Mplify Standard

Mplify 115.1

LSO Cantata and LSO Sonata Quote Management 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

The Quote allows the Buyer to submit a request to find out how much the installation of a new Product, an update to an existing Product, or a disconnect of an existing Product will cost.

This standard is intended to assist the implementation of the Quote functionality defined for the LSO Cantata and LSO Sonata Interface Reference Points (IRPs), for which requirements and use cases are defined in MEF 80 *Quote Management Requirements and Use Cases* [MEF 80] and it's *Amendment* [Mplify 80.0.1]. This standard consists of this document and complementary API definitions for Quote Management and Quote Notification.

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

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

commit id: aaa03d484f98664a5a14f4f54f47b675d7efb3b8

- `productApi/quote/quoteManagement.api.yaml`
- `productApi/quote/quoteNotification.api.yaml`

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

commit id: 83d6edd0c70386058a9af6e677c069b498671da7

- `productApi/quote/quoteManagement.api.yaml`
- `productApi/quote/quoteNotification.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 79.1
- MEF 80

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.	[MEF 55.1]
Budgetary Quote	A quote that is provided quickly and with very little analysis such that the Buyer can get an idea of how much the requested Product Offering could cost. Monthly Recurring Charges and Non-Recurring Charges, if specified, are subject to change.	[MEF 80]
Buyer	In the context of this document, denotes the organization or individual acting as the customer in a transaction over a Cantata (Customer <-> Service Provider) or Sonata (Service Provider <-> Partner) Interface.	This document; adapted from [MEF 55.1.1]
Completion State	A state a Quote is in when the Seller has completed processing their Quote. This is one of the following states: answered, approved.orderable, approved.orderableAlternate, cancelled, unableToProvide, rejected	This document
Currency	The unit of measurement in which a monetary cost is expressed. Currency is represented by currency codes defined in ISO 4217:2015.	[ISO4217]
Deferred Quote Response	A Seller's response to a Buyer's Create Quote Request whereby the Seller immediately acknowledges that the Create Quote Request was received and, over time, sends notifications to update the Buyer on the status and results of the Create Quote Request (assuming the Buyer has subscribed to receive the notifications). The Buyer can also poll the Seller for the Quote Request Response and status associated with the Create Quote Request.	[MEF 80]
Firm Quote	A quote provided to the Buyer based on a pre-order analysis. All	[MEF 80]

	Monthly Recurring Charges and Non-Recurring Charges specified on a Firm Quote are committed. A Firm Quote may expire at some date specified by the Seller.	
Firm - Subject to Feasibility Check Quote	A quote that is provided to the Buyer based on some, but not a complete, pre-order analysis. At this stage there is further analysis that the Seller can (and is willing) to undertake to provide a committed or firm price, but the Seller needs more time to complete this or the Seller may request that the Buyer use the Firm - Subject to Feasibility Check Quote to proceed to the Order process. Ordering may be possible based on the Firm - Subject to Feasibility Check Quote with some stipulations as to how the cost identified during delivery is addressed. The Monthly Recurring Charges specified in the Quote Response are final. Non-Recurring Charges specified in the Quote Response are subject to change.	[MEF 80]
Immediate Quote Response	A Seller's response to the Buyer's Create Quote Request, 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.	[MEF 80]
Recurring Charge	Charge for a product that is incurred by the Buyer each specified time interval for that product.	[MEF 80]
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 architectural constraints that, when applied as a whole, emphasizes scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems.	[REST]
Seller	In the context of this document, denotes the organization acting as the supplier in a transaction over a Cantata (Customer <-> Service Provider) or Sonata (Service Provider <-> Partner) Interface.	This document; adapted from [MEF 55.1.1]
Terminal State	A state in which the Quote is considered terminated, and no further actions may be taken by the Buyer. This is one of the following: accepted, cancelled, unableToProvide, declined, expired, rejected	This document

Table 2. Terminology

3. Compliance Levels

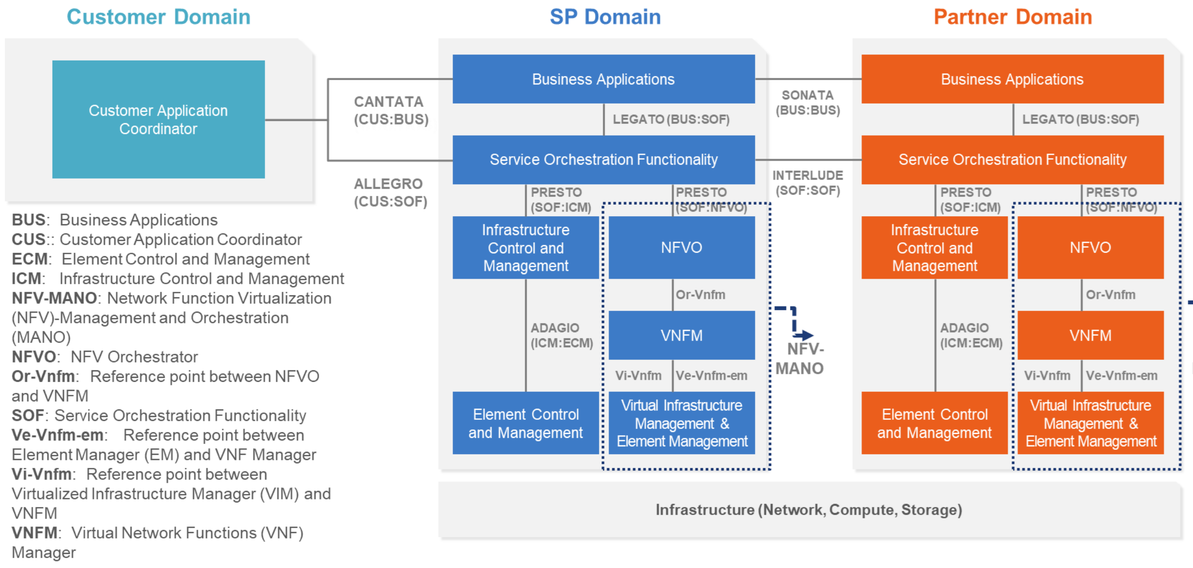
The key words **"MUST"**, **"MUST NOT"**, **"REQUIRED"**, **"SHALL"**, **"SHALL NOT"**, **"SHOULD"**, **"SHOULD NOT"**, **"RECOMMENDED"**, **"NOT RECOMMENDED"**, **"MAY"**, and **"OPTIONAL"** in this document are to be interpreted as described in BCP 14 ([RFC2119], [RFC8174]) when, and only when, they appear in all capitals, as shown here. All key words must be in bold text.

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

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 Quote Management functionality of the LSO Cantata Interface Reference Point (IRP) and LSO Sonata IRP as defined in the *MEF 55.1 Lifecycle Service Orchestration (LSO): Reference Architecture and Framework* [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
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
SONATA (BUS:BUS)
INTERLUDE (SOF:SOF)
NFV-MANO
Business Applications
LEGATO (BUS:SOF)
Service Orchestration Functionality
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 functionalities that allow an automated exchange of information between business applications of the Buyer (Customer or Service Provider) and Seller (Service Provider or Partner) Domains. Those are:

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

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

- Chapter 4 provides an introduction to Quote Management and its description in a broader context of Cantata and Sonata and their corresponding SDKs.
- Chapter 5 gives an overview of endpoints, resource model and design patterns.
- Use cases and flows are presented in Chapter 6.
- And finally, Chapter 7 complements previous sections with a detailed API description.

4.1. Description

The Quote Management API allows the Buyer to request a quote for the installation of one or more of the Seller's Product Offerings or for an action to be performed on an existing Product.

The API payloads exchanged between the Buyer and the Seller consist of product-independent and product-specific parts. The product-independent part is technically defined in this standard. The product-specific part is defined in the product specification standard of the concerned product. Both standards must be used in combination to validate the correctness of the payloads. Section 5.5 explains how to use product specifications as the Quote API payloads.

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 business requirements and use cases for Quote Management are defined in [MEF 80] and it's amendment [Mplify 80.0.1]. Product specifications are defined using JSON Schema (draft 7) standard [JS], whereas Quote API is defined using OpenAPI 3.0 [OAS-V3]. The payloads exchanged through Quote endpoints must comply with respective Product Specifications.

The API definition builds on *TMF648 Quote Management API REST Specification R19.0.1* [TMF648].

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)

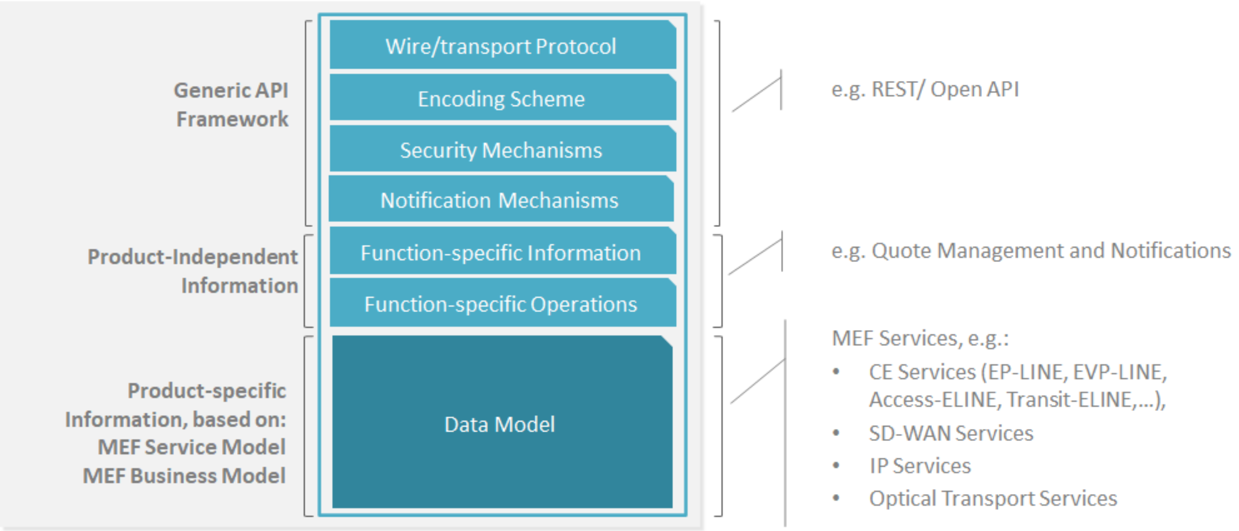

Generic API Framework
Product-Independent Information
Product-specific Information, based on: MEF Service Model MEF Business Model
Wire/transport Protocol
Encoding Scheme
Security Mechanisms
Notification Mechanisms
Function-specific Information
Function-specific Operations
Data Model
e.g. REST/ Open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

Figure 2. Cantata and Sonata API framework

The essential concept behind the framework is to decouple the common structure, information, and operations from the specific product information content.
Firstly, the Generic API Framework defines a set of design rules and patterns that are applied across all Cantata or Sonata APIs.
Secondly, the product-independent information of the framework focuses on a model of a particular Cantata or Sonata functionality and is agnostic to any of the product specifications. For example, this standard describes the Quote model and operations that allow performing quoting of any product that is aligned with either 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 Developer Guide does not define Mplify product specifications but can be used in combination with any product specifications defined by or compliant with Mplify.

4.5. High-Level Flow

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

The Quote is not required to be performed by the Buyer unless it is mandated by the Seller.

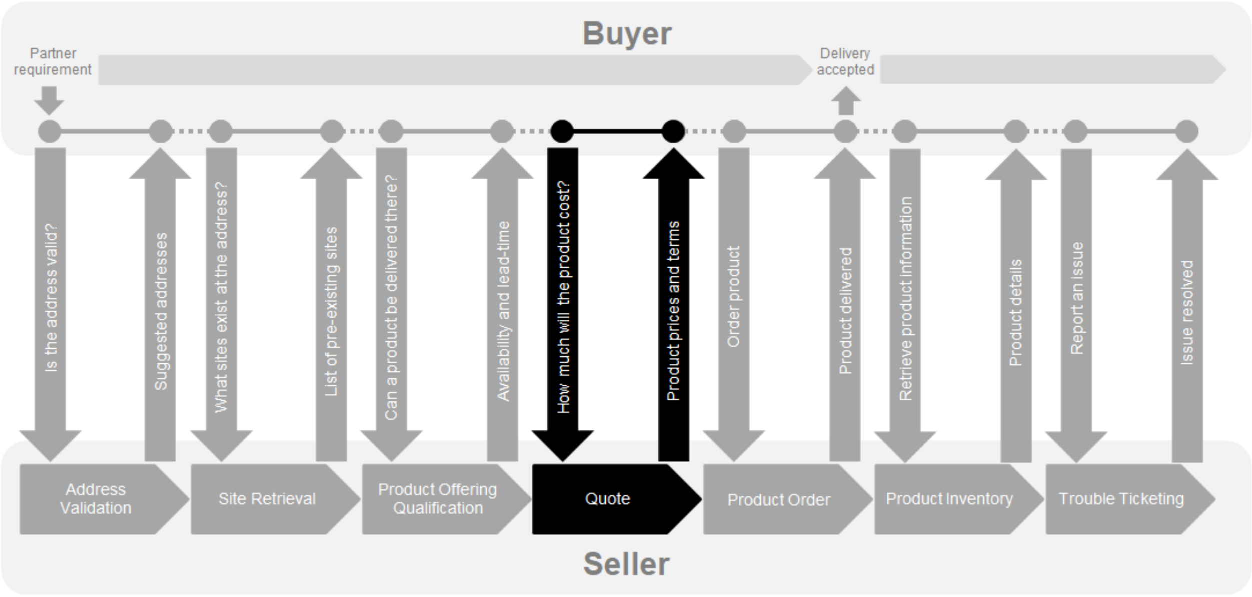


Figure 3. Cantata and Sonata End-to-End Function Flow

- Address Validation:
  - Allows the Buyer to retrieve address information from the Seller, including exact formats, for Geographic Addresses known to the Seller.
- Site Retrieval:
  - Allows the Buyer to retrieve Service Site information including exact formats for Service Sites known to the Seller.
- Product Offering Qualification (POQ):
  - Allows the Buyer to check whether the Seller can deliver a product or set of products from among their product offerings at the geographic address or a 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.

5. API Description

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

5.1. 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)
- 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`.

5.2. Use cases

Figure 4 presents a high-level use case diagram as specified in [MEF 80] in section 7.2. This picture aims to help understand the endpoint mapping. Use cases are described extensively in chapter 6

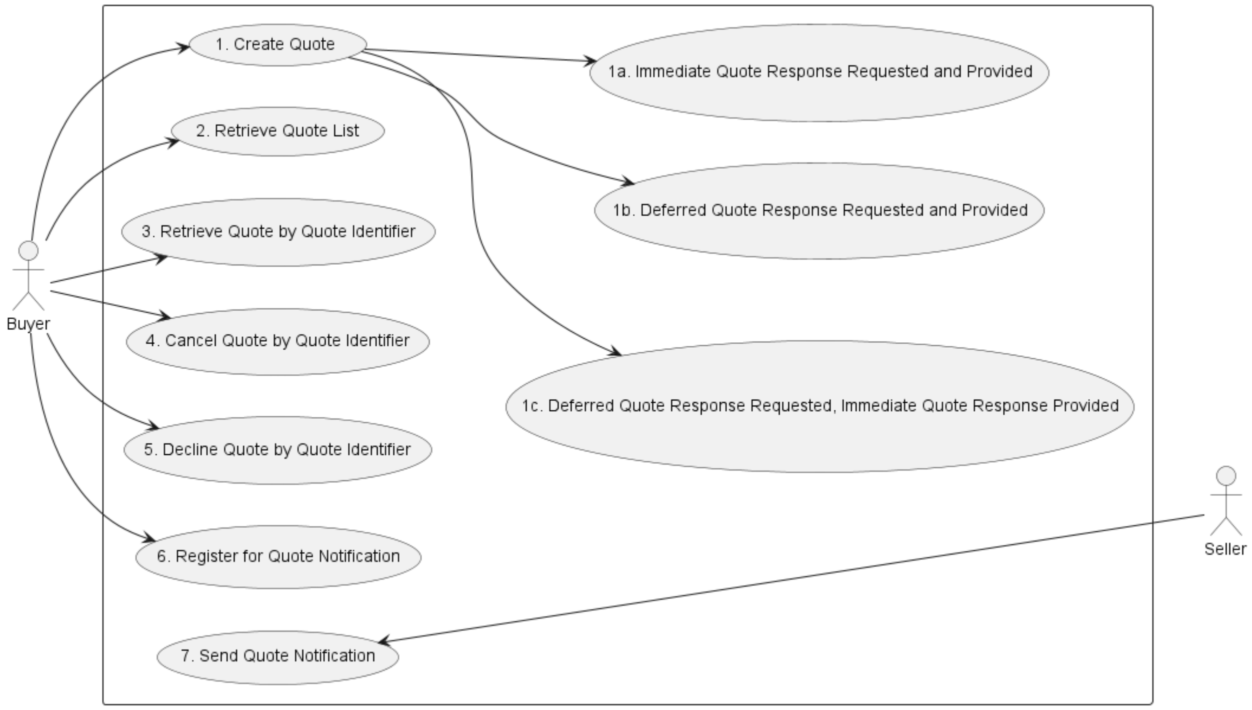

1. Create Quote
2. Retrieve Quote List
3. Retrieve Quote by Quote Identifier
4. Cancel Quote by Quote Identifier
5. Decline Quote by Quote Identifier
6. Register for Quote Notification
7. Send Quote Notification
1a. Immediate Quote Response Requested and Provided
1b. Deferred Quote Response Requested and Provided
1c. Deferred Quote Response Requested, Immediate Quote Response Provided
Buyer
Seller

Figure 4. Use cases

5.3. API Endpoint and Operation Description

5.3.1. Seller-side API Endpoints

Base URL for Cantata:

```
https://{{serverBase}}:{{port}}
{{?/seller_prefix}}/mefApi/cantata/quoteManagement/v4/
```

Base URL for Sonata:

```
https://{{serverBase}}:{{port}}
{{?/seller_prefix}}/mefApi/sonata/quoteManagement/v10/
```

The following API endpoints are implemented by the Seller and allow the Buyer to send Quote requests, retrieve existing Quotes or Quote details, and manage notification registrations. The endpoints and corresponding data model are defined in `productApi/quote/quoteManagement.api.yaml`.

API endpoint	Description	MEF 80 Use Case mapping
`POST /quote`	A request initiated by the Buyer to *create* new `Quote` and start the quotation process on the Seller's side.	UC 1: Create Quote (incl. 1a, 1b, 1c)
`GET /quote`	A request initiated by the Buyer to retrieve a list of `Quotes` from the Seller based on filter criteria provided as `query` parameters.	UC 2: Retrieve Quote List
`GET /quote/{{id}}`	A request initiated by the Buyer to retrieve full details of a specific `Quote` based on `id`_ provided as _`path`_ parameter	UC 3: Retrieve Quote by Quote Identifier
`POST /cancelQuote`	A request initiated by the Buyer to *cancel* existing `Quote`. The `Quote.id` is provided in the message body.	UC 4: Cancel Quote by Quote Identifier
`POST /declineQuote`	A request initiated by the Buyer to *decline* existing `Quote`. The `Quote.id` is provided in the message body.	UC 5: Decline Quote by Quote Identifier
`POST /hub`	A request initiated by the Buyer to instruct the Seller to send notifications of `Quote` and `QuoteItem` state change events.	UC 6: Register for Quote Notifications
`GET /hub/{{id}}`	A request initiated by the Buyer to retrieve the details of specific notification registration.	UC 6: Register for Quote Notification
`DELETE /hub/{{id}}`	A request initiated by the Buyer to instruct the Seller to stop sending notifications.	UC 6: Register for Quote Notification

Table 3. Seller side API endpoints

[R2] The Seller **MUST** support either Use Case 1a or 1b in a sense that both types of requests (immediate or deferred) **MUST** be supported yet only one of the response types (immediate or deferred) MAY be supported. [MEF80 R11]

[R3] The Seller **MUST** support Use Cases 3, 4, and 5.

[O1] The Seller **MAY** support Use Cases 1a, 1b, 1c, 2, and 6.

5.3.2. Buyer-side API Endpoints

Base URL for Cantata:

```
https://{{serverBase}}:{{port}}
{{?/seller_prefix}}/mefApi/cantata/quoteNotification/v4/
```

Base URL for Sonata:

```
https://{{serverBase}}:{{port}}
{{?/seller_prefix}}/mefApi/sonata/quoteNotification/v10/
```

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

API Endpoint	Description	MEF 80 Use Case Mapping
`POST /listener/quoteStateChangeEvent`	A request initiated by the Seller to notify the Buyer of the `Quote` state change.	UC 7: Send Quote Notification
`POST /listener/quoteItemStateChangeEvent`	A request initiated by the Seller to notify the Buyer of the `QuoteItem` state change.	UC 7: Send Quote Notification

Table 4. Buyer-side API endpoints

[O2] The Buyer **MAY** support Use Case 7.

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.

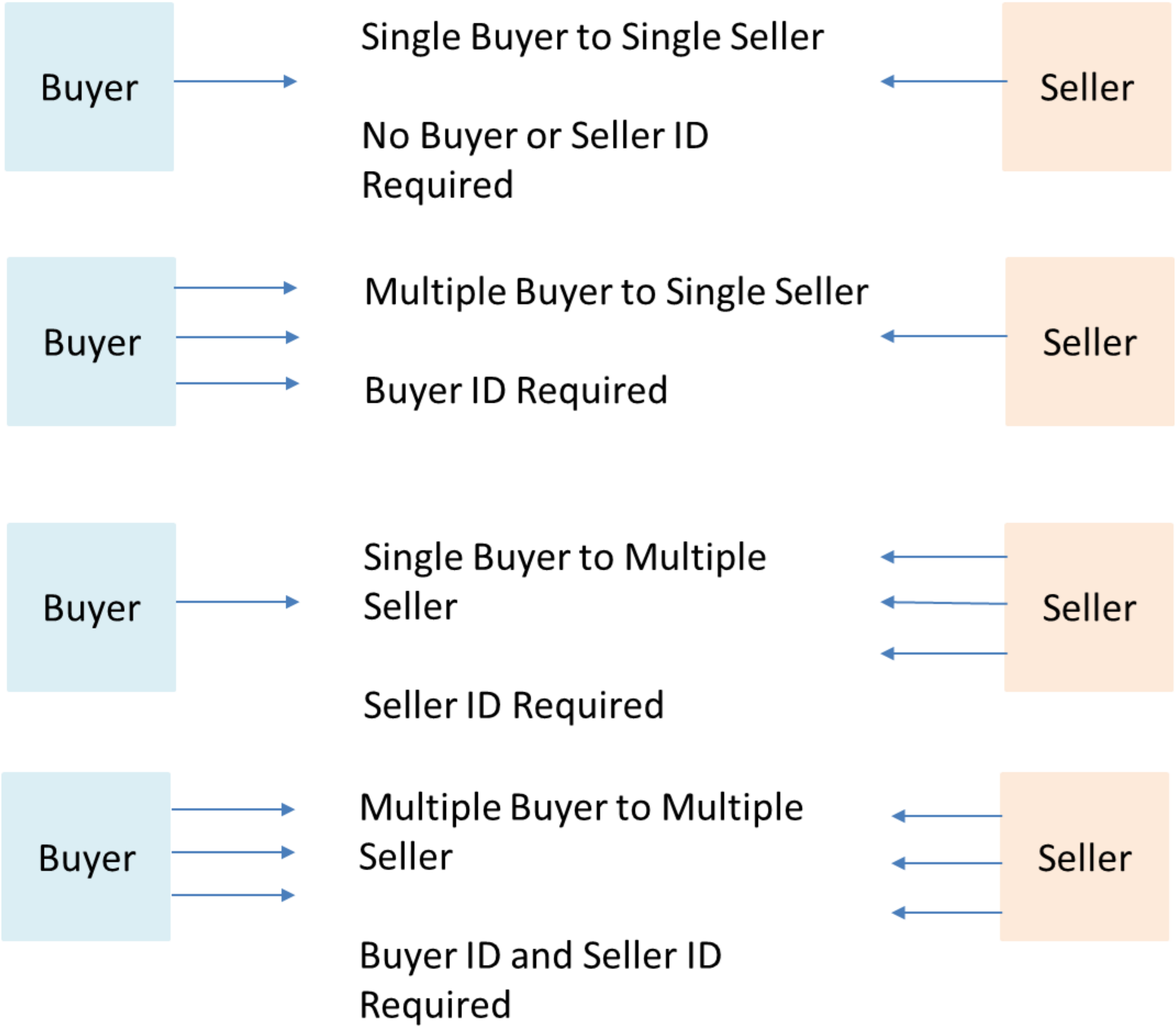

Buyer
Single Buyer to Single Seller
No Buyer or Seller ID Required
Seller
Buyer
Multiple Buyer to Single Seller
Buyer ID Required
Seller
Buyer
Single Buyer to Multiple Seller
Seller ID Required
Seller
Buyer
Multiple Buyer to Multiple Seller
Buyer ID and Seller ID Required
Seller

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.

[R4] 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]

[R5] 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. Integration of Product Specific Attributes

Product specifications are defined using JsonSchema format and are integrated into a Quote 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 must be introduced into the `productConfiguration` attribute of the `MEFProductRefOrValueQuote`.

Implementations might choose to integrate selected product specifications into 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 Quote 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 Quote 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.

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

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

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

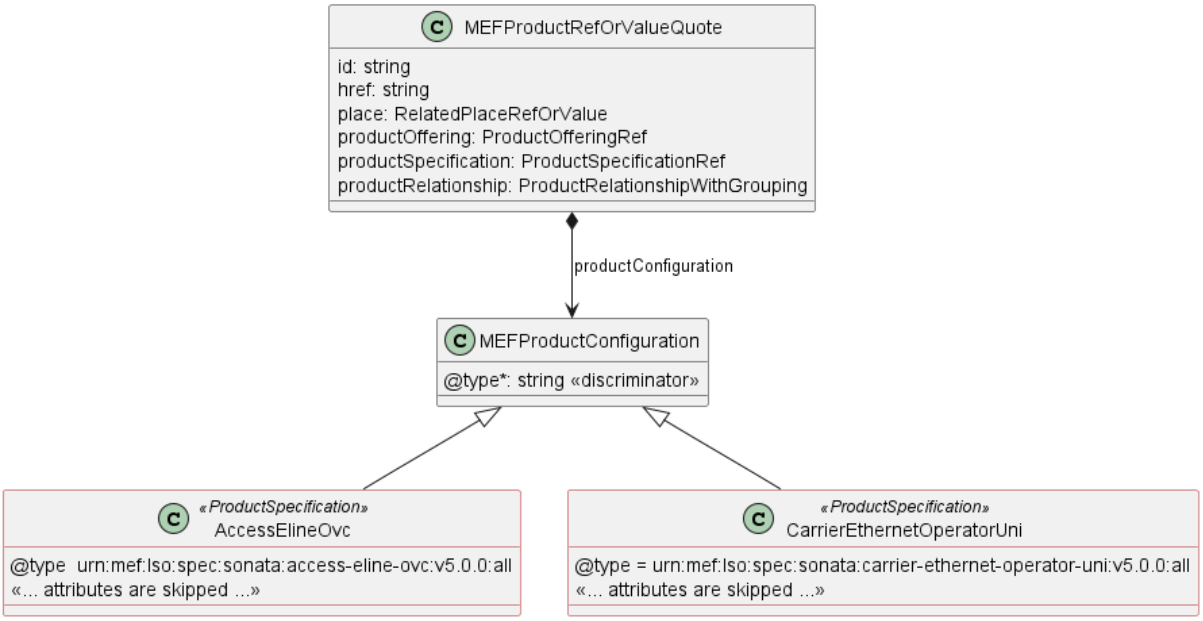

C MEFProductRefOrValueQuote
id: string
href: string
place: RelatedPlaceRefOrValue
productOffering: ProductOfferingRef
productSpecification: ProductSpecificationRef
productRelationship: ProductRelationshipWithGrouping
productConfiguration
C MEFProductConfiguration
@type*: string «discriminator»
C «ProductSpecification»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

Figure 6. The Extension Pattern

Figure 6 depicts two Mplify <<ProductSpecifications>> that represent Access E-Line and Operator UNI products. When these products are used in the Quote 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.

5.6. 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 7 depicts a simplified view of the defined relationships with other products and places.

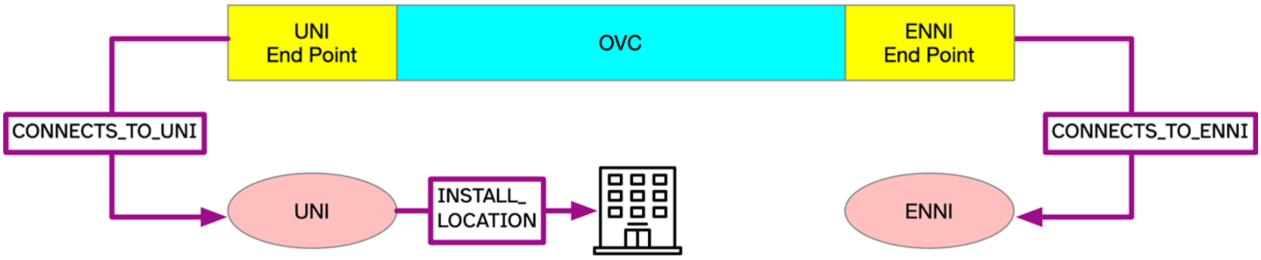


Figure 7. A Simplified View of Product and Place Relationships

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 7.

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.

5.7. Model Structural Validation

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

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

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

[R10] The Buyer and the Seller **MUST NOT** use any operation, entity or attribute that is not explicitly defined or allowed by this standard.

[R11] A product specification may define additional consistency rules and requirements that **MUST** be respected by implementations. [MEF80 R23]

These are defined for:

- required relation type, multiplicity to other items in the same quote request
- required relation type, multiplicity to entities in the Seller's product inventory
- related contact information roles that are to be defined at the item level
- relations to places (locations) and their roles that are to be defined at the item level

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 Interactions and Flows

This section provides a detailed insight into the API functionality, use cases, and flows. It starts with Table 5 presenting a list and short description of all business use cases then presents the variants of ent-to-end interaction flows, and in the following subchapters describes the API usage flow and examples for each of the use cases.

Use Case #	Use Case Name	Use Case Description
1	Create Quote	The Buyer requests a Quote from the Seller using one of the sub-use Cases below.
1a	Immediate Quote Response Requested and Provided	The Buyer requests a Quote from the Seller and requests an Immediate Quote Response.
1b	Deferred Quote Response Requested and Provided	The Buyer requests a Quote from the Seller and does not request an Immediate Quote Response. The Seller provides a Deferred Quote Response.
1c	Deferred Quote Response Requested, Immediate Quote Response Provided	The Buyer requests a Quote from the Seller and does not request an Immediate Quote Response. The Seller provides an Immediate Quote Response.
2	Retrieve Quote List	The Buyer requests a list of Quotes from the Seller based on Quote filter criteria.
3	Retrieve Quote by Quote Identifier	The Buyer requests detailed information related to a single Quote based on a Quote Identifier.
4	Cancel Quote by Quote Identifier	The Buyer requests to Cancel a Quote.
5	Decline Quote by Quote Identifier	The Buyer declines the Quote.
6	Register for Quote Notifications	The Buyer initiates a request to instruct the Seller to send notifications of Quote and/or Quote Item state changes
7	Send Quote Notification	Seller sends Notifications to the Buyer

Table 5. Use cases description

The detailed business requirements of each of the use cases are described in sections 7.2 and 8 of MEF 80 [MEF 80] and in [Mplify 80.0.1].

6.1. API Resource Schema Summary

This subchapter describes the most important entities from the resource model which can be found in the API specification. Each entity is a simple or composed type (with the use of `allOf` keyword for data type composition). A simple type defines a set of properties that might be of an object, primitive, or reference type.

[R12] If an entity is used in the request or response payload, all properties marked as required **MUST** be provided.

Section 6 provides examples of data model and API usage. For a detailed description and complete definition of the data model, please refer to API Details chapter.

6.1.1. Key Entities - Create request

Figure 8 presents the most important parts of the data model used during the Quote request (`POST /quote`) that is sent by a Buyer (see Section 5.3.1 for details). The model of the request message is a subset of the `Quote` model and contains only attributes that can (or must) be set by the Buyer. The Seller then enriches the entity in the response with additional information.

[R13] `Quote_Create` is the root entity of a quote request. It **MUST** contain one or more `QuoteItem_Create` [MEF80 R12].

Note: `Quote_Create` and `QuoteItem_Create` are entities used by the Buyer to make a request. `Quote` and `QuoteItem` are entities used by the Seller to provide a response. The request entities have a subset of attributes of the response entities. Thus for visibility of these shared attributes `Quote_Common` and `QuoteItem_Common` have been introduced. Though, these are not to be used directly in the exchange.

A `QuoteItem_Create` defines details of the product being subject of the quotation (in `MEFProductRefOrValueQuote` structure) and allows for the definition of additional information like related parties (`RelatedContactInformation`) or relations to other items (`QuoteItemRelationship`).

`MEFProductRefOrValueQuote` allows for the introduction of Mplify product-specific properties to the Quote payload. The extension mechanism is described in detail in Section 5.5.. `MEFProductRefOrValueQuote` may be also used to specify relations to places (using specializations of `RelatedPlaceRefOrQueryWithSubUnit`) and/or to a product that exists in the Seller's inventory (using `ProductRelationship`).

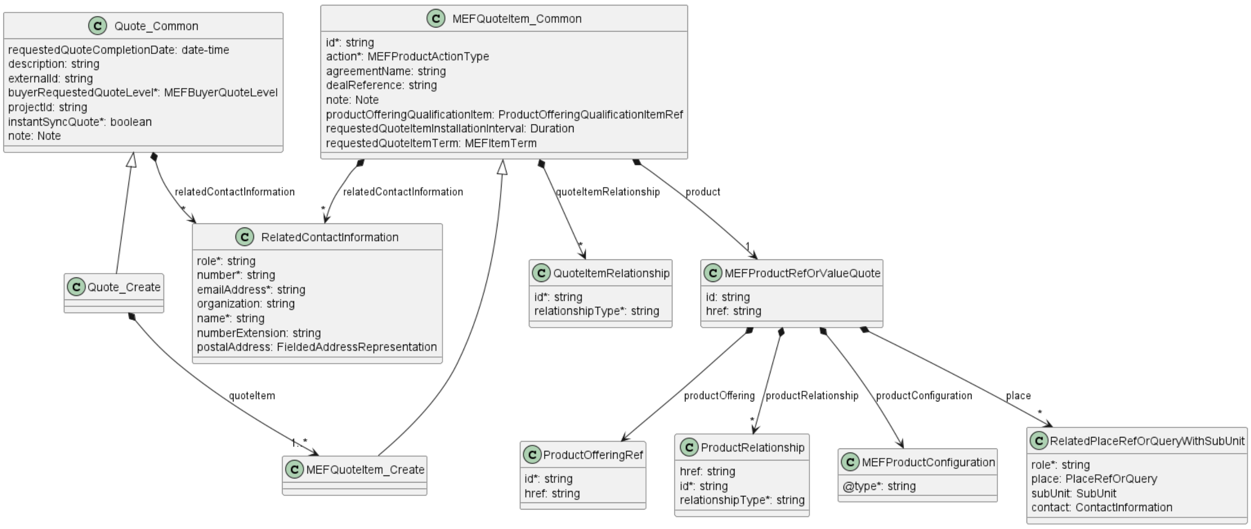


Figure 8. Key Entities - Create Request

6.1.2. Key Entities - Response

Figure 9. shows the most important data model parts used to provide a response to a Buyer's Create Quote (`POST /quote`) or to retrieve a `Quote` by identifier (`GET /quote/{{id}}`) request. Please note that the model differs only with the number of attributes for `Quote` and `QuoteItem` entities.

[R14] Any attribute set by the Buyer in the request **MUST NOT** be modified by the Seller in the response.

`Quote` is the root entity of a response and it contains one or more `QuoteItems`. For `Quote` and each of the `QuoteItems`, the Seller provides the state and, if applicable, the final quotation (`QuotePrice`) to a particular item from Buyer's request.

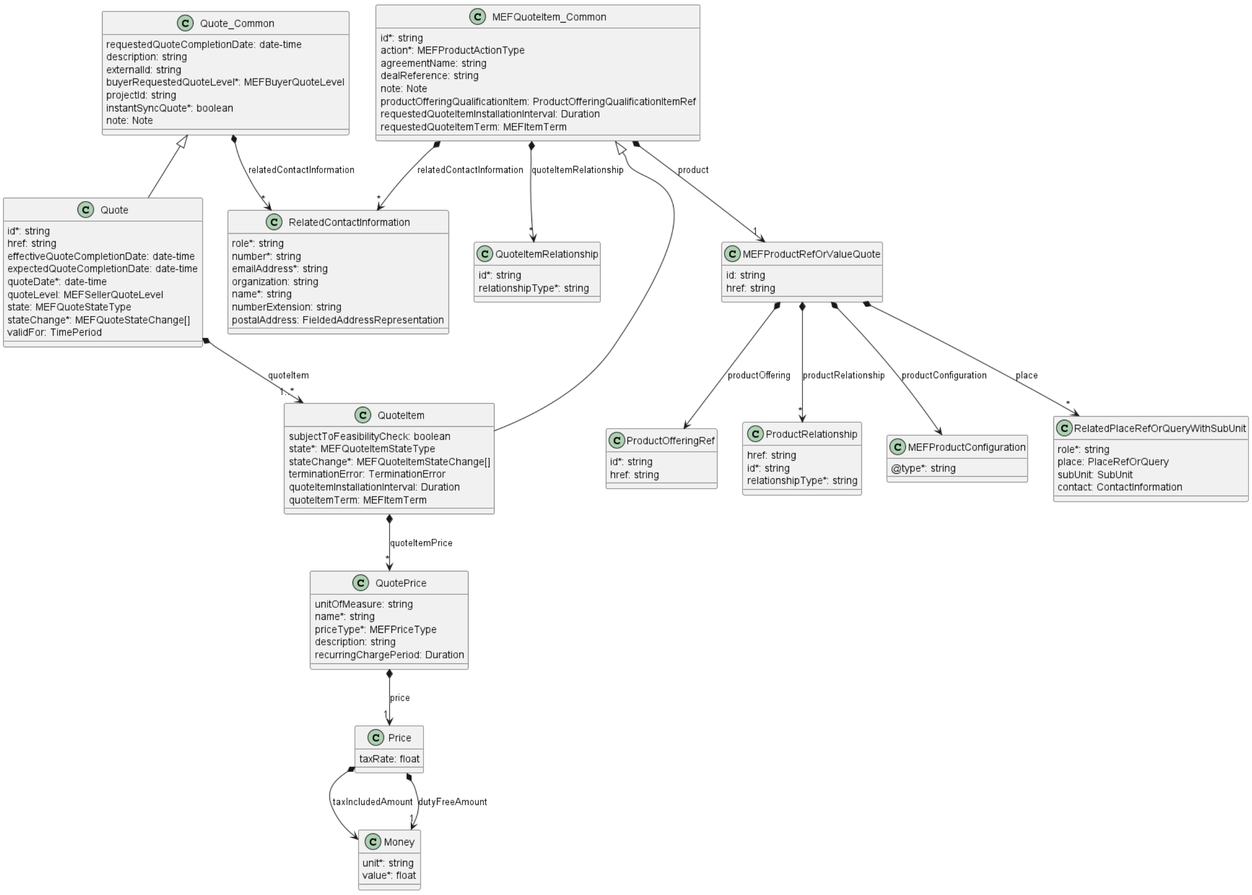

Quote_Common
requestedQuoteCompletionDate: date-time
description: string
externalId: string
buyerRequestedQuoteLevel*: MEFBuyerQuoteLevel
projectId: string
instantSyncQuote*: boolean
note: Note
MEFQuoteItem_Common
id*: string
action*: MEFProductActionType
agreementName: string
dealReference: string
note: Note
productOfferingQualificationItem: ProductOfferingQualificationItemRef
requestedQuoteItemInstallationInterval: Duration
requestedQuoteItemTerm: MEFItemTerm
relatedContactInformation
relatedContactInformation
quoteItemRelationship
product
Quote
id*: string
href: string
effectiveQuoteCompletionDate: date-time
expectedQuoteCompletionDate: date-time
quoteDate*: date-time
quoteLevel: MEFSellerQuoteLevel
state: MEFQuoteStateType
stateChange*: MEFQuoteStateChange[]
validFor: TimePeriod
RelatedContactInformation
role*: string
number*: string
emailAddress*: string
organization: string
name*: string
numberExtension: string
postalAddress: FieldedAddressRepresentation
QuoteItemRelationship
id*: string
relationshipType*: string
MEFProductRefOrValueQuote
id: string
href: string
quoteItem
productOffering
productRelationship
productConfiguration
place
QuoteItem
subjectToFeasibilityCheck: boolean
state*: MEFQuoteItemStateType
stateChange*: MEFQuoteItemStateChange[]
terminationError: TerminationError
quoteItemInstallationInterval: Duration
quoteItemTerm: MEFItemTerm
ProductOfferingRef
id*: string
href: string
ProductRelationship
href: string
id*: string
relationshipType*: string
MEFProductConfiguration
@type*: string
RelatedPlaceRefOrQueryWithSubUnit
role*: string
place: PlaceRefOrQuery
subUnit: SubUnit
contact: ContactInformation
quoteItemPrice
QuotePrice
unitOfMeasure: string
name*: string
priceType*: MEFPriceType
description: string
recurringChargePeriod: Duration
price
Price
taxRate: float
taxIncludedAmount
dutyFreeAmount
Money
unit*: string
value*: float

Figure 9. Key Entities - Response

6.1.3. Quote Process Flow

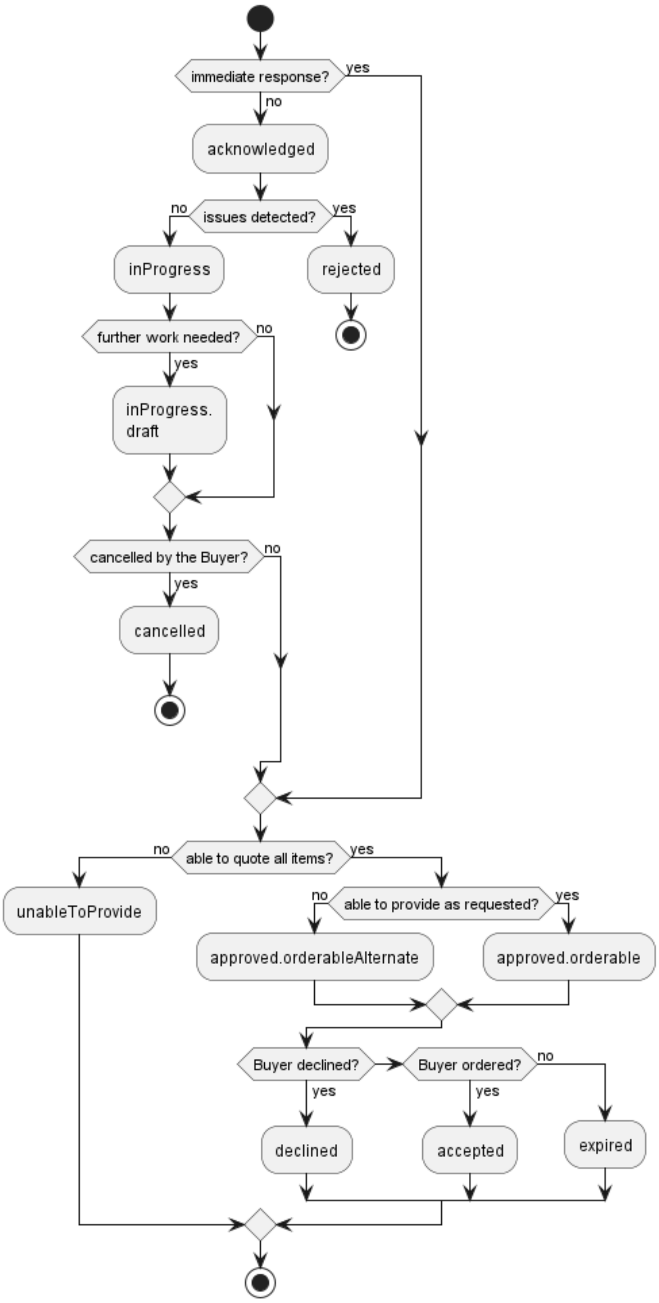


Figure 10. Quote firm flow activity diagram

The Buyer makes the decision about the kind of response that will be provided. The decision is rather implementation than request-dependent. After successful basic syntax checks, the response is sent back with all objects in the `acknowledged` state. If some issues were found an Error message must be returned. After moving the Quote to the `acknowledged` state the Seller has no technical possibility to send back an error message. In case any problems are found in the next business validation, the Quote must be moved to `rejected` state indicating the reason for rejection.

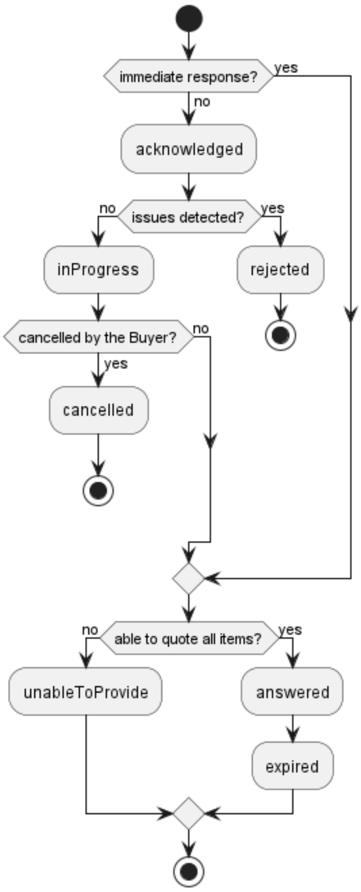


Figure 11. Quote budgetary flow activity diagram

The following tables present the possible combinations of Quote and QuoteItem states, as described in MEF 80 [MEF 80]:

Quote State\Quote Items State	abandoned	acknowledged	inProgress	inProgress.draft	approved.orderableAlternate	approved.orderable	rejected	unableToProvide
accepted					X	X		
acknowledged		X						
cancelled	X				X	X	X	X
declined					X	X		
expired					X	X		
inProgress		X	X	X	X	X		
inProgress.draft		X		X	X	X		
approved.orderableAlternate					X	X		
approved.orderable						X		
rejected	X						X	
unableToProvide	X				X	X		X

Table 6. Allowable Quote Item States per Quote State for FIRM Quote Level

Quote State\Quote Items State	abandoned	acknowledged	answered	inProgress	rejected	unableToProvide
acknowledged		X				
answered			X			
cancelled	X		X			
expired			X			
inProgress		X	X	X		
rejected	X				X	
unableToProvide	X		X			X

Table 7. Allowable Quote Item States per Quote State for BUDGETARY Quote Level

6.1.4. Quote Item Process Flow

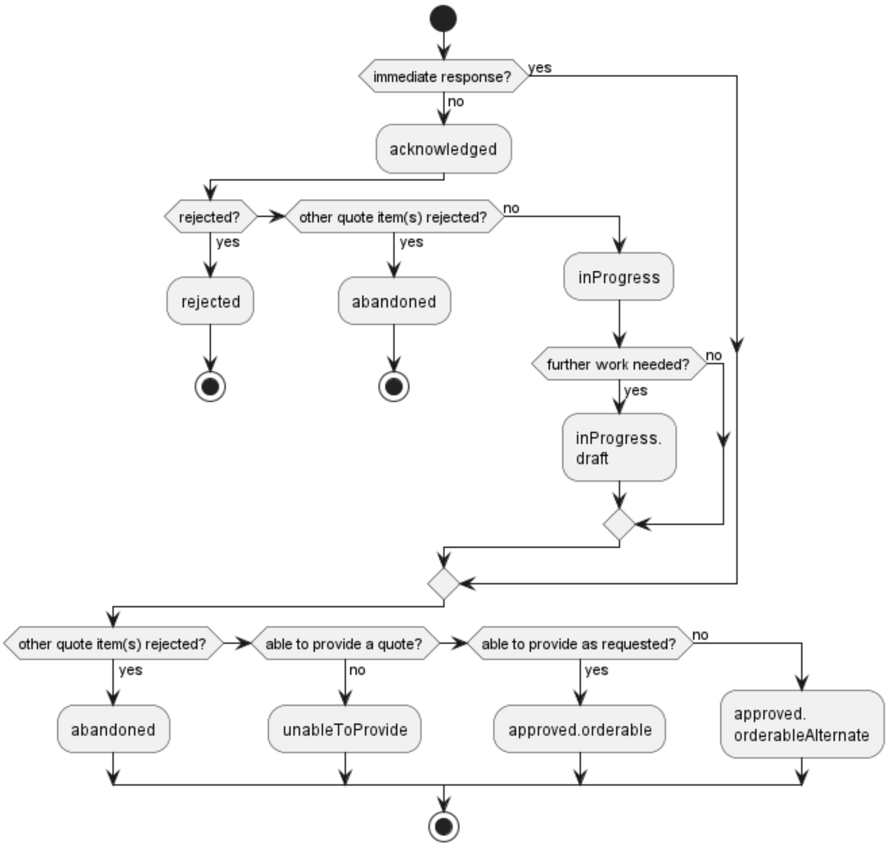


Figure 12. Quote Item firm quote flow activity

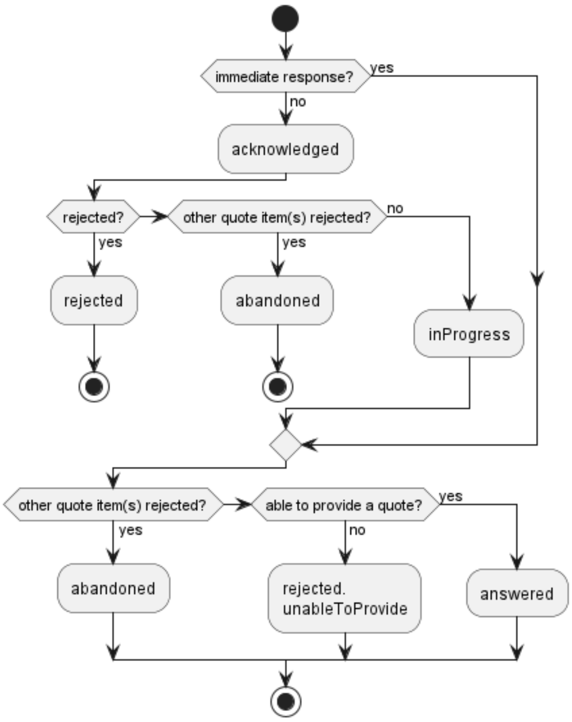


Figure 13. Quote Item budgetary quote flow activity

6.1.5. 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 `Quote Item's` attribute: `product.place` of type `RelatedPlaceRefOrQueryWithSubUnit`, which is presented in Figure 14.

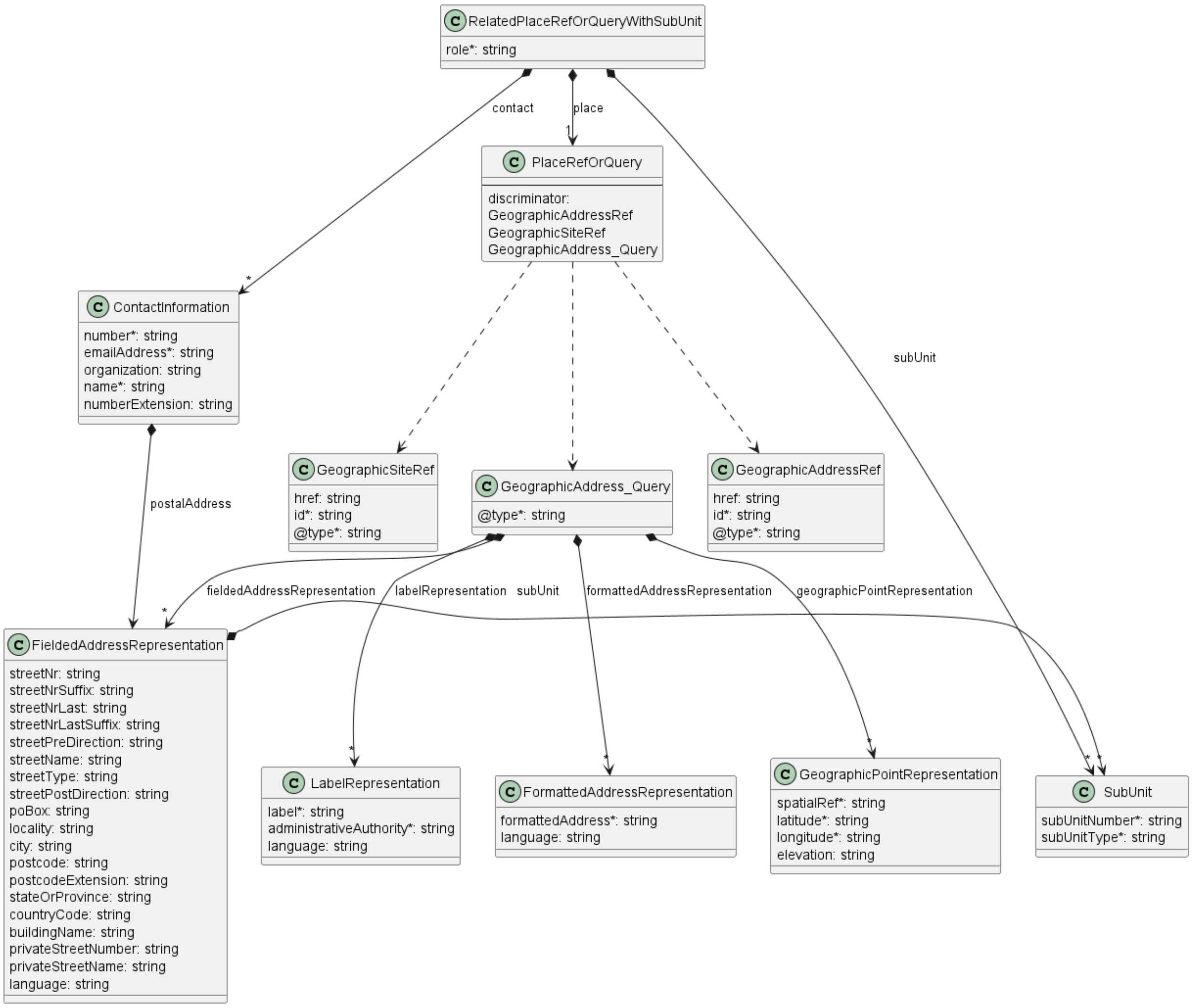


Figure 14. Data model - referring to a place

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 Quote 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 Quote inquiries 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.

[D1] For all Addresses that have not been validated, a Buyer **SHOULD** initiate a Validate Address request, to get the Seller's full details associated with this Address prior to submitting a Create Quote request. [MEF80 D1]

[D2] For all Addresses that have been validated, the Buyer **SHOULD** use the Seller's Address Identifier to describe the location when submitting a Create Quote request and when a Service Site does not exist. [MEF80 D2]

[D3] After validating an Address, a Buyer **SHOULD** initiate a Retrieve Service Site List request and obtain the Seller's Service Site Identifiers for all Service Sites at the validated Address prior to submitting a Create Quote request. [MEF80 D3]

[D4] Once the Buyer has obtained the Service Site Identifier of the Service Site, then for all subsequent operations related to this Service Site, the Buyer **SHOULD** use the Seller's Service Site Identifier to reference this Service Site when submitting a Create Quote request. [MEF80 D4]

6.2. Use case 1: Create Quote

There are two possible types of interaction: immediate and deferred.

1. The Seller responds immediately with the final results of the processing. This is called an Immediate Quote Response.
2. The Seller acknowledges that the request has been received, but will not complete processing it immediately, and send notifications to update the Buyer on the status (assuming the Buyer has subscribed to receive the notifications). This is called a Deferred Quote Response.

Their details are described in the following subchapters.

[R15] When providing responses to the API calls the Buyer and the Seller **MUST** provide relevant HTTP Response codes.

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. Use case 1a: Immediate Quote Response Requested and Provided

An immediate quote response can be requested by a Buyer using a mandatory `instantSyncQuote` flag set to `true`. If the Buyer's Create Quote request is not valid, the appropriate error code and description are returned in case the request doesn't pass the initial validation. In case of successful processing, the Seller responds with a `Quote` in one of the Completion States: `approved.orderable`, `answered` or `approved.orderableAlternate` to indicate success or `unableToProvide` to indicate that the Buyer did not provide enough information or the Seller is not able to provide the answer for any other reason.

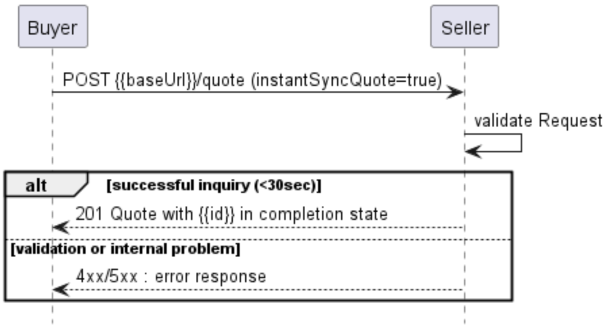

Buyer
Seller
POST {{baseUrl}}/quote (instantSyncQuote=true)
validate Request
alt
[successful inquiry (<30sec)]
201 Quote with {{id}} in completion state
[validation or internal problem]
4xx/5xx : error response

Figure 15. Use case 1a: Immediate Quote Response Requested and Provided

Figure 15 presents the basic synchronous use case flow while the one below adds the context when the Buyer decides to use additional asynchronous the Notification mechanism (and the Seller supports it). In this case, the Buyer must register to receive notifications before sending the Create Quote request. In case of Immediate Response, the received Quote will already be in the Completion State. The notification can be sent afterward if from a successful Completion State (`approved.orderable` or `approved.orderableAlternate`), the Quote will move to one of the Terminal States (`accepted`, `declined`, `expired`).

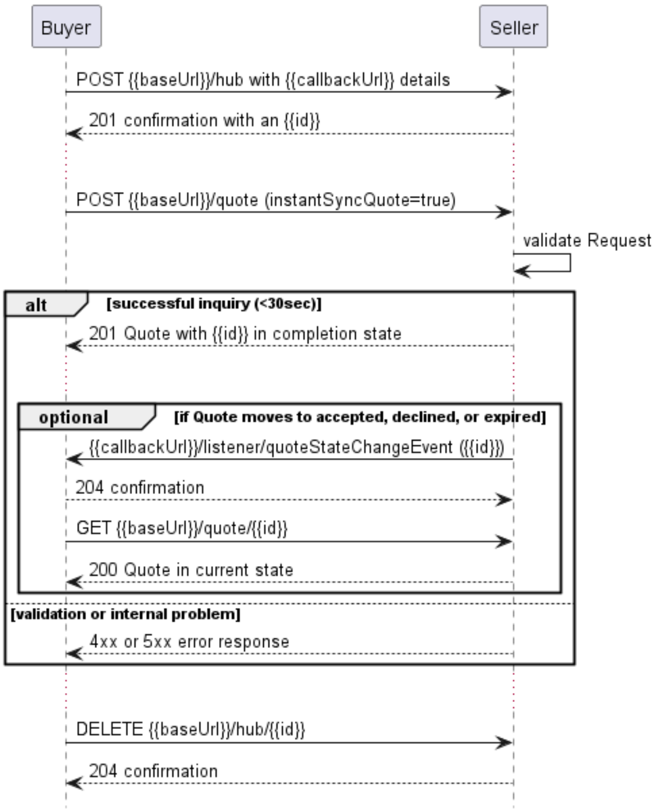

Buyer
Seller
POST {{baseUrl}}/hub with {{callbackUrl}} details
201 confirmation with an {{id}}
POST {{baseUrl}}/quote (instantSyncQuote=true)
validate Request
alt
[successful inquiry (<30sec)]
201 Quote with {{id}} in completion state
optional
[if Quote moves to accepted, declined, or expired]
{{callbackUrl}}/listener/quoteStateChangeEvent ({{id}})
204 confirmation
GET {{baseUrl}}/quote/{{id}}
200 Quote in current state
[validation or internal problem]
4xx or 5xx error response
DELETE {{baseUrl}}/hub/{{id}}
204 confirmation

Figure 16. Use case 1a: Immediate Quote Response Requested and Provided with Notification

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

6.2.2. Use case 1b: Deferred Quote Response Requested and Provided

A deferred quotation can be requested by using the `instantSyncQuote` flag set to `false`. The Seller responds with `Quote` (state `acknowledged` and `Quote.id` specified) and starts processing the request asynchronously. When the Buyer has registered for quote notifications, the Seller will send a quote state change notifications to the Buyer.

The Buyer may choose between two possible patterns to get details on the progress of his quote: polling and notification.

In the first case (Figure 17), the Buyer needs to poll periodically for the Quote to check its state until the Completion State is reached and optionally to check whether the state changed to one of the Terminal States

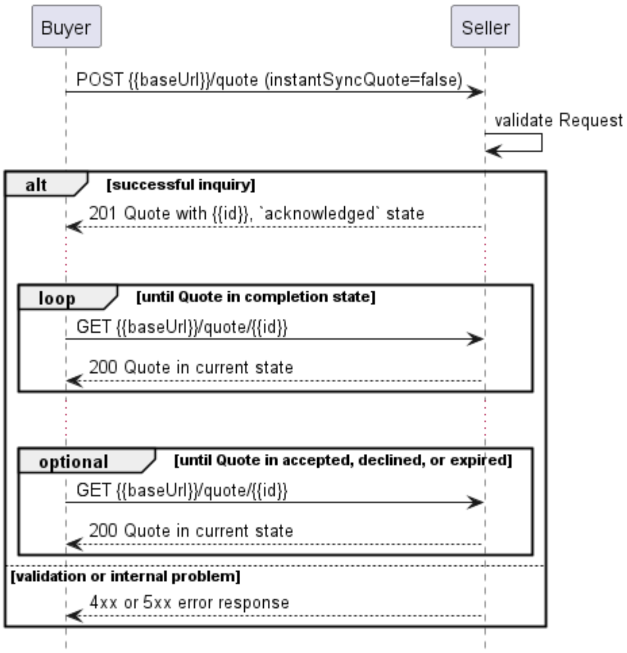

Buyer
Seller
POST {{baseUrl}}/quote (instantSyncQuote=false)
validate Request
alt
[successful inquiry]
201 Quote with {{id}}, `acknowledged` state
loop
[until Quote in completion state]
GET {{baseUrl}}/quote/{{id}}
200 Quote in current state
optional
[until Quote in accepted, declined, or expired]
GET {{baseUrl}}/quote/{{id}}
200 Quote in current state
[validation or internal problem]
4xx or 5xx error response

Figure 17. Use case 1b: Deferred Quote Response Requested and Provided - Polling pattern

To use the notifications mechanism (Figure 18), the Buyer needs to register for by providing a callback endpoint before sending the Quote Create request. The Seller sends notifications of Quote changes until the Terminal State is reached.

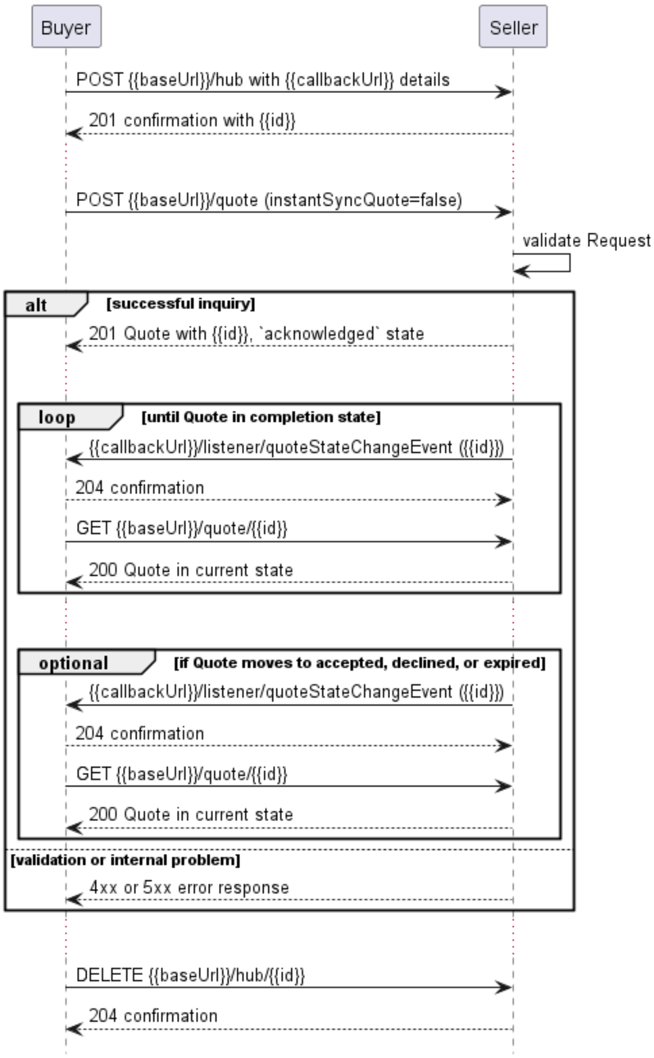

Buyer
Seller
POST {{baseUrl}}/hub with {{callbackUrl}} details
201 confirmation with {{id}}
POST {{baseUrl}}/quote (instantSyncQuote=false)
validate Request
alt
[successful inquiry]
201 Quote with {{id}}, `acknowledged` state
loop
[until Quote in completion state]
{{callbackUrl}}/listener/quoteStateChangeEvent ({{id}})
204 confirmation
GET {{baseUrl}}/quote/{{id}}
200 Quote in current state
optional
[if Quote moves to accepted, declined, or expired]
{{callbackUrl}}/listener/quoteStateChangeEvent ({{id}})
204 confirmation
GET {{baseUrl}}/quote/{{id}}
200 Quote in current state
[validation or internal problem]
4xx or 5xx error response
DELETE {{baseUrl}}/hub/{{id}}
204 confirmation

Figure 18. Use case 1b: Deferred Quote Response Requested and Provided - Notification pattern

6.2.3. Use case 1c: Deferred Quote Response Requested, Immediate Quote Response Provided

In this scenario, the Buyer does not request an Immediate Quote Response (`instantSyncQuote` equals `false`), but the Seller is able to provide one and does so by providing a synchronous response with a `Quote` in one of the Completion States.

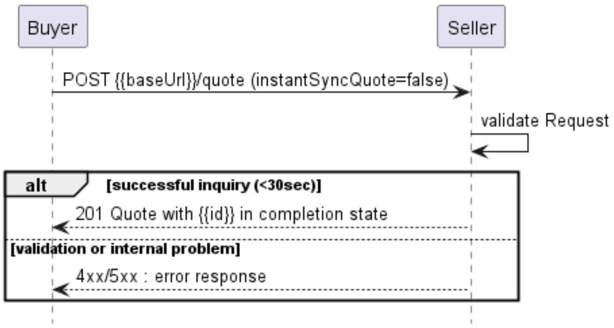

Buyer
Seller
POST {{baseUrl}}/quote (instantSyncQuote=false)
validate Request
alt
[successful inquiry (<30sec)]
201 Quote with {{id}} in completion state
[validation or internal problem]
4xx/5xx : error response

Figure 19. Use case 1c: Deferred Quote Response Requested, Immediate Quote Response Provided

Figure 19 presents the case where the Buyer didn't register for Quote Notifications.

Figure 20 presents the interaction between the Buyer and the Seller when the Buyer registered for Quote Notifications. Please note that in this case (just like in use case 1a) the Seller provides an immediate response. The Quote state change event will only be sent after some time when the Quote reaches a terminal state (e.g. `expired`).

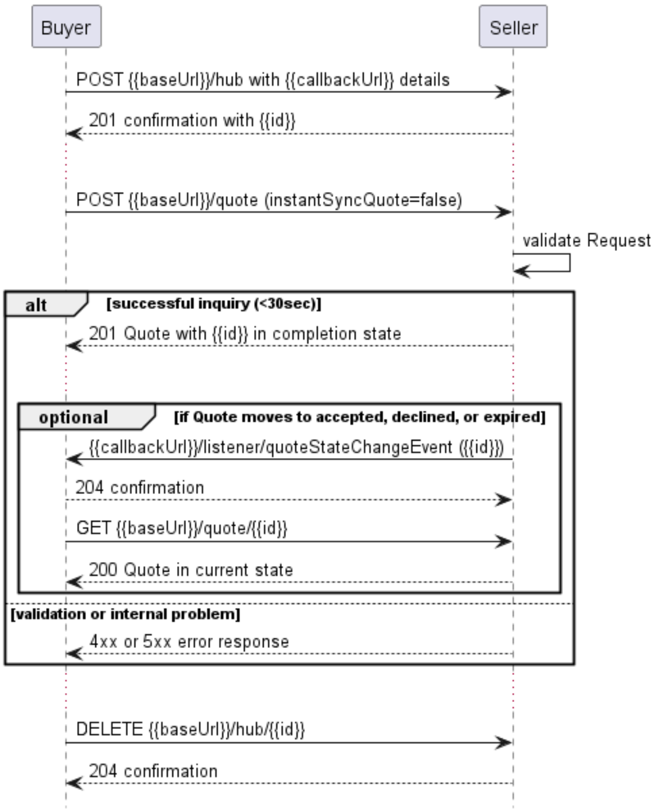

Buyer
Seller
POST {{baseUrl}}/hub with {{callbackUrl}} details
201 confirmation with {{id}}
POST {{baseUrl}}/quote (instantSyncQuote=false)
validate Request
alt
[successful inquiry (<30sec)]
201 Quote with {{id}} in completion state
optional
[if Quote moves to accepted, declined, or expired]
{{callbackUrl}}/listener/quoteStateChangeEvent ({{id}})
204 confirmation
GET {{baseUrl}}/quote/{{id}}
200 Quote in current state
[validation or internal problem]
4xx or 5xx error response
DELETE {{baseUrl}}/hub/{{id}}
204 confirmation

Figure 20. Use case 1c: Deferred Quote Response Requested, Immediate Quote Response Provided, with Notifications

6.2.4. Buyer's Quote request

To send a Quote request the Buyer uses the `createQuote` operation from the API: `POST /quote`. The Create Quote request model is common for Use Cases 1a, 1b, and 1c. For clarity, some of the Quote payload's attributes might be omitted to improve examples' readability.
The full list of attributes is available in Section 7 and in the API specification which is an integral part of this standard.

`Quote` Create

```
{
  "instantSyncQuote": false,
  "buyerRequestedQuoteLevel": "firm",
  "requestedQuoteCompletionDate": "2020-08-10T16:45:39.368Z",
  "description": "Buyer defined description ",
  "externalId": "buyerQuote-001", << Buyer understandable External Id >>
  "projectId": "buyerProject-001", << Buyer understandable Project Id >>
  "quoteItem": [
    {
      "id": "item-001",
      "action": "add",
      "product": { << product specific attributes and configuration, see 6.3.3 >>
      },
      "productOfferingQualificationItem": {
        "id": "poqItem-001",
        "productOfferingQualificationId": "32112300-0000-0000-0000-000000000394812"
      },
      "requestedQuoteItemTerm": {
        "duration": {
          "amount": 12,
          "units": "calendarMonths"
        },
        "endOfTermAction": "autoRenew",
        "name": "Yearly Subscription"
      },
      "relatedContactInformation": [
        {
          "emailAddress": "eddie.technical@buyer.mef.com",
          "name": "Eddie Technical",
          "number": "12-345-6789",
          "numberExtension": "1234",
          "role": "quoteItemTechnicalContact"
        }
      ]
    }
  ],
  "relatedContactInformation": [
    {
      "emailAddress": "john.example@buyer.mef.com",
      "name": "John Example",
      "number": "12-345-6789",
      "numberExtension": "1234",
      "role": "buyerContactInformation"
    }
  ]
}
```

[R16] In the create Quote request, the Buyer **MUST** provide the following attributes: [MEF80 R13]

- `buyerRequestedQuoteLevel`
- `instantSyncQuote`
- `quoteItem` (minimum 1)

[R17] If `instantSyncQuote` equals `false` the Buyer **MUST** also provide `relatedContactInformation[]` with an item of `role` equal `buyerContactInformation`. [MEF80 R16]

[R18] If `instantSyncQuote` equals `false` the Buyer **MUST** also provide `requestedQuoteCompletionDate`. [MEF80 R17], [MEF80 R18]

[O3] The Seller **MAY** decide to make the `productOfferingQualificationItem` mandatory for a Buyer Create Quote request. [MEF80 O5]

Note: 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.

[R19] For every `QuoteItem` in the create Quote request, the Buyer **MUST** specify the following attributes: [MEF80 R14], [MEF80 R13]

- `id`
- `action`
- `product`

[R20] If `instantSyncQuote` equals `false` the Buyer **MUST** also provide `relatedContactInformation[]` with an item of `role` equal `quoteItemTechnicalContact`. [MEF80 R15]

[R21] The QuoteItem content **MUST** follow the product specification-related requirements when specifying values for relatedContactInformation, quoteItemRelationship, product.place, and product.productRelationship attributes.

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.

[R22] When specifying the product.place, the Buyer **MUST** provide following attributes`: [*Mplify 80.0.1 A1-R1]

- place
- role

[R23] When specifying the product.place, with GeographicSiteRef the Buyer **MUST NOT** additionally provide subUnit to describe exactly where the Buyer wants the Product to be installed. [Mplify 80.0.1 A1-R3]

[R24] If the Buyer includes the requestedQuoteItemInstallationInterval in the Create Quote request and they completed a POQ prior to submitting the Create Quote request, the requestedQuoteItemInstallationInterval **MUST** be longer or equal to the interval returned in the referenced POQ Item or Alternate Product Offering Proposal. [MEF80 R22], [MEF80 CR2]

Note: if the Buyer includes the Requested Quote Item Installation Interval in the Create Quote request and they have not completed a POQ prior to submitting the Create Quote request, the Requested Quote Item Installation Interval may be any value allowed by the Seller

6.2.5. Seller's Response to a Create Quote Request

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

```
{
  "id": "00000000-0000-0000-0000-000000000123",
  "href": "{{baseUrl}}/quote/00000000-0000-0000-0000-000000000123",
  "state": "approved.orderable",
  "effectiveQuoteCompletionDate": "2020-08-10T16:45:20.421Z",
  "expectedQuoteCompletionDate": "2020-08-10T16:45:39.421Z",
  "quoteDate": "2020-08-10T16:40:33.422Z",
  "quoteLevel": "firm",
  "instantSyncQuote": "false", << as provided by the Buyer >>
  "buyerRequestedQuoteLevel": "firm", << as provided by the Buyer >>
  "requestedQuoteCompletionDate": "2020-08-10T16:45:39.368Z", << as provided by the Buyer >>
  "externalId": "buyerQuote-001", << as provided by the Buyer >>
  "projectId": "buyerProject-001", << as provided by the Buyer >>
  "stateChange": [
    {
      "changeDate": "2020-08-10T16:45:39.422Z",
      "state": "approved.orderable"
    },
    {
      "changeDate": "2020-08-10T16:42:39.422Z",
      "state": "inProgress.draft"
    },
    {
      "changeDate": "2020-08-10T16:40:39.422Z",
      "state": "inProgress"
    },
    {
      "changeDate": "2020-08-10T16:40:33.422Z",
      "state": "acknowledged"
    }
  ],
  "quoteItem": [
    {
      "id": "item-001",
      "action": "add",
      "state": "approved.orderable",
      "stateChange": [
        {
          "changeDate": "2020-08-10T16:45:39.422Z",
          "state": "approved.orderable"
        },
        {
          "changeDate": "2020-08-10T16:42:39.422Z",
          "state": "inProgress.draft"
        },
        {
          "changeDate": "2020-08-10T16:40:39.422Z",
          "state": "inProgress"
        },
        {
          "changeDate": "2020-08-10T16:40:33.422Z",
          "state": "acknowledged"
        }
      ],
      "subjectToFeasibilityCheck": false,
      "product": {<< as provided by the Buyer >>},
      "productOfferingQualificationItem": {
        "id": "poqItem-001",
        "productOfferingQualificationId": "32112300-0000-0000-0000-000000000394812"
      },
      "requestedQuoteItemTerm": {<< as provided by the Buyer >>
        "duration": {
          "amount": 12,
          "units": "calendarMonths"
        },
        "endOfTermAction": "autoRenew",
        "name": "Yearly Subscription"
      },
      "relatedContactInformation": [<< as provided by the Buyer >>
        {
          "emailAddress": "eddie.technical@buyer.mef.com",
          "name": "Eddie Technical",
          "number": "12-345-6789",
          "numberExtension": "1234",
          "role": "quoteItemTechnicalContact"
        }
      ],
      "quoteItemTerm": {
        "duration": {
          "amount": 12,
          "units": "calendarMonths"
        },
        "endOfTermAction": "autoRenew",
        "name": "Yearly Subscription"
      },
      "quoteItemPrice": [
        {
          "name": "Monthly Plan 25",
          "priceType": "recurring",
          "recurringChargePeriod": {
            "amount": 1,
            "units": "calendarMonths"
          },
          "price": {
            "taxRate": 16,
            "dutyFreeAmount": {
              "unit": "EUR",
              "value": 25
            },
            "taxIncludedAmount": {
              "unit": "EUR",
              "value": 29
            }
          }
        }
      ]
    }
  ],
  "relatedContactInformation": [
    {
      "emailAddress": "john.example@buyer.mef.com",
      "name": "John Example",
      "number": "12-345-6789",
      "role": "buyerContactInformation"
    },
    {
      "emailAddress": "kate.example@seller.mef.com",
      "name": "Kate Example",
      "number": "12-345-67890",
      "role": "sellerContactInformation"
    }
  ],
  "validFor": {
    "endDateTime": "2020-08-17T16:45:39.422Z"
  }
}
```

[R25] The Seller **MUST NOT** change the values of attributes specified by the Buyer. [MEF80 R36]

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

[R26] In the response, the Seller **MUST** provide the following attributes: [MEF80 R35], [MEF80 R44], [MEF80 R39], [MEF80 R40], [MEF80 R46], [MEF80 R58].

- `id`
- `state`
- `stateChange`
- `quoteDate`
- `relatedContactInformation` with added `role=sellerContactInformation`
- `quoteItem`
  - `quoteItem.state`
  - `quoteItem.stateChange`

[R27] In the response, the Seller **MUST** provide the `quoteItem.quoteItemInstallationInterval` attribute, if the `quoteItem.state` is `answered`, `approved.orderable`, or `approved.orderableAlternate`. [MEF80 R54], [MEF80 R67]

[R28] In the response, the Seller **MUST** provide the `quoteItem.quoteItemTerm` attribute, if the `quoteItem.state` is `answered`, `approved.orderable`, or `approved.orderableAlternate` and `quoteItem.action` is not `delete`.

[R29] Each item in `quoteItem` list **MUST** correspond to one and only one Quote Item in the Buyer's Create Quote request. [MEF80 R36], [MEF80 R45]

[R30] If the `Quote` is in Completion State that returns a quote (not `rejected` or `unableToProvide`) and the Buyer requested a `budgetary` level, the Seller **MUST** respond with `quoteLevel` equal to `budgetary`. [MEF80 R42]

[R31] If the `Quote` is in a Completion State that returns a quote (not `rejected` or `unableToProvide`) and the Buyer request a `firm` level, the Seller **MUST** respond with `quoteLevel` equal to `firm` or `firmSubjectToFeasibilityCheck`. [MEF80 40]

[R32] If the `quoteLevel` is `firmSubjectToFeasibilityCheck`, the Seller **MUST** specify the `subjectToFeasibilityCheck` equal to `true` attribute in their response for at least one `quoteItem`. [MEF80 R46]

[R33] The `stateChange` **MUST** include a full object's state history including the initial state (also in the Immediate Response).

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

[R34] If the Seller's Quote specifies `endOfTermAction` equal to `roll` for a `QuoteItem.quoteItemTerm`, then the Seller **MUST** specify the `rollInterval` for that Quote Item. [MEF80 R61]

[R35] If the Seller's Quote specifies `endOfTermAction` equal to `autoRenew` or `autoDisconnect`, then the Seller **MUST NOT** specify the `rollInterval` for that Quote Item. [MEF80 R62]

[R36] The grace period after auto-renewal during which the Buyer can disconnect the Product without penalty **MUST** be agreed between Seller and Buyer as part of onboarding if the Seller chooses to use the value `autoRenew` for the `endOfTermAction` attribute. [MEF80 R63]

[R37] The `quoteItemTerm.duration` specified in the Seller's Quote **MUST** be the closest term duration that the Seller offers to the Buyer's `requestedQuoteItemTerm.duration`. [MEF80 R59]

[O4] The `quoteItemTerm.duration` specified by the Seller in their Quote **MAY** be greater than, equal to, or less than the Buyer's `requestedQuoteItemTerm.duration`.[MEF80 O13]

[R38] If the `requestedQuoteItemTerm.duration` specified in the Seller's Quote is less than the Buyer's `requestedQuoteItemTerm.duration` and the `Quote` moves to the Orderable states, the `quoteItem.state` **MUST** be `approved.orderableAlternate`.[MEF80 R60]

[R39] When specifying the `quoteItem.quoteItemPrice` the Seller **MUST** provide the following attributes: [MEF80 R55]

- `name`
- `priceType`
- `price.dutyFreeAmount`

[R40] When specifying the `quoteItem.quoteItemPrice` the Seller **MUST** follow the combination of attributes presented in Table 8. [MEF80 R56], [MEF80 R57]

`priceType`	`recurring ChargePeriod`	`unitOfMeasure`	`price. dutyFreeAmount`	**Comments**
`recurring`	X		X	
`nonRecurring`			X	
`usageBased`		X	X	`price.dutyFreeAmount` is the charge per `unitOfMeasure`

Table 8. Price Type Required Information

6.2.6. Quote Item Specification Details

This section provides examples of how the `quoteItem` should look like depending on the desired `action`.

6.2.6.1. Quote Item Structure for `add` Action

When requesting a new Product (`action` equal to `add`) the Buyer needs to provide all of its configurations. The example below shows a request for Access E-Line product (type `urn:mef:lso:spec:sonata:access-eline-ovc:v5.0.0:all`).

```
{
  <<Quote attributes...>>
  "quoteItem": [
    {
      "id": "item-001",
      "action": "add",
      ...
      "product": {
        "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"
              }
            }
          }
        },
        "productOffering": {
          "id": "000073"
        },
        "productRelationship": [
          {
            "relationshipType": "CONNECTS_TO_ENNI",
            "id": "SP1_ENNI"
          }
        ]
      },
      "quoteItemRelationship": [
        {
          "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"
            }
          ]
        }
      }
    }
  ]
}
```

[R41] `MEFProductConfiguration` **MUST** be provided in the payload in case an item `action` is set to `add` [MEF80 R21].

[R42] `productOffering` - **MUST** be provided in the payload in case an item `action` is set to `add` [MEF80 R21]

[R43] The Buyer **MUST NOT** specify the `product.id` in the request when `action` equal to `add`. It is the Seller who assigns this id [MEF80 R27].

An Access E-Line product specification defines two mandatory relationship types that have to be specified in case of quoting an `add` action: `CONNECTS_TO_ENNI` and `CONNECTS_TO_UNI`.
The reference to an operator UNI product might use another Quote item or an existing product from the Seller's inventory. In this example, the UNI product is another item of the request with a unique identifier `item-002`. This Access E-Line product references an existing ENNI product which is uniquely identified with id `SP1_ENNI` in the Seller's inventory.

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 only a tiny subset of available Access E-Line attributes. It aims to explain the Product definition and relation patterns, not to focus on the product configurations themselves.

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

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

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

Both product specifications and product offerings are not negotiated and exchanged within Cantata and Sonata. They are agreed between the Buyer and the Seller during the onboarding process. After that, they are only referenced as in the example above.

6.2.6.2. Quote Item Structure for `modify` Action

The following example shows a request for a quotation of an existing Access E-Line Product modification (`action` equal to `modify`). In particular, changes to `cir` (Committed Information Rate) and `cirMax` (Maximum Committed Information Rate) values for `uniEp` bandwidth profiles are introduced.
The Access E-Line product exists in Seller's inventory and is identified as `AccessElineOVC-0001`.

[R44] If `action=modify` the Buyer **MUST** provide following attributes of `quoteItem.product`: [MEF80 R28]

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

[R45] The modify request **MUST** provide a full state of the `product` attributes, including values of (specified or empty) of `product.productRelationship` and `product.place`. [MEF80 R29]

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

[O5] The Seller **MAY** allow the Buyer to specify a different `product.productOffering` than the one of the existing Product.

[R46] If `product.productOffering` changes, it **MUST** be based on the same Product Specification` as the existing Product.

There is no possibility to send an update to single attributes. The Buyer must send a full product description (the whole `product.productConfiguration` section and if set previously or to be set: `product.productRelationship` and `product.place`), 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 an appropriate error response (422) must be returned to the Buyer.

Please also note, that in the add case, a reference to the UNI product used the quoteItemRelationship pointing to another quoteItem in the same Quote Request. This is because the UNI did not exist at that moment and was also a part of the quotation. In the case of quoting the update of an existing Access E-Line, the UNI is also existing and it must be referenced with the use of productRelationship. This example assumes that the UNI product is available in Seller's Inventory with the id=SP1_UNI.

```
{
  <<Quote attributes...>>
  "quoteItem": [
    {
      "id": "item-001",
      "action": "modify",
      ...
      "product": {
        "id": "AccessElineOVC-0001",
        "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"
              }
            }
          }
        },
        "productOffering": {
          "id": "000073"
        },
        "productRelationship": [
          {
            "relationshipType": "CONNECTS_TO_ENNI",
            "id": "SP1_ENNI"
          },
          {
            "relationshipType": "CONNECTS_TO_UNI",
            "id": "SP1_UNI"
          }
        ]
      },
      "relatedContactInformation": [
        {
          "number": "1-234-567-890",
          "emailAddress": "john@buyer.mef.com",
          "role": "buyerContactInformation",
          "name": "John Example"
        }
      ]
    }
  ]
}
```

6.2.6.3. Quote Item Structure for `delete` Action

The example below represents a single Quote request for deletion (`action` equals `delete`) of an existing Access E-Line product (`id=AccessElineOVC-0001`). It assumes that `instantSyncQuote=true` so no `relatedContactInformation` is provided.

```
{
  <<Quote attributes...>>
  "quoteItem": [
    {
      "id": "item-001",
      "action": "delete",
      "product": {
        "id": "AccessElineOVC-0001"
      }
    }
  ]
}
```

[R47] In the `delete` request the Buyer **MUST** provide the `quoteItem.product.id` attribute. [MEF80 R30]

[R48] In the `delete` request the Buyer **MUST NOT** provide any of the following attributes: [MEF80 R31]

- `quoteItem.productOfferingQualificationItem`
- `quoteItem.quoteItemRelationship`
- `quoteItem.product.*` - product attributes other than `id`

6.3. Use Case 2: Retrieve Quote List

The Buyer can retrieve a list of `Quotes` by using a `GET /quote` operation with desired filtering criteria.

[O6] The Buyer **MAY** use any of the following query parameters to query for the Quote list: [MEF80 O17]

- `state`
- `quoteLevel`
- `externalId`
- `projectId`
- `quoteDate.gt`
- `quoteDate.lt`
- `requestedQuoteCompletionDate.gt`
- `requestedQuoteCompletionDate.lt`
- `expectedQuoteCompletionDate.gt`
- `expectedQuoteCompletionDate.lt`
- `effectiveQuoteCompletionDate.gt`
- `effectiveQuoteCompletionDate.lt`

[O7] The Buyer **MAY** use a combination of attributes to avoid getting an `Error422` with `tooManyRecords` code.

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/quoteManagement/v10/quote?
state=approved.orderable&quoteLevel=firm&limit=20&offset=0
```

The example above shows a Buyer's request to get the first twenty `Quotes` that are in `approved.orderable` state and with `firm` level. The correct response (HTTP code `200`) contains a list of `Quote_Find` objects matching the criteria in the response body. To get more details (e.g. the item level information), the Buyer has to query a specific `Quote` by id.

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`

[D5] The Seller **SHOULD** support the pagination mechanism.

[CR1]<[D5] Seller **MUST** use either `X-Total-Count` or `X-Pagination-Throttled` to indicate that the page was truncated and additional results are available.

[R49] The Seller **MUST** put the following attributes into the `Quote_Find` object in the response: [MEF80 R77]

- `id`
- `effectiveQuoteCompletionDate`
- `expectedQuoteCompletionDate`
- `externalId`
- `projectId`

- `quoteDate`
- `quoteLevel`
- `requestedQuoteCompletionDate`
- `state`

In case no items match the criteria an empty list is returned.

[R50] In case of too many matching items are found (the definition of 'too many' is up to Seller's discretion), the Seller **MUST** return an `Error422` with `code=tooManyRecords`.

In that case, the Buyer can change the filter criteria and/or repeat the query with pagination in order to avoid receiving the `tooManyRecords` error.

Below you can find a response with 2 matching entities:

```
[
  {
    "id": "00000000-0000-0000-0000-000000000123",
    "effectiveQuoteCompletionDate": "2020-08-10T16:45:20.421Z",
    "expectedQuoteCompletionDate": "2020-08-10T16:45:39.421Z",
    "externalId": "BuyerId-00112233",
    "projectId": "Project-ABCDEF",
    "quoteDate": "2020-08-10T16:40:33.422Z",
    "quoteLevel": "firm",
    "requestedQuoteCompletionDate": "2020-08-10T16:45:39.368Z",
    "state": "approved.orderable"
  },
  {
    "id": "00000000-1212-3434-0000-987600000abc",
    "effectiveQuoteCompletionDate": "2020-09-112T08:25:20.421Z",
    "expectedQuoteCompletionDate": "2020-09-112T08:25:39.421Z",
    "externalId": "BuyerId-99887766",
    "projectId": "Project-ZYX",
    "quoteDate": "2020-09-112T08:20:33.422Z",
    "quoteLevel": "firm",
    "requestedQuoteCompletionDate": "2020-09-112T08:25:39.368Z",
    "state": "approved.orderable"
  }
]
```

6.4. Use Case 3: Retrieve Quote by Quote Identifier

The Buyer can get detailed information about the Quote from the Seller by using a `GET /quote/{{id}}` operation. In case `id` does not allow to find a `Quote` in Seller's Inventory, an error response `404` must be returned. The payload returned in the response includes all the attributes Buyer has provided while sending a Quote request. The attributes provided by the Seller depend on the status of the `Quote` and may require some time to be set.

[R51] If `quoteLevel` equals `firm` then the response **MUST** specify attributes as shown in Table 9 and Table 10. [MEF80 R81]

Please note that for readability purposes following tables (9, 10, 11, and 12) do not show attributes specified by the Buyer that must be echoed back by the Seller without any change. Attributes required to be provided by the Seller are shown by an "R", Required if Populated by the Seller shown by a "PR", or Optional to be provided by the Seller or the Buyer shown by an "O".

	accepted	acknowledged	cancelled	declined	expired	inProgress	inProgress.draft	approved.orderable	approved.orderableAlternate	rejected	unableToProvide
id	R	R	R	R	R	R	R	R	R	R	R
state	R	R	R	R	R	R	R	R	R	R	R
stateChange	R	R	R	R	R	R	R	R	R	R	R
quoteDate	R	R	R	R	R	R	R	R	R	R	R
expectedQuoteCompletionDate	O	O	O	O	O	R	R	O	O	O	O
validFor	O			O	O			R	R		
effectiveQuoteCompletionDate			R					R	R		R
quoteLevel	R		PR	R	R		R	R	R		
note	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR
relatedContactInformation (role=sellerContactInformation)	R	E	R	R	R	R	R	R	R	E	R

Table 9. Seller Response to Query by ID, FIRM Quote Level, Quote Attributes

	accepted	acknowledged	cancelled	declined	expired	inProgress	inProgress.draft	approved.orderable	approved.orderableAlternate	rejected	unableToProvide
subjectToFeasibilityCheck	R	PR	PR	R	R		R	R	R		
note	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR
state	R	R	R	R	R	R	R	R	R	R	R
stateChange	R	R	R	R	R	R	R	R	R	R	R
price	R			R	R		R	R	R		
quoteItemTerm	R			R	R		R	R	R		
quoteItemInstallationInterval	R			R	R		R	R	R		
terminationError				PR							R

Table 10. Seller Response to Query by ID, FIRM Quote Level, QuoteItem Attributes

[R52] If `quoteLevel` equals `budgetary` then the response must specify attributes as shown in Table 11 and Table 12. [MEF80 R82]

	answered	acknowledged	cancelled	expired	inProgress	rejected	unableToProvide
subjectToFeasibilityCheck	NA	NA	NA	NA	NA	NA	NA
note	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	
state	R	R	R	R	R	R	R
stateChange	R	R	R	R	R	R	R
price	R			R			
quoteItemTerm	R			R			
quoteItemInstallationInterval	R			R			
terminationError						PR	PR

Table 11. Seller Response to Query by ID, BUDGETARY Quote Level, Quote attributes

	answered	acknowledged	cancelled	expired	inProgress	rejected	unableToProvide
subjectToFeasibilityCheck	NA	NA	NA	NA	NA	NA	NA
note	E / PR	E / PR	E / PR	E / PR	E / PR	E / PR	
state	R	R	R	R	R	R	R
stateChange	R	R	R	R	R	R	R
price	R			R			
quoteItemTerm	R			R			
quoteItemInstallationInterval	R			R			
terminationError						PR	PR

Table 12. Seller Response to Query by ID, BUDGETARY Quote Level, QuoteItem attributes

In the example below, the `Quote` is in the `approved.orderable` state.

[R53] The `Quote` can be in `approved.orderable` state only when all its `QuoteItems` are in the `approved.orderable` state as well [MEF80 R70].

[D6] When moving the Quote to `rejected` or `unableToProvide` state, the Seller **SHOULD** use the `quoteItem.terminationError` to describe the reason for item processing failure.

The Seller's response to an inquiry is valid for one week (`validFor.endDateTime`). The `stateChange` lists the history of the `Quote` state.

```
{
  "id": "00000000-0000-0000-0000-000000000123",
  "href": "{{baseUrl}}/quote/00000000-0000-0000-0000-000000000123",
  "state": "approved.orderable",
  "quoteLevel": "firm",
  "instantSyncQuote": false,
  "buyerRequestedQuoteLevel": "firm",
  "effectiveQuoteCompletionDate": "2020-08-10T16:45:20.421Z",
  "quoteDate": "2020-08-10T16:40:33.422Z",
  "validFor": {
    "endDateTime": "2020-08-17T16:45:39.422Z"
 },
  "quoteItem": [
 {
      "state": "approved.orderable",
      "subjectToFeasibilityCheck": false,
      "id": "item-001",
      "action": "add"
      << some attributes are omitted >>
 },
 {
      "state": "approved.orderable",
      "subjectToFeasibilityCheck": false,
      "id": "item-002",
      "action": "add"
      << some attributes are omitted >>
 }
 ],
  "stateChange": [
 {
      "changeDate": "2020-08-10T16:45:39.422Z",
      "state": "approved.orderable"
 },
 {
      "changeDate": "2020-08-10T16:42:39.422Z",
      "state": "inProgress.draft"
 },
 {
      "changeDate": "2020-08-10T16:40:39.422Z",
      "state": "inProgress"
 },
 {
      "changeDate": "2020-08-10T16:40:33.422Z",
      "state": "acknowledged"
 }
 ],
  "relatedContactInformation": [
 {
      "emailAddress": "john.example@buyer.mef.com",
      "name": "John Example",
      "number": "12-345-6789",
      "role": "buyerContactInformation"
 },
 {
      "emailAddress": "kate.example@seller.mef.com",
      "name": "Kate Example",
      "number": "12-345-67890",
      "role": "sellerContactInformation"
 }
 ]
}
```

6.5. Use case 4: Cancel Quote by Quote Identifier

The Buyer may decide to cancel a Quote request that is in progress (`inProgress` or `inProgress.draft` states). A `cancelQuote` operation from the `POST /cancelQuote` endpoint must be used to do so.

[R54] The Seller **MUST** provide the ability for a Buyer to cancel a `Quote` when the `Quote` is in the `inProgress` or `inProgress.draft` state. [MEF80 R1].

The message body contains only two attributes:

- `quoteId` - mandatory one to point which `Quote` must be canceled
- `reason` - to optionally specify the cause of the cancellation.

```
{
  "quoteId": "00000000-0000-0000-0000-000000000123",
  "reason": "My requirements have changed I do not need this Quote to finish processing."
}
```

The Seller responds with the same body. No `id` is added. The cancellation request does not create any trackable object in the Seller's system that can be further tracked or monitored by the Buyer. The Seller is obliged to process such a request and cancel the `Quote`.

The reason for having a separate `POST` endpoint for operation on the `Quote`, even when it is not technically required, is keeping the pattern consistency with Ordering API, where operations like `cancelOrder` are long-lasting processes that can be tracked by the Buyer and might not necessarily end up with success.

Figure 21 presents an example of the Cancel use case flow:

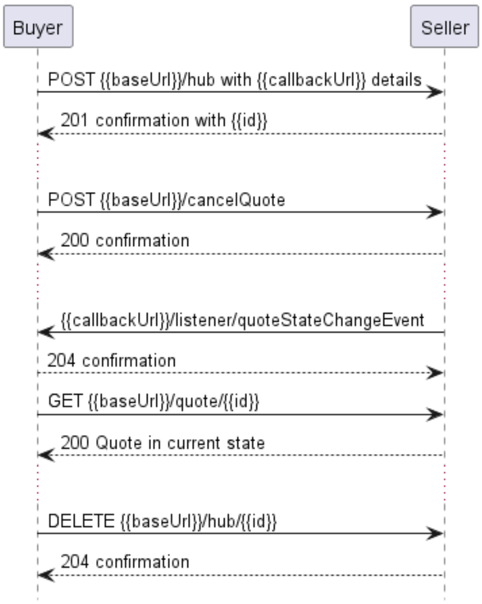

Buyer
Seller
POST {{baseUrl}}/hub with {{callbackUrl}} details
201 confirmation with {{id}}
POST {{baseUrl}}/cancelQuote
200 confirmation
{{callbackUrl}}/listener/quoteStateChangeEvent
204 confirmation
GET {{baseUrl}}/quote/{{id}}
200 Quote in current state
DELETE {{baseUrl}}/hub/{{id}}
204 confirmation

Figure 21 Use case 4, 5: Cancel or Decline Quote by Quote Identifier

***Note*:** If the Buyer requests cancellation of a Quote that does not belong to them, the Seller should respond that the Quote does not exist (`Error422` with `code` equal to `referenceNotFound`)

6.6. Use case 5: Decline Quote by Quote Identifier

The Buyer may also decide to decline a `Quote` provided by the Seller (`approved.orderable` or `approved.orderableAlternate` states). A `declineQuote` operation from the `POST /declineQuote` endpoint must be used to do so.

[R55] The Seller **MUST** provide the ability for a Buyer to decline a `Quote` when the `Quote` is in the `approved.orderable` or `approved.orderableAlternate` state. [MEF80 R2].

The Decline Request has the same body and rules of usage as the Cancel Request mentioned in the section above.

Note: Declining a Quote is optional. The Buyer may as well leave the response unattended and let it time out.

Note: There is no endpoint to move the Quote to `accepted` state. The Buyer accepts a Quote by placing an Order (using Order API) that refers to it. In that case, it is the Seller who makes the transition to the `accepted` state.

6.7. Use case 6: Register for Quote Notifications

The Seller communicates with the Buyer with Notifications provided that:

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

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

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

The usage of a combination of these attributes fulfills the [MEF80 R84], [MEF80 R85], and [MEF80 R86] requirements.

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=quoteStateChangeEvent"
}
```

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

```
eventType=quoteStateChangeEvent,quoteItemStateChangeEvent
```

or

```
eventType=quoteStateChangeEvent&eventType=quoteItemStateChangeEvent
```

Note: There are only 2 event types in the Quote API so eventually there is no need to request both in the query - an empty query may be used as well.

[R56] The Buyer **MUST** provide the `callback` during notification registration.

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

[O8] The Seller **MAY** support the ability for the Buyer to Register for Quote Notification. [MEF80 O19]

[CR2]<[O8] If the Seller supports the ability for the Buyer to Register for Quote Notifications, they **MUST** support sending Quote Notifications. [MEF80 CR3<O19]

[R57] If the Seller does not support notifications, they **MUST** return an error message (`Error501`) to the Buyer indicating that notifications are not supported.

[R58] `eventType` is the only attribute that the Seller **MUST** support in the query.

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

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

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

- `https://buyer.mef.com/listenerEndpoint/mefApi/sonata/productOfferingQualificationNotification/v10/listener/quoteStateChangeEvent`

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.

[R59] 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/quoteManagement/v10/hub/00000000-0000-0000-0000-000000000678
```

[R60] 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 need to be removed and replaced by the new `EventSubscription` with adjusted `query` attribute.

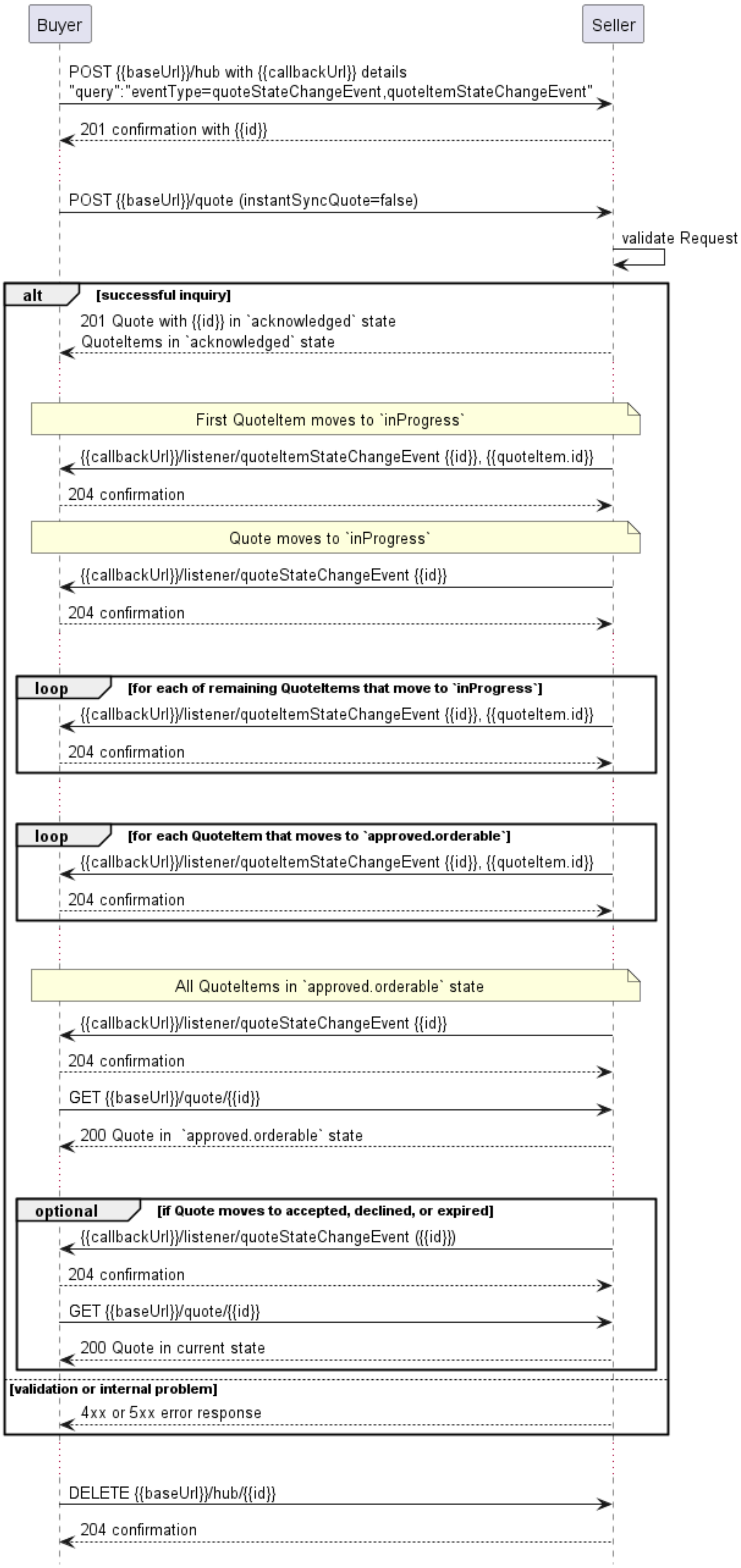
Note: The above note concludes that the Buyer cannot update the existing `EventSubscription`. Every kind of update is done by subscription replacement.

6.8. Use case 7: Send Quote Notification

Notifications are used to asynchronously inform the Buyer about the `Quote.state` or `QuoteItem.state` attributes change. The Seller's synchronous response to a Create Quote request is considered to act as a Create Notification so there is no explicit Create Notification type. The next notification must be sent when the `state` changes compared to the previously sent one.

For the sake of readability, all previous flow diagrams presented only cases of using only the `quoteStateChangeEvent`. Figure 22 presents the end-to-end sequence of communication in Use

Case 1b - Deferred Quote Response Requested and Provided with Buyer's subscription to both `quoteStateChangeEvent` and `quoteItemStateChangeEvent` event types.



Buyer
Seller
POST {{baseUrl}}/hub with {{callbackUrl}} details
"query":"eventType=quoteStateChangeEvent,quoteItemStateChangeEvent"
201 confirmation with {{id}}
POST {{baseUrl}}/quote (instantSyncQuote=false)
validate Request
alt
[successful inquiry]
201 Quote with {{id}} in `acknowledged` state
QuoteItems in `acknowledged` state
First QuoteItem moves to `inProgress`
{{callbackUrl}}/listener/quoteItemStateChangeEvent {{id}}, {{quoteItem.id}}
204 confirmation
Quote moves to `inProgress`
{{callbackUrl}}/listener/quoteStateChangeEvent {{id}}
204 confirmation
loop
[for each of remaining QuoteItems that move to `inProgress`]
{{callbackUrl}}/listener/quoteItemStateChangeEvent {{id}}, {{quoteItem.id}}
204 confirmation
loop
[for each QuoteItem that moves to `approved.orderable`]
{{callbackUrl}}/listener/quoteItemStateChangeEvent {{id}}, {{quoteItem.id}}
204 confirmation
All QuoteItems in `approved.orderable` state
{{callbackUrl}}/listener/quoteStateChangeEvent {{id}}
204 confirmation
GET {{baseUrl}}/quote/{{id}}
200 Quote in `approved.orderable` state
optional
[if Quote moves to accepted, declined, or expired]
{{callbackUrl}}/listener/quoteStateChangeEvent ({{id}})
204 confirmation
GET {{baseUrl}}/quote/{{id}}
200 Quote in current state
[validation or internal problem]
4xx or 5xx error response
DELETE {{baseUrl}}/hub/{{id}}
204 confirmation

Figure 22. Use case 1b: Deferred Quote Response Requested and Provided with QuoteItem Notifications

After a successful Notification subscription, the Buyer sends a Create Quote request asking for a deferred response. The Seller responds with Quote and all items in `acknowledged` state. When the first Quote Item moves to `inProgress`, a `quoteItemStateChangeEvent` is sent. Immediately the Quote also changes its state to `inProgress` and the `quoteStateChangeEvent` is sent. Then the rest (if any) of the Quote Items are processed. When particular items are done processing they reach the `approved.orderable` state. Once all are successfully done, the Quote also changes state to `approved.orderable`. The Buyer will likely now ask for the Quote details.

The events are sent only after a synchronous response to the create Quote request was provided.

[O9] The Seller **MAY** support sending Quote Notifications. [MEF80 O20]

[R6\10] The Seller **MUST** support sending Quote Notifications if a deferred response is supported.

[CR3]<[O9] The Seller **MUST** be able to send Quote Notifications to Buyers for both Immediate and Deferred Quote Responses. [MEF80 CR4<O20]

[CR4]<[O9] If the Buyer has registered for Quote State notifications, Notifications for Immediate Quote Responses **MUST** be sent to indicate when a Quote has changed from the `approved.orderableAlternate` or `approved.orderable` states to one of the Terminal States (`declined`, `expired`, or `accepted`). [MEF80 CR5<O20]

[CR5]<[O9] The Seller **MUST** send Quote Notifications to Buyers who have registered for Quote Notifications [MEF80 CR7<O20].

[CR6]<[O9] The Seller **MUST NOT** send Quote Notifications to Buyers who have not registered for Quote Notifications [MEF80 CR6<O20].

Seller sends notifications about `Quote` or `QuoteItem` state change events. `Quote` state change event might look like:

```
{
  "eventId": "event-001",
  "eventType": "quoteStateChangeEvent",
  "eventTime": "2020-08-10T16:40:39.422Z",
  "event": {
    "id": "00000000-0000-0000-0000-000000000123",
    "state": "inProgress"
  }
}
```

`QuoteItemStateChangeEvent` example:

```
{
  "eventId": "event-002",
  "eventType": "quoteItemStateChangeEvent",
  "eventTime": "2020-08-10T16:40:39.422Z",
  "event": {
    "id": "00000000-0000-0000-0000-000000000123",
    "quoteItemId": "item-001",
    "state": "inProgress"
  }
}
```

Note: The body of the event carries only the Quote and/or Quote Item `id` and `state`. The Buyer needs to query Quote by `id` to get details.

[R62] The Seller **MUST** provide the following attributes of `Event`:

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

[R63] The Seller **MUST** provide the following attributes of `QuoteStateChangeEventPayload` when sending `QuoteStateChangeEvent`: [MEF80 CR8<O20]

- `id`
- `state`

[R64] The Seller **MUST** provide the following attributes of `QuoteItemStateChangeEventPayload` when sending `QuoteItemStateChangeEvent`: [MEF80 CR8<O20], [MEF80 CR8<O21]

- `id`
- `quoteItemId`
- `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 Quote API uses the error responses as depicted and described below.

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

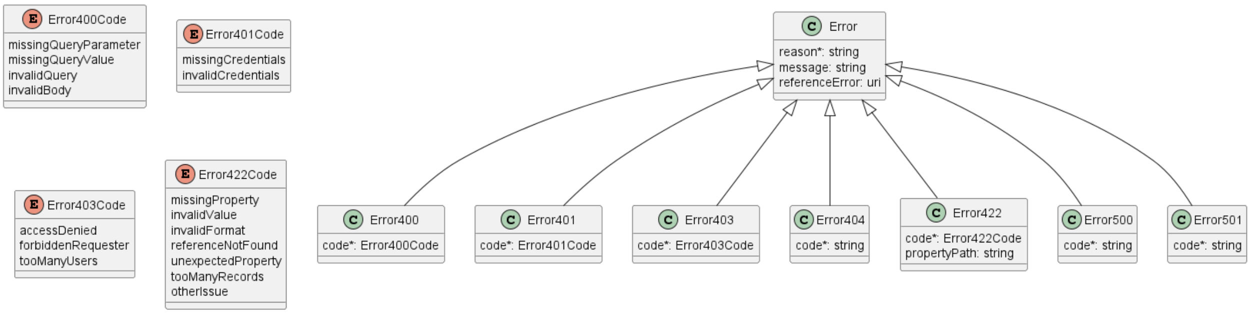


Figure 23. Data model types to represent an erroneous response

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. (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 24 presents the whole Quote Management data model The data types, requirements related to them and mapping to MEF 80 specifications are discussed later in this section.

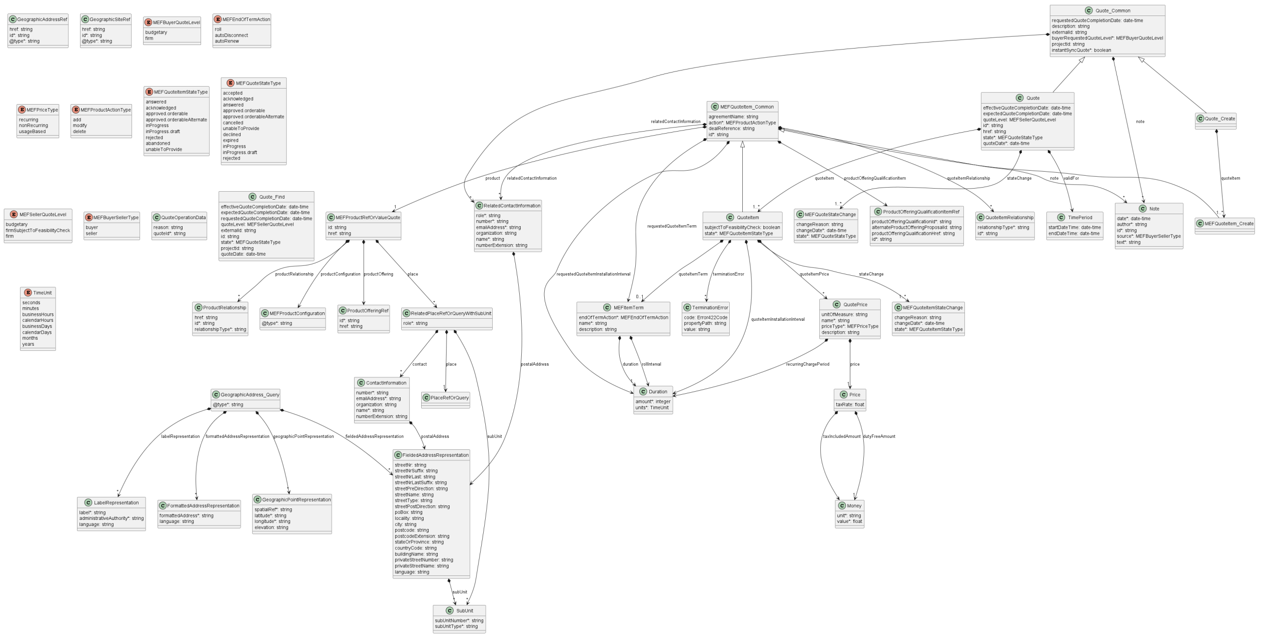


Figure 24. Quote Management Data Model

7.2.1. Quote

7.2.1.1 Type Quote_Common

Description: Quote can be used to negotiate service and product acquisition or modification between a customer and a service provider. Quote contains a list of quote items, a reference to a customer, a list of productOfferings, and attached prices and conditions.

Name	Type	M/O	Description	MEF 80
note	Note[]	O	Free form text associated with the quote. Only useful in processes involving human interaction. Not applicable for the automated process.	Note
requestedQuote-CompletionDate	date-time *format = date-time*	O	This is requested date - from quote requester - to get a complete response for this quote	Requested Quote Completion Date
relatedContact-Information	RelatedContact-Information[]	O	Party playing a role for this quote. If `instantSyncQuote` equals `false` then the Buyer MUST specify Buyer Contact Information ('role: buyerContactInformation') and the Seller MUST specify Seller Contact Information ('role: sellerContactInformation')	Buyer Contact Information (role` equals `buyerContactInformation), Seller Contact Information (role` equals `sellerContactInformation)
description	string	O	Description of the quote	Description

Name	Type	M/O	Description	MEF 80
externalId	string	O	ID given by the consumer and only understandable by him (to facilitate his searches afterwards)	Buyer Quote Identifier
buyerRequested-QuoteLevel	MEFBuyer-QuoteLevel	M	An indication of whether the Buyer's Quote request is for a Quote of Budgetary or Firm level.	Buyer Requested Quote Level
projectId	string	O	An identifier that is used to group Quotes that represent a unit of functionality that is important to a Buyer. A Project can be used to relate multiple Quotes together.	Project Identifier
instantSyncQuote	boolean	M	If this flag is set to true, the Buyer requests an immediate Quote to be provided in the response to the creation of a Quote.	Immediate Quote Response

7.2.1.2. Type Quote_Create

Description: Quote can be used to negotiate service and product acquisition or modification between a customer and a service provider. Quote contains a list of quote items, a reference to a customer, a list of productOfferings, and attached prices and conditions.

Inherits from:

- Quote_Common

Name	Type	M/O	Description	MEF 80
quoteItem	MEFQuoteItem_Create[] *minItems = 1*	M	An item of the quote - used to describe an operation on a product to be quoted	Quote Item

7.2.1.3. Type Quote

Description: Quote can be used to negotiate service and product acquisition or modification between a customer and a service provider. Quote contains a list of quote items, a reference to a customer, a list of productOfferings, and attached prices and conditions.

Inherits from:

- Quote_Common

Name	Type	M/O	Description	MEF 80

Name	Type	M/O	Description	MEF 80
effectiveQuote-CompletionDate	date-time *format = date-time*	O	Date when the Quote State was set to one of the Completion States.	Quote Completion Date
expectedQuote-CompletionDate	date-time *format = date-time*	O	This is the date provided by the Seller to indicate the date by which the Quote is expected to reach a Quote Completion State	Expected Quote Completion Date
validFor	TimePeriod	O	Quote validity period. For use in the context of this attribute, only the endDateTime attribute must be used.	Valid Until Date

Name	Type	M/O	Description	MEF 80
quoteLevel	MEFSellerQuoteLevel	O	An indication of whether the Seller's Quote Response is Budgetary, Firm - Subject to Feasibility Check, or Firm. The Seller Quote Level is provided by the Seller when responding to a Quote request. This represents the lowest Quote Item Level of all Quote Items included in the Quote.	Seller Quote Level
quoteItem	QuoteItem[] *minItems = 1*	M	An item of the quote - it is used to describe an operation on a product to be quoted	Quote Item
stateChange	MEFQuoteStateChange[] *minItems = 1*	M	State change for the Quote	Quote ACCEPTED Date, Quote IN_PROGRESS Date, Quote IN_PROGRESS_DRAFT Date, Quote Completion State Date, Quote CANCELLED Date, Quote DECLINED Date, Quote EXPIRED Date, Quote REJECTED Date
id	string	M	Unique identifier - attributed by quoting system	Seller Quote Identifier

Name	Type	M/O	Description	MEF 80
href	string	O	Hyperlink representing this Quote. Hyperlink MAY be used when providing a response by the Seller	Not represented in MEF 80
state	MEFQuoteStateType	M	The state of the Quote.	Quote State
quoteDate	date-time *format = date-time*	M	Date and time when the quote was created	Quote Request Date

7.2.1.4. `enum` MEFQuoteStateType

Description: Possible values for the status of a Quote. Following mapping has been used between `MEFQuoteStateType` and MEF 80:

QuoteStateType	MEF 80
accepted	ACCEPTED
acknowledged	ACKNOWLEDGED
answered	ANSWERED
approved.orderable	ORDERABLE
approved.orderableAlternate	ORDERABLE_ALTERNATE
declined	DECLINED
expired	EXPIRED
cancelled	CANCELLED
unableToProvide	UNABLE_TO_PROVIDE
inProgress	IN_PROGRESS
inProgress.draft	IN_PROGRESS_DRAFT
rejected	REJECTED

7.2.1.5. `enum` MEFBuyerQuoteLevel

Description: An indication of whether the Buyer's Quote Request is for a Budgetary or Firm Quote Level. Set by the Buyer. Buyer Requested Quote Level contains the possible values and may be set by the Buyer on the Request. All Quote Items in a Quote have the same Quote Level.

Value	MEF 80
budgetary	BUDGETARY
firm	FIRM

7.2.1.6. `enum` MEFSellerQuoteLevel

Description: An indication of whether the Seller's Quote Response is Budgetary, Firm - Subject to Feasibility Check, or Firm. The Seller Quote Level is provided by the Seller when responding to a Quote request. This represents the lowest Quote Item Level of all Quote Items included in the Quote.

Value	MEF 80
budgetary	BUDGETARY
firmSubjectToFeasibilityCheck	FIRM_SUBJECT_TO_FEASIBILITY_CHECK
firm	FIRM

7.2.1.7. Type MEFQuoteStateChange

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

Name	Type	M/O	Description	MEF 80
changeReason	string	O	Additional comment related to state change	Not represented in MEF 80
changeDate	date-time *format = date-time*	M	The date when the state was reached	Not represented in MEF 80
state	MEFQuoteStateType	M	The state reached at the change date	Not represented in MEF 80

7.2.1.8. Type MEFQuoteItemStateChange

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

Name	Type	M/O	Description	MEF 80
changeReason	string	O	Additional comment related to state change	Not represented in MEF 80
changeDate	date-time *format = date-time*	M	The date when the state was reached	Not represented in MEF 80
state	MEFQuoteItemStateType	M	The state reached at the change date	Not represented in MEF 80

7.2.1.9. Type Quote_Find

Description: This class represents a single list item for the response of `listQuote` operation.

Name	Type	M/O	Description	MEF 80

Name	Type	M/O	Description	MEF 80
effectiveQuoteCompletionDate	date-time *format = date-time*	O	Date when the Quote State was set to one of the Completion States	Quote Completion Date
expectedQuoteCompletionDate	date-time *format = date-time*	O	This is the date provided by the Seller to indicate expected Quote completion date	Expected Quote Completion Date
requestedQuoteCompletionDate	date-time *format = date-time*	O	This is requested date - from quote requester - to get a complete response for this quote	Requested Quote Completion Date
quoteLevel	MEFSellerQuoteLevel	O	The level of the Quote provided by the Seller. This represents the lowest Quote Item Level of all Quote Items included in the Quote.	Quote Level
externalId	string	O	ID given by the consumer and only understandable by him (to facilitate his searches afterward)	Buyer Quote Identifier
id	string	O	Unique identifier - attributed by quoting system	Seller Quote Identifier
state	MEFQuoteStateType	M	The state of the Quote.	Quote State

Name	Type	M/O	Description	MEF 80
projectId	string	O	An identifier that is used to group Quotes that represent a unit of functionality that is important to a Buyer. A Project can be used to relate multiple Quotes together.	Project Identifier
quoteDate	date-time *format = date-time*	O	Date and time when the quote was created	Quote Request Date

7.2.2. Quote Item

7.2.2.1 Type MEFQuoteItem_Common

Description: Quote items describe an action to be performed on a productOffering or a product in order to get pricing elements and conditions.

Name	Type	M/O	Description	MEF 80
requested-QuoteItemTerm	MEFItemTerm	O	The terms of the Quote Item. Used to describe a term (also known as commitment) for a Quote Item. Each Quote Item in a Quote Request could have a different Requested Quote Item Term. The Buyer specifies the longest term that they would accept. The Buyer may be willing to accept a shorter term. If the Seller responds with a term longer than the Buyer's request, it is treated as an alternate response.	Requested Quote Item Term

Name	Type	M/O	Description	MEF 80
note	Note[]	O	Free form text associated with the quote item. Only useful in processes involving human interaction. Not applicable for the automated process.	Quote Item Notes
product	MEFProduct-RefOrValueQuote	M	The Buyer's existing Product for which the quote is being requested.	Product Identifier, Product Specific Attributes
productOffering-QualificationItem	ProductOffering-QualificationItemRef	O	A reference to a previously done POQ with item specified	POQ
relatedContact-Information	RelatedContact-Information[]	O	Contact information of an individual or organization playing a role for this Quote. If `instantSyncQuote` equals `false` then 'Quote Item Technical Contact' must be specified (`role: quoteItem-TechnicalContact`).	Quote Item Location Contact (role: quoteItem-LocationContact), Quote Item Technical Contact (role: quoteItem-TechnicalContact)
agreementName	string	O	Name of the agreement. The name is unique between the Buyer and the Seller.	Agreement
action	MEFProduct-ActionType	M	Product action to be applied to this Quote Item. This corresponds to the Order Item Action when an associated product is ordered.	Quote Item Product Action
dealReference	string	O	A pre-agreed pricing modifier reference that the Seller is offering to the Buyer which will impact the price.	Quote Item Deal Reference

Name	Type	M/O	Description	MEF 80
id	string	M	Identifier of the quote item (generally it is a sequence number 01, 02, 03, ...)	Quote Item Reference Number
requestedQuoteItem-InstallationInterval	Duration	O	The installation interval requested by the Buyer.	Requested Quote Item Installation Interval
quoteItem-Relationship	QuoteItem-Relationship[]	O	A relationship from item within a quote	Quote Item Relationship

7.2.2.2. Type MEFQuoteItem_Create

Description: A quote item describes an action to be performed on a productOffering or a product in order to get pricing elements and condition. The modeling pattern introduces the MEFQuoteItem_Common supertype to aggregate attributes that are common to both QuoteItem and MEFQuoteItem_Create. In this case the create type has a subset of attributes of the response type and does not introduce any new, thus the MEFQuoteItem_Create type has an empty definition.

Inherits from:

- MEFQuoteItem_Common

7.2.2.3. Type QuoteItem

Description: Quote items describe an action to be performed on a productOffering or a product in order to get pricing elements and conditions.

Inherits from:

- MEFQuoteItem_Common

Name	Type	M/O	Description	MEF 80
terminationError	TerminationError[]	O	When the Seller cannot process the Quote Item Request, the Seller returns a text-based list of reasons here.	Quote Item Termination Error

Name	Type	M/O	Description	MEF 80
quoteItemInstallationInterval	Duration	O	Quote Item Installation Interval as proposed by the Seller for the Quote.	Quote Item Installation Interval
subjectToFeasibilityCheck	boolean	O	For a Firm Quote Level indicates if the pricing requires a Feasibility Check. The Seller indicates if the Quote Item requires a Feasibility Check. This is not used for a Budgetary Quote Level.	Subject to Feasibility Check
quoteItemTerm	MEFItemTerm[] *maxItems = 1*	O	Quote Item Term as defined by the Seller and part of the Quote for the Quote Item.	Quote Item Term
state	MEFQuoteItemStateType	M	The state of the Quote Item.	Quote Item State
stateChange	MEFQuoteItemStateChange[] *minItems = 1*	M	State change for the Quote	Not represented in MEF 80
quoteItemPrice	QuotePrice[]	O	Price for this quote item	Quote Item Price

7.2.2.4. `enum` MEFProductActionType

Description: Product action to be applied to the Quote Item. This corresponds to the Order Item Action when an associated product is ordered.

MEFProductActionType	MEF 80
add	INSTALL
modify	CHANGE
delete	DISCONNECT

7.2.2.5. `enum` MEFQuoteItemStateType

Description: Possible values for the status of a QuoteItem. Following mapping has been used between `MEFQuoteItemStateType` and MEF 80:

MEFQuoteItemStateType	MEF 80
answered	ANSWERED
acknowledged	ACKNOWLEDGED
approved.orderable	ORDERABLE
approved.orderableAlternate	ORDERABLE_ALTERNATE
inProgress	IN_PROGRESS
inProgress.draft	IN_PROGRESS_DRAFT
abandoned	ABANDONED
rejected	REJECTED
unableToProvide	UNABLE_TO_PROVIDE

7.2.2.6. Type ProductOfferingQualificationItemRef

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

Name	Type	M/O	Description	MEF 80
productOfferingQualificationId	string	M	Unique identifier of related Product Offering Qualification.	POQ Identifier
alternateProductOfferingProposalId	string	O	A unique identifier for the Alternate Product Offering Proposal assigned by the Seller, if the referenced product offering qualification comes from an alternate proposal.	Alternate Product Proposal Identifier
productOfferingQualificationHref	string	O	Reference of the related Product Offering Qualification.	Not represented in MEF 80
id	string	M	Id of an item of a product offering qualification	POQ Item Identifier

7.2.2.7. 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. The id of the Product offering is assigned by the Seller. The Buyer and the Seller exchange information about offerings' ids during the onboarding process.

Name	Type	M/O	Description	MEF 80
id	string	M	unique identifier of the Product Offering.	Quote.Product Offering Identifier
href	string	O	Hyperlink to a Product Offering in Sellers catalog. In case Seller is not providing catalog capabilities this field is not used. The catalog API definition is provided by the Seller to Buyer during onboarding Hyperlink MAY be used when providing response by the Seller Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a requestHyperlink reference	Not represented in MEF 80

7.2.2.8. Type QuoteItemRelationship

Description: Used to describe the relationship between quote items. These relationships could have an impact on pricing and conditions

Name	Type	M/O	Description	MEF 80
relationshipType	string	M	Relationship type as relies on, bundles, etc... Specifies the nature of the relationship to the related Quote Items. 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 Relationship Nature values are defined in the Product Specification.	Quote Item Relationship Nature
id	string	M	ID of the related quote item (must be in the same quote)	Quote Item Relationship Identifier

7.2.2.9. Type MEFItemTerm

Description: The terms of the Quote Item. Used to describe a term (also known as commitment) for a Quote Item. Each Quote Item in a Quote Request could have a different Requested Quote Item Term. The Buyer specifies the longest term that they would accept. The Buyer may be willing to accept a shorter term. If the Seller responds with a term longer than the Buyer's request, it is treated as an alternate response.

Name	Type	M/O	Description	MEF 80
duration	Duration	M	Duration of the term	Quote Item Term Duration
endOfTermAction	MEFEndOfTermAction	M	The action that needs to be taken by the Seller once the term expires	Seller End of Term Action
name	string	M	Name of the term	Quote Item Term Name
description	string	O	Description of the term	Quote Item Term Description
rollInterval	Duration	O	The recurring period that the Buyer is willing to pay to the end of upon disconnecting the Product after the original term has expired. If `endOfTermAction` is equal to `roll` then `rollInterval` MUST be specified. If `endOfTermAction` is equal to `autoRenew` or `autoDisconnect`, then `rollInterval` MUST NOT be specified.	Roll Interval

7.2.2.10. enum MEFEndOfTermAction

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

Value	MEF 80
roll	ROLL
autoDisconnect	AUTO_DISCONNECT
autoRenew	AUTO_RENEW

7.2.2.11. Type QuotePrice

Description: Description of price and discount awarded

Name	Type	M/O	Description	MEF 80
unitOfMeasure	string	O	Unit of Measure if price depending on it (Gb, SMS volume, etc..) if Quote Item Price Type equals usageBased	Quote Item Price Unit Of Measure
price	Price	M	The associated price	Quote Item Price Amount
name	string	M	Name of the price	Quote Item Price Name
priceType	MEFPriceType	M	Indicates if the price is for recurring, non-recurring, or usage based charges	Quote Item Price Type
description	string	O	Description of the price	Quote Item Price Description
recurringChargePeriod	Duration	O	Used for a recurring charge to indicate a period	Quote Item Price Recurring Charge Period

7.2.2.12. Type Price

Description: Provides all amounts (tax included, duty-free, tax rate) and used currency of a Price

Name	Type	M/O	Description	MEF 80
taxRate	float *format = float*	O	Price Tax Rate. Unit: [%]. E.g. value 16 stand for 16% tax.	Price Tax Rate.
taxIncludedAmount	Money	O	All taxes included amount (expressed in the given currency)	Price Tax Included Amount
dutyFreeAmount	Money	M	All taxes excluded amount (expressed in the given currency)	Price Duty Free Amount

7.2.2.13. enum MEFPriceType

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

Value	MEF 80
recurring	RECURRING
nonRecurring	NON_RECURRING
usageBased	USAGE_BASED

7.2.3. Product representation

7.2.3.1. Type MEFProductRefOrValueQuote

Description: One or more services sold to a Buyer by a Seller. A particular Product Offering defines the technical and commercial attributes and behaviors of a Product.

Name	Type	M/O	Description	MEF 80
id	string	O	The unique identifier of an in-service Product that is the quotation's subject. This field MUST be populated if an item `action` is either `modify` or `delete`. This field MUST NOT be populated if an item `action` is `add`.	Product Identifier
href	string	O	Hyperlink to the product in Seller's inventory that is the quotation's subject. Hyperlink MAY be used when providing a response by the Seller. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request	Not represented in MEF 80
place	RelatedPlace-RefOrQueryWithSubUnit[]	O	A list of places that are related to the Product. For example an installation location	Quote Item Location and Quote Item Location Type
productConfiguration	MEFProductConfiguration	O	Technical attributes for the Product that would be delivered to fulfill the Quote Item.	Product Specific Attributes

Name	Type	M/O	Description	MEF 80
productOffering	ProductOfferingRef	O	A particular Product Offering defines the technical and commercial attributes and behaviors of a Product.	Product Offering Identifier
productRelationship	ProductRelationship[]	O	A list of references to existing products that are related to the Product that would be delivered to fulfill the Quote Item	Product Relationships

7.2.3.2. Type MEFProductConfiguration

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

Name	Type	M/O	Description	MEF 80
@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 the Mplify product, this is the URN provided in the Product Specification.	Not represented in MEF 80

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 same relationshipType the Seller determines if a list is supported as defined in the Product Specification.

Name	Type	M/O	Description	MEF 80
href	string	O	Hyperlink to the product in Seller's inventory that is referenced Hyperlink MAY be used when providing a response by the Seller Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request	Not represented in MEF 80
id	string	M	unique identifier of the related Product	Related Product Identifier

Name	Type	M/O	Description	MEF 80
relationshipType	string	M	Specifies the type (nature) of the relationship to the related Product. The nature of required relationships varies for Products of different types. For example, a UNI or ENNI Product may not have any relationships, but an Access E-Line may have two mandatory relationships (related to the UNI on one end and the ENNI on the other). More complex Products such as multipoint IP or Firewall Products may have more complex relationships. As a result, the allowed and mandatory `relationshipType` values are defined in the Product Specification.	Product Relationship Nature

7.2.4. Place representation

7.2.4.1. Type RelatedPlaceRefOrQueryWithSubUnit

Description: Allows pointing to a place by referring 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	MEF 80
place	PlaceRefOrQuery	M		Quote Item Location
role	string	M	Role of this place. The values that can be specified here are described by Product Specification (e.g. "INSTALL_LOCATION").	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	Quote Item Place Sub Units
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.	Quote Item Place Contact

7.2.4.2. Type PlaceRefOrQuery

Description: A place described by reference to 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	MEF 80
fieldedAddress-Representation	FieldedAddress-Representation[]	O	A list of Fielded Address representations	Installation Place Representations
formattedAddress-Representation	FormattedAddress-Representation[]	O	A list of Formatted Address representations	Installation Place Representations
geographicPoint-Representation	GeographicPoint-Representation[]	O	A list of Geographic Point Address representations	Installation Place Representations
label-Representation	Label-Representation[]	O	A list of Label Address representations	Installation Place Representations
@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	MEF 80
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

Name	Type	M/O	Description	MEF 80
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
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	MEF 80
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	MEF 80
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	MEF 80
label	string	M	The unique reference to an Geographic Address assigned by the Administrative Authority.	Installation Place Label
administrativeAuthority	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	MEF 80
href	string	O	Hyperlink to the referenced Address. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in MEF 80
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 MEF 80

7.2.4.9. Type GeographicSiteRef

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

Name	Type	M/O	Description	MEF 80
href	string	O	Hyperlink to the referenced Site. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in MEF 80
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 MEF 80

7.2.4.10. Type SubUnit

Description: Allows for sub unit identification

Name	Type	M/O	Description	MEF 80
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	MEF 80
query	string	O	This attribute is used to define to which type of events to register to. Example: "query":"eventType = quoteStateChangeEvent". To subscribe for more than one event type, put the values separated by comma: `eventType=quoteStateChangeEvent, quoteItemStateChangeEvent`. The possible values are enumerated by the 'QuoteEventType' in quoteNotification.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 Buyer Notification API (quoteNotification.api.yaml). This property is appended with the base path and notification resource path specified in that API to construct an URL to which notification is sent. E.g. for "callback": "http://buyer.mef.com/listenerEndpoint", the state change event notification will be sent to: `http://buyer.mef.com/listenerEndpoint /mefApi/sonata/quoteNotification/v10 /listener/quoteStateChangeEvent`	Return Address Information

7.2.5.2. Type EventSubscription

Description: Sets the communication endpoint address the service instance must use to deliver notification information

Name	Type	M/O	Description	MEF 80
id	string	M	Id of the listener	Not represented in MEF 80
callback	string	M	The callback being registered.	Return Address Information
query	string	O	additional data to be passed	List of Notification Types

7.2.6. Type QuoteOperationData

The `QuoteOperationData` is a common type for both Cancel or Decline requests that can be sent by using `/cancelQuote` or `/declineQuote` endpoints.

Description: Request for operation on an existing Quote (cancel or decline)

Name	Type	M/O	Description	MEF 80
reason	string	O	Allows the Buyer to specify a reason for the Cancel or Decline Quote request.	Reason

Name	Type	M/O	Description	MEF 80
quoteId	string	M	Unique (within the Seller quoting domain) identifier for the quote, as attributed by the Seller.	Seller Quote Identifier

7.2.7. Common

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

7.2.7.1. Type Duration

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

Name	Type	M/O	Description	MEF 80
amount	integer *minimum = 0*	M	Duration (number of seconds, minutes, hours, etc.)	Duration Value
units	TimeUnit	M	Time unit enumerated	Duration Unit

7.2.7.2. Type Money

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

Name	Type	M/O	Description	MEF 80
unit	string	M	Currency (ISO4217 norm uses 3 letters to define the currency)	Currency
value	float *format = float*	M	A positive floating point number	Value

7.2.7.3. Type Note

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

Name	Type	M/O	Description	MEF 80
date	date-time *format = date-time*	M	Date of the note	Note Date
author	string	M	Author of the note	Note Author
id	string	M	Identifier of the note within its containing entity (may or may not be globally unique, depending on provider implementation)	Not represented in MEF 80
source	MEFBuyerSellerType	M	Indicates if the note is from Buyer or Seller	Note source
text	string	M	Text of the note	Note Text

7.2.7.4. `enum` MEFBuyerSellerType

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

Value	MEF 80
buyer	BUYER
seller	SELLER

7.2.7.5. 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	MEF 80
number	string	M	Phone number	Contact Phone Number
emailAddress	string	M	Email address	Contact Email Address
postalAddress	FieldedAddressRepresentation	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

7.2.7.6. 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	MEF 80
role	string	M	A role of the particular contact in the request	Not represented in MEF 80
number	string	M	Phone number	Contract Phone Number

Name	Type	M/O	Description	MEF 80
emailAddress	string	M	Email address	Contact email Address
postalAddress	FieldedAddressRepresentation	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

The related contact information can be defined at a Quote or a Quote Item level. In both cases, it is allowed to provide a list of party role information. The `role` attribute is used to provide a reason the particular party information is used. It can result from MEF 80 requirements (e.g. Seller Contact Information) or from the Product Specification requirements.

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

- Seller Contact Information: `role` equal to `sellerContactInformation`
- Buyer Contact Information: `role` equal to `buyerContactInformation`
- Quote Item Technical Contact: `role` equal to `quoteItemTechnicalContact`

7.2.7.7. 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).

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

7.2.7.8. Type TimePeriod

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

Name	Type	M/O	Description	MEF 80
startDateTime	date-time *format = date-time*	O	Start of the time period, using IETC-RFC-3339 format. If you define a start, you must also define an end	Not represented in MEF 80
endDateTime	date-time *format = date-time*	O	End of the time period, using IETC-RFC-3339 format	Quote.Valid Until Date

7.2.7.9. `enum` TimeUnit

Description: Represents a unit of time.

Value	MEF 80
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 25 presents the Quote Notification data model. The data types, requirements related to them, and mapping to MEF 80 are discussed later in this section.

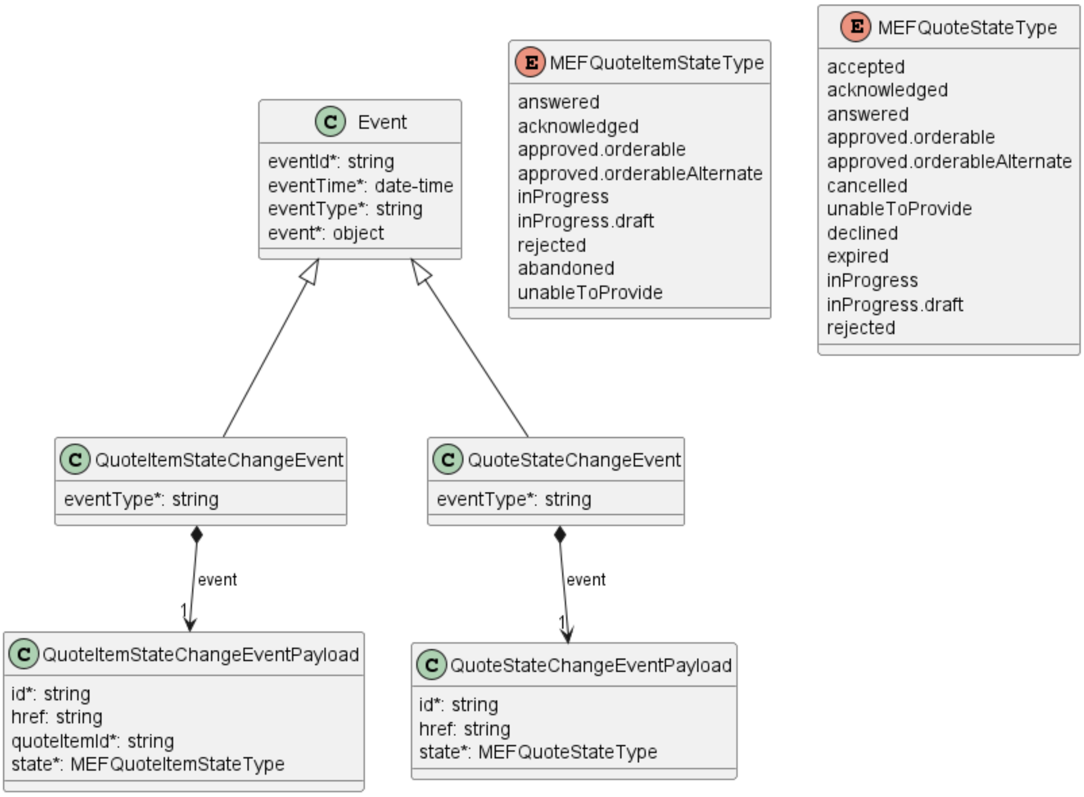


Figure 25. Quote Notification Data Model

The Quote Management data model is used to construct requests and responses of the API endpoints described in Section 5.1.2.

7.3.1. Type Event

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

Name	Type	M/O	Description	MEF 80
eventId	string	M	Id of the event	Not represented in MEF 80
eventTime	date-time *format = date-time*	M	Date-time when the event occurred	Not represented in MEF 80
eventType	string	M	The type of the notification.	Notification Type
event	object	M	The event linked to the involved resource object	Quote State, Quote Item State

7.3.2. Type QuoteStateChangeEvent

Description: QuoteStateChangeEvent structure

Inherits from:

- Event

Name	Type	M/O	Description	MEF 80

Name	Type	M/O	Description	MEF 80
eventType	string	M	Indicates the type of product offering qualification event.	Notification Type
event	QuoteStateChangeEventPayload	M	A reference to the Quote that is source of the notification.	Quote State

7.3.3. Type QuoteStateChangeEventPayload

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

Name	Type	M/O	Description	MEF 80
id	string	M	The Quote unique identifier.	Quote Identifier
href	string	O	Link to the Quote	Not represented in MEF 80
state	MEFQuoteStateType	M	The state reached at change date	Quote State

7.3.4. Type QuoteItemStateChangeEvent

Description: QuoteItemStateChangeEvent structure

Inherits from:

- Event

Name	Type	M/O	Description	MEF 80
eventType	string	M	Indicates the type of product offering qualification event.	Notification Type
event	QuoteItemStateChangeEventPayload	M	A reference to the Quote that is source of the notification.	Quote Item State

7.3.5. Type QuoteItemStateChangeEventPayload

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

Name	Type	M/O	Description	MEF 80
id	string	M	The Quote unique identifier.	Quote Identifier
href	string	O	Link to the Quote	Not represented in MEF 80
quoteItemId	string	M	ID of the Quote Item (within the Quote) which state change triggered the event	Quote Item Identifier

Name	Type	M/O	Description	MEF 80
state	MEFQuoteItemStateType	M	The state reached at change date	Quote Item State

8. References

- ISO4217 International Standards Organization ISO 4217:2015, 2015
- 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 50.1, MEF Services Lifecycle Process Flows, August 2017.
- 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 80, Quote Management Requirements and Use Cases, July 2021
- Mplify 80.0.1, Amendment to MEF 80: Quote Management 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 https://tools.ietf.org/html/rfc7231
- RFC 8174, Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words, May 2017
- TMF 630 TMF630 API Design Guidelines 4.0.1
- TMF 648 TMF648 Quote Management API REST Specification R19.0.1 Doc

Appendix A Acknowledgments

Mike **BENCHECK**

Tomasz **CHMAL**

Pankaj **BODADE**

Michał **ŁĄCZYŃSKI**

Jack **PUGACZEWSKI**

Patrick **ROOSEN**

Fahim **SABIR**