Mplify Standard

Mplify 87.1

LSO Cantata and LSO Sonata Product Offering Qualification API - Developer Guide

November 2025

Disclaimer

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

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

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

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

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

Copyright

Table of Contents

List of Contributing Members

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

Member
Amartus
Colt Technology Services
Proximus

Table 1. Contributing Members

1. Abstract

This standard is intended to assist in the implementation of the Product Offering Qualification (POQ) function defined for the LSO Cantata and LSO Sonata Interface Reference Points (IRPs), for which requirements and use cases are defined in Mplify 79.1 *Product Offering Qualification Management Business Requirements and Use Cases* [Mplify 79.1].

POQ allows the Buyer to determine whether it is feasible for the Seller to deliver a particular Product Offering with a given configuration to a particular place if applicable.

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

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

commit id: aaa03d484f98664a5a14f4f54f47b675d7efb3b8

- `productApi/serviceability/offeringQualification/productOfferingQualificationManagement.api.yaml`
- `productApi/serviceability/offeringQualification/productOfferingQualificationNotification.api.yaml`

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

commit id: 83d6edd0c70386058a9af6e677c069b498671da7

- `productApi/serviceability/offeringQualification/productOfferingQualificationManagement.api.yaml`
- `productApi/serviceability/offeringQualification/productOfferingQualificationNotification.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 Mplify or external documents.

In addition, terms defined in the standards referenced below are included in this document by reference and are not repeated in the table below:

- MEF 55.1
- MEF 55.1.1
- Mplify 150

Term	Description	Reference
POQ	Product Offering Qualification	[Mplify79.1]
POQ Item	Product Offering Qualification Item	[Mplify79.1]

Table 2. Abbreviations

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]
Build	Build is when a Seller or Seller's 3rd Party needs to perform additional physical work to connect to the Installation Place or Service Site. Build work can require approvals from local government, landlords, or legal teams. When Build is needed during fulfillment, this may result in an extended cycle time. The following requirements are associated with a POQ being classed as Build: Wayleave is needed (legal agreements) Installation of fiber Installation of radio antenna Installation of inside wiring from common equipment area to customer New network entry point Digging and additional approvals from concerned authorities	[Mplify79.1]
Buyer	In the context of this document, denotes the organization acting as the customer in a transaction over a Cantata (Customer <-> Service Provider) or Sonata (Service Provider <-> Partner) Interface Reference Point.	This document; adapted from [MEF55.1.1]
Deferred Response	A Seller's response to a Buyer's request whereby the Seller immediately acknowledges that the request was received, and, over time, sends notifications to update the Buyer on the status and results of the request (assuming the Buyer has subscribed	[Mplify79.1]

	to receive the notifications). The Buyer can also poll the Seller for the results and status associated with the request.	
End Customer	The actual user of the Product that contracts for the Product with the Buyer or the Buyer's representative.	[Mplify79.1]
Immediate Response	A Seller's response to the Buyer whereby the Seller responds immediately with the results of the request or indicates that the request cannot be processed. The maximum time to provide an Immediate Response is for further study but is expected to be less than 30 seconds.	[Mplify79.1]
Installation Interval	The estimated minimum interval that the Seller requires in their standard process to complete the delivery of a Product from the time the order is placed, and any precedents have been completed.	[Mplify79.1]
OFF-NET with Build	The place does not have an existing connection to the Seller's network, but the Seller can connect via a partner who is willing to build out their network to connect to the place (refer to the definition of Build). Some providers may refer to this as Near-Net.	[Mplify79.1]
OFF-NET without Build	A place that does not have an existing connection to the Seller's network, but the Seller can connect via a 3rd party's existing connection. This may require augmentation of additional equipment, but no Build is needed for the connection.	[Mplify79.1]
ON-NET with Build	A place that does not have an existing connection to the Seller's network, but to which the Seller is willing to build out their network to connect to the place (refer to the definition of Build). Some providers may refer to this as Near-Net.	[Mplify79.1]
ON-NET without Build	A place that has an existing connection to the Seller's network. This may require the augmentation of additional equipment.	[Mplify79.1]
OpenAPI	RESTful API Documentation Specification for machine-readable interface files for describing, producing, consuming, and visualizing RESTful web services.	[OAS-V3]
Product Offering Qualification	One or more POQ Items formulated into a request made by a Buyer to a Seller.	[Mplify79.1]
Product Offering Qualification Item	An individual article included in a POQ that describes a Product of a particular type (Product Offering). The objective is to determine if it is feasible for the Seller to deliver this item as described and for the Seller to inform the Buyer of the estimated time interval to complete this delivery.	[Mplify79.1]
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.	[Mplify150]
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.	[Mplify150]
REST API	Representational State Transfer. REST provides a set of	[REST]

	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.	
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 Reference Point.	This document; adapted from [MEF55.1.1]

Table 3. Terminology

3. Compliance Levels

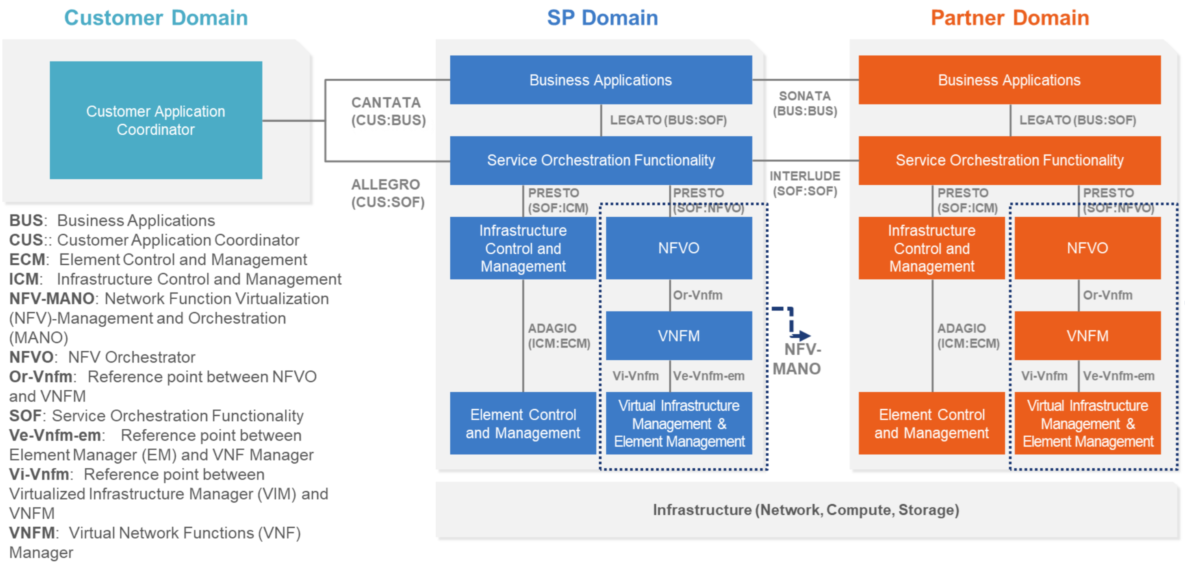
The key words **"MUST"**, **"MUST NOT"**, **"REQUIRED"**, **"SHALL"**, **"SHALL NOT"**, **"SHOULD"**, **"SHOULD NOT"**, **"RECOMMENDED"**, **"NOT RECOMMENDED"**, **"MAY"**, and **"OPTIONAL"** in this document are to be interpreted as described in BCP 14 ([RFC 2119], [RFC 8174]) when, and only when, they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

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

A paragraph preceded by **[CRa]<** specifies a conditional mandatory requirement that **MUST** be followed if the condition(s) following the "<" have been met. For example, **"[CR1]<[D38]"** indicates that Conditional Mandatory Requirement 1 must be followed if Desirable Requirement 38 has been met. A paragraph preceded by **[CDb]<** specifies a Conditional Desirable Requirement that **SHOULD** be followed if the condition(s) following the "<" have been met. A paragraph preceded by **[COc]<**specifies a Conditional Optional Requirement that **MAY** be followed if the condition(s) following the "<" have been met.

4. Introduction

This standard specification document describes the Application Programming Interface (API) for Product Offering Qualification 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* [MEF 55.1]. 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
SONATA (BUS:BUS)
INTERLUDE (SOF:SOF)
PRESTO (SOF:ICM)
PRESTO (SOF:NFVO)
Infrastructure Control and Management
NFVO
Or-Vnfm
ADAGIO (ICM:ECM)
VNFM
Vi-Vnfm
Ve-Vnfm-em
Element Control and Management
Virtual Infrastructure Management & Element Management
NFV-MANO
Infrastructure (Network, Compute, Storage)
BUS: Business Applications
CUS:: Customer Application Coordinator
ECM: Element Control and Management
ICM: Infrastructure Control and Management
NFV-MANO: Network Function Virtualization (NFV)-Management and Orchestration (MANO)
NFVO: NFV Orchestrator
Or-Vnfm: Reference point between NFVO and VNFM
SOF: Service Orchestration Functionality
Ve-Vnfm-em: Reference point between Element Manager (EM) and VNF Manager
Vi-Vnfm: Reference point between Virtualized Infrastructure Manager (VIM) and VNFM
VNFM: Virtual Network Functions (VNF) Manager

Figure 1. The LSO Reference Architecture

Cantata and Sonata IRPs define pre-ordering and ordering operations that allow 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
- Quote
- Product Offering Availability and Pricing Discovery
- Product Inventory
- Product Ordering
- Trouble Ticketing
- Billing

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

- Section 4 gives a technical introduction to POQ functionality.
- Section 5 provides an overview of the operations, data models, and design patterns of API definition.
- Section 6 focuses on API interactions with the help of end-to-end sequence diagrams and usage examples.
- Section 7 complements Section 5 with an in-depth API description.

4.1. Description

The Product Offering Qualification (POQ) API allows a Buyer to:

- Determine whether it is feasible for the Seller to deliver a particular Product based on a Product Offering with a given configuration to a particular place (if applicable).
- Find out alternative Product Offerings if any are available
- Retrieve an overview of existing POQs
- Retrieve details of a specified POQ
- Register for and receive notifications

The API payloads exchanged between a Buyer and the Seller during the POQ execution consist of product-independent and product-specific parts. The product-independent part is defined in this standard. The product-specific part is defined in the product specifications of the concerned product. Both definitions must be used in combination to validate the correctness of the requests. Section 5.6 explains how to use product-specific definitions with the POQ API definition.

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 indicate a variable to be substituted with a correct value.

4.3. Relation to Other Documents

The requirements and use cases for POQ functionality are defined in Mplify 79.1 [Mplify79.1]. Product specifications are defined using JSON Schema (draft 7) standard [JS], whereas POQ API is defined using OpenAPI 3.0 standard [OAS-V3]. The product payloads exchanged through POQ endpoints must comply with respective product specifications. This standard is based on TMF 679 API as specified by *TMF679 Product Offering Qualification API REST Specification* [TMF679].

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 (Mplify product specification data model)

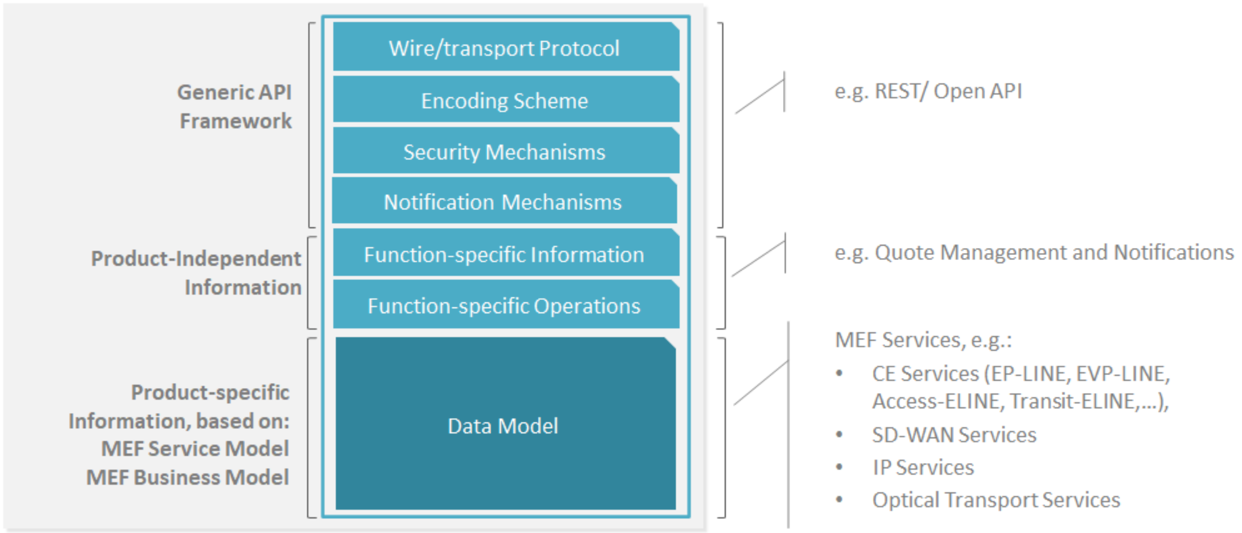

Generic API Framework
Wire/transport Protocol
Encoding Scheme
Security Mechanisms
Notification Mechanisms
e.g. REST/ Open API
Product-Independent Information
Function-specific Information
Function-specific Operations
e.g. Quote Management and Notifications
Product-specific Information, based on: MEF Service Model MEF Business Model
Data Model
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 part 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 describes Product Offering Qualification model and operations that allow performing qualifications of any product that is aligned with one of Mplify or custom product specifications. Finally, the product-specific information part of the framework focuses on Mplify product specifications that define business-relevant attributes and requirements for trading Mplify subscriber and Mplify operator services. This standard does not define Mplify product specifications, however, can be used along with any product specification defined by or compliant with Mplify.

4.5. High-Level Flow

Product Offering Qualification is part of a broader Cantata and Sonata process flow. Figure 3. below shows a high-level diagram to get a good understanding of the process and Product Offering Qualification's position within it.

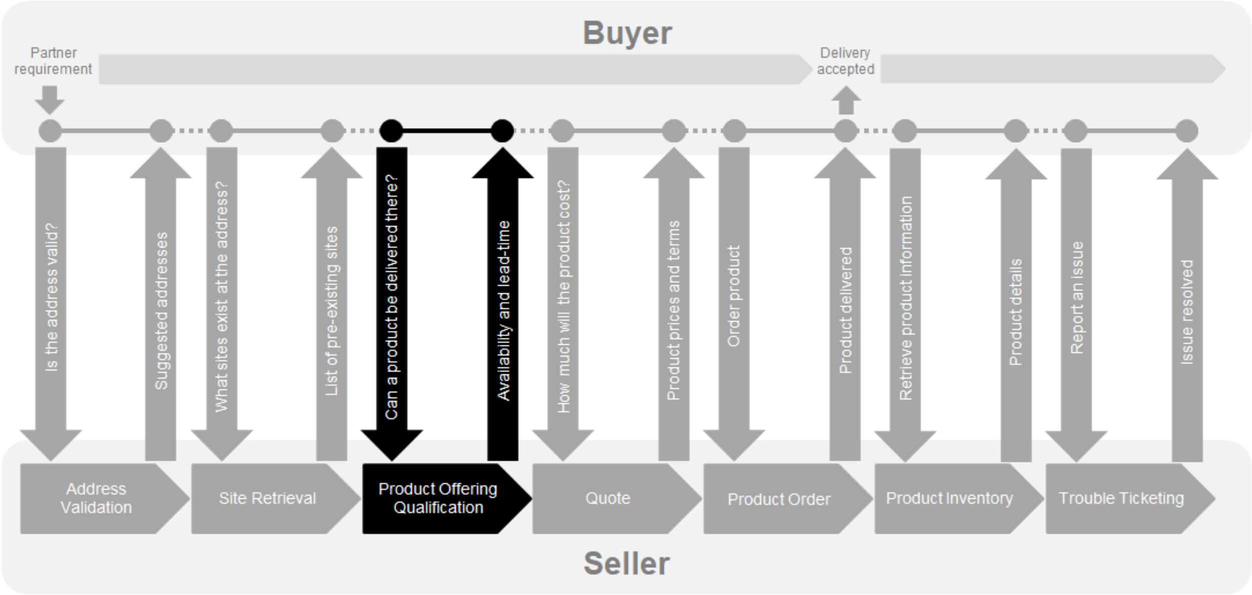


Figure 3. Cantata and Sonata Flow

- Address Validation:
    - Allows the Buyer to retrieve address information from the Seller, including exact formats, for Geographic Addresses known to the Seller.
- Site Retrieval:
    - Allows the Buyer to retrieve Geographic Site information including exact formats for Geographic 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 Geographic 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.

Note that this is not a comprehensive list of APIs available in Cantata and Sonata IRPs.

This document focuses on the function shown in black. The Buyer asks if a specific configuration of a Product Offering can be provided at a Geographic Address or Geographic Site by the Seller. The Seller responds with the technical feasibility of fulfilling a Product with the lead-time required to complete the fulfillment. This function is not required to be performed by the Buyer unless it is mandated by the Seller.

5. API Description

This section discusses the API structure and design patterns. It starts with a description of available REST endpoints. Then, an overview of the API data model is given together with the description of the extension 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. Pre-Requisites

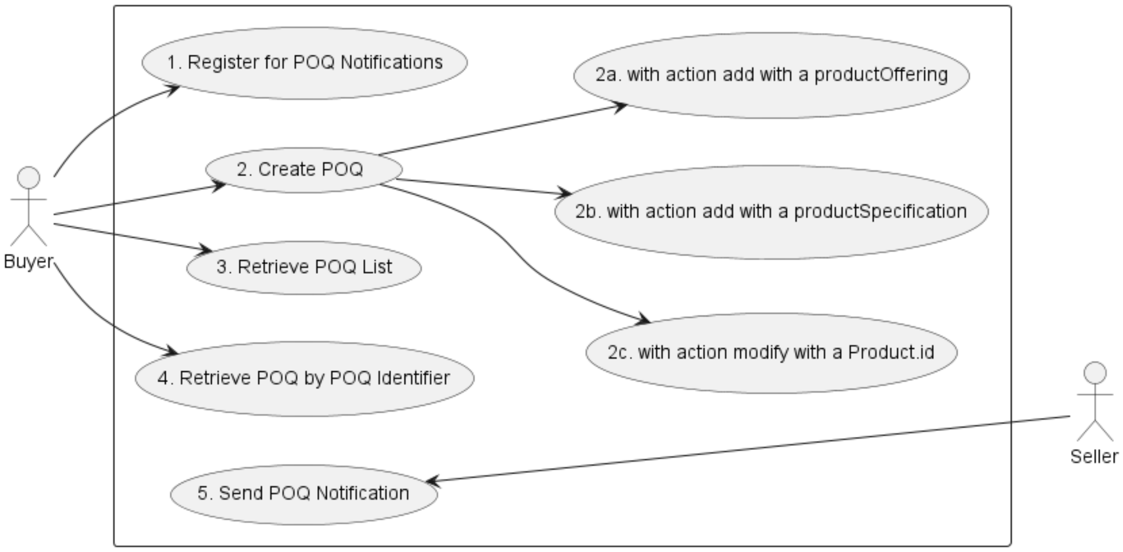
Prior to establishing an API communication, the Buyer and the Seller need to agree on the following, during the so-called onboarding process:

- Commercial contract and terms
- Authentication method
- Supported Geographic Address Representations (only when `GeographicAddress.id` is not supported)
- Supported response type (Immediate, Deferred, both)
- If the Alternate Product Proposal is of the same Product Specification as the Buyer's request
- Notification support

[R1] The Seller and the Buyer **MUST** agree on the method used to identify a place, either by `GeographicAddressRef`, `GeographicSiteRef` or `GeographicAddress_Query`. [Mplify79.1 R1]

5.2. Use cases

Figure 4 presents a use case diagram. It aims to help understand the endpoint mapping. Use cases are described extensively in chapter 6



1. Register for POQ Notifications
2a. with action add with a productOffering
2. Create POQ
2b. with action add with a productSpecification
Buyer
3. Retrieve POQ List
2c. with action modify with a Product.id
4. Retrieve POQ by POQ Identifier
Seller
5. Send POQ Notification

Figure 4. Use cases

5.3. Resource/endpoint Description

5.3.1. Seller Side Endpoints

Base URL for Cantata: `https://{{serverBase}}:{{port}}{{?/seller_prefix}}/mefApi/cantata/productOfferingQualification/v2/`

Base URL for Sonata: `https://{{serverBase}}:{{port}}{{?/seller_prefix}}/mefApi/sonata/productOfferingQualification/v8/`

The following API endpoints are implemented by the Seller and allow the Buyer to send POQ create requests, retrieve existing POQs or POQ details, and manage notification registrations. The endpoints and corresponding data model are defined in `productApi/serviceability/offeringQualification/productOfferingQualificationManagement.api.yaml`.

API endpoint	Description	Mplify 79.1 Use case Mapping
`POST /productOfferingQualification`	A request initiated by the Buyer to determine whether the Seller can feasibly deliver a particular Product Offering(s) per desired configuration	UC 2: Create Product Offering Qualification (incl. 2a, 2b, 2c)
`GET /productOfferingQualification`	A request initiated by the Buyer to retrieve a list of POQs from the Seller based on a set of POQ filter criteria.	UC 3: Retrieve POQ List
`GET /productOfferingQualification/{{id}}`	A request initiated by the Buyer to retrieve full details of a single Product Offering Qualification based on a POQ identifier.	UC 4: Retrieve POQ by Identifier
`POST /hub`	A request initiated by the Buyer to instruct the Seller to send notifications of specified type(s)	UC 1: Register for POQ Notifications
`GET /hub/{{id}}`	A request initiated by the Buyer to retrieve the details of the notification subscription.	UC 1: Register for POQ Notifications
`DELETE /hub/{{id}}`	A request initiated by the Buyer to instruct the Seller to stop sending notifications.	UC 1: Register for POQ Notifications

Table 4. Seller side endpoints

[R2] A Buyer **MUST** be able to initiate Use Cases 2a, 2c, and 4. [Mplify79.1 R2]

[D1] A Buyer **SHOULD** be able to initiate Use Case 2b. [Mplify79.1 D1]

[D2] A Buyer **SHOULD** be able to initiate Use Case 1 and/or Use Case 3. [Mplify79.1 D2]

[R3] The Seller **MUST** be able to provide an Immediate Response or a Deferred Response for Use Cases 2a and 2c. [Mplify79.1 R3]

[CR1]<[D1] The Seller **MUST** be able to provide an Immediate Response or a Deferred Response for Use Case 2b. [Mplify79.1 CR1<D1]

[R4] If a Deferred Response is provided, the Seller **MUST** support the ability for the Buyer to register for Notifications (Use Case 1) and to generate Notifications (Use Case 5) when the Buyer has registered for them. [Mplify79.1 R4]

5.3.2. Buyer Side Endpoints

Base URL for Cantata:

`https://{{serverBase}}:{{port}}{{?/buyer_prefix}}/mefApi/cantata/productOfferingQualificationNotification/v2/`

Base URL for Sonata:

`https://{{serverBase}}:{{port}}{{?/buyer_prefix}}/mefApi/sonata/productOfferingQualificationNotification/v8/`

The following API endpoints are implemented by the Buyer and are used by the Seller to send POQ-related notifications. The endpoints and corresponding data model are defined in `productApi/serviceability/offeringQualification/productOfferingQualificationNotification.api.yaml`

API endpoint	Description	Mplify 79.1 Use case Mapping
`POST /listener/poqStateChangeEvent`	A request initiated by the Seller to notify Buyer on POQ state change	UC 5: Send POQ Notification
`POST /listener/poqItemStateChangeEvent`	A request initiated by the Seller to notify Buyer on POQ Item state change	UC 5: Send POQ Notification

Table 5. Buyer side endpoints

5.4. Specifying the Buyer ID and the Seller ID

A business entity willing to represent multiple Buyers or multiple Sellers must follow requirements of [Mplify 150] 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.

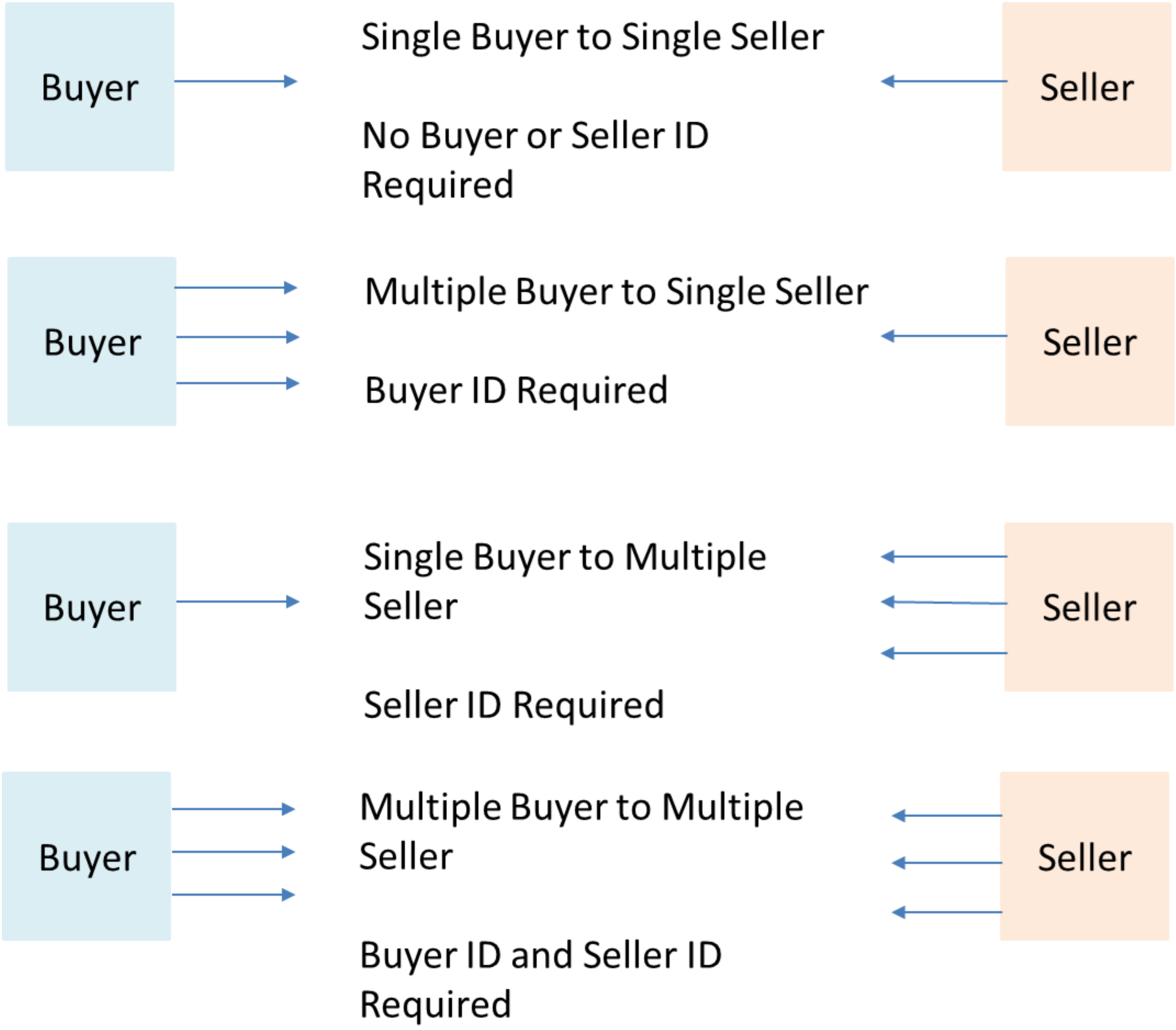

Buyer
Single Buyer to Single Seller
No Buyer or Seller ID Required
Seller
Buyer
Multiple Buyer to Single Seller
Buyer ID Required
Seller
Buyer
Single Buyer to Multiple Seller
Seller ID Required
Seller
Buyer
Multiple Buyer to Multiple Seller
Buyer ID and Seller ID Required
Seller

Figure 5. Buyer ID and Seller ID Examples

> As shown in Figure 5, if a Requesting Entity representing a single Buyer is doing business with a Responding Entity representing a single Seller, Buyer and Seller IDs are not required to be passed between the two entities. If a Requesting Entity representing more than one Buyer is doing business with a Responding Entity representing a single Seller, the Buyer ID is required to be passed between the two entities. If a Requesting Entity representing a single Buyer is doing business with a Responding entity representing multiple Sellers, the Seller ID is required to be passed between the two entities. If a Requesting Entity representing multiple Buyers is doing business with a Responding Entity representing multiple Sellers, both the Buyer ID and the Seller ID are required to be passed between the entities.
>
> 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 the Requesting Entity knows:
>
> - the list of Buyers the Requesting Entity represents when interacting with this Responding Entity; and
> - the list of Sellers that this Responding Entity represents to this Requesting Entity.
>
> It is also assumed that the Responding Entity knows:
>
> - the list of Sellers that this Responding Entity represents to this Requesting Entity and

> - the list of Buyers the Requesting Entity represents when interacting with this Responding Entity.

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

[R5] If the Requesting Entity has the authority to represent more than one Buyer the request **MUST** include buyerId that identifies the Buyer being represented. [Mplify150 R62]

[R6] If the Responding Entity represents more than one Seller to this Buyer the request **MUST** include sellerId that identifies the Seller with whom this request is associated. [Mplify150 R63]

5.5. Data Model - Key Entities

The sections below describe the most important entities (aka data types) from the data model which can be found in the API specification (definitions section). Each entity is a simple or composed type (using the allOf keyword for data types composition). A simple type defines a set of properties that might be of an object, primitive, or reference type.

[R7] If an entity is used in the request or response payload all properties marked as required **MUST** be provided.

A detailed description of the data types is provided in Section 7 and the OpenAPI definition. The examples that illustrate the usage of the data model are included in Section 6.

5.5.1. Request Key Entities

Figure 6 depicts a view of the data model that is used in the Product Offering Qualification request (POST /productOfferingQualification) that is sent by a Buyer (see Section 5.3.1 for details).

ProductOfferingQualification_Create is the root entity of a product offering qualification request. It contains one or more ProductOfferingQualificationItem_Create. A POQ item defines an item inquiry details (in MEFProductRefOrValue structure) and allows for the definition of related contact information (relatedContactInformation) or relations to other items (qualificationItemRelationship). MEFProductRefOrValue allows for the introduction of Mplify product-specific properties to the POQ payload. The extension mechanism is described in detail in Section 5.6.. Also, a MEFProductRefOrValue.place may be used to specify relations to place(s) or/and MEFProductRefOrValue.productRelationship to specify relations product that exists in the Seller's inventory.

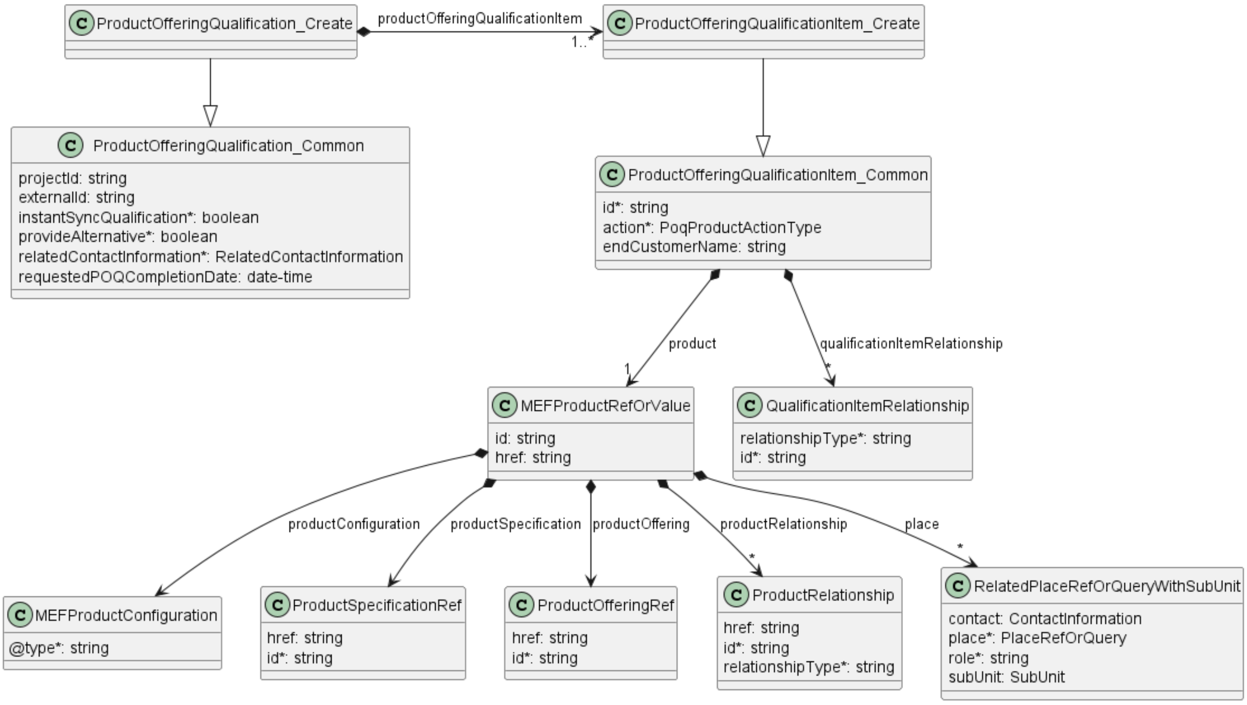

ProductOfferingQualification_Create
productOfferingQualificationItem
1..*
ProductOfferingQualificationItem_Create
ProductOfferingQualification_Common
projectId: string
externalId: string
instantSyncQualification*: boolean
provideAlternative*: boolean
relatedContactInformation*: RelatedContactInformation
requestedPOQCompletionDate: date-time
ProductOfferingQualificationItem_Common
id*: string
action*: PoqProductActionType
endCustomerName: string
product
1
qualificationItemRelationship
*
MEFProductRefOrValue
id: string
href: string
QualificationItemRelationship
relationshipType*: string
id*: string
productConfiguration
productSpecification
productOffering
productRelationship
*
place
*
MEFProductConfiguration
@type*: string
ProductSpecificationRef
href: string
id*: string
ProductOfferingRef
href: string
id*: string
ProductRelationship
href: string
id*: string
relationshipType*: string
RelatedPlaceRefOrQueryWithSubUnit
contact: ContactInformation
place*: PlaceRefOrQuery
role*: string
subUnit: SubUnit

Figure 6. Data model - request key entities

5.5.2. Response Key Entities

Figure 7 depicts a view of the data model that is used to provide a response to a Buyer's Product Offering Qualification request (`POST /productOfferingQualification`) or to retrieve POQ by an identifier (`GET /productOfferingQualification/{{id}}`).

`ProductOfferingQualification` is the root entity of a response and it is managed by the Seller. `ProductOfferingQualification` extends `ProductOfferingQualification_Common` (which represents Buyer's request) with a number of attributes, i.e. unique identifier or state information.

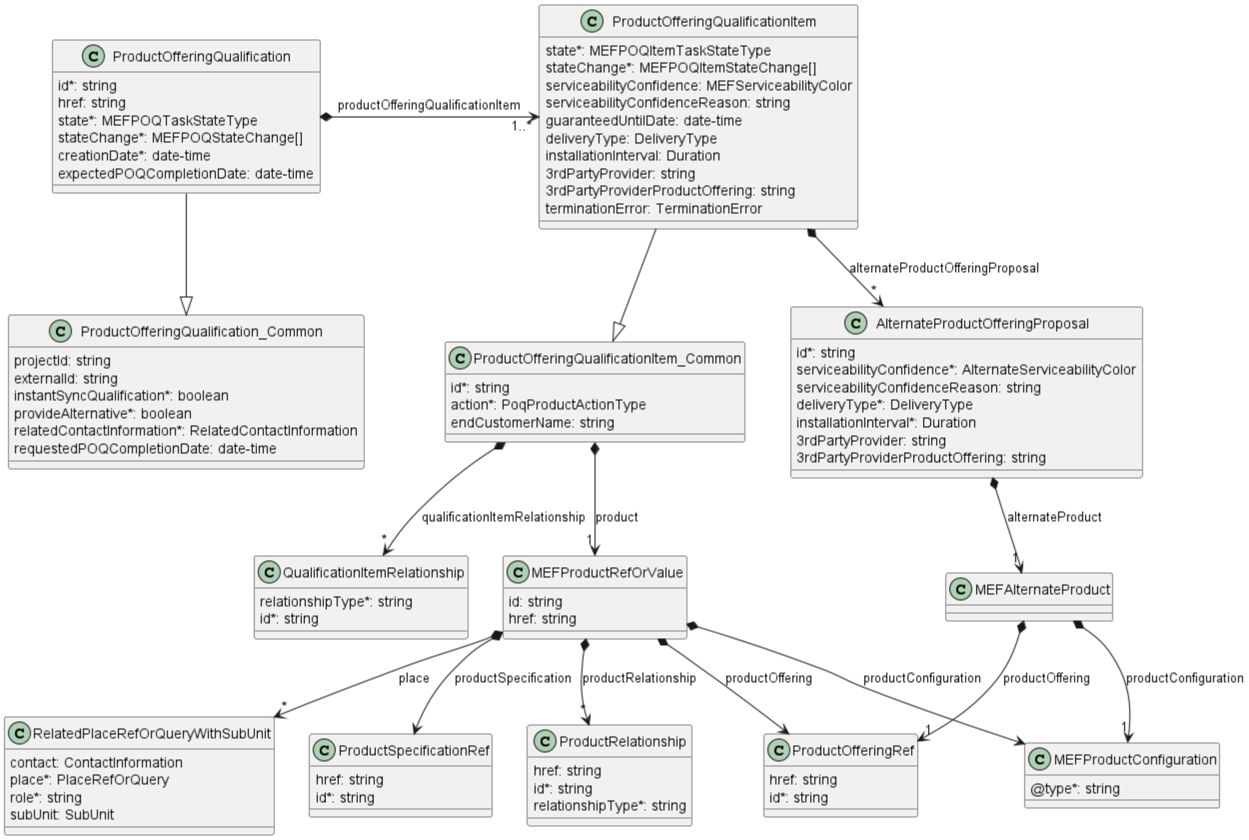

ProductOfferingQualification
id*: string
href: string
state*: MEFPOQTaskStateType
stateChange*: MEFPOQStateChange[]
creationDate*: date-time
expectedPOQCompletionDate: date-time
productOfferingQualificationItem
1..*
ProductOfferingQualificationItem
state*: MEFPOQItemTaskStateType
stateChange*: MEFPOQItemStateChange[]
serviceabilityConfidence: MEFServiceabilityColor
serviceabilityConfidenceReason: string
guaranteedUntilDate: date-time
deliveryType: DeliveryType
installationInterval: Duration
3rdPartyProvider: string
3rdPartyProviderProductOffering: string
terminationError: TerminationError
alternateProductOfferingProposal
*
ProductOfferingQualification_Common
projectId: string
externalId: string
instantSyncQualification*: boolean
provideAlternative*: boolean
relatedContactInformation*: RelatedContactInformation
requestedPOQCompletionDate: date-time
ProductOfferingQualificationItem_Common
id*: string
action*: PoqProductActionType
endCustomerName: string
AlternateProductOfferingProposal
id*: string
serviceabilityConfidence*: AlternateServiceabilityColor
serviceabilityConfidenceReason: string
deliveryType*: DeliveryType
installationInterval*: Duration
3rdPartyProvider: string
3rdPartyProviderProductOffering: string
qualificationItemRelationship
*
product
1
alternateProduct
1
QualificationItemRelationship
relationshipType*: string
id*: string
MEFProductRefOrValue
id: string
href: string
MEFAlternateProduct
place
*
productSpecification
productRelationship
*
productOffering
productConfiguration
productOffering
1
productConfiguration
1
RelatedPlaceRefOrQueryWithSubUnit
contact: ContactInformation
place*: PlaceRefOrQuery
role*: string
subUnit: SubUnit
ProductSpecificationRef
href: string
id*: string
ProductRelationship
href: string
id*: string
relationshipType*: string
ProductOfferingRef
href: string
id*: string
MEFProductConfiguration
@type*: string

Figure 7. Data model - response key entities

5.5.3. Product Offering Qualification Process Flow

This chapter specifies the POQ process states and possible transitions.

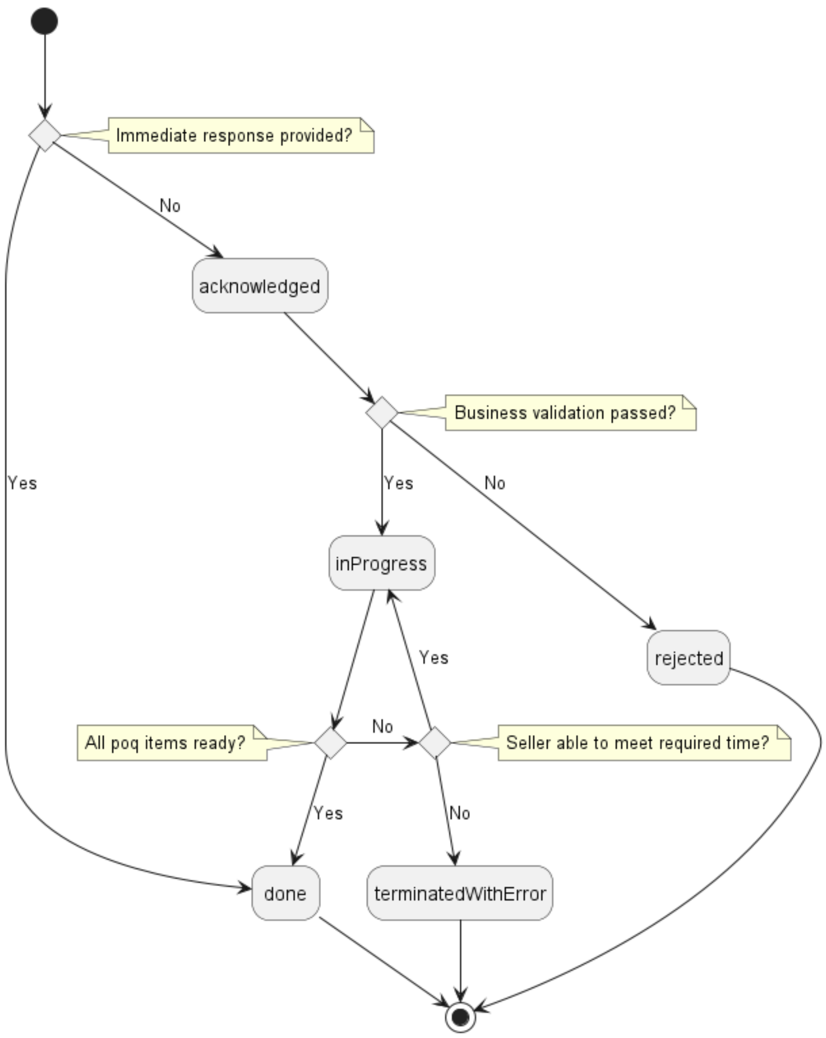


Figure 8. Product Offering Qualification Process Flow

Table 6 presents detailed descriptions of states and mapping between `MEFPOQTaskStateType` and Mplify 79.1.

MEFPOQTaskStateType	Mplify 79.1	Description
acknowledged	ACKNOWLEDGED	A request has been received by the Seller, has passed basic validation, and the id was assigned. For an Immediate response, the POQ moves directly to the `done` state and does not pass through `acknowledged`.
inProgress	IN_PROGRESS	The POQ is currently being worked by the Seller.
done	READY	The POQ has been internally approved by the Seller. Reached when all items are in a `done` state. It does not imply that the Seller can deliver all POQ Items in this POQ. It only means that the POQ has been completed.

MEFPOQTaskStateType	Mplify 79.1	Description
rejected	REJECTED	A POQ was submitted, and it has failed at least one of the business validation checks the Seller performs after it reached the `acknowledged` state.
terminatedWithError	UNABLE_TO_MEET_TIME	The Seller is unable to provide a response in the timeframe required by the Buyer (e.g. if an immediate response or a response date is set but cannot be met by the Seller).

Table 6. Product Offering Qualification States

If a POQ request does not pass an initial (syntax) validation the appropriate error response is returned to the Buyer. In case a POQ request fails business rules validation the HTTP response code is `422` and a list of validation problems is returned. Otherwise, the POQ is assigned a unique identifier. In case of a deferred response, the POQ gets the `acknowledged` state assigned and is returned in the response. In case of an immediate response, the POQ moves directly to `done` once the processing is done. POQ reaches the `done` state only if all items are in the `done` state as well. If an evaluation of any items concludes in the state `rejected` the POQ reaches the `rejected` state. If the POQ is processed asynchronously (Deferred Response) it can reach `terminatedWithError` if the Seller is not able to complete all items qualification by the deadline specified by the Buyer in `requestedPOQCompletionDate`.

[R8] If the Seller provides an Immediate Response, the Seller **MUST** support POQ `done` state and its associated state transitions as specified in Figure 8. [Mplify79.1 R94]

[R9] If the Seller provides a Deferred Response, the Seller **MUST** support all POQ States and their associated state transitions as specified in Figure 8. [Mplify79.1 R95]

[R10] The state of the POQ **MUST** be `done` only if *all* items are in `done` state. [Mplify79.1 R97]

[R11] The state of the POQ **MUST** be `terminatedWithError` when *at least one* item is in the `terminatedWithError` state.

[R12] The state of the POQ **MUST** be `rejected` when *at least one* item is in a `rejected` state.

[R13] When the POQ state moves to `terminatedWithError` or `rejected` or all POQ Items that are not `done` **MUST** move to `done.abandoned`. [Mplify79.1 R102]

[R14] The state of the POQ **MUST** be `inProgress` only if *at least one* item is in the `inProgress` state and *none* of the items is in `terminatedWithError` or `rejected` state. [Mplify79.1 R98]

5.5.4. Product Offering Qualification Item Process Flow

Figure 9. depicts a process flow for a POQ Item lifecycle.

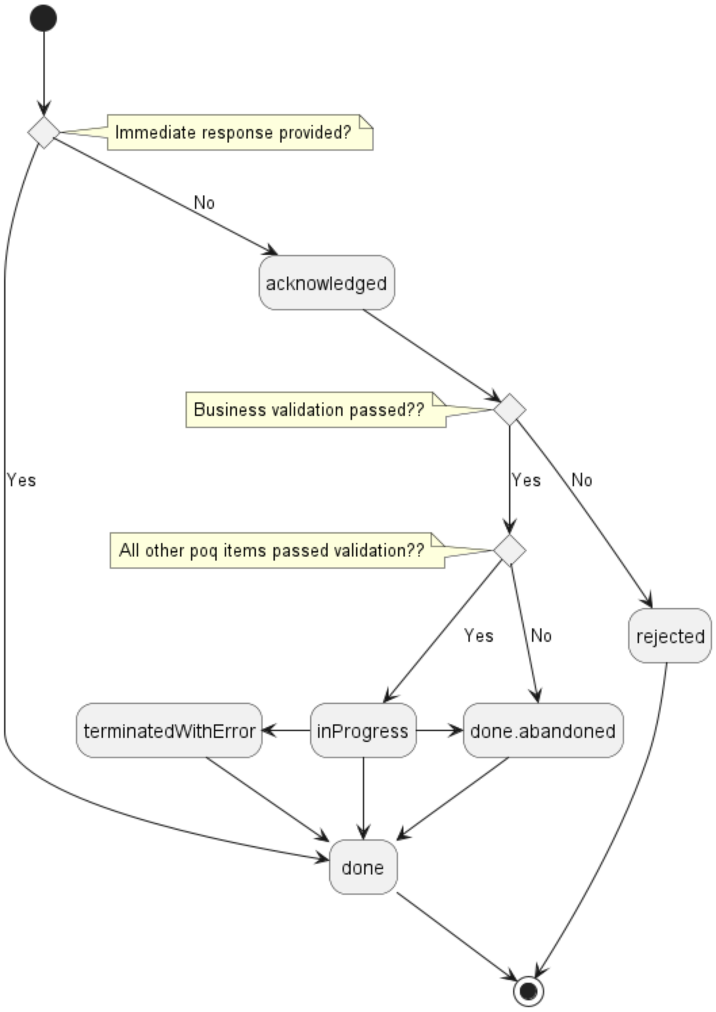

Immediate response provided?
No
Yes
acknowledged
Business validation passed??
Yes
No
All other poq items passed validation??
Yes
No
rejected
terminatedWithError
inProgress
done.abandoned
done

Figure 9. POQ Item Process Flow

Table 7 presents detailed descriptions of states and mapping between `MEFPOQItemTaskStateType` and Mplify 79.1.

MEFPOQItemTaskStateType	Mplify 79.1	Description
acknowledged	ACKNOWLEDGED	A request has been received by the Seller and has passed basic validation. For an Immediate response, the POQ moves directly to the `done` state and does not pass through `acknowledged`.
inProgress	IN_PROGRESS	The Seller is working on a POQ item response and the answer is not ready yet
done.abandoned	ABANDONED	Applied to a POQ Item in case the final state is not reached and POQ is moved to the final state other than `done`

MEFPOQItemTaskStateType	Mplify 79.1	Description
done	READY	The POQ Item has been internally approved by the Seller. This state does not imply that the Seller is able to deliver the requested item. It only means that the response for this POQ Item is complete.
rejected	REJECTED	A POQ Item has failed the business validation checks the Seller performs after it reaches the `acknowledged` state.
terminatedWithError	UNABLE_TO_MEET_TIME	The Seller is unable to provide a POQ Item response in the timeframe required by the Buyer (e.g. if an immediate response or a response date is set but cannot be met by the Seller). When a POQ Item goes to `terminatedWithError`, all POQ Items that are not `done` move to `done.abandoned`.

Table 7. Product Offering Qualification Item States

In the Immediate Response, if successful, the POQ item goes directly to `done` state. In the case of a Deferred Response, `acknowledged` is the initial state of an item. The item reaches `inProgress` once the Seller starts processing it. If there is any other item that reaches the `terminatedWithError` or `rejected` state the currently processed or not yet processed items are abandoned (`done.abandoned`). If the Seller was able to successfully complete the processing of the item the `done` state is assigned.

[R15] If the Seller provides an Immediate Response, the Seller **MUST** support POQ Item `done` state and its associated state transitions as specified in Figure 9. [Mplify79.1 R100]

[R16] If the Seller provides a Deferred Response, the Seller **MUST** support all POQ Item States and their associated state transitions as specified in Figure 9. [Mplify79.1 R101]

Table 8 presents the dependencies between the POQ state and the POQ Item state. The header represents the POQ state and the rows represent the POQ Item state.

POQ Item state \ POQ state	`acknowledged`	`inProgress`	`done`	`rejected`	`terminatedWithError`
`done.abandoned`	0	0	0	0 OR MORE BUT NOT ALL	0 OR MORE

POQ Item state \ POQ state	acknowledged	inProgress	done	rejected	terminatedWithError
acknowledged	ALL	0 OR MORE BUT NOT ALL	0	0	0
inProgress	0	1 OR MORE	0	0	0
done	0	0 OR MORE BUT NOT ALL	ALL	0 OR MORE BUT NOT ALL	0 OR MORE BUT NOT ALL
rejected	0	0	0	1 OR MORE	0
terminatedWithError	0	0	0	0	1 OR MORE

Table 8. POQ State to POQ Item State Dependency Matrix

[R17] The POQ na POQ Item state interactions **MUST** must follow rules defined in Table 8. [Mplify79.1 R99]

5.5.5. Providing the place information

When required by product specification, the Buyer must point to the place where the Product is to be provided. This is done with the use of the POQ Item's attribute: `product.place` of type `RelatedPlaceRefOrQueryWithSubUnit`, which is presented in Figure 10.

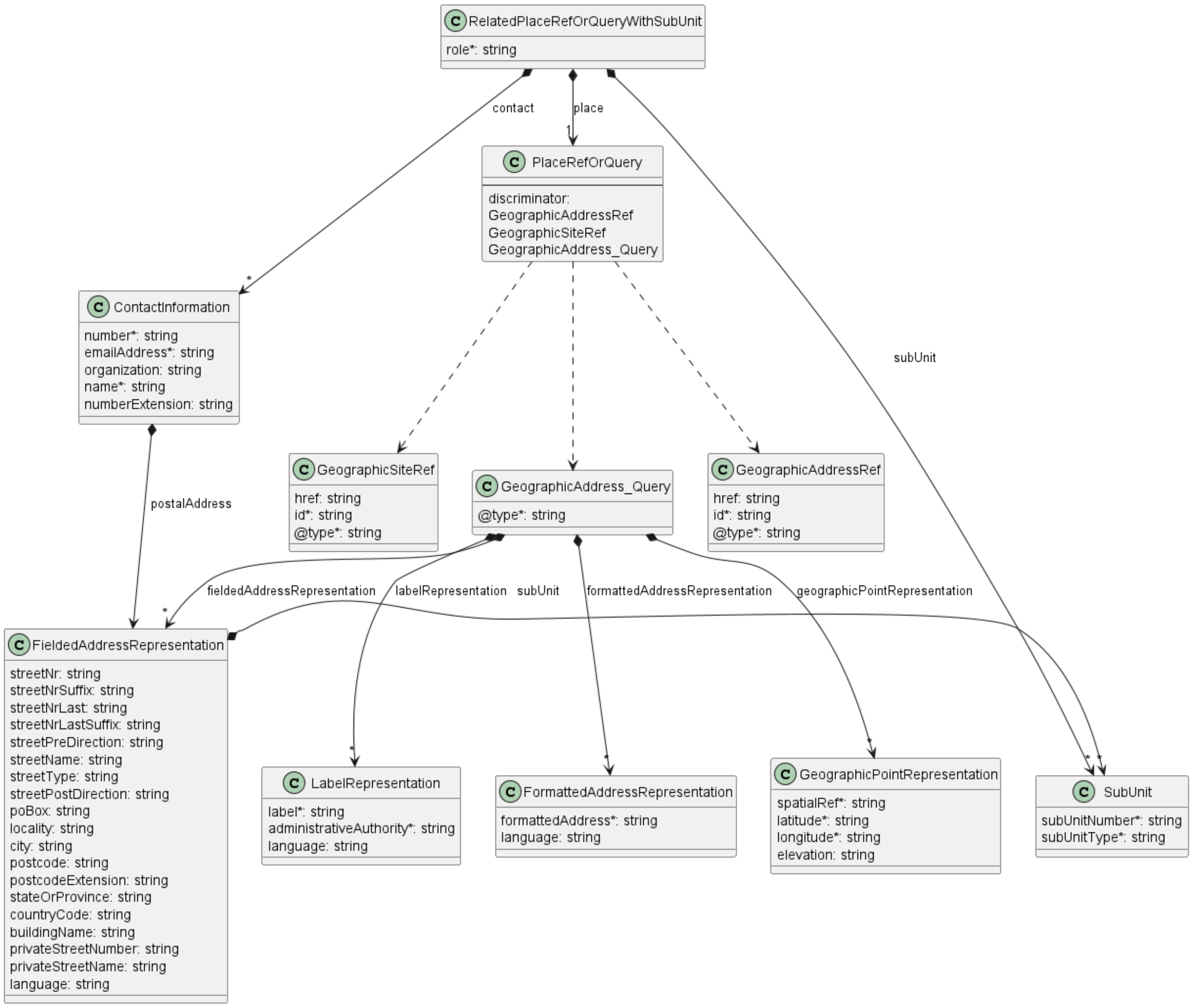


Figure 10. Data model - referring to a place

The `role` defines the function that the place plays for a given Product. The name of the role to be provided is strictly defined by the product specification. Usually, it is `INSTALL_LOCATION`.

`contact` provides additional information about the person to contact to get access to this place in case such access is required to complete the evaluation of this POQ Item.

`place` is where the actual place is pointed. The attribute is of type `PlaceRefOrQuery` which is an abstract class that can be of one of three types: `GeographicAddressRef`, `GeographicSiteRef`, or `GeographicAddress_Query`. The first two are simple identifiers to reference a `GeographicAddress` or `GeographicSite` respectively. The Buyer usually first validates the `GeographicAddress` and gets its identifier from the Seller and then optionally retrieves `GeographicSite` information for that address. In the unlikely case that the Seller does not provide the Address Validation API and the Buyer is not able to obtain the address identifier in any other way, the `GeographicAddressQuery` type might be used. It contains lists of Geographic Address Representations to provide the address information by value. There are four types of Geographic Address Representations:

- `FieldedAddressRepresentation`
- `FormattedAddressRepresentation`
- `LabelRepresentation`
- `GeographicPointRepresentation`

The Buyer may use one or more of these representations to describe a single desired place. The Buyer must provide sufficient clarity that allows the Seller to match to precisely one place. For this reason, the success rate of POQs is significantly better when identifiers are used.

In case when there is no desired `GeographicSite` object in the Seller's system, or `GeographicAddress` precision is not sufficient, the Buyer may use the `subUnit` attribute to provide more detailed information about the precise location of the installation. This information may be used by the Seller to create an instance of a `GeographicSite` with the same `subUnit` attribute value.

The `GeographicAddress` model together with its above-mentioned representations and respective requirements are defined by Mplify 121.1 (chapter 5.3). That standard is the owner of those definitions. This API specification contains a model of `GeographicAddress` but does not define it. Any further changes of these types will update the API specification, but will not be reflected in this document.

The mandatory `@type` attribute of `GeographicSiteRef`, `GeographicAddressRef` and `GeographicAddress_Query` is used as a discriminator to unambiguously identify the intended type when using in the context of the `oneOf` section of `PlaceRefOrQuery` type.

5.6. Integration of Product Specific Attributes

Product specifications are defined using JsonSchema format and are integrated into a POQ payload using a standard 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 Mplify standard product specifications is in URN format and is assigned by Mplify. This identifier is provided as root schema `$id` and in product specification documentation. Use of non-Mplify standard product definitions is allowed. In such a case, the schema identifier must be agreed between the Buyer and the Seller.

The example below shows a header of a Product Specification schema, where `"$id": urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all` is the above-mentioned URN:

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

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

Product-specific attributes can be introduced into `MEFProductRefOrValue` (defined by the Buyer) or into `AlternateProductOfferingProposal` (may be defined by the Seller while responding to POQ) using `MEFAlternateProduct`. Each of these types introduces 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 the data model during development. In such cases an integrated data model is built and product specifications are in an inheritance relationship with `MEFProductConfiguration` as described in 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 POQ API with a number of Mplify 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:carrier-ethernet-operator-uni:v5.0.0:all:
'#/components/schemas/CarrierEthernetOperatorUni'
      urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all: '#/components/schemas/AccessElineOvc'
    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 POQ Request. The named type must
        be a subclass of MEFProductConfiguration.
      type: string
```

```
AccessElineOvc:
  allOf:
    - $ref: '#/components/schemas/MEFProductConfiguration'
    - $ref: '#/components/schemas/AccessElineOvcCommon'
    - properties:
        uniEp:
          $ref: '#/components/schemas/AccessElineOvcEndPoint'
          description:
            MEF 26.2 sec. 16 - The OVC EP object for the OVC EP at the UNI. The
            UNI OVC End Point must be included in the Access E-Line Product.
        enniEp:
          $ref: '#/components/schemas/AccessElineOvcEndPoint'
          description:
            MEF 26.2 sec. 16 - The OVC EP object for the OVC EP at the ENNI. The
            ENNI OVC End Point must be included in the Access E-Line Product.
```

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

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

[R18] `MEFProductConfiguration` type is an extension point that **MUST** be used to integrate product specifications' properties into a request/response payload.

[R19] The `@type` property of `MEFProductConfiguration` **MUST** be used to specify the type of the extending entity.

[R20] Product attributes specified in the payload must conform to the product specification indicated by the `@type` property.

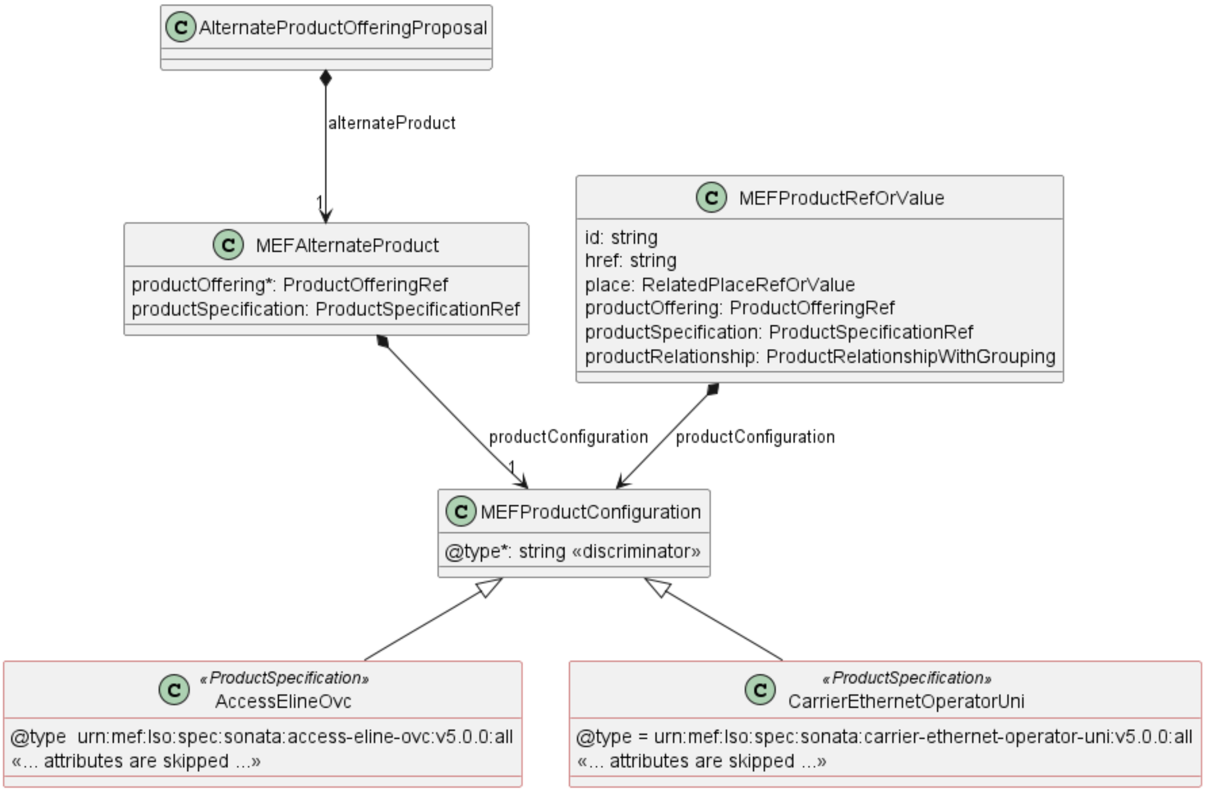

AlternateProductOfferingProposal
alternateProduct
1
MEFAlternateProduct
productOffering*: ProductOfferingRef
productSpecification: ProductSpecificationRef
MEFProductRefOrValue
id: string
href: string
place: RelatedPlaceRefOrValue
productOffering: ProductOfferingRef
productSpecification: ProductSpecificationRef
productRelationship: ProductRelationshipWithGrouping
productConfiguration
1
productConfiguration
MEFProductConfiguration
@type*: string «discriminator»
«ProductSpecification»
AccessElineOvc
@type urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all
«... attributes are skipped ...»
«ProductSpecification»
CarrierEthernetOperatorUni
@type = urn:mef:lso:spec:sonata:carrier-ethernet-operator-uni:v5.0.0:all
«... attributes are skipped ...»

Figure 11. The Extension Pattern

Figure 11 depicts two Mplify `<<ProductSpecifications>>` that represent Access E-Line and Operator UNI products. When these products are used in the POQ payload the `@type` of `MEFProductConfiguration` takes `"urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all"` or `"urn:mef:lso:spec:sonata:carrier-ethernet-operator-uni:v5.0.0:all"` value to indicate which product specification should be used to interpret a set of product-specific attributes included in the payload.

The *all* suffix after the product type name in the URN comes from the approach that the product schemas may differ depending on the API they are used with. `all` means that this schema is applicable to all contexts.

This document uses samples of Access E-Line Product specification definitions to construct API payload examples in Section 6.

Note: The Access E-Line product is valid only in the Sonata context. It is used only for the explanation of the rules of combining the product-agnostic (envelope) and product-specific (payload) parts of the APIs. The examples do not represent full and consistent product configurations, they are not normative and are not kept up to date with their respective standards. It is out of the scope of this document to explain the details of any product.

5.7. Model Structural Validation

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

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

[R21] Implementations **MUST** use payloads that conform to these definitions.

[R22] The Buyer and the Seller **MUST NOT** use any operation, entity or attribute that is not explicitly defined or allowed by this standard.

[R23] The API payloads **MUST** conform to any consistency rules and requirements defined by respective Product Specifications.

These are defined for:

- relations to other items in the same product offering qualification request (e.g. required relation type, multiplicity)
- relations to entities from the inventory managed by the Seller
- related contact information that is to be defined at an item level
- relations to places (locations) that are to be defined at an item level

5.8. Security Considerations

There must be an authentication mechanism whereby a Seller can be assured who a Buyer is and vice-versa. There must also be authorization mechanisms in place to control what a particular Buyer or Seller is allowed to do and what information may be obtained. However, the definition of the exact security mechanism and configuration is outside the scope of this document. Security considerations are standardized by *LSO API Security Profile* [MEF 128.1].

6. API Interaction & Flows

This section discusses the most important aspects of end-to-end interactions that result in completed product qualification inquiries. It starts with a description of product specifications which are used in the remainder of the section in example payloads. Then the end-to-end flows are presented for the immediate and deferred interaction patterns. Next, the structure of the POQ request and response is described. This part highlights different variants of POQ interactions for different item action types, place definitions, and alternative responses. Finally, the mechanism of notifications is discussed.

6.1. Sample Product Specification

The Sonata SDK contains product specification definitions, from which Access E-Line [MEF 106] is used in the payload samples in this section. They are located in the SDK at:

```
\productSchema\carrierEthernet\operatorEthernet\accessEline\accessElineOvc.yaml
\productSchema\carrierEthernet\operatorEthernet\carrierEthernetOperatorUni\carrierEt
hernetOperatorUni.yaml
```

Figure 12 depicts a simplified view of the defined relationships with other products and places.

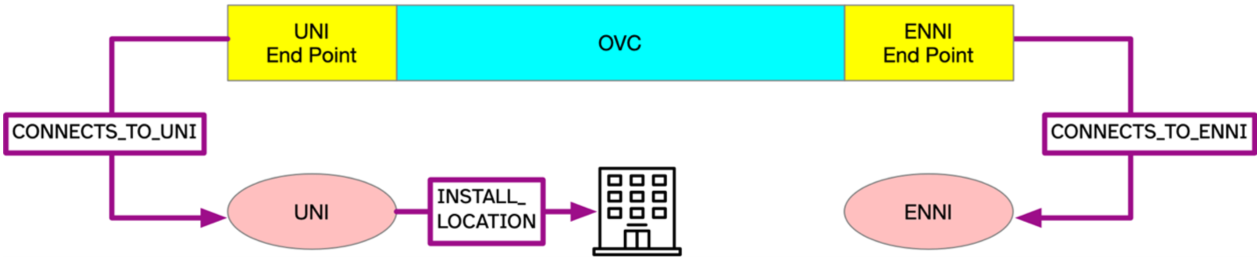


Figure 12. A simplified view of Product and Place Relationships

Product specifications define a number of product-related and envelope-related requirements. Sample envelope-related requirements for Access E-Line:

- for an Access E-Line OVS product two mandatory relationship roles must be specified, one with the operator ENNI (`CONNECTS_TO_ENNI`) and a second with the operator UNI (`CONNECTS_TO_UNI`).
- in the case of a `modify` action, product relationships must have the same value as in the `add` action. They must not be changed
- for an operator UNI product a place relationship (`INSTALL_LOCATION`) must be specified
- in the case of a `modify` action, place relationships must have the same value as in the `add` action. They must not be changed

The product relationship (`product.productRelationship`) and the place relationship (`product.place`) are presented in Figure 12.

In case some of both product-related or envelope-related requirements are violated the Seller returns an error response to the Buyer which indicates specific functional errors. These errors are listed in the response body (a list of `Error422` entries) for HTTP `422` response.

6.2. Interaction Patterns

To complete the POQ inquiry three interaction patterns can be used depending on Buyer/Seller side capabilities.

[R24] When providing responses to the API calls the Buyer and the Seller **MUST** provide relevant HTTP Response codes. [Mplify79.1 R8], [Mplify79.1 R27], [Mplify79.1 R36], [Mplify79.1 R54], [Mplify79.1 R63], [Mplify79.1 R77], [Mplify79.1 R83]

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*.

6.2.1. Immediate Response

Immediate response can be requested by a Buyer using `instantSyncQualification` flag set to `true`. In case of successful processing, the Seller will respond with POQ in a `done` state (indicating success). Otherwise, the appropriate error code and description are returned in case the payload doesn't pass initial validation. Please note that the `terminatedWithError` state is not supported in the immediate response case. It is only used during the Deferred Response pattern when the Seller is unable to provide a response in the time frame required by the Buyer.

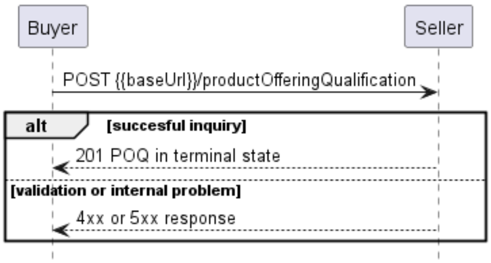

Buyer
Seller
POST {{baseUrl}}/productOfferingQualification
alt
[succesful inquiry]
201 POQ in terminal state
[validation or internal problem]
4xx or 5xx response

Figure 13. Immediate response

6.2.2. Deferred Response with Polling

A deferred response can be requested by a Buyer using `instantSyncQualification` flag set to `false`. The Seller responds with partial `POQ` (including at least `id` and `state=acknowledged`) and starts processing the request asynchronously. The Buyer polls the POQ until the final status is reached by the POQ using the POQ identifier specified by the Seller in response.

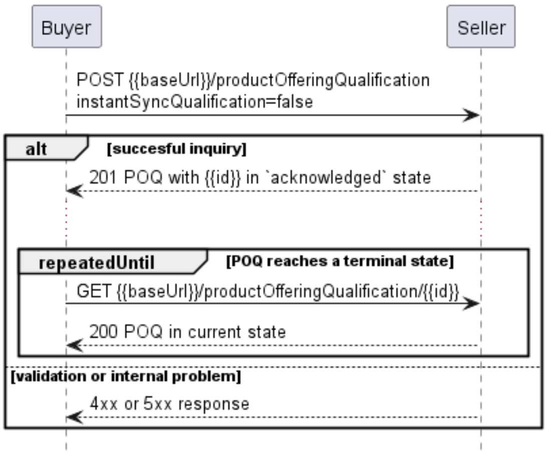

Buyer
Seller
POST {{baseUrl}}/productOfferingQualification
instantSyncQualification=false
alt
[succesful inquiry]
201 POQ with {{id}} in `acknowledged` state
repeatedUntil
[POQ reaches a terminal state]
GET {{baseUrl}}/productOfferingQualification/{{id}}
200 POQ in current state
[validation or internal problem]
4xx or 5xx response

Figure 14. Deferred response with polling

6.2.3. Deferred Response with Notifications

In this variant of the deferred response, the notifications mechanism is used. First, the Buyer registers for notifications providing a callback endpoint. Then the Buyer requests for qualification. The Seller sends notifications on POQ and POQ Item State changes until the final POQ status is reached.

When the Buyer registers for POQ notifications this registration will be valid until the Buyer unsubscribes from the POQ notifications. This implies that for any POQ request the Buyer sends to the Seller, and for which they request a Deferred Response, in the time frame between the registration for the POQ notifications and unsubscribing from the same POQ notifications by the Buyer, the Seller will have to inform the Buyer of changes using the POQ notifications.

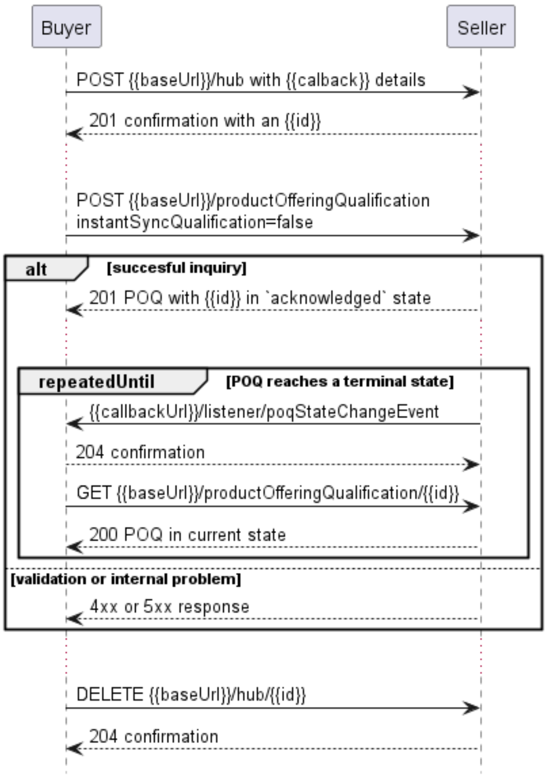

Buyer
Seller
POST {{baseUrl}}/hub with {{calback}} details
201 confirmation with an {{id}}
POST {{baseUrl}}/productOfferingQualification
instantSyncQualification=false
alt
[succesful inquiry]
201 POQ with {{id}} in `acknowledged` state
repeatedUntil
[POQ reaches a terminal state]
{{callbackUrl}}/listener/poqStateChangeEvent
204 confirmation
GET {{baseUrl}}/productOfferingQualification/{{id}}
200 POQ in current state
[validation or internal problem]
4xx or 5xx response
DELETE {{baseUrl}}/hub/{{id}}
204 confirmation

Figure 15. Deferred response with notifications

6.3. Use Case 2: Create Product Offering Qualification

To send a POQ request a Buyer must use createProductOfferingQualification operation (`POST {{baseUrl}}/productOfferingQualification`) which represents Use Case 2 (including sub-use cases). In the remainder of this section, some of the POQ payloads attributes might be omitted to simplify examples' content. The full list of attributes is available in Section 7 and in the API specification which is an integral part of this standard. Use cases 2a, 2b, and 2c differ in the details on the POQ Item level, so first the common POQ level will be described, and the the item level for each use case.

The sub-use cases of Use Case 2 are as follows:

- 2a: Create POQ Item with `action=add` and referring a particular Product Offering (`product.productOffering`)
- 2b: Create POQ Item with `action=add` and providing only Product Specification (`product.productSpecification`)
- 2c: Create POQ Item with `action=modify`, referring existing product to modify by providing `product.id`.

The Buyer uses 2a when they know exactly the Product Offering they want to order. Use Case 2b allows the Buyer only to provide the Product Specification to receive the Product Offering(s) from the Seller.

6.3.1. Buyer Create POQ Request

Here is the example of the Buyer's Create POQ request. It allows deferred response to be provided by `2024-11-06T09:36:05.668Z`. It requests the qualification of the creation of 2 products: Access E-Line OVC and Carrier Ethernet Operator UNI, and provides all required relationships, as described in Figure 12.

The Buyer also requests providing alternative proposals (`provideAlternative=true`).

The Access E-Line Product Offering is identified as `000073` in the Seller's Product Catalog. This specification describes the structure and requirements defined for this product which should be validated. An Access E-Line product specification defines two mandatory relationship types that have to be specified: `CONNECTS_TO_ENNI` and `CONNECTS_TO_UNI`. This Access E-Line product references an existing ENNI product that is uniquely identified with id `SP1_ENNI` in the Seller's inventory. The reference to a UNI product might use another POQ item or an existing product from the Seller's Product Inventory. This example assumes that the UNI product is another item of the POQ request with a unique identifier `item-002`.

The place is not provided as the 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 the `productConfiguration` node. This example uses a subset of available Access E-Line attributes.

The UNI product refers to Product Offering with `id=000074` and provides the required `place` relationship with `role=INSTALL_LOCATION` and referencing an address by an identifier with `GeographicAddressRef`.

```
{
  "instantSyncQualification": false,
  "requestedPOQCompletionDate": "2024-11-12T09:36:05.668Z",
  "provideAlternative": true,
  "externalId": "BuyerPoq-00001",
  "projectId": "BuyerProjectX",
  "relatedContactInformation": [
    {
      "emailAddress": "john.example@buyer.mef.com",
      "name": "John Example",
      "number": "12-345-6789",
      "numberExtension": "1234",
      "organization": "Buyer Co.",
      "role": "buyerContactInformation"
    }
  ],
  "productOfferingQualificationItem": [
    {
      "id": "item-001",
      "action": "add",
      "product": {
        "productOffering": {
          "id": "000073"
        },
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all",
          "ceVlanIdPreservation": "PRESERVE",
          "maximumFrameSize": 1526,
          "listOfClassOfServiceNames": ["low"],
          "enniEp": {
            "identifier": "SP1_ENNI-EP1",
            "ingressClassOfServiceMap": {
              "mapType": "ENDPOINT",
              "map_M": "low",
              "l2cp_P": {
                "l2cpIdentifier": {
                  "l2cpProtocolType": "LLC",
                  "llcAddressOrEtherType": 66
                },
                "l2cpCosName": "low"
              }
            }
          },
          "uniEp": {
            "identifier": "NewYork_UNI-EP1",
            "ingressBandwidthProfilePerClassOfServiceName": [
              {
                "classOfServiceName": "low",
                "bwpFlow": {
                  "cir": {
                    "irValue": 0,
                    "irUnits": "MBPS"
                  },
                  "cirMax": {
                    "irValue": 0,
                    "irUnits": "MBPS"
                  },
                  "eir": {
                    "irValue": 10,
                    "irUnits": "GBPS"
                  },
                  "eirMax": {
                    "irValue": 10,
                    "irUnits": "GBPS"
                  }
                }
              }
            ],
            "ingressClassOfServiceMap": {
              "mapType": "ENDPOINT",
              "map_M": "low",
              "l2cp_P": {
                "l2cpIdentifier": {
                  "l2cpProtocolType": "LLC",
                  "llcAddressOrEtherType": 66
                },
                "l2cpCosName": "low"
              }
            }
          }
        },
        "productRelationship": [
          {
            "relationshipType": "CONNECTS_TO_ENNI",
            "id": "SP1_ENNI"
          }
        ]
      },
      "qualificationItemRelationship": [
        {
          "relationshipType": "CONNECTS_TO_UNI",
          "id": "item-002"
        }
      ]
    },
    {
      "id": "item-002",
      "action": "add",
      "product": {
        "productOffering": {
          "id": "000074"
        },
        "place": [
          {
            "place": {
              "@type": "GeographicAddressRef",
              "id": "NewYorkAddress-id-1"
            },
            "role": "INSTALL_LOCATION",
            "contact": [
              {
                "number": "+12-345-678-90",
                "emailAddress": "LocationContact@buyer.mef.com",
                "name": "Location Contact"
              }
            ]
          }
        ],
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:sonata:carrier-ethernet-operator-uni:v5.0.0:all",
          "defaultCeVlanId": 4094,
          "maximumNumberOfEndPoints": 6,
          "lagLinkMeg": "DISABLED",
          "linkAggregation": "NONE",
          "tokenShare": "ENABLED",
          "maximumServiceFrameSize": 1522,
          "listOfPhysicalLinks": [
            {
              "id": "01",
              "physicalLink": "10GBASE_SR",
              "uniConnectorGender": "SOCKET",
              "synchronousEthernet": "ENABLED",
              "uniConnectorType": "SC",
              "precisionTiming": "DISABLED"
            }
          ]
        }
      }
    }
  ]
}
```

[R25] The Buyer **MUST** specify following attributes: [Mplify79.1 R9], [Mplify79.1 R12]

- `instantSyncQualification`
- `provideAlternative`
- `relatedContactInformation` with an item with `role=buyerContactInformation`
- at least one `productOfferingQualificationItem`

[R26] For Use Case 2b the Buyer **MUST** specify `provideAlternative` attribute as `true`. [Mplify79.1 R10]

[R27] The Buyer **MUST** specify following attributes of `RelatedContactInformation`: [Mplify79.1 R22]

- `emailAddress`
- `name`
- `number`
- `role`

During the onboarding, the Seller may require to provide an additional contact `role`.

Note: It is up to Seller's discretion on how to react in case the Buyer provides a contact `role` that is not listed by this standard or agreed upon during the onboarding. Preferably the Seller should return an error with a message stating which `roles` are accepted. It may also be ignored.

[R28] The `requestedPOQCompletionDate` **MUST** be specified when `instantSyncQualification=false`, [Mplify79.1 R11]

[R29] The Buyer **MUST** specify following attributes of `productOfferingQualificationItem`: [Mplify79.1 R17], [Mplify79.1 R48]

- `id`
- `action`
- `product.productConfiguration`

[R30] A relationship between the POQ Item and an already activated Product **MUST** use the `productRelationship` attribute to detail the relationship. [Mplify79.1 R23], [Mplify79.1 R50]

[R31] When specifying the `productRelationship`, the Buyer **MUST** provide following attributes`: [Mplify79.1 R25], [Mplify79.1 R52]

- `id`

- relationshipType

[R32] A relationship between the POQ Item and other POQ Items **MUST** use the qualificationItemRelationship attribute to detail the relationship. [Mplify79.1 R24], [Mplify79.1 R51]

[R33] When specifying the qualificationItemRelationship, the Buyer **MUST** provide following attributes`: [Mplify79.1 R26], [Mplify79.1 R53]

- id
- relationshipType

Some Product Specifications allow providing a list of related products even for relationship types whose final cardinality is 1. These act like a list of candidates. For example, the Buyer may include a list of ENNIs between the Buyer and Seller as related Products. The ENNIs in the list might need to all be in the same Geographic Area as defined by the Seller (same city, same county, etc.). The Seller uses any of the ENNIs in the list to respond to the POQ Request.

[R34] If the Product Specification mandates a Place, the product.place **MUST** be used to detail the place relationship. [Mplify79.1 R18]

[R35] When specifying the product.place, the Buyer **MUST** provide the following attributes: [Mplify79.1 R19]

- place
- role
- contact

[O1] When specifying the product.place, with GeographicAddressRef or GeographicAddress_Query the Buyer **MAY** additionally provide subUnitto describe exactly where the Buyer wants the Product to be installed. [Mplify79.1 O4]

[R36] When specifying the ContactInformation, the Buyer **MUST** provide following attributes`: [Mplify79.1 R22]

- emailAddress
- name
- number

6.3.1.1. Request for add action

Requirements in this section apply to the Buyer providing productOfferingQualificationItem with the add action.

[R37] If action=add the Buyer **MUST** provide exactly one of product.productOffering (Use Case 2a) or product.productSpecification (Use Case 2b). [Mplify79.1 R17]

[R37] If action=add the Buyer **MUST NOT** provide product.id.

6.3.1.2. Request for modify action

Requirements in this section apply to the Buyer requesting the modify action.

The example below represents a single POQ request item to evaluate a modification of an existing (action modify) Access E-Line product. The product is referred to with `id=AccessElineOVC-0001` and a new full `product` representation. The desire is to set the `cir` (Committed Information Rate) from `0 GBPS` to `1 GBPS` so that the Buyer can have 1 GBPS of the bandwidth guaranteed.

```
{
  "id": "item-001",
  "action": "modify",
  "product": {
    "id": "AccessElineOVC-0001",
    "productOffering": {
      "id": "000073"
    },
    "productConfiguration": {
      "@type": "urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all",
      "ceVlanIdPreservation": "PRESERVE",
      "maximumFrameSize": 1526,
      "listOfClassOfServiceNames": ["low"],
      "enniEp": {
        "identifier": "SP1_ENNI-EP1",
        "ingressClassOfServiceMap": {
          "mapType": "ENDPOINT",
          "map_M": "low",
          "l2cp_P": {
            "l2cpIdentifier": {
              "l2cpProtocolType": "LLC",
              "llcAddressOrEtherType": 66
            },
            "l2cpCosName": "low"
          }
        }
      },
      "uniEp": {
        "identifier": "NewYork_UNI-EP1",
        "ingressBandwidthProfilePerClassOfServiceName": [
          {
            "classOfServiceName": "low",
            "bwpFlow": {
              "cir": {
                "irValue": 1,
                "irUnits": "GBPS"
              },
              "cirMax": {
                "irValue": 1,
                "irUnits": "GBPS"
              },
              "eir": {
                "irValue": 10,
                "irUnits": "GBPS"
              },
              "eirMax": {
                "irValue": 10,
                "irUnits": "GBPS"
              }
            }
          }
        ],
        "ingressClassOfServiceMap": {
          "mapType": "ENDPOINT",
          "map_M": "low",
          "l2cp_P": {
            "l2cpIdentifier": {
              "l2cpProtocolType": "LLC",
              "llcAddressOrEtherType": 66
            },
            "l2cpCosName": "low"
          }
        }
      }
    },
    "productRelationship": [
      {
        "relationshipType": "CONNECTS_TO_ENNI",
        "id": "SP1_ENNI"
      },
      {
        "relationshipType": "CONNECTS_TO_UNI",
        "id": "SP1_UNI"
      }
    ]
  }
}
```

[R39] If `action=modify` the Buyer **MUST** provide following attributes of `poqItem.product`: [Mplify79.1 R48]

- `product.id`
- `product.productOffering`
- `product.productConfiguration`

[R40] The modify request **MUST** provide a full state of the `product` attributes, including values of (specified or empty) of `product.productOffering`, `product.productRelationship`, `product.productSpecification`, and `product.place` as they are available in the inventory for a given product instance.

[O2] The Seller **MAY** allow the Buyer to specify a different `product.productOffering` than the one of the existing Product. [Mplify79.1 O10]

[R41] If `product.productOffering` changes, it **MUST** be based on the same `product.productSpecification` as the existing Product. [Mplify79.1 R49]

There is no possibility to send an update to single attributes. The Buyer must send a full `product` representation, which means all attributes that represent the desired state, even if some of them do not change.
If the Seller does not allow for some of the attributes to change (e.g. because of Product Offering or technical constraints) an appropriate error response (`422`) must be returned to the Buyer.

The Product Specification defines if the relationships to products or places can be changed.

Note, that since the example contains only one POQ item it has to refer to the UNI product by `product.productRelationship`, instead of `qualificationItemRelationship` like in the create request where the UNI was provided as an item in the same POQ.

6.3.2. Seller's Response to Create POQ request

[R42] The Seller **MUST** echo back all attributes contained in the Buyer's request (not changed). [Mplify79.1 R34], [Mplify79.1 R37], [Mplify79.1 R61], [Mplify79.1 R64]

The following snippet provides an example of an immediate response. For sake of readability only Seller settable attributes are shown.

```
{
  "id": "1234-5678-9000",
  "href": "{{baseUrl}}/productOfferingQualification/1234-5678-9000",
  "creationDate": "2024-11-07T10:39:26.245Z",
  "state": "done",
  "stateChange": [
    {
      "changeDate": "2024-11-07T10:39:26.245Z",
      "state": "done"
    }
  ],
  "relatedContactInformation": [
    {
      "emailAddress": "john.example@buyer.mef.com",
      "name": "John Example",
      "number": "12-345-6789",
      "numberExtension": "1234",
      "organization": "Buyer Co.",
      "role": "buyerContactInformation"
    },
    {
      "emailAddress": "anna.seller@seller.mef.com",
      "name": "Anna Seller",
      "number": "98-765-4321",
      "organization": "Seller",
      "role": "sellerContactInformation"
    }
  ],
  "productOfferingQualificationItem": [
    {
      "id": "item-001",
      "state": "done",
      "stateChange": [
        {
          "changeDate": "2024-11-07T10:39:26.245Z",
          "state": "done"
        }
      ],
      "serviceabilityConfidence": "yellow",
      "serviceabilityConfidenceReason": "There needs to be a site survey done to verify the possibility of serving a 10 GBPS connection",
      "installationInterval": {
        "amount": 10,
        "units": "businessDays"
      },
      "guaranteedUntilDate": "2024-12-07T10:39:26.245Z",
      "alternateProductOfferingProposal": [
        {
          "id": "altItem-001",
          "serviceabilityConfidence": "green",
          "serviceabilityConfidenceReason": "1 GBPS connection can be provisioned with current network configuration",
          "installationInterval": {
            "amount": 1,
            "units": "businessDays"
          },
          "guaranteedUntilDate": "2024-12-07T10:39:26.245Z",
          "alternateProduct": {
            "productOffering": {
              "id": "000166"
            },
            "productConfiguration": {
              "@type": "urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all",
              "ceVlanIdPreservation": "PRESERVE",
              "maximumFrameSize": 1526,
              "listOfClassOfServiceNames": ["low"],
              "enniEp": {
                "identifier": "SP1_ENNI-EP1",
                "ingressClassOfServiceMap": {
                  "mapType": "ENDPOINT",
                  "map_M": "low",
                  "l2cp_P": {
                    "l2cpIdentifier": {
                      "l2cpProtocolType": "LLC",
                      "llcAddressOrEtherType": 66
                    },
                    "l2cpCosName": "low"
                  }
                }
              },
              "uniEp": {
                "identifier": "NewYork_UNI-EP1",
                "ingressBandwidthProfilePerClassOfServiceName": [
                  {
                    "classOfServiceName": "low",
                    "bwpFlow": {
                      "cir": {
                        "irValue": 0,
                        "irUnits": "MBPS"
                      },
                      "cirMax": {
                        "irValue": 0,
                        "irUnits": "MBPS"
                      },
                      "eir": {
                        "irValue": 1,
                        "irUnits": "GBPS"
                      },
                      "eirMax": {
                        "irValue": 1,
                        "irUnits": "GBPS"
                      }
                    }
                  }
                ],
                "ingressClassOfServiceMap": {
                  "mapType": "ENDPOINT",
                  "map_M": "low",
                  "l2cp_P": {
                    "l2cpIdentifier": {
                      "l2cpProtocolType": "LLC",
                      "llcAddressOrEtherType": 66
                    },
                    "l2cpCosName": "low"
                  }
                }
              }
            },
            "productRelationship": [
              {
                "relationshipType": "CONNECTS_TO_ENNI",
                "id": "SP1_ENNI"
              }
            ]
          },
          "deliveryType": "onNetWithoutBuild"
        }
      ],
      "deliveryType": "onNetWithBuild"
    },
    {
      "id": "item-002",
      "state": "done",
      "stateChange": [
        {
          "changeDate": "2024-11-07T10:39:26.245Z",
          "state": "done"
        }
      ],
      "serviceabilityConfidence": "green",
      "serviceabilityConfidenceReason": "We can serve as requested",
      "installationInterval": {
        "amount": 1,
        "units": "businessWeek"
      },
      "guaranteedUntilDate": "2024-12-07T10:39:26.245Z",
      "deliveryType": "onNetWithBuild"
    }
  ]
}
```

[R43] When providing a response, the Seller **MUST** provide the additional following attributes of `ProductOfferingQualification`: [Mplify79.1 R29], [Mplify79.1 R33], [Mplify79.1 R38], [Mplify79.1 R55], [Mplify79.1 R59], [Mplify79.1 R60], [Mplify79.1 R65]

- `id`
- `state`
- `stateChange`
- `creationDate`
- `productOfferingQualificationItem`
  - `state`
  - `stateChange`

[R44] The `stateChange` **MUST** include a full object's state history including the initial state (also in the Immediate Response).

[R45] The ProductOfferingQualification.id **MUST** be unique within the Seller's system. [Mplify79.1 R32], [Mplify79.1 R58]

[R46] Each item in the productOfferingQualificationItem list in the response **MUST** correspond to an item from a list in the request. [Mplify79.1 R27], [Mplify79.1 R42], [Mplify79.1 R69]

[R47] The Seller response to a Use Case 2a request **MUST** match the Product Offering specified in the request. [Mplify79.1 R15]

[R48] The Seller response to a Use Case 2b request **MUST** match the Product Specification provided in the request. [Mplify79.1 R13], [Mplify79.1 R16]

Note: This POQ Item response may have a serviceabilityConfidence=red if there are no matching Product Offerings with a serviceabilityConfidence of green or yellow. In that case, the Seller also does not provide the product.productOffering attribute.

[R49] The Seller response to Use Case 2b **MUST** include the Product Offering with the highest serviceabilityConfidence possible (i.e. green, or yellow if not green). [Mplify79.1 R14]

6.3.2.1. Deferred response

Requirements in this section apply to Seller providing a Deferred Response.

[R50] The Seller **MUST** provide following attributes' values: [Mplify79.1 R30], [Mplify79.1 R31], [Mplify79.1 R56], [Mplify79.1 R57]

- state=acknowledged
- for all productOfferingQualificationItems: state=acknowledged

[O3] The Seller **MAY** provide an immediate answer even when the instantSyncQualification flag is set to false. [Mplify79.1 O5].

6.3.2.2. Immediate response

Requirements in this section apply to Seller providing an Immediate Response.

[R51] The Seller **MUST** provide the following attribute of ProductOfferingQualificationItem: [Mplify79.1 R43], [Mplify79.1 R71]

- serviceabilityConfidence

[R52] The Seller **MUST** provide following attributes' values: [Mplify79.1 R38], [Mplify79.1 R39], [Mplify79.1 R40], [Mplify79.1 R65], [Mplify79.1 R66], [Mplify79.1 R67], [Mplify79.1 R70]

- state=done
- relatedContactInformation with added item with role=sellerContactInformation
- productOfferingQualificationItem.state=done

[R53] When attribute serviceabilityConfidence is set to green or yellow the Seller **MUST** provide the following attributes of ProductOfferingQualificationItem: [Mplify79.1 R44], [Mplify79.1 R72], [Mplify79.1 R73]

- installationInterval

- `deliveryType`
- `product.productOffering`

Note: The Buyer and Seller may agree on a method where a confidence of `yellow` is returned with a specific lead time value that can be interpreted by the Buyer as the Seller saying that they can provide the Product Offering but cannot provide a valid lead time in their response.

[D2] The Seller **SHOULD** specify the `guaranteedUntilDate` in the response [Mplify79.1 D3], [Mplify79.1 D5]

[O4] If the `deliveryType` is `offNetWithBuild` or `offNetWithoutBuild`, the Seller **MAY** provide the `3rdPartyProvider` and `3rdPartyProviderProductOffering`. [Mplify79.1 O6], [Mplify79.1 O12]

6.3.2.3. Alternative Product Offering Proposals

Alternate Product Offering Proposals represent other Product Offerings that the Seller is proposing to meet the needs of the Buyer. In the example in section 6.3.2, the Buyer requested a Product Offering with `id=000073` and configuration of 10 GBPS of not guaranteed bandwidth. The Seller can potentially provide that but a site survey is needed to verify that. That is why `serviceabilityConfidence-yellow` is provided. However the Seller can quickly provide a slower connection (1 GBPS) and an ALternative is proposed with Product Offering `id=000166` and relevant product configuration. The Seller may specify any number of Alternate Product Offering Proposals in response to one POQ Item.

[O5] In Use Cases 2a and 2c the Seller **MAY** specify `alternateProductOfferingProposal` only when all of the following are true: [Mplify79.1 O7], [Mplify79.1 O11], [Mplify79.1 O13]

- the Buyer has set `provideAlternate` to `true`
- the Seller has determined that the `serviceabilityConfidence` for this item is `yellow` or `red`
- the Seller has alternate Product Offerings (e.g., similar but lower bandwidth) that may be adequate

[CR2]<[O5] If the Buyer has requested to provide alternatives, but the Seller cannot find any, then the Seller **MUST** return `alternateProductOfferingProposal` array with 0 items.

[O6] The Seller response to Use Case 2b **MAY** include additional Product Offerings as Alternate Offerings regardless of the `serviceabilityConfidence` of the Product Offering. [Mplify79.1 O2]

[O7] The Seller's response provided as Alternate Product Offering Proposals **MAY** be of a different Product Offering or Product Specification than requested by the Buyer. [Mplify79.1 O3], [Mplify79.1 O9], [Mplify79.1 O15]

[R54] When specifying the `alternateProductOfferingProposal`, the Seller **MUST** provide following attributes`: [Mplify79.1 R45], [Mplify79.1 R46], [Mplify79.1 R74], [Mplify79.1 R75]

- `id`
- `serviceabilityConfidence`
- `installationInterval`
- `alternateProduct`
- `deliveryType`

[D3] The Seller **SHOULD** specify the `guaranteedUntilDate` attribute. [Mplify79.1 D4], [Mplify79.1 D6]

[R55] The `alternateProductOfferingProposal.id` **MUST** be unique within the POQ Item. [Mplify79.1 R47], [Mplify79.1 R76]

[O8] If the `deliveryType` is `offNetWithBuild` or `offNetWithoutBuild`, the Seller **MAY** provide the `3rdPartyProvider` and `3rdPartyProviderProductOffering`. [Mplify79.1 O8], [Mplify79.1 O14]

6.4. Use Case 3: Retrieve POQ list

A Buyer can retrieve a list of the POQs by using `GET /productOfferingQualification/` operation with desired filtering criteria.

[O9] The Buyer **MAY** use any of the following query parameters to query for the Product Offering Qualification list. [Mplify79.1 O16]

- `state`
- `creationDate.gt`
- `creationDate.lt`
- `requestedPOQCompletionDate.lt`
- `requestedPOQCompletionDate.gt`
- `externalId`
- `projectId`

[O10] The Buyer **MAY** use a combination of attributes to avoid getting an `Error422` with `tooManyRecords` code.

[D4] The Seller **SHOULD** support the pagination mechanism.

The Buyer may also ask for pagination of the response when the number of results is too big. The following query attributes related to pagination can be provided:

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

```
https://serverRoot/mefApi/sonata/productOfferingQualification/v8/productOfferingQualification?
state=inProgress&limit=20&offset=0
```

The example above shows a Buyer's request to get the first twenty Product Offering Qualifications that are in progress from a possible long list within the response.

The Seller returns a list of elements that comply with the requested `limit`. If the requested `limit` is higher than the supported list size then the smaller list of results 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`

[CR3]<[D4] Seller **MUST** use either `X-Total-Count` or `X-Pagination-Throttled` to indicate that the page was truncated and additional results are available.

In the response, the Seller returns all POQs matching these filtering criteria (3 in the example).

```
[
  {
    "id": "1234-5678-9000",
    "creationDate": "2024-11-15T10:39:26.245Z",
    "state": "inProgress",
    "requestedPOQCompletionDate": "2024-11-06T09:36:05.668Z",
    "externalId": "BuyerPoq-00001",
    "projectId": "BuyerProjectX"
  },
  {
    "id": "97975e56-b6ba-40d4-b9b3-dab2b0e58279",
    "creationDate": "2024-09-18",
    "state": "inProgress",
    "requestedPOQCompletionDate": "2024-09-20"
  },
  {
    "id": "79de3367-ce55-4e9a-952c-3c16e715bb7f",
    "creationDate": "2024-09-26",
    "state": "inProgress",
    "requestedPOQCompletionDate": "2024-09-30",
    "externalId": "BuyerPoq-00124"
  }
]
```

[R56] The Seller **MUST** return zero or more `ProductOfferingQualification_Find` objects in the response. [Mplify79.1 R79]

[R57] For each `ProductOfferingQualification_Find` returned, the Seller **MUST** specify values (if present) of the following attributes: [Mplify79.1 R80]

- `id`
- `state`
- `creationDate`
- `requestedPOQCompletionDate`
- `externalId`
- `projectId`

[R58] In case of too many matching records are found (the definition of 'too many' is up to Seller's discretion), the Seller **MUST** return an `Error422` with `code` equal to `tooManyRecords`.

To see full details of a particular POQ the Buyer must retrieve the POQ by the identifier as described in the section below.

6.5. Use Case 4: Retrieve POQ by identifier

POQ information can be retrieved from the Seller using the GET `/productOfferingQualification/{{id}}` operation. The correct payload returned in the response includes all the attributes that the Buyer has provided while sending the POQ request and all that the Seller has added during the request processing.

POQ can be in an intermediate (`acknowledged`, `inProgress`) or one of the final states (`done`, `rejected`, `terminatedWithError`). There are different requirements, depending on the POQ state.

The example below shows a possible response for a POQ being in the middle of the processing. The POQ is in the `inProgress` state, with the timestamps of state transitions captured in `stateChange`. One item is in the `inProgress` state and the second is still in the `acknowledged`. Both are still not done processing so they do not have the "response" attributes like `alternateProductOfferingProposal`, `serviceabilityConfidence`, `serviceabilityConfidenceReason`, `installationInterval`, `guaranteedUntilDate`, and `deliveryType` set yet.

[R59] The Buyer **MUST** provide the `id` of the `ProductOfferingQualification` in the query. [Mplify79.1 R81]

Request:

```
https://serverRoot/mefApi/sonata/productOfferingQualification/v8/productOfferingQualification/1234-5678-9000
```

Response:

```
{
  "id": "1234-5678-9000",
  "href": "{{baseUrl}}/productOfferingQualification/1234-5678-9000",
  "creationDate": "2024-11-07T10:39:26.245Z",
  "expectedPOQCompletionDate": "2024-11-10T10:39:26.245Z",
  "state": "inProgress",
  "stateChange": [
    {
      "changeDate": "2024-11-07T10:39:26.245Z",
      "state": "acknowledged"
    },
    {
      "changeDate": "2024-11-07T15:39:26.245Z",
      "state": "inProgress"
    }
  ],
  "instantSyncQualification": false,
  "requestedPOQCompletionDate": "2024-11-12T09:36:05.668Z",
  "provideAlternative": true,
  "externalId": "BuyerPoq-00001",
  "projectId": "BuyerProjectX",
  "relatedContactInformation": [
    {
      "emailAddress": "john.example@buyer.mef.com",
      "name": "John Example",
      "number": "12-345-6789",
      "numberExtension": "1234",
      "organization": "Buyer Co.",
      "role": "buyerContactInformation"
    },
    {
      "emailAddress": "anna.seller@seller.mef.com",
      "name": "Anna Seller",
      "number": "98-765-4321",
      "organization": "Seller",
      "role": "sellerContactInformation"
    }
  ],
  "productOfferingQualificationItem": [
    {
      "id": "item-001",
      "action": "add",
      "state": "inProgress",
      "stateChange": [
        {
          "changeDate": "2024-11-07T10:39:26.245Z",
          "state": "acknowledged"
        },
        {
          "changeDate": "2024-11-08T15:39:26.245Z",
          "state": "inProgress"
        }
      ],
      "product": {
        "productOffering": {
          "id": "000073"
        },
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all",
          "ceVlanIdPreservation": "PRESERVE",
          "maximumFrameSize": 1526,
          "listOfClassOfServiceNames": ["low"],
          "enniEp": {
            "identifier": "SP1_ENNI-EP1",
            "ingressClassOfServiceMap": {
              "mapType": "ENDPOINT",
              "map_M": "low",
              "l2cp_P": {
                "l2cpIdentifier": {
                  "l2cpProtocolType": "LLC",
                  "llcAddressOrEtherType": 66
                },
                "l2cpCosName": "low"
              }
            }
          },
          "uniEp": {
            "identifier": "NewYork_UNI-EP1",
            "ingressBandwidthProfilePerClassOfServiceName": [
              {
                "classOfServiceName": "low",
                "bwpFlow": {
                  "cir": {
                    "irValue": 0,
                    "irUnits": "MBPS"
                  },
                  "cirMax": {
                    "irValue": 0,
                    "irUnits": "MBPS"
                  },
                  "eir": {
                    "irValue": 10,
                    "irUnits": "GBPS"
                  },
                  "eirMax": {
                    "irValue": 10,
                    "irUnits": "GBPS"
                  }
                }
              }
            ],
            "ingressClassOfServiceMap": {
              "mapType": "ENDPOINT",
              "map_M": "low",
              "l2cp_P": {
                "l2cpIdentifier": {
                  "l2cpProtocolType": "LLC",
                  "llcAddressOrEtherType": 66
                },
                "l2cpCosName": "low"
              }
            }
          }
        },
        "productRelationship": [
          {
            "relationshipType": "CONNECTS_TO_ENNI",
            "id": "SP1_ENNI"
          }
        ]
      },
      "qualificationItemRelationship": [
        {
          "relationshipType": "CONNECTS_TO_UNI",
          "id": "item-002"
        }
      ]
    },
    {
      "id": "item-002",
      "state": "acknowledged",
      "stateChange": [
        {
          "changeDate": "2024-11-07T10:39:26.245Z",
          "state": "acknowledged"
        }
      ],
      "product": {
        "productOffering": {
          "id": "000074"
        },
        "place": [
          {
            "place": {
              "@type": "GeographicAddressRef",
              "id": "NewYorkAddress-id-1"
            },
            "role": "INSTALL_LOCATION",
            "contact": [
              {
                "number": "+12-345-678-90",
                "emailAddress": "LocationContact@buyer.mef.com",
                "name": "Location Contact"
              }
            ]
          }
        ],
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:sonata:carrier-ethernet-operator-uni:v5.0.0:all",
          "defaultCeVlanId": 4094,
          "maximumNumberOfEndPoints": 6,
          "lagLinkMeg": "DISABLED",
          "linkAggregation": "NONE",
          "tokenShare": "ENABLED",
          "maximumServiceFrameSize": 1522,
          "listOfPhysicalLinks": [
            {
              "id": "01",
              "physicalLink": "10GBASE_SR",
              "uniConnectorGender": "SOCKET",
              "synchronousEthernet": "ENABLED",
              "uniConnectorType": "SC",
              "precisionTiming": "DISABLED"
            }
          ]
        }
      }
    }
  ]
}
```

The Seller receives the request and returns all information (including all POQ items) for the POQ with the requested `id`.

The following requirements apply to the `ProductOfferingQualification` provided in Seller's response.

[R60] The Seller **MUST** provide the following attributes: [Mplify79.1 R84]

- `id`
- `state`
- `stateChange`
- `creationDate`
- `instantSyncQualification`
- `provideAlternative`
- `relatedContactInformation`
- `productOfferingQualificationItem`

The following requirements apply to all `ProductOfferingQualificationItems` provided in Seller's response.

[R61] Each item in the `productOfferingQualificationItem` list in the response **MUST** correspond to an item from a list in the request. [Mplify79.1 R88]

[R62] The Seller **MUST** provide the following attributes of `ProductOfferingQualificationItem`: [Mplify79.1 R86], [Mplify79.1 R88]

- `id`
- `state`

- `stateChange`
- `action`
- `product`

[R63] The `stateChange` **MUST** include a full object's state history including the initial state (also in the Immediate Response).

[R64] If `state=done` the Seller **MUST** provide the following attributes of `ProductOfferingQualificationItem`: [Mplify79.1 R82]

- `serviceabilityConfidence`
- `relatedContactInformation` with items:
  - `role=buyerContactInformation`
  - `role=sellerContactInformation`

[R65] When attribute `serviceabilityConfidence` is set to `green` or `yellow` the Seller **MUST** provide the following attributes of `ProductOfferingQualificationItem`: [Mplify79.1 R82]

- `installationInterval`
- `deliveryType`
- `product.productOffering`

[D5] If `state=done` the Seller **SHOULD** specify the `guaranteedUntilDate`. [Mplify79.1 D7]

[R66] If `state=rejected` the Seller **MUST** only provide (aside from buyer set) the following attributes of `ProductOfferingQualificationItem`: [Mplify79.1 R90]

- `state`
- `stateChange`

6.6. Notifications

Notifications are used to asynchronously inform the Buyer about:

- `ProductOfferingQualification.state` attribute value change,
- `ProductOfferingQualificationItem.state` attribute value change

Notifications are sent from Seller to Buyer in case:

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

The state change notifications are sent only in the Deferred scenario as in the Immediate scenario once the response to the POQ create request is provided there are no further state changes.

6.6.1. Use Case 1: Register for POQ Notifications

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.

By using a simple request:

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

The Buyer subscribes for notification of all types of events.

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

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

[R67] The Buyer **MUST** provide the `callback` during notification registration. [Mplify79.1 R7]

The `query` formatting complies with 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 the `eventType` attribute to be supported.

[R68] `eventType` is the only attribute that the Seller **MUST** support in the `query`.

[R69] If the Seller does not support notifications, they **MUST** return an error message (`Error501`) to the Buyer indicating that notifications are not supported.

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

```
{
  "callback": "https://buyer.mef.com/listenerEndpoint",
  "id": "1659bc83-d334-4de4-aa60-0818e4060ae1",
  "query": "eventType=poqStateChangeEvent"
}
```

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

- `https://buyer.mef.com/listenerEndpoint/mefApi/sonata/productOfferingQualificationNotification/v8/listener/poqStateChangeEvent`

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.

[R70] In the `unregisterListener` operation, the Buyer **MUST** provide the `id` of the registered `EventSubscription` that originates from the Seller.

The example below shows an exemplary unregister call sent by the Buyer to the Seller:

```
http://seller.mef.com:8080/mefApi/sonata/productOfferingQualification/v8/hub/1659bc83-d334-4de4-aa60-0818e4060ae1
```

[R71] In the successful scenario the Seller **MUST** respond with an empty body and HTTP code `204`.

The Buyer can unregister only the whole EventSubscription, regardless of the provided query. In the case when the Buyer e.g. resigns from specific types of events (or changes the callback address), the existing EventSubscription that includes undesired notification types that needs to be removed and replaced by the new EventSubscription with adjusted query attribute.

Note: The above note concludes that the Buyer cannot update the existing EventSubscription. Every kind of update is done by subscription replacement.

6.6.2. Use Case 5: Send POQ Notification

Figure 16 presents the model of Events.

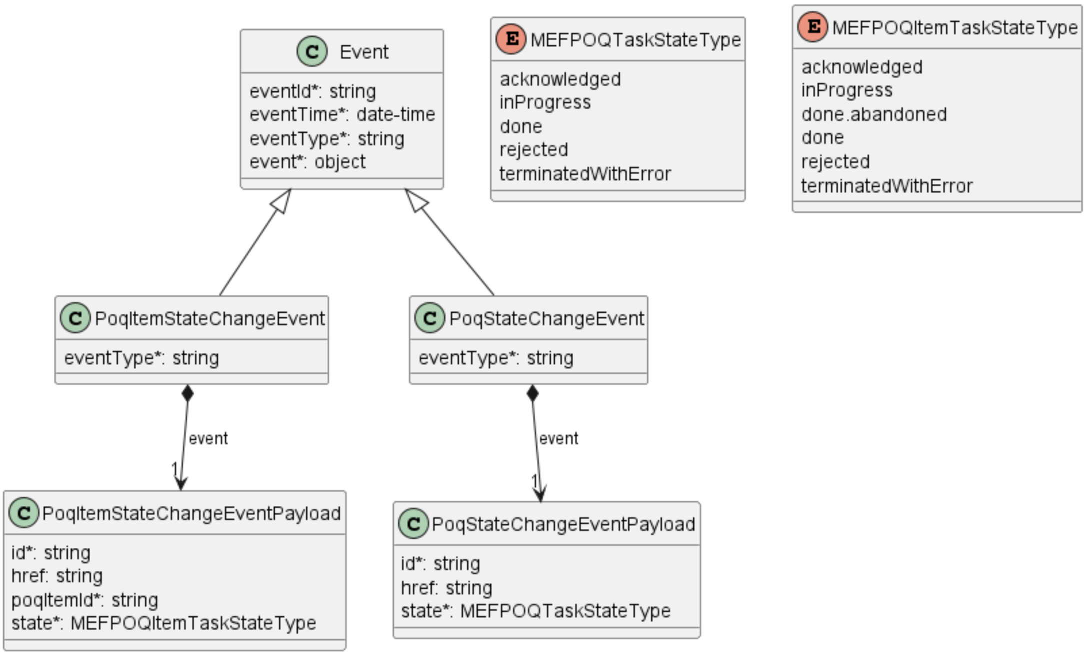


Figure 16. Event Model

An example of a POQ state change event might look like this:

```
{
  "eventId": "event-001",
  "eventType": "poqStateChangeEvent",
  "eventTime": "2024-08-07T01:07:42.7030052+01:00",
  "event": {
    "id": "00000000-0000-0000-0000-000000000b01",
    "state": "inProgress"
  }
}
```

[R72] If a Buyer has registered for notifications, the Seller **MUST** generate notifications to the Buyer. [Mplify79.1 R5]

[R73] If the Buyer has not registered for notifications, the Seller **MUST NOT** generate notifications to the Buyer. [Mplify79.1 R6]

[R74] Seller **MUST** send events only to Buyers who have registered to receive such notifications. [Mplify79.1 R90].

[R75] The state change notifications **MUST** be sent only in the Deferred Response scenario. [Mplify79.1 R91]

There are no state changes in the Immediate scenario.

[R76] The Seller **MUST** send a notification to all of the targets specified by the Buyer in their Register for POQ Notifications request. [Mplify79.1 R92]

[R76] means, that the Buyer may have multiple subscriptions listeners registered per single event type.

[R77] The Seller **MUST** provide the following attributes of `Event`:

- `event`
- `eventId`
- `eventTime`
- `eventType`

[R78] The Seller **MUST** provide the following attributes of `PoqStateChangeEventPayload` when sending `PoqStateChangeEvent`: [Mplify79.1 R92]

- `id`
- `state`

[R79] The Seller **MUST** provide the following attributes of `PoqItemStateChangeEventPayload` when sending `PoqItemStateChangeEvent`: [Mplify79.1 R93]

- `id`
- `poqItemId`
- `state`

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 POQ API uses the error responses depicted and described below.

Implementations can use HTTP error codes not specified in this standard in compliance with rules defined in RFC 7231 [RFC7231]. In such a case, the error message body structure might be aligned with the `Error`.

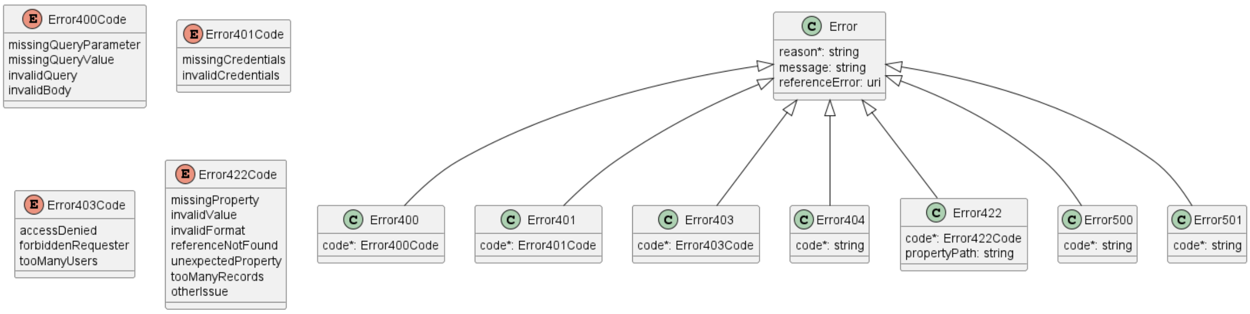


Figure 17. Data model types to represent an erroneous response

7.1.1.1. Type Error

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

Name	Type	Description
reason*	string *maxLength = 255*	Text that explains the reason for the error. This can be shown to a client user.
message	string	Text that provides mode details and corrective actions related to the error. This can be shown to a client user.
referenceError	uri *format = uri*	URL pointing to documentation describing the error

7.1.1.2. Type Error400

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

Inherits from:

- Error

Name	Type	Description

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

7.1.1.3. `enum` Error400Code

Description: One of the following error codes:

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

7.1.1.4. Type Error401

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

Inherits from:

- Error

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

7.1.1.5. `enum` Error401Code

Description: One of the following error codes:

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

7.1.1.6. Type Error403

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

Inherits from:

- Error

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

7.1.1.7. enum Error403Code

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

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

7.1.1.8. Type 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

7.1.1.9. 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*	Error422Code	One of the following error codes: - missingProperty: The property the Seller has expected is not present in the payload - invalidValue: The property has an incorrect value - invalidFormat: The property value does not comply with the expected value format - referenceNotFound: The object referenced by the property cannot be identified in the Seller system - unexpectedProperty: Additional property, not expected by the Seller has been provided - tooManyRecords: the number of records to be provided in the response exceeds the Seller's threshold. - otherIssue: Other problem was identified (detailed information provided in a reason)

Name	Type	Description
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).

7.1.1.10. enum Error422Code

Description: One of the following error codes:

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

7.1.1.11. 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.

7.1.1.12. Type Error501

Description: Not Implemented. Used in case Seller is not supporting an optional operation (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

7.2. Management API Data model

Figure 18 presents the Product Offering Qualification Management data model. The data types, requirements related to them, and mapping to MEF 79.1 specification are discussed later in this section.

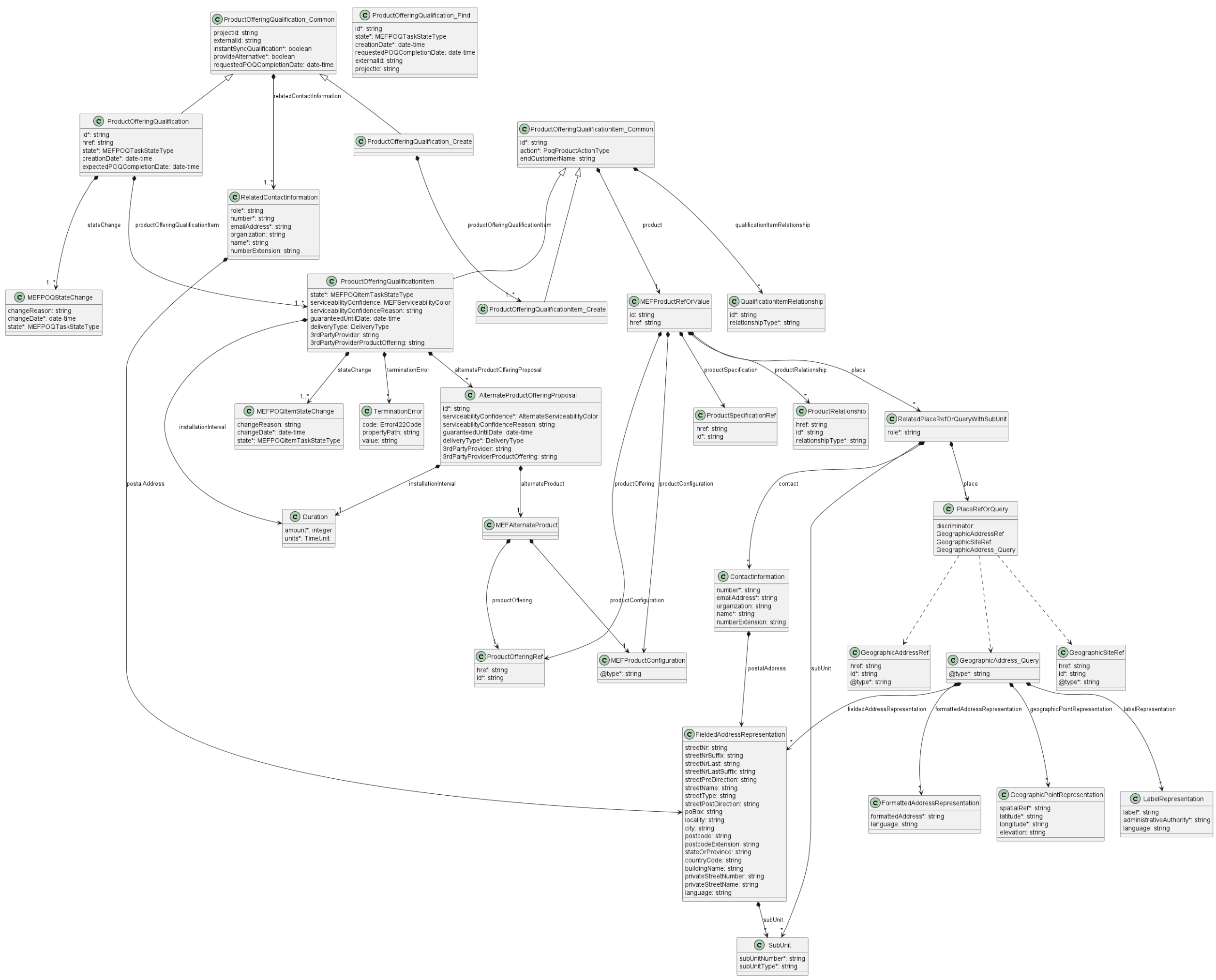


Figure 18. Product Offering Qualification Management Data Model

The POQ Management data model is used to construct requests and responses of the API endpoints described in Section 5.1.1

7.2.1 Product Offering Qualification

7.2.1.1. Type ProductOfferingQualification_Common

Description: Defines a set of POQ attributes that might be used by the Buyer and cannot be modified by the Seller. The `relatedContactInformation` entries provided by the Buyer cannot be changed by the Seller, however the Seller might append related contact information to that list.`

Name	Type	M/O	Description	Mplify 79.1
projectId	string	O	An identifier that is used to group Product Offering Qualifications that represent a unit of functionality that is important to a Buyer. A Project can be used to relate multiple Product Offering Qualifications together.	Project Identifier

Name	Type	M/O	Description	Mplify 79.1
externalId	string	O	An identifier that is used to group things that represent a unit of functionality that is important to a Buyer (unique for the Buyer). A Project can be used to relate multiple requests together such as POQ requests, Product Orders, etc.	External Identifier
instantSyncQualification	boolean	M	If this flag is set to true, the Buyer requires an Immediate Response to this request. If the Seller is unable to provide an Immediate Response, the Seller is to reply with an appropriate error.	Immediate Response Only
provideAlternative	boolean	M	If set to true it means that the wishes to receive alternative solutions. The Seller may provide Alternative Product Offering Configurations in the response such as a Product Offering with a lower bandwidth than requested. If "false" the Seller is to reply only with exact matches.	Provide Alternate
requestedPOQCompletionDate	date-time *format = date-time*	O	The desired date by which the Buyer wants to receive a response to the Create Product Offering Qualification request. If the Seller cannot meet the expected date, the Seller may choose to reject the request using the `terminatedWithError` state.	Expected Response Date
relatedContactInformation	RelatedContactInformation[] *minItems = 1*	M	Party playing a role for this qualification. Buyer Contact Information' **MUST** be provided in the request ('role=buyerContactInformation') and 'Seller Contact Information' **MUST** be provided in the response ('role=sellerContactInformation')	Allows for specifying Buyer and Seller Contact Information

7.2.1.2. Type ProductOfferingQualification_Create

Description: Represents a request formulated by the Buyer that is composed of product offering qualification items.

Inherits from:

- ProductOfferingQualification_Common

Name	Type	M/O	Description	Mplify 79.1
productOffering-QualificationItem	ProductOffering-QualificationItem_Create[] *minItems = 1*	M	The Product Offering Qualification is composed of Product Offering Qualification Items. This is the list of associated Product Offering Qualification Items.	Product Offering Qualification Items

7.2.1.3. Type ProductOfferingQualification

Description: Represents a response to the Buyer's POQ inquiry. This type defines a set of attributes that are assigned by the Seller while processing the request. A POQ response is a combination of attributes defined here with common attributes that are sent in the request. This type is used in response to an immediate request and POQ retrieval by an identifier.

Inherits from:

- ProductOfferingQualification_Common

Name	Type	M/O	Description	Mplify 79.1
id	string	M	The identifier of the Product Offering Qualification that is unique within this Seller.	POQ Identifier
href	string	O	Hyperlink to this POQ. 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 Mplify 79.1
state	MEFPOQ-TaskStateType	M	The state of the qualification	POQ State
creationDate	date-time *format = date-time*	M	Date when the POQ was created within the Seller's system	POQ Create Date

Name	Type	M/O	Description	Mplify 79.1
expectedPOQ-CompletionDate	date-time *format = date-time*	O	The date the Seller expects to provide qualification result. Set by the Seller in case of providing a deferred response when the POQ is in an acknowledged or inProgress state.	Not represented in Mplify 79.1
productOfferingQualificationItem	ProductOffering-QualificationItem[] *minItems = 1*	M	One or more of Product Offering Qualification Items. It MUST contain exactly one entry for each item in the POQ request.	Product Offering Qualification Items
stateChange	MEFPOQ-StateChange[] *minItems = 1*	M	A log of all state transitions for the POQ. It must be in sync with the most recent POQ Request state.	Not represented in Mplify 79.1

7.2.1.4. `enum` MEFPOQTaskStateType

Description: These values represent the valid states through which the product offering qualification can transition.

The following mapping has been used between `MEFPOQTaskStateType` and Mplify 79.1:

MEFPOQTaskStateType	Mplify 79.1	Description

MEFPOQTaskStateType	Mplify 79.1	Description
acknowledged	ACKNOWLEDGED	A request has been received by the Seller, has passed basic validation, and the id was assigned. For an Immediate response, the POQ moves directly to the `done` state and does not pass through `acknowledged`.
inProgress	IN_PROGRESS	The POQ is currently being worked by the Seller.
done	READY	The POQ has been internally approved by the Seller. Reached when all items are in `done` state. It does not imply that the Seller can deliver all POQ Items in this POQ. It only means that the POQ has been completed.
rejected	REJECTED	A POQ was submitted, and it has failed at least one of the business validation checks the Seller performs after it reached the `acknowledged` state.
terminatedWithError	UNABLE_TO_MEET_TIME	The Seller is unable to provide a response in the timeframe required by the Buyer (e.g. if an immediate response or a response date is set but cannot be met by the Seller).

7.2.1.5. Type MEFPOQStateChange

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

Name	Type	M/O	Description	Mplify 79.1
changeReason	string	O	Additional comment related to state change	Not represented in Mplify 79.1
changeDate	date-time *format = date-time*	M	The date on when the state was reached	Not represented in Mplify 79.1
state	MEFPOQTaskStateType	M	The state reached at the change date	Not represented in Mplify 79.1

7.2.1.6. Type ProductOfferingQualification_Find

Description: This class represent a single list item for the response of `listProductOfferingQualification` operation.

Name	Type	M/O	Description	Mplify 79.1
id	string	M	The POQ Request's unique identifier.	POQ Identifier
state	MEFPOQTask-StateType	M	Current state of the POQ	POQ State
creationDate	date-time *format = date-time*	M	Date when the POQ was created within the Seller's system	POQ Create Date
requestedPOQ-CompletionDate	date-time *format = date-time*	O	The latest date the POQ completion is expected by the Buyer, if specified by the Buyer.	Requested Response Date
externalId	string	O	ID given by the consumer and only understandable by him (to facilitate his searches afterwards)	External Identifier
projectId	string	O	The project ID specified by the Buyer in the POQ Request, if any.	Project Identifier

7.2.2. Product Offering Qualification Item

7.2.2.1. Type ProductOfferingQualificationItem_Common

Description: Common attributes shared between a POQ request and response. These attributes are provided by the Buyer and must not be modified by the Seller.

Name	Type	M/O	Description	Mplify 79.1
id	string	M	Id of this POQ item which is unique within the POQ. Assigned by the Buyer.	Product Offering Qualification Item Identifier
action	PoqProductActionType	M	The POQ Item Action associated with this POQ item. `add` means that this POQ Item being evaluated is a completely new deployment. `modify` means that this is a change to an exist-ing Product (e.g., to increase the bandwidth).	POQ Activity

Name	Type	M/O	Description	Mplify 79.1
product	MEFProductRefOrValue	M	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` to point which product instance to update and to what state.	Related to Product Specific Attributes, Product Relationships
qualification-ItemRelationship	Qualification-ItemRelationship[]	O	A list of references to related POQ items in this POQ	POQ Item Relationships
endCustomerName	string	O	The name of the End Customer. The actual user of the Product that contracts for the Product with the Buyer or the Buyer's representative.	POQ Item End Customer Name

7.2.2.2. `enum` PoqProductActionType

Description: Action to be performed on the Product Item.

The following mapping has been used between `PoqProductActionType` and Mplify 79.1:

PoqProductActionType	Mplify 79.1
add	INSTALL
modify	CHANGE

7.2.2.3. 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	M/O	Description	Mplify 79.1

Name	Type	M/O	Description	Mplify 79.1
href	string	O	Hyperlink to a Product Offering. The catalog is provided by the Seller to the Buyer during onboarding. Hyperlink MAY be used by the Seller in responses Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Product Offering Identifier
id	string	M	id of a Product Offering. The Buyer and the Seller exchange information about offerings' ids during the onboarding process.	Product Offering Identifier

7.2.2.4. Type QualificationItemRelationship

Description: The relationship between product offering qualification items that can be used to validate business rules between POQ items.

Name	Type	M/O	Description	Mplify 79.1
id	string	M	An identifier of the targeted POQ item within the same POQ request	Related POQ Item Identifier
relationshipType	string	M	Specifies the type (nature) of the relationship to the related Product. The nature of required relationships varies for Products of different types. For example, a UNI or ENNI Product may not have any relationships, but an Access E-Line may have two mandatory relationships (related to the UNI on one end and the ENNI on the other). More complex Products such as multipoint IP or Firewall Products may have more complex relationships. As a result, the allowed and mandatory `relationshipType` values are defined in the Product Specification.	Relationship Nature

7.2.2.5. Type ProductOfferingQualificationItem_Create

Description: This structure serves as a request for a product offering qualification item. A product qualification item is an individual article included in a POQ that describes a Product of a particular type (Product Offering) being delivered to the geographic address or a service site specified by the Buyer. The objective is to determine if it is feasible for the Seller to deliver this item as described and for the Seller to inform the Buyer of the estimated time interval to complete this delivery. The modelling pattern introduces the `Common` supertype to aggregate attributes that are common to both `ProductOfferingQualificationItem` and `ProductOfferingQualificationItem_Create`. It happens that it is the Create type has a subset of attributes of the response type and does not introduce any new, thus the `Create` type has an empty definition.

Inherits from:

- ProductOfferingQualificationItem_Common

7.2.2.6. Type ProductOfferingQualificationItem

Description: An individual article included in a POQ that describes a Product of a particular type (Product Offering) being delivered to a specific geographical location. The objective is to determine if it is feasible for the Seller to deliver this item as described and for the Seller to inform the Buyer of the estimated time interval to complete this delivery.

Inherits from:

- ProductOfferingQualificationItem_Common

Name	Type	M/O	Description	Mplify 79.1
state	MEFPOQItem-TaskStateType	M	Current state of an item	POQ Item State
stateChange	MEFPOQItem-StateChange[] *minItems = 1*	M	A log of all state transitions for the POQ Item. It must be in sync with the most recent POQ Item's state.	Not represented in Mplify 79.1
serviceability-Confidence	MEFServiceability-Color	O	The level of confidence of the Seller to be able to service the request. When the item state is \`done\` the Seller **MUST** provide a value. It **MUST NOT** be populated for other states.	POQ Confidence Level
serviceability-ConfidenceReason	string	O	A free text description of the reason a particular Serviceability Confidence is being provided.	Not represented in Mplify 79.1
installationInterval	Duration	O	The estimated minimum interval that the Seller requires in their standard process to complete the delivery of this Product from the time the order is placed and any precedents have been completed. When attribute \`serviceabilityConfidence\` is set to \`green\` or \`yellow\` the Seller **MUST** populate this attribute. **MUST NOT** be specified if 'state' is 'terminatedWithError' or \`done.abandoned\`.	Installation Interval (Value + Unit)
guaranteed-UntilDate	date-time *format = date-time*	O	Date until the Seller guarantees the qualification result. **MUST NOT** be specified if 'state' is 'terminatedWithError' or 'done.abandoned'.	Guaranteed Until Date

Name	Type	M/O	Description	Mplify 79.1
alternateProduct-OfferingProposal	AlternateProduct-OfferingProposal[]	O	A list of zero or more alternative Product Offerings that the Seller is proposing to the Buyer. The Seller MAY specify items if: 1) the Buyer has set `provideAlternate=true`; 2) the Seller has determined that the POQ Confidence Level for this item is `yellow` or `red`; and 3) The Seller has alternate Products (e.g. similar but lower bandwidth) that may be adequate. If the Buyer has requested to provide alternatives, but the Seller cannot find any, then the Seller **MUST** return `alternateProductOfferingProposal` array with 0 items.	Alternate Product Proposals
deliveryType	DeliveryType	O	The mechanism used to deliver the Product to the place of its installation.	Delivery Type
3rdPartyProvider	string	O	If the Delivery Type is `offNet*` it specifies what 3rd party provider is used to connect to that place of installation.	3rd Party Provider
3rdPartyProvider-ProductOffering	string	O	If the Delivery Type is `offNet*`, it specifies the 3rd party provider's offering is used to connect to that place of installation.	3rd Party Provider Product Offering
terminationError	TerminationError[]	O	A list of text-based reasons the Seller MUST provide when the request cannot be processed. When item state is `terminatedWithError` the Seller **MUST** provide at least one termination error.	Termination Error

7.2.2.7. enum MEFServiceabilityColor

Description: A color that indicates confidence to service the request. When the item state is done the Seller **MUST** provide a value. It **MUST NOT** be populated for other states.

ServiceabilityColor	Description
green	The Seller is highly confident that they can deliver the Product Offering. It is not expected that the Seller will change this after an order is placed.

ServiceabilityColor	Description
yellow	The Seller is confident that they can deliver the Product Offering subject to a feasibility check. The delivery of the Product Offering is subject to checks that the Seller may not carry out until later in the pre-sales/fulfillment process. The Seller cannot be highly confident that they can deliver the Product Offering until those checks are performed. There is a possibility that the Seller may determine that they will not be able to deliver the Product Offering.
red	The Seller cannot deliver the Product Offering as specified. The Seller may or may not have an alternate Product Offerings that could be substituted.

7.2.2.8. `enum` AlternateServiceabilityColor

Description: A color that indicates confidence to service the request.

ServiceabilityColor	Description
green	The Seller is highly confident that they can deliver the Product Offering. It is not expected that the Seller will change this after an order is placed.
yellow	The Seller is confident that they can deliver the Product Offering subject to a feasibility check. The delivery of the Product Offering is subject to checks that the Seller may not carry out until later in the pre-sales/fulfillment process. The Seller cannot be highly confident that they can deliver the Product Offering until those checks are performed. There is a possibility that the Seller may determine that they will not be able to deliver the Product Offering.

Value	Mplify 79.1
green	GREEN
yellow	YELLOW

7.2.2.9. `enum` MEFPOQItemTaskStateType

Description: POQ item states - The specific states are managed by the Seller based on its processing and/or based on the Buyer's action.

MEFPOQItemTaskStateType	Mplify 79.1	Description
acknowledged	ACKNOWLEDGED	A request has been received by the Seller and has passed basic validation. For an Immediate response, the POQ moves directly to `done` state and does not pass through `acknowledged`.
inProgress	IN_PROGRESS	The Seller is working on a POQ item response and the answer is not ready yet

MEFPOQItemTaskStateType	Mplify 79.1	Description
done.abandoned	ABANDONED	Applied to a POQ Item in case the final state is not reached and POQ is moved to the final state other than `done`
done	READY	The POQ Item has been internally approved by the Seller. This state does not imply that the Seller is able to deliver the requested item. It only means that the response for this POQ Item is complete.
rejected	REJECTED	A POQ Item has failed the business validation checks the Seller performs after it reaches the `acknowledged` state.
terminatedWithError	UNABLE_TO_MEET_TIME	The Seller is unable to provide a POQ Item response in the timeframe required by the Buyer (e.g. if an immediate response or a response date is set but cannot be met by the Seller). When a POQ Item goes to `terminatedWithError`, all POQ Items that are not `done` move to `done.abandoned`.

7.2.2.10. Type MEFPOQItemStateChange

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

Name	Type	M/O	Description	Mplify 79.1
changeReason	string	O	Additional comment related to state change	Not represented in Mplify 79.1
changeDate	date-time *format = date-time*	M	The date on when the state was reached	Not represented in Mplify 79.1
state	MEFPOQItemTaskStateType	M	The state reached at the change date	Not represented in Mplify 79.1

7.2.2.11. `enum` DeliveryType

Description: The mechanism used to deliver the Product to the place of its installation.

Delivery Type	Description
offNetWithBuild	A place does not have an existing connection to the Seller's network, but the Seller can connect via a partner who is willing to build out their network to connect to the place (refer to definition of Build). Some providers may refer to this as Near-Net.
offNetWithoutBuild	A place that does not have an existing connection to the Seller's network, but the Seller can connect via a 3rd party's existing connection. This may require augmentation of additional equipment, but no Build is needed for the connection.
onNetWithBuild	A place that does not have an existing connection to the Seller's network, but to which the Seller is willing to build out their network to connect to the place (refer to definition of Build). Some providers may refer to this as Near-Net.
onNetWithoutBuild	A place that has an existing connection to the Seller's network. This may require the augmentation of additional equipment.

Value	Mplify 79.1
offNetWithBuild	OFF-NET w/Build
offNetWithoutBuild	OFF-NET w/o Build
onNetWithBuild	ON-NET w/Build
onNetWithoutBuild	ON-NET w/o Build

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

7.2.3. Product representation

7.2.3.1. Type MEFProductRefOrValue

Description: Used by the Buyer to point to existing and/or describe the desired shape of the product. In the case of the `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.

Name	Type	M/O	Description	Mplify 79.1
id	string	O	The unique identifier of an in-service Product that is the qualification's subject. This field MUST be populated if an item `action` is either `modify`. This field MUST NOT be populated if an item `action` is `add`.	Product Identifier
href	string	O	Hyperlink to the referenced Product. Hyperlinks may be used by the Seller in responses. Hyperlinks MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in Mplify 79.1
product-Specification	Product-SpecificationRef	O	The identifier of a Product Specification associated with this POQ Item. This identifier is specified by the Seller and is established between the Buyer and Seller prior to issuing any POQ requests.	Not represented in 79
product-Offering	Product-OfferingRef	O	The identifier for a Product Offering associated with this POQ Item. This identifier is specified by the Seller and is established between the Buyer and Seller prior to issuing any POQ requests.	Product Offering Identifier

Name	Type	M/O	Description	Mplify 79.1
product-Configuration	MEFProduct-Configuration	O	The technical attributes for the Product that would be delivered to fulfill this POQ Item. This essentially specifies the values for attributes defined in the Product Specification or Product Offering. MEFProductConfiguration is used to specify the product-specific payload. This field MUST be populated if an item \`action\` is \`add\` or \`modify.	Product Specific Attributes
productRelationship	ProductRelationship[]	O	A list of references to existing products that are related to the Product that would be delivered to fulfill the POQ Item.	Product Relationships
place	RelatedPlace-RefOrQueryWithSubUnit[]	O	A list of locations that are related to the Product. For example an installation location	POQ Item Location and POQ Item Location Type

7.2.3.2. Type MEFProductConfiguration

Description: The extension point to provide product-specific payload. The @type is used as a discriminator.

Name	Type	M/O	Description	Mplify 79.1
@type	string	M	The name of the type that uniquely identifies the type of the product that is the subject of the POQ Request. In the case of Mplify product this is the URN provided in the Product Specification.	Not represented in Mplify 79.1

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. When the Buyer provides multiple ProductRelationships of the same relationshipType the Seller determines if a list is supported as defined in the Product Specification.

Name	Type	M/O	Description	Mplify 79.1

Name	Type	M/O	Description	Mplify 79.1
href	string	O	Hyperlink to the product in Seller's inventory that is referenced Hyperlink MAY be used when providing a response by the Seller Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request	Not represented in Mplify 79.1
id	string	M	unique identifier of the related Product	Related Product Identifier
relationshipType	string	M	Specifies the type (nature) of the relationship to the related Product. The nature of required relationships varies for Products of different types. For example, a UNI or ENNI Product may not have any relationships, but an Access E-Line may have two mandatory relationships (related to the UNI on one end and the ENNI on the other). More complex Products such as multipoint IP or Firewall Products may have more complex relationships. As a result, the allowed and mandatory `relationshipType` values are defined in the Product Specification.	Relationship Nature

7.2.3.4. Type AlternateProductOfferingProposal

Description: If in request the Buyer has requested to have alternate product proposals, then this class represents a single proposal. All properties are assigned by the Seller.

Name	Type	M/O	Description	Mplify 79.1
id	string	M	A unique identifier for this Alternate Product Offering Proposal assigned by the Seller.	Alternate Product Proposal Identifier

Name	Type	M/O	Description	Mplify 79.1
serviceability-Confidence	Alternate-ServiceabilityColor	M	The level of confidence of the Seller to be able to service the request with the Alternate Product Offering. Note that this response is only an evaluation of the technical feasibility of delivery independent of when the Product Offering can be delivered. When the item state is `done` the Seller **MUST** provide a value. It **MUST NOT** be populated for other states.	Alternate Product Offering POQ Confidence Level
serviceability-ConfidenceReason	string	O	The reason for the Serviceability Confidence Reason value.	Alternate Product Offering POQ Confidence Reason
installationInterval	Duration	M	The estimated minimum interval that the Seller requires in their standard process to complete the delivery of this product from the time the order is placed and any precedents have been completed.	Installation Interval Value and Installation Interval Unit
alternateProduct	MEFAlternateProduct	M	Alternate product proposal	related to Product Specific Attributes
guaranteedUntilDate	date-time *format = date-time*	O	Date until the Seller guarantees the qualification result.	Guaranteed Until Date
deliveryType	DeliveryType	M	The mechanism used to deliver the Product to the place of its installation.	Delivery Type
3rdPartyProvider	string	O	If the Delivery Type is `offNet*` it specifies what 3rd party provider is used to connect to that place of installation.	3rd Party Provider

Name	Type	M/O	Description	Mplify 79.1
3rdPartyProvider-ProductOffering	string	O	If the Delivery Type is `offNet*`, it specifies the 3rd party provider's offering is used to connect to that place of installation.	3rd Party Provider Product

7.2.3.5. Type MEFAlternateProduct

Description: An alternative Product Offering that the Seller is proposing to the Buyer.

Name	Type	M/O	Description	Mplify 79.1
productOffering	ProductOfferingRef	M	A reference to the alternate product offering.	Product Offering Identifier
productConfiguration	MEFProductConfiguration	M	MEFProductConfiguration is used to provide product-specific payload. The @type is used as a discriminator.	Product Specific Attributes

7.2.3.6. Type ProductSpecificationRef

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

Name	Type	M/O	Description	Mplify 79.1
href	string	O	Hyperlink to a Product Specification in the Seller's catalog. The catalog is provided by the Seller to the Buyer during onboarding. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in Mplify 79.1
id	string	M	Unique identifier of the product specification	Not represented in Mplify 79.1

7.2.4. Place representation

7.2.4.1. Type RelatedPlaceRefOrQueryWithSubUnit

Description: Allows pointing to a place by referring to a GeographicAddress, GeographicSite, or providing GeographicAddress by value. It also provides additional information like the `role` the place plays for given Product, subUnit to provide more detailed information about the precise location of the installation and contact needed access to this place.

Name	Type	M/O	Description	Mplify 79.1
place	PlaceRefOrQuery	M		POQ Item Place
role	string	M	Role of this place. The values that can be specified here are described by Product Specification (e.g. "INSTALL_LOCATION").	POQ Item Place Role
subUnit	SubUnit[]	O	A list of zero or more sub units included within the boundary of the `place` for this POQ Item. This is a list to allow complex sub-unit information such as SUITE 42 ROOM A	POQ Item Place Sub Unit List
contact	ContactInformation[]	O	The person to call to get access to this place in case such access is required to complete the evaluation of this POQ Item.	POQ Item Place Contact

7.2.4.2. Type PlaceRefOrQuery

Description: A place described by reference to a Geographic Address, Geographic Site or by Geographic Address Representations.

7.2.4.3. Type GeographicAddress_Query

Description: A list of representations being a subset of Geographic Address entity. This is to be used when providing a list of representations to validate or search for a Geographic Address

Name	Type	M/O	Description	Mplify 79.1
fieldedAddressRepresentation	FieldedAddressRepresentation[]	O	A list of Fielded Address representations	Installation Place Representations
formattedAddressRepresentation	FormattedAddressRepresentation[]	O	A list of Formatted Address representations	Installation Place Representations
geographicPointRepresentation	GeographicPointRepresentation[]	O	A list of Geographic Point Address representations	Installation Place Representations
labelRepresentation	LabelRepresentation[]	O	A list of Label Address representations	Installation Place Representations

Name	Type	M/O	Description	Mplify 79.1
@type	string	M	Used to unambiguously designate the class type when using `oneOf`	Not represented in Mplify 150

7.2.4.4. Type FieldedAddressRepresentation

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.

Name	Type	M/O	Description	Mplify 79.1
streetNr	string	O	Number identifying a specific property on a public street. It may be combined with streetNrLast for ranged addresses.	Street Number
streetNrSuffix	string	O	The first street number suffix (in a street number range) or the suffix for the street number if there is no range	Street Number Suffix
streetNrLast	string	O	Last number in a range of street numbers allocated to an Address	Street Number Last
streetNrLastSuffix	string	O	Last street number suffix for a ranged Address	Street Number Last Suffix
streetPreDirection	string	O	The direction of the street that appears before the Street Name	Street Pre-Direction
streetName	string	O	Name of the street or other street type	Street Name
streetType	string	O	The type of street (e.g., alley, avenue, boulevard, brae, crescent, drive, highway, lane, terrace, parade, place, tarn, way, wharf)	Street Type
streetPostDirection	string	O	A modifier denoting a relative direction that appears after the Street Name.	Street Post-Direction
poBox	string	O	Number identifying a specific location in a post office.	PO Box Number
locality	string	O	An area of defined or undefined boundaries within a local authority or other legislatively defined area.	Locality
city	string	O	City in which the Address is located.	City

Name	Type	M/O	Description	Mplify 79.1
postcode	string	O	A descriptor for a postal delivery area used to speed and simplify the delivery of mail (also known as zip code)	Postal Code
postcodeExtension	string	O	The extension used on a postal code. Note: there are different use codes for this attribute depending upon the country.	Postal Code Extension
stateOrProvince	string	O	The State or Province in which the Address is located.	State or Province
countryCode	string *minLength = 2* *maxLength = 2*	O	Country in which the Address is located, defined using two characters as defined in ISO 3166	Country
subUnit	SubUnit[]	O	The Sub Unit represented as a list. This is a list to allow complex sub-unit information such as SUITE 42 ROOM A	Sub Units
buildingName	string	O	The well-known name of a building that is located at this Address (e.g., where there is one Address for a campus).	Building Name
privateStreetNumber	string	O	Street number on a private street within the Address.	Private Street Number
privateStreetName	string	O	Private streets internal to a property (e.g., a university) may have internal names that are not recorded by the land title office.	Private Street Name
language	string *minLength = 2* *maxLength = 2*	O	The language in which the address is expressed. It MUST use the ISO 639:2023 two letter code 639:2023	Language

7.2.4.5. Type FormattedAddressRepresentation

Description: A freeform text representation agreed to by the Buyer and Seller.

Name	Type	M/O	Description	Mplify 79.1
formattedAddress	string	M	A formatted Address Representation that contains a non-fielded address.	Formatted Address
language	string *minLength = 2* *maxLength = 2*	O	The language in which the address is expressed. Based on ISO 639:2023	Language

7.2.4.6. Type GeographicPointRepresentation

Description: A GeographicPointRepresentation defines a geographic point through coordinates.

Name	Type	M/O	Description	Mplify 79.1
spatialRef	string	M	The spatial reference system used to determine the coordinates. The system used and the value of this field are to be agreed during the onboarding process.	Spatial Reference
latitude	string	M	The latitude expressed in the format specified by the `spacialRef`	Latitude
longitude	string	M	The longitude expressed in the format specified by the `spacialRef`	Longitude
elevation	string	O	The elevation expressed in the format specified by the `spacialRef`	Elevation

7.2.4.7. Type LabelRepresentation

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

Name	Type	M/O	Description	Mplify 79.1
label	string	M	The unique reference to a Geographic Address assigned by the Administrative Authority.	Installation Place Label
administrative-Authority	string	M	The organization or standard from the organization that administers this Geographic Address Label ensuring it is unique within the Administrative Authority.	Administrative Authority
language	string *minLength = 2* *maxLength = 2*	O	The language in which the label is expressed. Based on ISO 639:2023	Language

7.2.4.8. Type GeographicAddressRef

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

Name	Type	M/O	Description	Mplify 79.1
href	string	O	Hyperlink to the referenced Address. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in Mplify 79.1

Name	Type	M/O	Description	Mplify 79.1
id	string	M	Identifier of the referenced Geographic Address. This identifier is assigned during a successful address validation request (Geographic Address Management API)	Installation Place Identifier
@type	string	M	Used to unambiguously designate the class type when using `oneOf`	Not represented in Mplify 79.1

7.2.4.9. Type GeographicSiteRef

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

Name	Type	M/O	Description	Mplify 79.1
href	string	O	Hyperlink to the referenced Site. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in Mplify 79.1
id	string	M	Identifier of the referenced Geographic Site.	Site Identifier
@type	string	M	Used to unambiguously designate the class type when using `oneOf`	Not represented in Mplify 79.1

7.2.4.10. Type SubUnit

Description: Allows for sub unit identification

Name	Type	M/O	Description	Mplify 79.1
subUnitNumber	string	M	The discriminator used for the subunit, often just a simple number but may also be a range.	Sub Unit Name
subUnitType	string	M	The type of subunit e.g. BERTH, FLAT, PIER, SUITE, SHOP, TOWER, UNIT, WHARF.	Sub Unit Type

7.2.5. Notification registration

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

7.2.5.1. Type EventSubscriptionInput

Description: This class is used to register for Notifications.

Name	Type	M/O	Description	Mplify 79.1
query	string	O	This attribute is used to define which type of events to register to. Example: "query":"eventType = poqStateChangeEvent". To subscribe for more than one event type, put the values separated by a comma: `eventType=poqStateChangeEvent,-poqItemStateChangeEvent`. The possible values are enumerated by the 'POQEventType' in productOfferingQualificationNotification.api.yaml. An empty query is treated as specifying no filters - ending in subscription for all event types.	List of Notification Types
callback	string	M	This callback value must be set to *host* property from the Buyer ProductOfferingQualification Notification API (productOfferingQualificationNotification.api.yaml). This property is appended with the base path and notification resource path specified in that API to construct a URL to which notification is sent. E.g. for the "callback": "https://buyer.mef.com/listenerEndpoint", the create event notification will be sent to: `https://buyer.mef.com/listenerEndpoint/-mefApi/sonata/productOfferingQualificationNotification/-v8/listener/poqStateChangeEvent`	Return Address Information

7.2.5.2. Type EventSubscription

Description: This resource is used to manage notification subscriptions.

Name	Type	M/O	Description	Mplify 79.1
query	string	O	The value provided by the Buyer in `EventSubscriptionInput` during notification registration	List of Notification Types
callback	string	M	The value provided by the Buyer in `EventSubscriptionInput` during notification registration	Return Address Information
id	string	M	An identifier of this Event Subscription assigned by the Seller when a resource is created.	Not represented in Mplify 79.1

7.2.6. Common

7.2.6.1. Type RelatedContactInformation

Description: Contact data for a person or organization that is involved in the product offering qualification. In a given context it is always specified by the Seller (e.g. Seller Contact Information) or by the Buyer.

Name	Type	M/O	Description	Mplify 79.1

Name	Type	M/O	Description	Mplify 79.1
role	string	M	The role of the particular contact in the request	Not represented in Mplify 79.1
number	string	M	Phone number	Contract Phone Number
emailAddress	string	M	Email address	Contact email Address
postalAddress	FieldedAddress-Representation	O	Identifies the postal address of the person or office to be contacted.	Contact Postal Address
organization	string	O	The organization or company that the contact belongs to	Contact Organization
name	string	M	Name of the contact	Contact Name
numberExtension	string	O	Phone number extension	Contract Phone Number Extension

7.2.6.1. Type ContactInformation

Description: Contact data for a person or organization that is involved in the product offering qualification. In a given context it is always specified by the Seller (e.g. Seller Contact Information) or by the Buyer.

Name	Type	M/O	Description	Mplify 79.1
number	string	M	Phone number	Contact Phone Number
emailAddress	string	M	Email address	Contact Email Address
postalAddress	FieldedAddress-Representation	O	Identifies the postal address of the person or office to be contacted.	Contact Postal Address
organization	string	O	The organization or company that the contact belongs to	Contact Organization
name	string	M	Name of the contact	Contact Name
numberExtension	string	O	Phone number extension	Contact Phone Number Extension

7.2.6.2. Type Duration

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

Name	Type	M/O	Description	Mplify 79.1
amount	integer *minimum = 0*	M	Duration (number of seconds, minutes, hours, etc.)	Duration Value

Name	Type	M/O	Description	Mplify 79.1
units	TimeUnit	M	Time unit enumerated	Duration Unit

7.2.6.3. enum TimeUnit

Description: Represents a unit of time.

Value	Mplify 79.1
seconds	SECONDS
minutes	MINUTES
businessHours	BUSINESS_HOURS
calendarHours	CALENDAR_HOURS
businessDays	BUSINESS_DAYS
calendarDays	CALENDAR_DAYS
months	MONTHS
years	YEARS

7.3. Notification API Data model

Figure 19 presents the Product Offering Qualification Notification data model.

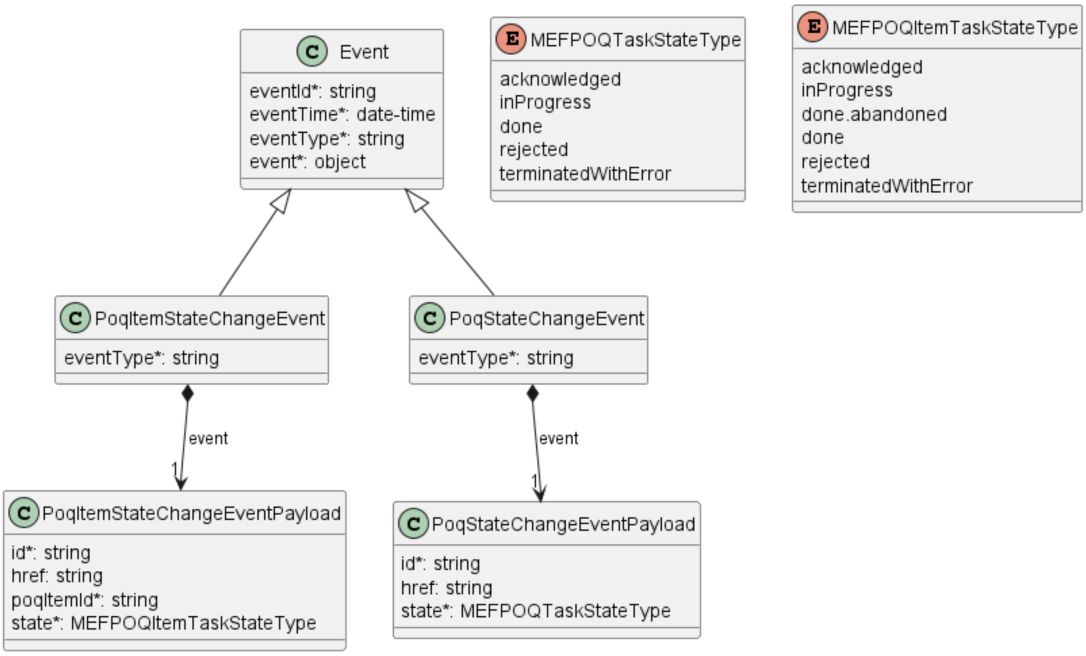


Figure 19. Product Offering Qualification Notifications Data Model

7.3.1. Type Event

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

Name	Type	M/O	Description	Mplify 79.1
eventId	string	M	Id of the event	Not represented in Mplify 79.1

Name	Type	M/O	Description	Mplify 79.1
eventTime	date-time *format = date-time*	M	Date-time when the event occurred	Not represented in Mplify 79.1
eventType	string	M	The type of the notification.	Notification Type
event	object	M	The event linked to the involved resource object	Not represented in Mplify 79.1

7.3.2. Type PoqStateChangeEvent

Description: PoqStateChangeEvent structure

Inherits from:

- Event

Name	Type	M/O	Description	Mplify 79.1
eventType	string	M	Indicates the type of product offering qualification event.	Notification Type
event	PoqStateChangeEventPayload	M	A reference to the POQ that is source of the notification.	POQ State

7.3.3. Type PoqStateChangeEventPayload

Description: A reference to the POQ that is the source of the notification.

Name	Type	M/O	Description	Mplify 79.1
id	string	M	The POQ unique identifier.	Product Offering Qualification Identifier
href	string	O	Link to the POQ	Not represented in Mplify 79.1
state	MEFPOQTaskStateType	M	The state reached at change date	POQ State

7.3.4. Type PoqItemStateChangeEvent

Description: PoqItemStateChangeEvent structure

Inherits from:

- Event

Name	Type	M/O	Description	Mplify 79.1
eventType	string	M	Indicates the type of product offering qualification event.	Notification Type
event	PoqItemStateChangeEventPayload	M	A reference to the POQ that is source of the notification.	POQ Item State

7.3.5. Type PoqItemStateChangeEventPayload

Description: A reference to the POQ that is the source of the notification.

Name	Type	M/O	Description	Mplify 79.1
id	string	M	The POQ unique identifier.	Product Offering Qualification Identifier
href	string	O	Link to the POQ	Not represented in Mplify 79.1
poqItemId	string	M	ID of the Poq Item (within the Poq) which state change triggered the event	Product Offering Qualification Item Identifier
state	MEFPOQItem-TaskStateType	M	The state reached at change date	POQ Item State

7.3.6. enum MEFPOQTaskStateType

Description: These values represent the valid states through which the product offering qualification can transition.

The following mapping has been used between MEFPOQTaskStateType and Mplify 79.1:

MEFPOQTaskStateType	Mplify 79.1	Description
acknowledged	ACKNOWLEDGED	A request has been received by the Seller, has passed basic validation and the id was assigned. For an Immediate response, the POQ moves directly to the done state and does not pass through acknowledged.
inProgress	IN_PROGRESS	The POQ is currently being worked by the Seller.
done	READY	The POQ has been internally approved by the Seller. Reached when all items are in done state. It does not imply that the Seller can deliver all POQ Items in this POQ. It only means that the POQ has been completed.
rejected	REJECTED	A POQ was submitted, and it has failed at least one of the business validation checks the Seller performs after it reached the acknowledged state.
terminatedWithError	UNABLE_TO_MEET_TIME	The Seller is unable to provide a response in the timeframe required by the Buyer (e.g., if an immediate response or a response date is set but cannot be met by the Seller).

7.3.7. enum MEFPOQItemTaskStateType

Description: POQ item states - The specific states are managed by the Seller based on its processing and/or based on the Buyer's action.

MEFPOQItemTaskStateType	Mplify 79.1	Description
acknowledged	ACKNOWLEDGED	A request has been received by the Seller and has passed basic validation. For an Immediate response, the POQ moves directly to `done` state and does not pass through `acknowledged`.
inProgress	IN_PROGRESS	The Seller is working on a POQ item response and the answer is not ready yet
done.abandoned	ABANDONED	Applied to a POQ Item in case the final state is not reached and POQ is moved to the final state other than `done`
done	READY	The POQ Item has been internally approved by the Seller. This state does not imply that Seller is able to deliver requested item. It only means that the response for this POQ Item is complete.
rejected	REJECTED	A POQ Item has failed the business validation checks the Seller performs after it reached the `acknowledged` state.
terminatedWithError	UNABLE_TO_MEET_TIME	The Seller is unable to provide a POQ Item response in the timeframe required by the Buyer (e.g., if an immediate response or a response date is set but cannot be met by the Seller). When a POQ Item goes to `terminatedWithError`, all POQ Items that are not `done` move to `done.abandoned`.

8. References

- JSON, JSON Schema: A Media Type for Describing JSON Documents and associated documents, by Austin Wright and Henry Andrews, March 2018. Copyright © 2018 IETF Trust and the persons identified as the document authors. All rights reserved.
- MEF 55.1, Lifecycle Service Orchestration (LSO): Reference Architecture and Framework, February 2021
- MEF 55.1.1, Amendment to MEF 55.1: Reference Architecture and Framework - Terminology, June 2023
- Mplify 79.1, Product Offering Qualification Management Business Requirements and Use Cases, June 2025
- MEF 106, LSO Sonata Access E-Line Product Schemas and Developer Guide, February 2023
- Mplify 121.1, LSO Cantata and LSO Sonata Address Management API - Developer Guide, July 2025
- MEF 128.1, LSO API Security Profile, April 2024
- Mplify 150, Installation Place and Service Site Management Business Requirements and Use Cases, June 2025
- OAS-V3, February 2020
- REST, Fielding, Roy Thomas, Architectural Styles and the Design of Network-based Software Architectures (Ph.D.).
- RFC 2119, Key words for use in RFCs to Indicate Requirement Levels, March 1997
- RFC 3986 Uniform Resource Identifier (URI): Generic Syntax, January 2005
- RFC 7231, Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content, June 2014
- RFC 8174, Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words, May 2017
- TMF 679, TMF679 Product Offering Qualification API REST Specification, June 2019

Appendix A Acknowledgments

Mike **BENCHECK**

Tomasz **CHMAL**

Pankaj **BODADE**

Michał **ŁĄCZYŃSKI**

Jack **PUGACZEWSKI**

Patrick **ROOSEN**

Fahim **SABIR**