re specified in that API to construct an URL to which notification is sent. E.g. for "callback": "https://buyer.co/listenerEndpoint", the product order state change event notification will be sent to: `https://buyer.co/listenerEndpoint/mefApi/sonata/productOrderNotification/v7/listener/productOrderS

7.2.8.2. Type EventSubscription

Description: This resource is used to respond to notification subscription.

Name	Type	Description	MEF 57.2
query	string	The value provided by the Buyer in \`EventSubscriptionInput\` during notification registration	Not represented in MEF 57.2
callback*	string	The value provided by the Buyer in \`EventSubscriptionInput\` during notification registration	Notification Target Information
id*	string	An identifier of this Event Subscription assigned by the Seller when a resource is created.	Not represented in MEF 57.2

7.2.9. Common

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

7.2.9.1. Type Duration

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

Name	Type	Description	MEF 57.2
amount*	integer	Duration (number of seconds, minutes, hours, etc.)	Duration Value
units*	TimeUnit	Time unit type	Duration Unit

7.2.9.2. `enum` MEFAcceptedRejectedType

Description: Indicator of acceptance

Value	MEF 57.2	Description
accepted	ACCEPTED	
rejected	REJECTED	

7.2.9.3. Type MEFBillingAccount

Description: References the billing arrangement that a buyer has with a seller that provides products to the customer.

Name	Type	Description	MEF 57.2
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
billingContact	RelatedContactInformation	Contact allow to capture contact information. It is used to capture billing account contact information.	Billing Contact
agreementName	string	The name of the Agreement which is referenced for the Product Order Item.	Agreement Name

7.2.9.4. `enum` MEFBuyerSellerType

Description: Indicates if the note is from Buyer or Seller.

Value	MEF 57.2	Description
buyer	BUYER	
seller	SELLER	

7.2.9.5. `enum` MEFChargeableTaskStateType

Description: The states as defined by TMF622 [11] and extended to meet MEF requirements.

Value	MEF 57.2	Description
acknowledged	ACKNOWLEDGED	
done	DONE	
done.declined	DONE.DECLINED	
inProgress.assessingCharge	IN_PROGRESS.ASSESSING_CHARGE	
rejected	REJECTED	

7.2.9.6. `enum` MEFChargePeriod

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

Value	MEF 57.2	Description
hour	HOUR	
day	DAY	
week	WEEK	
month	MONTH	
year	YEAR	

7.2.9.7. `enum` MEFEndOfTermAction

Description: The action the Seller will take once the term expires.

Value	MEF 57.2	Description
roll	ROLL	
autoDisconnect	AUTO_DISCONNECT	
autoRenew	AUTO_RENEW	

7.2.9.8. Type MEFItemTerm

Description: The term of the Item

Name	Type	Description	MEF 57.2

Name	Type	Description	MEF 57.2
description	string	Description of the term	Quote Item Term Description
duration	Duration	Duration of the term	Quote Item Term Duration
endOfTermAction	MEFEndOfTermAction	The action that needs to be taken by the Seller once the term expires	Seller End of Term Action
name	string	Name of the term	Quote Item Term Name
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.	Roll Interval

7.2.9.9. `enum` MEFPriceType

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

Value	MEF 57.2	Description
recurring	RECURRING	
nonRecurring	NON_RECURRING	
usageBased	USAGE_BASED	

7.2.9.10. Type Money

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

Name	Type	Description	MEF 57.2
unit	string	Currency (ISO4217 norm uses 3 letters to define the currency)	Currency
value	float	A positive floating point number	Value

7.2.9.11. Type Note

Description: Extra information about a given entity. Only useful in processes involving human interaction. Not applicable for the automated process.

Name	Type	Description	MEF 57.2
date*	date-time	Date the Note was created	Note Date
author*	string	Author of the note	Note Author
id*	string	Identifier of the note within its containing entity (may or may not be globally unique, depending on provider implementation)	Not represented in MEF 57.2
source*	MEFBuyerSellerType	Indicates if the note is from Buyer or Seller	Note source

Name	Type	Description	MEF 57.2
text*	string	Text of the note	Note Text

7.2.9.12. 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 57.2
taxRate	float	Price Tax Rate. Unit: [%]. E.g. value 16 stand 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

7.2.9.13. Type RelatedContactInformation

Description: Contact information of an individual or organization playing a role for this Order Item. 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 57.2
emailAddress*	string	Email address	Contact email Address
name*	string	Name of the contact	Contact Name
number*	string	Phone number	Contract Phone Number
numberExtension	string	Phone number extension	Contract Phone Number Extension
organization	string	The organization or company that the contact belongs to	Contact Organization
role*	string	A role the party plays in a given context.	Not represented in MEF 57.2
postalAddress	FieldedAddress	Identifies the postal address of the person or office to be contacted.	Contact Postal Address

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` equal to `sellerContact`
- Buyer Contact Information: `role` equal to `buyerContactInformation`

7.2.9.14. Type TerminationError

Description: This indicates an error that caused an Item to be terminated. The code and propertyPath should be used like in Error422.

Name	Type	Description
code	string	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).
value	string	Text to describe the reason of the termination.

7.2.9.15. `enum` TimeUnit

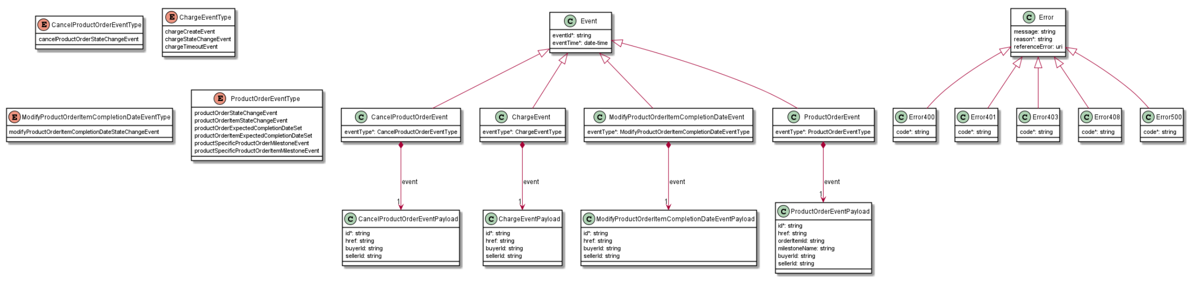
Description: Represents a unit of time. Reference: MEF 57.2 (Sn 9.22)

Value	MEF 57.2
calendarMonths	CALENDAR_MONTHS
calendarDays	CALENDAR_DAYS
calendarHours	CALENDAR_HOURS
calendarMinutes	CALENDAR_MINUTES
businessDays	BUSINESS_DAYS
businessHours	BUSINESS_HOURS
businessMinutes	BUSINESS_MINUTES

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

7.3. Notification API Data model

Figure 33 presents the Product Order Management Notification data model. The data types, requirements related to them and mapping to MEF 57.2 are discussed later in this section.



E CancelProductOrderEventType
cancelProductOrderStateChangeEvent
E ChargeEventType
chargeCreateEvent
chargeStateChangeEvent
chargeTimeoutEvent
E ModifyProductOrderItemCompletionDateEventType
modifyProductOrderItemCompletionDateStateChangeEvent
E ProductOrderEventType
productOrderStateChangeEvent
productOrderItemStateChangeEvent
productOrderExpectedCompletionDateSet
productOrderItemExpectedCompletionDateSet
productSpecificProductOrderMilestoneEvent
productSpecificProductOrderItemMilestoneEvent
C Event
eventId*: string
eventTime*: date-time
C CancelProductOrderEvent
eventType*: CancelProductOrderEventType
C ChargeEvent
eventType*: ChargeEventType
C ModifyProductOrderItemCompletionDateEvent
eventType*: ModifyProductOrderItemCompletionDateEventType
C ProductOrderEvent
eventType*: ProductOrderEventType
event
1
C CancelProductOrderEventPayload
id*: string
href: string
buyerId: string
sellerId: string
C ChargeEventPayload
id*: string
href: string
buyerId: string
sellerId: string
C ModifyProductOrderItemCompletionDateEventPayload
id*: string
href: string
buyerId: string
sellerId: string
C ProductOrderEventPayload
id*: string
href: string
orderItemId: string
milestoneName: string
buyerId: string
sellerId: string
C Error
message: string
reason*: string
referenceError: uri
C Error400
code*: string
C Error401
code*: string
C Error403
code*: string
C Error408
code*: string
C Error500
code*: string

Figure 33. Product Order Management Notification Data Model

This data model is used to construct requests and responses of the API endpoints described in Section 5.2.2.

7.3.1. Type Event

Description: Event class is used to describe information structure used for notification.

Name	Type	Description	MEF 57.2
eventId*	string	Id of the event	Not represented in MEF 57.2
eventTime*	date-time	Date-time when the event occurred	Not represented in MEF 57.2

7.3.2. Type ProductOrderEvent

Description:

Inherits from:

- Event

Name	Type	Description	MEF 57.2
eventType*	ProductOrderEventType	Indicates the type of the event.	Not represented in MEF 57.2
event*	ProductOrderEventPayload	A reference to the Product Order that is source of the notification.	Not represented in MEF 57.2

7.3.3. Type ProductOrderEventPayload

Description: The identifier of the Product Order and/or Order Item being subject of this event.

Name	Type	Description	MEF 57.2
id*	string	ID of the Product Order	Not represented in MEF 57.2
href	string	Hyperlink to access the Product Order	Not represented in MEF 57.2
orderItemId	string	ID of the Product Order Item (within the Product Order) which state change triggered the event. Mandatory for Product Order Item related events.	Not represented in MEF 57.2
milestoneName	string	The name of the Milestone that was reached by give Product Order or Product Order Item. Mandatory for Product Specific Milestone reached events.	Not represented in MEF 57.2
buyerId	string	The unique identifier of the organization that is acting as the a Buyer. MUST be specified in the request only when the responding represents more than one Buyer. Reference: MEF 79 (Sn 8.8)	Not represented in MEF 57.2

Name	Type	Description	MEF 57.2
sellerId	string	The unique identifier of the organization that is acting as the Seller. MUST be specified in the request only when requester entity represents more than one Seller. Reference: MEF 79 (Sn 8.8)	Not represented in MEF 57.2

7.3.4. enum ProductOrderEventType

Description: Indicates the type of Product Order event.

Value	MEF 57.2
productOrderStateChangeEvent	PRODUCT_ORDER_STATE_CHANGE_EVENT
productOrderItemStateChangeEvent	PRODUCT_ORDER_ITEM_STATE_CHANGE_EVENT
productOrderExpectedCompletionDateSet	PRODUCT_ORDER_EXPECTED_COMPLETION_DATE_SET
productOrderItemExpectedCompletionDateSet	PRODUCT_ORDER_ITEM_EXPECTED_COMPLETION_DATE_
productSpecificProductOrderMilestoneEvent	PRODUCT_SPECIFIC_PRODUCT_ORDER_MILESTONE_EVEN
productSpecificProductOrderItemMilestoneEvent	PRODUCT_SPECIFIC_PRODUCT_ORDER_ITEM_MILESTONE_

7.3.5. Type CancelProductOrderEvent

Description:

Inherits from:

- Event

Name	Type	Description	MEF 57.2
eventType*	CancelProductOrderEventType	Indicates the type of the event.	Not represented in MEF 57.2
event*	CancelProductOrderEventPayload	A reference to the object that is source of the notification.	Not represented in MEF 57.2

7.3.6. Type CancelProductOrderEventPayload

Description: The identifier of the Cancel Product Order being subject of this event.

Name	Type	Description	MEF 57.2
id*	string	ID of the Cancel Product Order	Not represented in MEF 57.2
href	string	Hyperlink to access the Cancel Product Order	Not represented in MEF 57.2

Name	Type	Description	MEF 57.2
buyerId	string	The unique identifier of the organization that is acting as the a Buyer. MUST be specified in the request only when the responding represents more than one Buyer. Reference: MEF 79 (Sn 8.8)	Not represented in MEF 57.2
sellerId	string	The unique identifier of the organization that is acting as the Seller. MUST be specified in the request only when requester entity represents more than one Seller. Reference: MEF 79 (Sn 8.8)	Not represented in MEF 57.2

7.3.7. enum CancelProductOrderEventType

Description: Indicates the type of Cancel Product Order event.

Value	MEF 57.2
cancelProductOrderStateChangeEvent	CANCEL_PRODUCT_ORDER_STATE_CHANGE_EVENT

7.3.8. Type ModifyProductOrderItemCompletionDateEvent

Description:

Inherits from:

- Event

Name	Type	Description	MEF 57.2
eventType*	ModifyProductOrderItemCompletionDateEventType	Indicates the type of the event.	Not represented in MEF 57.2
event*	ModifyProductOrderItemCompletionDateEventPayload	A reference to the object that is source of the notification.	Not represented in MEF 57.2

7.3.9. Type ModifyProductOrderItemCompletionDateEventPayload

Description: The identifier of the Modify Product Order Item Completion Date being subject of this event.

Name	Type	Description	MEF 57.2
id*	string	ID of the Modify Product Order Item Completion Date	Not represented in MEF 57.2
href	string	Hyperlink to access the Modify Product Order Item Completion Date	Not represented in MEF 57.2

Name	Type	Description	MEF 57.2
buyerId	string	The unique identifier of the organization that is acting as the a Buyer. MUST be specified in the request only when the responding represents more than one Buyer. Reference: MEF 79 (Sn 8.8)	Not represented in MEF 57.2
sellerId	string	The unique identifier of the organization that is acting as the Seller. MUST be specified in the request only when requester entity represents more than one Seller. Reference: MEF 79 (Sn 8.8)	Not represented in MEF 57.2

7.3.10. enum ModifyProductOrderItemCompletionDateEventType

Description: Indicates the type of Modify Product Order Item Completion Date event.

Value	MEF 57.2
modifyProductOrderItemCompletionDateStateChangeEvent	MODIFY_PRODUCT_ORDER_ITEM_COMPLETION_I

7.3.11. Type ChargeEvent

Description:

Inherits from:

- Event

Name	Type	Description	MEF 57.2
eventType*	ChargeEventType	Indicates the type of the event.	Not represented in MEF 57.2
event*	ChargeEventPayload	A reference to the object that is source of the notification.	Not represented in MEF 57.2

7.3.12. Type ChargeEventPayload

Description: The identifier of the Charge being subject of this event.

Name	Type	Description	MEF 57.2
id*	string	ID of the Charge	Not represented in MEF 57.2
href	string	Hyperlink to access the Charge	Not represented in MEF 57.2
buyerId	string	The unique identifier of the organization that is acting as the a Buyer. MUST be specified in the request only when the responding represents more than one Buyer. Reference: MEF 79 (Sn 8.8)	Not represented in MEF 57.2

Name	Type	Description	MEF 57.2
sellerId	string	The unique identifier of the organization that is acting as the Seller. MUST be specified in the request only when requester entity represents more than one Seller. Reference: MEF 79 (Sn 8.8)	Not represented in MEF 57.2

7.3.13. `enum` ChargeEventType

Description: Indicates the type of Charge event.

Value	MEF 57.2
chargeCreateEvent	CHARGE_CREATE_EVENT
chargeStateChangeEvent	CHARGE_STATE_CHANGE_EVENT
chargeTimeoutEvent	CHARGE_TIMEOUT_EVENT

8. References

- [ISO4217] Currency Codes International Standards Organization ISO 4217:2015, 2015
- [JS] JsonSchema specifications
- [MEF55.1] MEF 55.1 Lifecycle Service Orchestration (LSO): Reference Architecture and Framework, February 2021
- [MEF57.2] MEF 57.2 Product Order Management Requirements and Use Cases, Product Order Management, Draft (R2), May 2021
- [MEF79] MEF 79, Address, Service Site, and Product Offering Qualification Management, Requirements and Use Cases, November 2019
- [MEF80] MEF 80, Quote Management Requirements and Use Cases, Draft (R6), May 2021
- [MEF106] MEF W106 LSO Sonata Product Specification - Access E-Line - Product Schema Guide, Working Draft, June 2021
- [OAS-V3] Open API 3.0, February 2020
- [REST] Chapter 5: Representational State Transfer (REST) Fielding, Roy Thomas, Architectural Styles and the Design of Network-based Software Architectures (Ph.D.).
- [RFC3986] RFC 3986 Uniform Resource Identifier (URI): Generic Syntax, January 2005
- [RFC7231] RFC 7231, Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content, June 2014 https://tools.ietf.org/html/rfc7231
- [TMF622] TMF 622 Product Ordering API REST Specification R19.0.1, November 2019
- [TMF630] TMF 630 TMF630 API Design Guidelines 4.0.1