MEF Standard

MEF 115

LSO Cantata and LSO Sonata Quote Management API – Developer Guide

May 2022

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 MEF Forum (MEF) is not responsible for any errors. MEF does not assume responsibility to update or correct any information in this publication. No representation or warranty, expressed or implied, is made by MEF concerning the completeness, accuracy, or applicability of any information contained herein and no liability of any kind shall be assumed by MEF as a result of reliance upon such information.

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

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

a) any express or implied license or right to or under any patent, copyright, trademark or trade secret rights held or claimed by any MEF member which are or may be associated with the ideas, techniques, concepts or expressions contained herein; nor

b) any warranty or representation that any MEF members will announce any product(s) and/or service(s) related thereto, or if such announcements are made, that such announced product(s) and/or service(s) embody any or all of the ideas, technologies, or concepts contained herein; nor

c) any form of relationship between any MEF member and the recipient or user of this document.

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

Table of Contents

1 List of Contributing Members 1

2 Abstract 1

3 Terminology and Abbreviations 2

4 Compliance Levels 5

5 Introduction 6

5.1 Description 7
5.2 Conventions in this Document 7
5.3 Relation to Other Documents 8
5.4 Approach 8
5.5 High-Level Flow 9

6 API Description 10

6.1 High-level Use Cases 10
6.2 API Endpoint and Operation Description 10
6.2.1 Seller Side API Endpoints 10
6.2.2 Buyer Side API Endpoints 11
6.3 Specifying the Buyer ID and the Seller ID 12
6.4 Integration of Product Specifications into Quote Management API 13
6.5 Sample Product Specification 15
6.6 Model Structural Validation 17
6.7 Security Considerations 18

7 API Interaction & Flows 18

7.1 API Resource Schema Summary 19
7.1.1 Key Entities – Create Request 19
7.1.2 Key Entities – Response 21
7.2 Use Case 1: Create Quote 21
7.2.1 Use Case 1a: Immediate Quote Response Requested and Provided 22
7.2.2 Use Case 1b: Deferred Quote Response Requested and Provided 23
7.2.3 Use Case 1c: Deferred Quote Response Requested, Immediate Quote Response Provided 25
7.2.4 Buyer's Quote Request 26
7.2.5 Seller's Response to a Create Quote Request 28
7.2.6 Quote Item Specification Details 32
7.2.7 Specifying Place Details 39
7.3 Use Case 2: Retrieve Quote List 42
7.4 Use Case 3: Retrieve Quote by Quote Identifier 43
7.5 Use Case 4: Cancel Quote by Quote Identifier 47
7.6 Use Case 5: Decline Quote by Quote Identifier 48
7.7 Use Case 6: Register for Quote Notifications 48
7.8 Use Case 7: Send Quote Notification 50

8 API Details 54

8.1 API Patterns 54
8.1.1 Indicating Errors 54
8.1.2 Response Pagination 57

8.2 Management API Data Model 58
8.2.1 Quote 58
8.2.2 Quote Item 66
8.2.3 Product Representation 74
8.2.4 Place Representation 76
8.2.5 Notification Registration 80
8.2.6 Type QuoteOperationData 81
8.2.7 Common 82
8.3 Notification API Data Model 84
8.3.1 Type Event 85
8.3.2 Type QuoteEvent 85
8.3.3 enum QuoteEventType 85
9 References 86

List of Figures

Figure 1 – The LSO Reference Architecture 6
Figure 2 – Cantata and Sonata API Framework 8
Figure 3 – Cantata and Sonata End-to-End Function Flow 9
Figure 4 – Use Cases 10
Figure 5 – The Extension Pattern with Sample Product Specific Extensions 15
Figure 6 – A Simplified View on the Access E-Line Product Specification Data Model 16
Figure 7 – A Simplified View on the UNI Product Specification Data Model 17
Figure 8 – Key Entities – Create Request 20
Figure 9 – Key Entities – Response 21
Figure 10 – Use Case 1a: Immediate Quote Response Requested and Provided 22
Figure 11 – Use Case 1a: Immediate Quote Response Requested and Provided with Notification 23
Figure 12 – Use Case 1b: Deferred Quote Response Requested and Provided – Polling Pattern 24
Figure 13 – Use Case 1b: Deferred Quote Response Requested and Provided – Notification Pattern 24
Figure 14 – Use Case 1c: Deferred Quote Response Requested, Immediate Quote Response Provided 25
Figure 15 – Use Case 1c: Deferred Quote Response Requested, Immediate Quote Response Provided, with Notifications 26
Figure 16 – Use Case 4, 5: Cancel or Decline Quote by Quote Identifier 47
Figure 17 – Use Case 1b: Deferred Quote Response Requested and Provided with QuoteItem Notifications 51
Figure 18 – Use Case 6, 7: Register and Send Notifications 53
Figure 19 – Data Model Types to Represent an Erroneous Response 54
Figure 20 – Quote Management Data Model 58
Figure 21 – Quote Firm Flow Activity Diagram 63
Figure 22 – Quote Budgetary Flow Activity Diagram 64
Figure 23 – Quote Item Firm Quote Flow Activity 70
Figure 24 – Quote Item Budgetary Quote Flow Activity 70
Figure 25 – Data Model Types Representing a Place 76
Figure 26 – Quote Notification Data Model 84

List of Tables

Table 1 – Terminology and Abbreviations 4
Table 2 – Seller Side API Endpoints 11
Table 3 – Buyer Side API Endpoints 12
Table 4 – Use Case Descriptions 19
Table 5 – Seller Response to Query by ID, FIRM Quote Level, Quote Attributes 44
Table 6 – Seller Response to Query by ID, FIRM Quote Level, QuoteItem Attributes 44
Table 7 – Seller Response to Query by ID, BUDGETARY Quote Level, Quote Attributes 45
Table 8 – Seller Response to Query by ID, BUDGETARY Quote Level, QuoteItem Attributes 45
Table 9 – Type Error 54
Table 10 – Type Error400 55
Table 11 – Type Error401 55
Table 12 – Type Error403 55
Table 13 – Type Error404 56
Table 14 – Type Error422 56
Table 15 – Type Error500 57
Table 16 – Type Error501 57
Table 17 – Type Quote_Common 59
Table 18 – Type Quote_Create 59
Table 19 – Type Quote 60
Table 20 – enum MEFQuoteStateType 62
Table 21 – Allowable Quote Item States per Quote State for FIRM Quote Level 64
Table 22 – Allowable Quote Item States per Quote State for BUDGETARY Quote Level 65
Table 23 – enum MEFBuyerQuoteLevel 65
Table 24 – enum MEFSellerQuoteLevel 66
Table 25 – Type MEFQuoteStateChange 66
Table 26 – Type MEFQuoteItem_Common 67
Table 27 – Type QuoteItem 68
Table 28 – enum MEFProductActionType 68
Table 29 – enum MEFQuoteItemStateType 69
Table 30 – Type ProductOfferingQualificationItemRef 71
Table 31 – Type ProductOfferingRef 71
Table 32 – Type QuoteItemRelationship 72
Table 33 – Type MEFItemTerm 72
Table 34 – enum MEFEndOfTermAction 72
Table 35 – Price Type Required Information 73
Table 36 – Type QuotePrice 73
Table 37 – Type Price 73
Table 38 – enum MEFPriceType 74
Table 39 – enum MEFChargePeriod 74
Table 40 – Type MEFProductRefOrValueQuote 74
Table 41 – Type MEFProductConfiguration 75
Table 42 – Type ProductRelationshipWithGrouping 75
Table 43 – Type RelatedPlaceRefOrValue 77
Table 44 – Type FieldedAddress 77
Table 45 – Type FormattedAddress 78

Table 46 – Type MEFGeographicPoint 78
Table 47 – Type GeographicSubAddress 79
Table 48 – GeographicAddressRef 79
Table 49 – GeographicSiteRef 79
Table 50 – GeographicAddressLabel 80
Table 51 – Type MEFSubUnit 80
Table 52 – Type EventSubscriptionInput 81
Table 53 – Type EventSubscription 81
Table 54 – Type QuoteOperationData 81
Table 55 – Type Duration 82
Table 56 – Type Money 82
Table 57 – Type Note 82
Table 58 – Type MEFBuyerSellerType 82
Table 59 – Type RelatedContactInformation 83
Table 60 – Type TerminationError 83
Table 61 – Type TimePeriod 84
Table 62 – Type TimeUnit 84
Table 63 – Type Event 85
Table 64 – Type QuoteEvent 85
Table 65 – Type QuoteEventType 85

1 List of Contributing Members

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

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

2 Abstract

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 [11]. 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: 415ef5ad60d07cf6bc87c36c684217e98cb9936e

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

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

commit id: 6a27c0b1a237753ea234a6b4ad10798da837d2e6

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

3 Terminology and Abbreviations

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

Term	Definition	Reference
Application Program Interface (API)	In the context of LSO, API describes one of the Management Interface Reference Points based on the requirements specified in an Interface Profile, along with a data model, the protocol that defines operations on the data and the encoding format used to encode data according to the data model. In this document, API is used synonymously with REST API.	MEF 55.1 [9]
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 [11]
Buyer	In the context of this document, denotes the organization or individual acting as the customer in a transaction over a Cantata (Customer ↔ Service Provider) or Sonata (Service Provider ↔ Partner) Interface.	This document, adapted from MEF 80 [11]

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 [7].	ISO 4217:2015 [7]
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 [11]

Term	Definition	Reference
Firm Quote	A quote provided to the Buyer based on a pre-order analysis. All 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.	MEF 80 [11]
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 cost identified during delivering 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 [11]
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 [11]
Recurring Charge	Charge for a product that is incurred by the Buyer each specified time interval for that product.	MEF 80 [11]
Requesting Entity	The business organization that is acting on behalf of one or more Buyers. In the most common case, the Requesting Entity represents only one Buyer and these terms are then synonymous.	MEF 79 [10]
Responding Entity	The business organization that is acting on behalf of one or more Sellers. In the most common case, the Responding Entity represents only one Seller and these terms are then synonymous.	MEF 79 [10]
Representational State Transfer Application Program Interface	REST provides a set of architectural constraints that, when applied as a whole, emphasizes scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems.	Fielding [8]
REST API	Representational State Transfer Application Program Interface	Fielding [8]

Term	Definition	Reference
Seller	In the context of this document, denotes the organization acting as the supplier in a transaction over a Cantata (Customer ↔ Service Provider) or Sonata (Service Provider ↔ Partner) Interface.	This document; adapted from MEF 80 [11]

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:
	`cancelled`
	`unableToProvide`
	`declined`
	`expired`
	`rejected`
		This document

Table 1 – Terminology and Abbreviations

4 Compliance Levels

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 (RFC 2119 [2], RFC 8174 [6]) when, and only when, they appear in all capitals, as shown here. All key words must be in bold text.

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

5 Introduction

This standard specification document describes the Application Programming Interface (API) for Product 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* [9]. The LSO Reference Architecture is shown in Figure 1 with both IRPs highlighted.

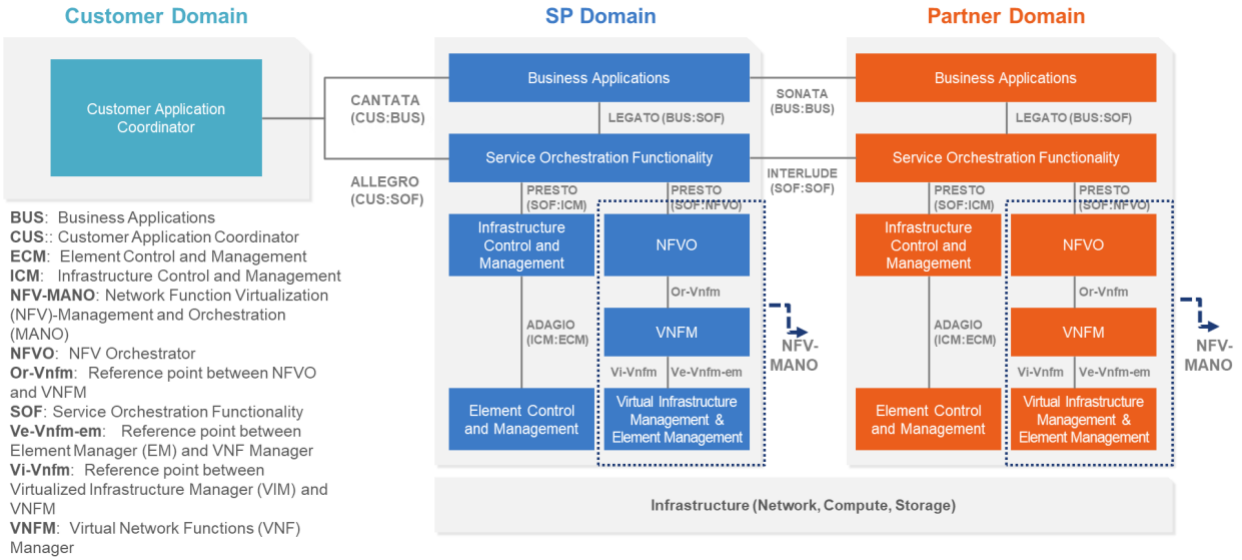

Figure 1 – The LSO Reference Architecture

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

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

The business requirements and use cases for Quote Management are defined in MEF 80 *Quote Management Requirements and Use Cases* [11].

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

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

5.1 Description

The 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. Product Offerings are defined in MEF 79 *Address, Service Site, and Product Offering Qualification Management, Requirements and Use Cases* [10].

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

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

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

Product specifications are defined using JSON Schema (draft 7) standard [1], whereas Quote API is defined using OpenAPI 3.0 [12]. The payloads exchanged through Quote endpoints must comply with the Product specification schema as well as with MEF 80 [11] requirements for Quote Management.

5.2 Conventions in this Document

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

5.3 Relation to Other Documents

The requirements and use cases for Quote Management are defined in MEF 80 [11]. The API definition builds on *TMF648 Quote Management API REST Specification R19.0.1* [14]. Quote Use Cases must support the use of any of MEF product specifications.

5.4 Approach

As presented in Figure 2, both Cantata and Sonata API frameworks consists of three structural components:

- Generic API framework
- Product-independent information (Function-specific information and Function-specific operations)
- Product-specific information (MEF product specification data model)

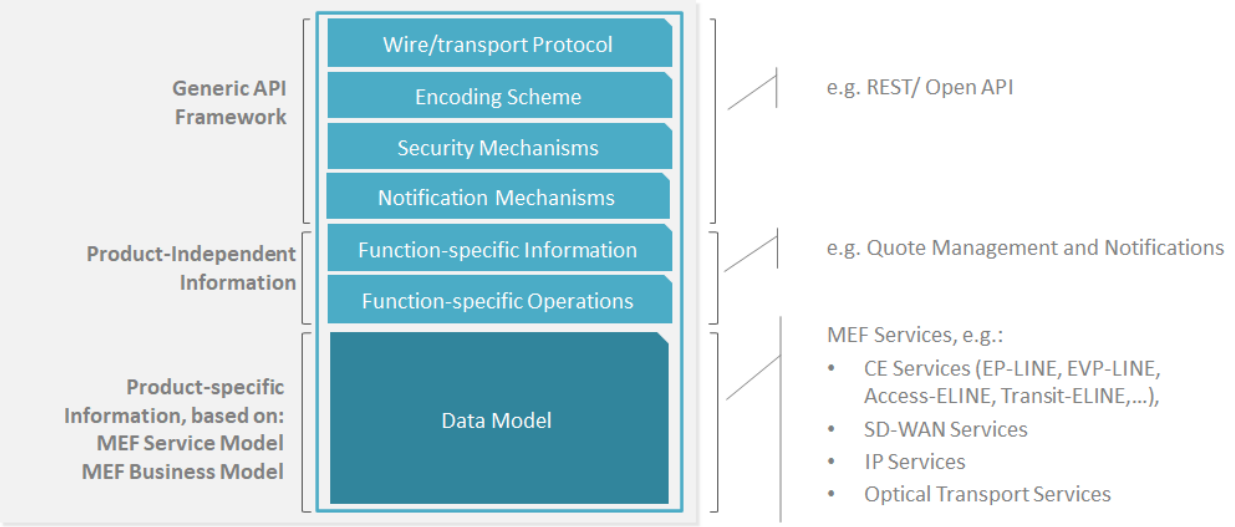

Figure 2 – Cantata and Sonata API Framework

The essential concept behind the framework is to decouple the common structure, information, and operations from the specific product information content. Firstly, the Generic API Framework defines a set of design rules and patterns that are applied across all Cantata or Sonata APIs. Secondly, the product-independent information of the framework focuses on a model of a particular Cantata or Sonata functionality and is agnostic to any of the product specifications. For example, this standard is describing the Quote model and operations that allow performing quoting of any product that is aligned with either MEF or custom product specifications. Finally, the product-specific information part of the framework focuses on MEF product specifications that define business-relevant attributes and requirements for trading MEF subscriber and MEF operator services.

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

5.5 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.

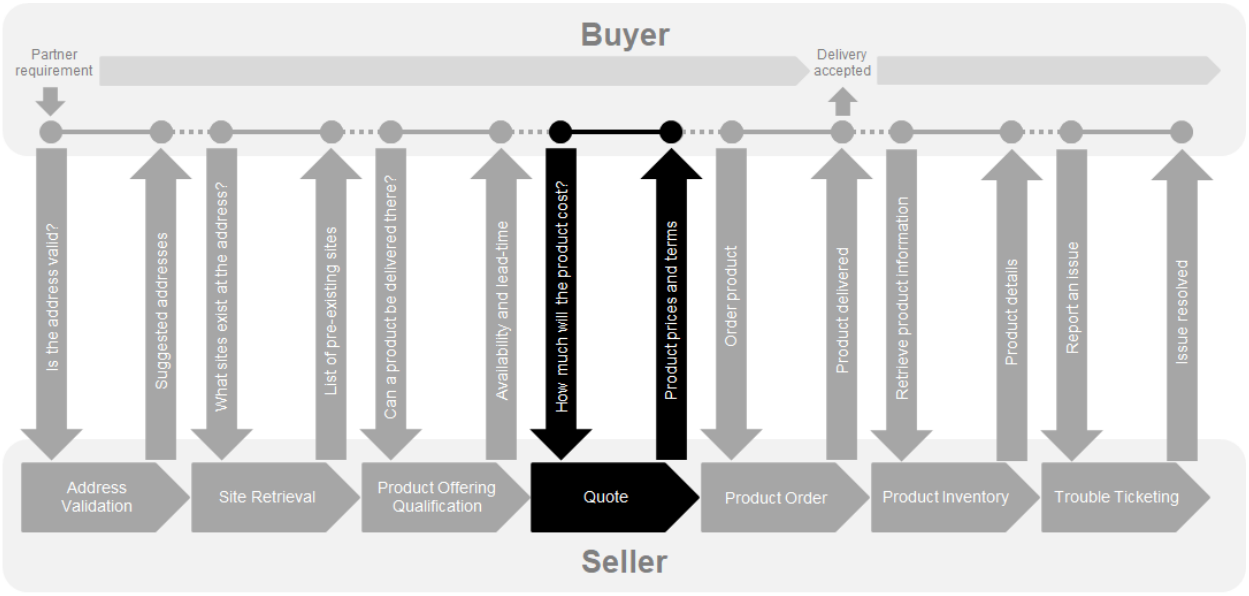

Figure 3 – Cantata and Sonata End-to-End Function Flow

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

- Allows the Buyer to create, retrieve, and update Trouble Tickets as well as receive notifications about Incidents' and Trouble Tickets' updates. This allows managing issues and situations that are not part of normal operations of the Product provided by the Seller.

6 API Description

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

6.1 High-level Use Cases

Figure 4 presents a high-level use case diagram as specified in MEF 80 [11] in Section 7.2. This picture aims to help understand the endpoint mapping. Use cases are described extensively in Section 7.

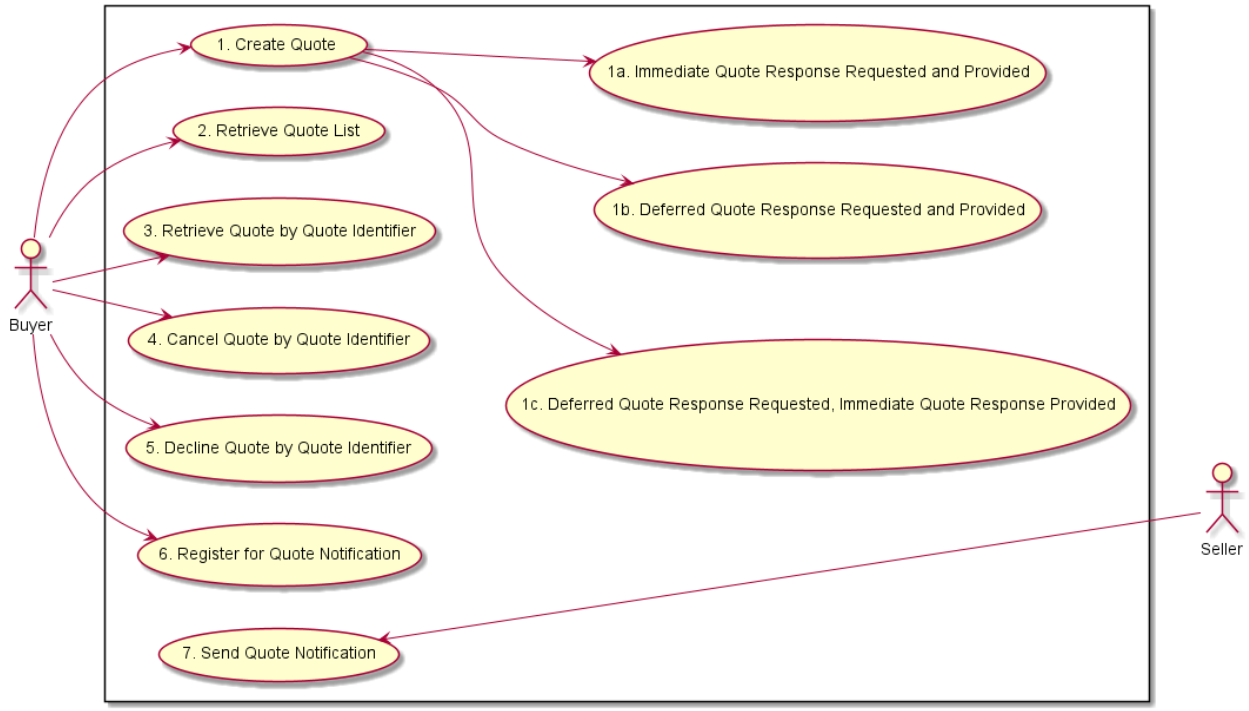

Figure 4 – Use Cases

6.2 API Endpoint and Operation Description

6.2.1 Seller Side API Endpoints

Base URL for Cantata:

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

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

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.	Use Case 1: Create Quote Use Case 1a: Immediate Quote Response Requested and Provided Use Case 1b: Deferred Quote Response Requested and Provided Use Case 1c: Deferred Quote Response Requested, Immediate Quote Response Provided
`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.	Use Case 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.	Use Case 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.	Use Case 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.	Use Case 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.	Use Case 6: Register for Quote Notifications
`DELETE /hub/{{id}}`	A request initiated by the Buyer to instruct the Seller to stop sending notifications.	Use Case 6: Register for Quote Notifications

Table 2 – Seller Side API Endpoints

6.2.2 Buyer Side API Endpoints

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

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

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 on `Quote` state change.	Use Case 7: Send Quote Notification
`POST /listener/quoteItemStateChangeEvent`	A request initiated by the Seller to notify the Buyer on `QuoteItem` state change.	Use Case 7: Send Quote Notification

Table 3 – Buyer Side API Endpoints

6.3 Specifying the Buyer ID and the Seller ID

A business entity willing to represent multiple Buyers or multiple Sellers must follow requirements of MEF 79 [10] Section 8.8, which states:

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

In the API the `buyerId` and `sellerId` are represented as query parameters in each operation defined in `quoteManagement.api.yaml` and as attributes of an event (`QuoteEvent`) as described in `quoteNotification.api.yaml`.

[R1] If the Requesting Entity has the authority to represent more than one Buyer the request **MUST** include the `buyerId` query parameter that identifies the Buyer being represented. MEF 79 R80 [10].

[R2] If the Requesting Entity represents precisely one Buyer with the Responding Entity, the request **MUST NOT** specify the `buyerId`. MEF 79 R81 [10].

[R3] If the Responding Entity represents more than one Seller to this Buyer the request **MUST** include the `sellerId` query parameter that identifies the Seller with whom this request is associated. MEF 79 R82 [10].

[R4] If the Responding Entity represents precisely one Seller to this Buyer, the request **MUST NOT** specify the `sellerId`. MEF 79 R83 [10].

[R5] If the `buyerId` or `sellerId` attributes were specified in the request, the same attributes **MUST** be used in the notification payload.

6.4 Integration of Product Specifications into Quote Management API

Product specifications are defined using JSON Schema (draft 7) [1] format and are integrated into the `Quote` payload using the TMF extension pattern.

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

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

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

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

Product-specific attributes are introduced via the `MEFProductRefOrValue` (defined by the Buyer). This entity has the `productConfiguration` attribute of type `MEFProductConfiguration` which is used as an extension point for product-specific attributes.

Implementations might choose to integrate selected product specifications to data model during development. In such a case an integrated data model is built and product specifications are in an inheritance relationship with `MEFProductConfiguration` as described in the OAS specification [12]. This pattern is called **Static Binding**. The SDK is additionally shipped with a set of API definitions that statically bind all product-related APIs (POQ, Quote, Order, Inventory) with all corresponding product specifications available in the release. The snippets below present an example of a static binding of the envelope API with a number of MEF product specifications, from both the `MEFProductConfiguration` and the product specification point of view:

```
MEFProductConfiguration:
  description:
    MEFProductConfiguration is used as an extension point for MEF specific
```

```
    product/service payload. The `@type` attribute is used as a discriminator
  discriminator:
    mapping:
      urn:mef:lso:spec:sonata:AccessElineOvc:v1.0.0:quote:
'#/components/schemas/AccessElineOvcQuote_v1.0.0'
      urn:mef:lso:spec:cantata-sonata:SubscriberUni:v1.0.0:quote:
'#/components/schemas/SubscriberUniQuote_v1.0.0'
      urn:mef:lso:spec:cantata-sonata:EplEvc:v1.0.0:quote:
'#/components/schemas/EplEvcQuote_v1.0.0'
      urn:mef:lso:spec:sonata:OperatorUNI:v1.0.0:quote:
'#/components/schemas/OperatorUNIQuote_v1.0.0'
    propertyName: '@type'
  properties:
    '@type':
      description:
        The name of the type, defined in the JSON schema specified above, for
        the product that is the subject of the POQ Request. The named type must
        be a subclass of MEFProductConfiguration.
      type: string

AccessElineOvcQuote_v1.0.0:
  allOf:
    - $ref: '#/components/schemas/MEFProductConfiguration'
    - description:
        OVC Service Attributes control the behavior observable at and between
        External Interfaces to the Carrier Ethernet Network (CEN). The
        behaviors are achieved by the Network Operator and the Operator's
        client (the Service Provider in this case) agreeing on the value for
        each of the Service Attributes.
```

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

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

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

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

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

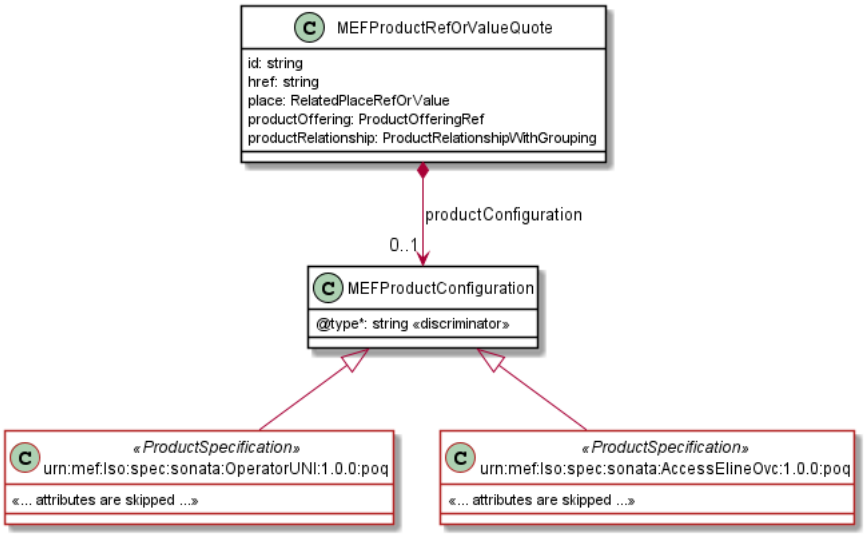

Figure 5 – The Extension Pattern with Sample Product Specific Extensions

Figure 5 presents two MEF `<<ProductSpecifications>>` that represent Access E-Line Operator UNI and OVC products. When these products are used as a Quote payload the `@type` of `MEFProductConfiguration` takes the `"urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:quote"` or `"urn:mef:lso:spec:sonata:OperatorUNI:1.0.0:quote"` value to indicate which product specification should be used to interpret a set of product-specific attributes included in the payload. An example of a product definition inside the `QuoteItem` is presented in Section 7.2.6.1.

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

6.5 Sample Product Specification

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

```
\productSchema\carrierEthernet\accessEline\quote\accessElineOvc.yaml
\productSchema\carrierEthernet\carrierEthernetOperatorUni\quote\carrierEthernetOperatorUni.yaml
```

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

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

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

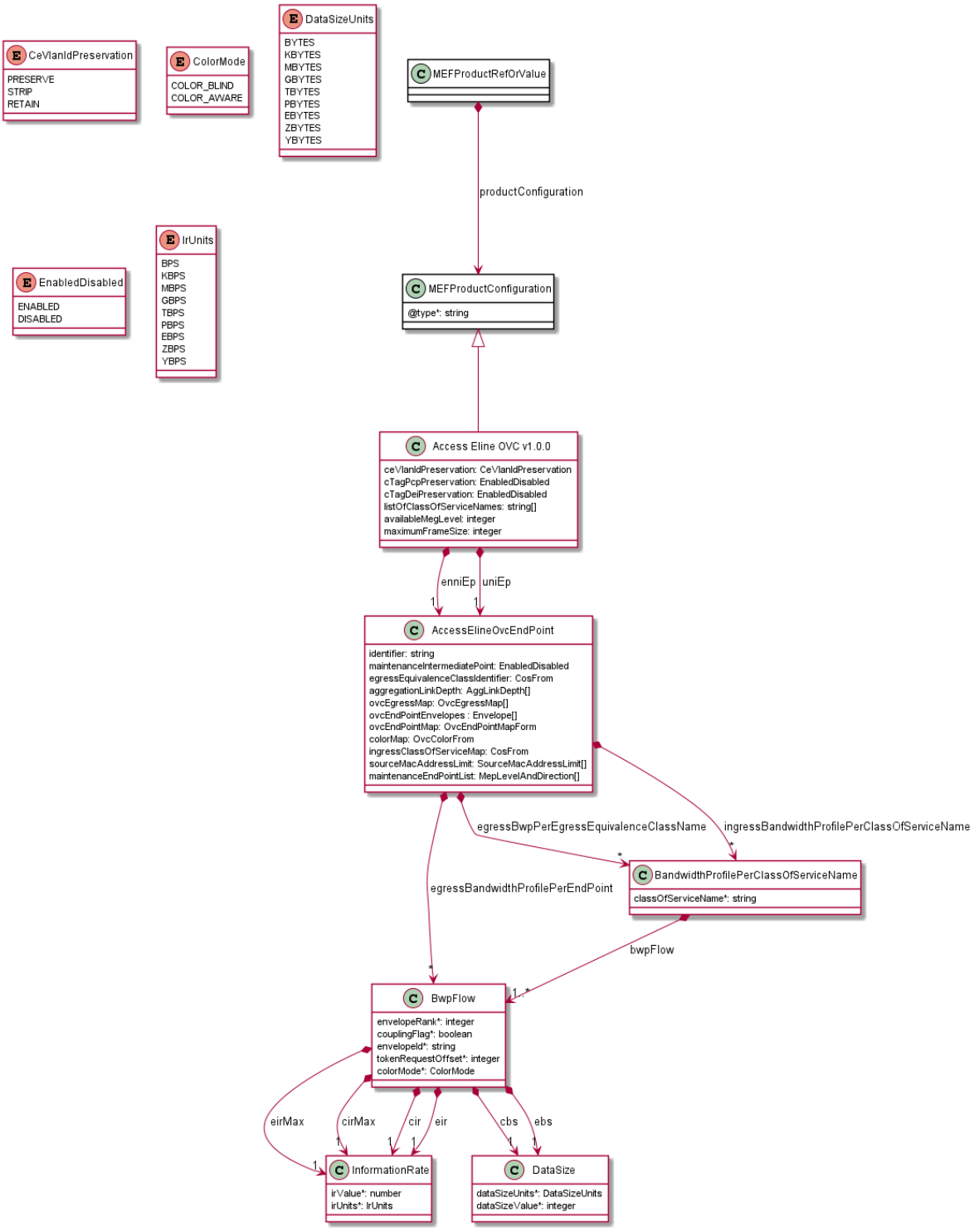

Figure 6 – A Simplified View on the Access E-Line Product Specification Data Model

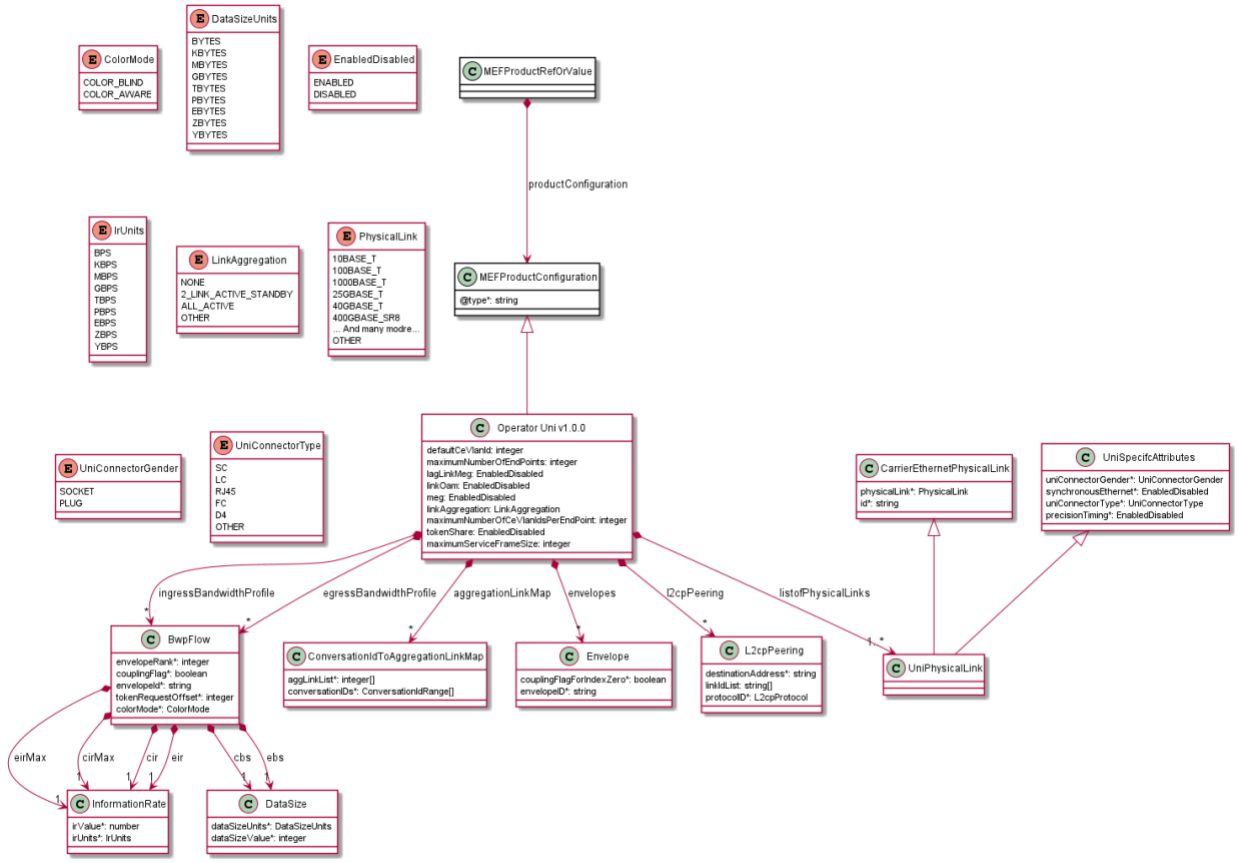

Figure 7 – A Simplified View on the UNI Product Specification Data Model

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

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

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

6.6 Model Structural Validation

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

- OpenAPI version 3.0 [12] for the product-agnostic part of the payload
- JSON Schema (draft 7) [1] for the product-specific part of the payload

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

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

- required relation type, multiplicity to other 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 MEF 80 R23 [11]

6.7 Security Considerations

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

7 API Interaction and Flows

This section provides a detailed insight into the API functionality, use cases, and flows. It starts with Table 4 presenting a list and short description of all business use cases then presents the variants of end-to-end interaction flows, and in following subsections 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 4 – Use Case Descriptions

The detailed business requirements of each of the use cases are described in Sections 7.2 and 8 of MEF 80 [11].

7.1 API Resource Schema Summary

This subsection 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 the `allOf` keyword for data types composition). A simple type defines a set of properties that might be of an object, primitive, or reference type.

[R11] If an entity is used in the request or response payload, all properties marked as required **MUST** be provided.

Section 7 provides examples of data model and API usage. For a detailed description and complete definition of the data model, please refer to Section 8.

7.1.1 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 6.2.1 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.

[R12] `Quote_Create` is the root entity of a quote request. It **MUST** contain one or more `QuoteItem_Create` MEF 80 R12 [11].

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(s) being subject of quotation (in a `MEFProductRefOrValue` structure) and allows for the definition of additional information like related parties (`RelatedContactInformation`) or relations to other items (`QuoteItemRelationship`).

`MEFProductRefOrValue` allows for the introduction of MEF product-specific properties to the Quote payload. The extension mechanism is described in detail in Section 6.4. `MEFProductRefOrValue` may be also used to specify relations to places (using specializations of `RelatedPlaceOrValue`) and/or to a product that exists in the Seller's inventory (using `ProductRelationship`).

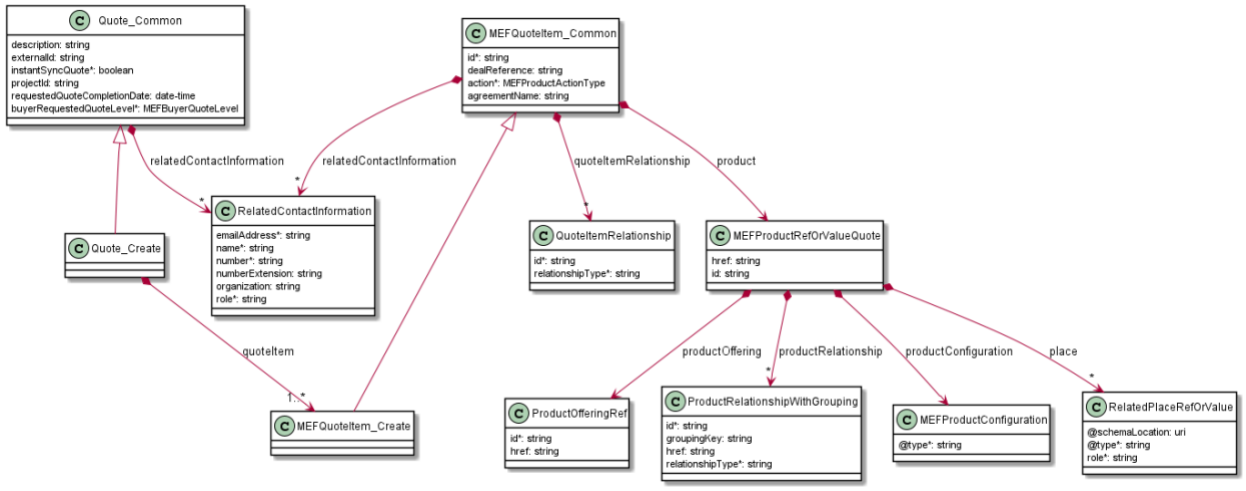

Figure 8 – Key Entities – Create Request

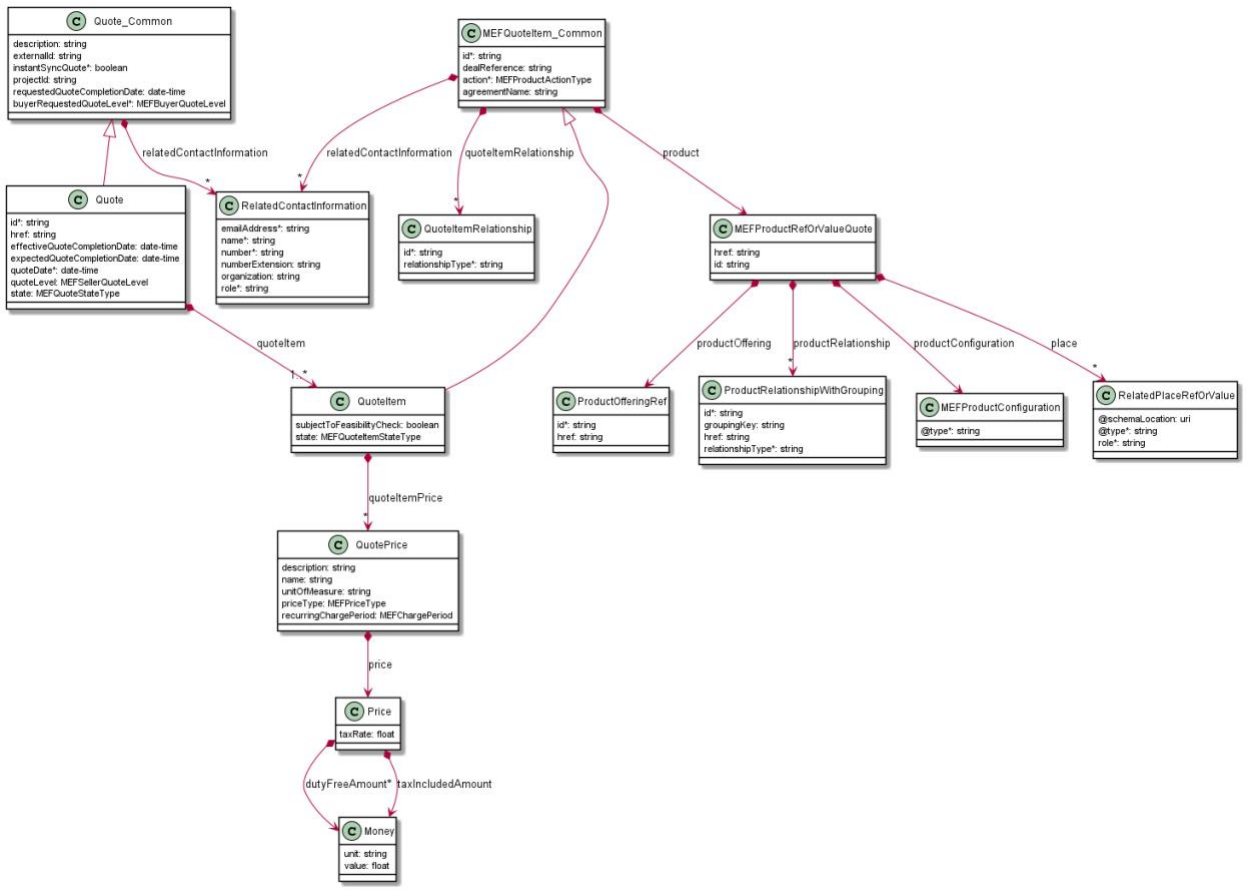

Figure 9 – Key Entities – Response

7.1.2 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.

[R13] 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.

7.2 Use Case 1: Create Quote

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

- The Seller responds immediately with the final results of the processing. This is called an **Immediate Quote Response**.
- 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 subsections.

7.2.1 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.

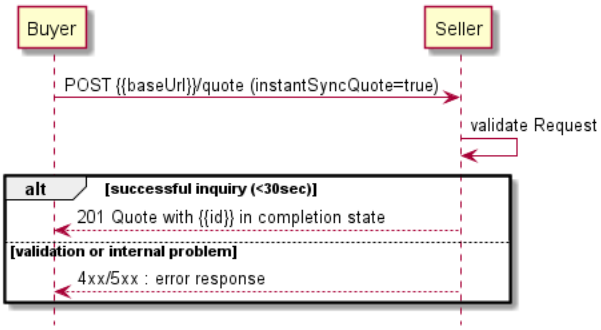

Figure 10 – Use Case 1a: Immediate Quote Response Requested and Provided

The figure above presents the basic synchronous use case flow while the one below adds the context when the Buyer decides to use the additional asynchronous 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`).

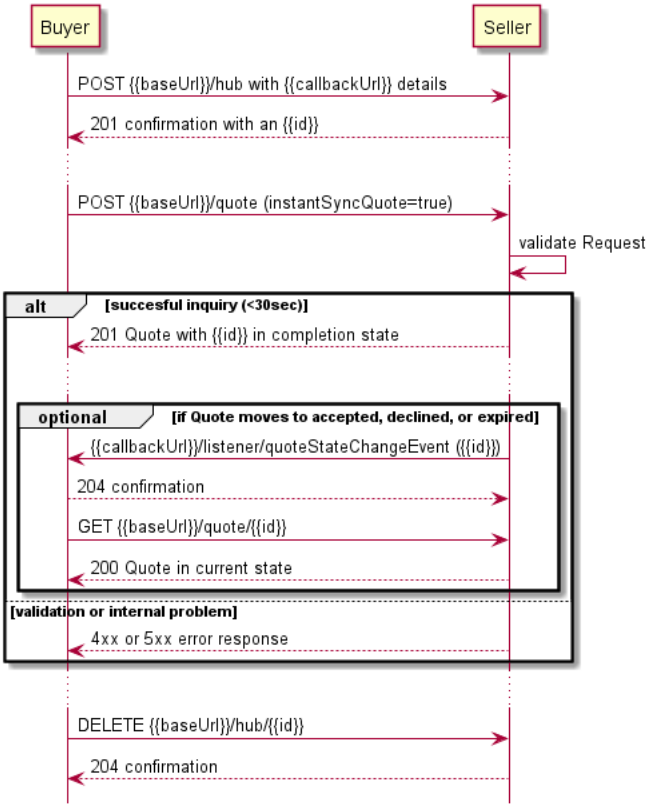

Figure 11 – Use Case 1a: Immediate Quote Response Requested and Provided with Notification

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 to also to all further use case flow diagrams with notifications.

[R14] 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. MEF 80 R11 [11]

7.2.2 Use Case 1b: Deferred Quote Response Requested and Provided

A deferred quotation can be requested by using `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.

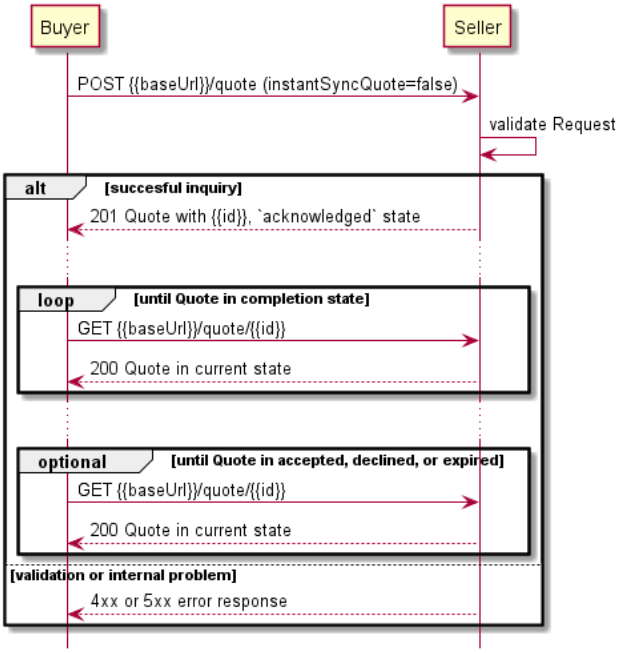

Figure 12 – Use Case 1b: Deferred Quote Response Requested and Provided – Polling Pattern

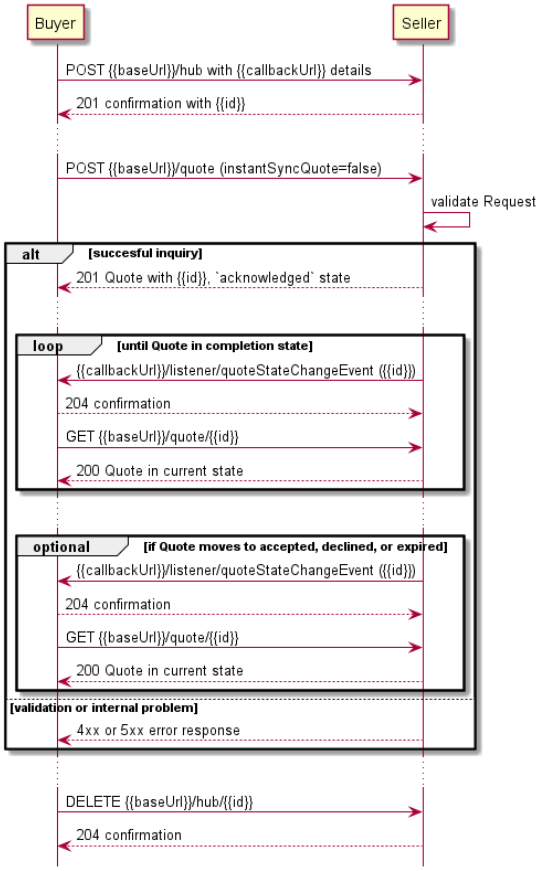

Figure 13 – Use Case 1b: Deferred Quote Response Requested and Provided – Notification Pattern

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 12), 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.

To use the notifications mechanism (Figure 13), 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. The state change is an update from the state (all attributes are considered, not only the `Quote.state`) that was sent as a first response to a Create Quote request.

7.2.3 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.

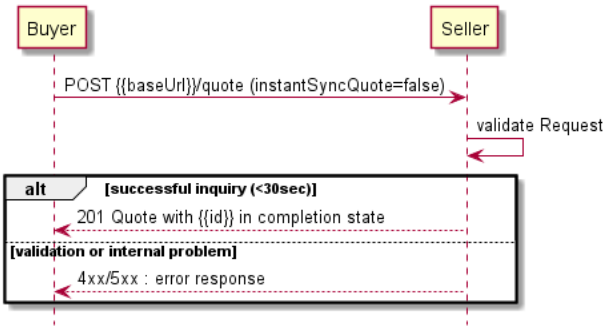

Figure 14 – Use Case 1c: Deferred Quote Response Requested, Immediate Quote Response Provided

Figure 14 presents the case where the Buyer didn't register for Quote Notifications.

Figure 15 presents the interaction between Buyer and 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 will reach a terminal state (e.g. `expired`).

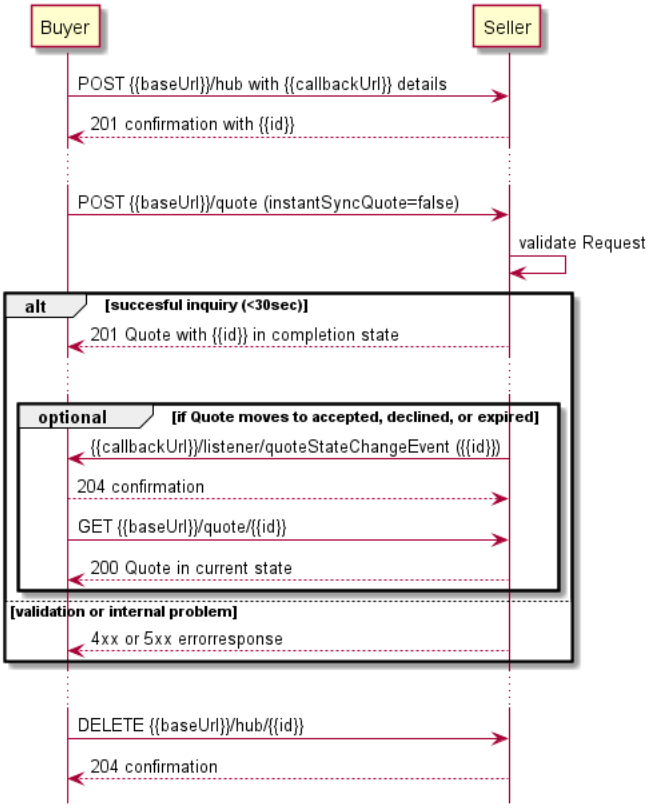

Figure 15 – Use Case 1c: Deferred Quote Response Requested, Immediate Quote Response Provided, with Notifications

7.2.4 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 the examples' readability. The full list of attributes is available in Section 8 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": "john.example@example.com",
      "name": "John Example",
      "number": "12-345-6789",
      "numberExtension": "1234",
      "role": "buyerContactInformation"
    }
  ]
}
```


The Buyer **MUST** specify:

[R15] `buyerRequestedQuoteLevel` – to inform whether `budgetary` or `firm` quotation level is required (MEF 80 R13 [11]). Please refer to the glossary for more details.

[R16] `instantSyncQuote` – to request the *Immediate* (`true`) or *Deferred* (`false`) response (MEF 80 R13 [11]).

[R17] at least one `quoteItem` (MEF 80 R13 [11]).

[R18] if `instantSyncQuote` equals `false` the `relatedContactInformation[]` with an item of `role` equal to `buyerContactInformation` to specify the required *Buyer Contact Information* (MEF 80 R16 [11]).

[R19] if `instantSyncQuote` equals `false` the `requestedQuoteCompletionDate` (MEF 80 R17 [11]) to set the deadline for the Seller to provide the pricing. If `instantSyncQuote` equals `true`, this is ignored by the Seller (MEF 80 R18 [11]).

[O1] The Seller **MAY** decide to make the `productOfferingQualificationItem` mandatory for a Buyer Create Quote request (MEF 80 O5 [11]).

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.

For every `QuoteItem` the Buyer **MUST** specify:

[R20] `id` – to identify `quoteItem` locally within a `Quote`. For example, a sequence number (`01`, `02`, `03`...) (MEF 80 R14 [11]).

[R21] `action` – to specify what kind of operation on a product is to be quoted (MEF 80 R14 [11]).

Note: The values correspond to `orderItem.action` that will be set during ordering of quoted product(s).

[R22] `product` – a relation to an instance of a product or product configuration (MEF 80 R13 [11]).

[R23] if `instantSyncQuote` equals `false` the `relatedContactInformation[]` with an item of `role` equal to `quoteItemTechnicalContact` to specify the required Quote Item Technical Contact Information (MEF 80 R15 [11]).

[R24] if `instantSyncQuote` equals `false` and the Quote Item requires a location, `relatedContactInformation[]` with an item of `role` equal to `quoteItemLocationContact` to specify the required Quote Item Location Contact Information (MEF 80 R24 [11]).

[R25] The `QuoteItem` content **MUST** follow the product specification related requirements when specifying values for `relatedContactInformation` and `quoteItemRelationship` attributes.

7.2.5 Seller's Response to a Create Quote Request

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

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": [
  {
    "state": "approved.orderable",
    "subjectToFeasibilityCheck": false,
    "id": "item-001",
    "action": "add",
    "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"
    },
    "quoteItemTerm": {
      "duration": {
        "amount": 12,
        "units": "calendarMonths"
      },
      "endOfTermAction": "autoRenew",
      "name": "Yearly Subscription"
    },
    "quoteItemPrice": [
      {
        "name": "Monthly Plan 25",
        "priceType": "recurring",
        "recurringChargePeriod": "month",
```

```
          "price": {
            "taxRate": 16,
            "dutyFreeAmount": {
              "unit": "EUR",
              "value": 25,
            },
            "taxIncludedAmount": {
              "unit": "EUR",
              "value": 29,
            },
          },
        }
      ],
    }
  ],
  "relatedContactInformation": [
    {
      "emailAddress": "john.example@example.com",
      "name": "John Example",
      "number": "12-345-6789",
      "role": "buyerContactInformation"
    },
    {
      "emailAddress": "kate.example@example.com",
      "name": "Kate Example",
      "number": "12-345-67890",
      "role": "sellerContactInformation"
    }
  ]
  "validFor": {
    "endDateTime": "2020-08-17T16:45:39.422Z",
  }
}
```


[R26] As mentioned earlier, the Seller **MUST NOT** change the values of attributes specified by the Buyer (MEF 80 R36 [11]).

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

In response, the Seller **MUST** provide:

[R27] `id` (MEF 80 R35 [11]),

[R28] `state` – one of the states (MEF 80 R44 [11]),

[R29] `quoteDate` (MEF 80 R39 [11]),

[R30] *Seller Contact Information* by adding a `relatedContactInformation` with `role` equal to `sellerContactInformation` (MEF 80 R40 [11]),

[R31] `quoteItem.state` (MEF 80 R46 [11]),

[R32] `quoteItem.quoteItemTerm` (MEF 80 R58 [11]),

If the Seller provides a successful response with `Quote` in completion state:

[R33] when the Buyer request a `budgetary` level, the Seller **MUST** respond with `quoteLevel` equal to `budgetary` (MEF 80 R42 [11]),

[R34] when the Buyer request a `firm` level, the Seller **MUST** respond with `quoteLevel` equal to `firm` or `firmSubjectToFeasibilityCheck` (MEF 80 R40 [11]),

[R35] if the `quoteLevel` is `firmSubjectToFeasibilityCheck`, the Seller **MUST** specify the `subjectToFeasibilityCheck` equal to `true` attribute in their response for at least one `quoteItem` (MEF 80 R46 [11]),

[R36] `stateChange` – to specify the history of the `Quote`'s state transitions.

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

[R37] 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 (MEF 80 R61 [11]).

[R38] If the Seller's Quote specifies `endOfTermAction` equal to `autoRenew` or `autoDisconnect`, then the Seller **MUST NOT** specify the `rollInterval` for that Quote Item (MEF 80 R62 [11]).

[R39] 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 (MEF 80 R63 [11]).

[R40] 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` (MEF 80 R59 [11]).

[O2] The `quoteItemTerm.duration` specified by the Seller in their Quote **MAY** be greater than, equal to, or less than the Buyer's `requestedQuoteItemTerm.duration` (MEF 80 O13 [11]).

[R41] 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` (MEF 80 R60 [11]).

7.2.6 Quote Item Specification Details

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

7.2.6.1 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:AccessElineOvc:1.0.0:quote`).

```
{
  <<Quote attributes...>>
  "quoteItem": [
    {
      "id": "item-001",
      "action": "add",
      ...
      "product": {
        "@type": "MEFProductRefOrValue",
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:quote",
          "enniEp": {
            "ingressBandwidthProfilePerClassOfServiceName": [
                {
                    "classOfServiceName": "silver",
                    "bwpFlow": [
                        {
                            "envelopeRank": 1,
                            "couplingFlag": false,
                            "envelopeName": "defaultENNI",
                            "tokenRequestedOffset": 0,
                            "colorMode": "COLOR_BLIND",
                            "cir": {
                                "irValue": 20,
                                "irUnits": "MBPS"
                            },
                            "cbs": {
                                "dataSizeValue": 50,
                                "dataSizeUnits": "KBYTES"
                            },
                            "eir": {
                                "irValue": 0,
                                "irUnits": "BPS"
                            },
                            "ebs": {
                                "dataSizeValue": 0,
                                "dataSizeUnits": "BYTES"
```

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

```
                            "irUnits": "BPS"
                          },
                        }
                      ]
                    }
                ]
              }
            },
            "productOffering": {
              "id": "000073"
            },
            "productRelationship": [
              {
                "relationshipType": "ENNI_REFERENCE",
                "id": "00000000-0000-000a-0000-000000000001"
              }
            ]
          },
          "quoteItemRelationship": [
            {
              "relationshipType": "UNI_REFERENCE",
              "id": "item-002"
            }
          ],
          "relatedContactInformation": [
            {
              "number": "1-234-567-890",
              "emailAddress": "john@example.com",
              "role": "buyerContactInformation",
              "name": "John Example"
            }
          ]
        },
        {
          "id": "item-002",
          "action": "add"
          <<Quote Item with UNI Product configuration that the E-Line OVC refers to>>
        }
      ]
    }
```


[R42] `MEFProductConfiguration` **MUST** be provided in the payload if an item `action` is set to `add` (MEF 80 R21 [11]).

[R43] `productOffering` – **MUST** be provided in the payload if an item `action` is set to `add` (MEF 80 R21 [11]).

[R44] The Buyer **MUST NOT** specify the `product.id` in the request when `action` equal to `add`. It is the Seller who assigns this `id` (MEF 80 R27 [11]).

The Access E-Line product specification is identified as `00000000-0000-0000-0000-0000000007e3` in the Seller's catalog.

An Access E-Line product specification defines two mandatory relationship types that have to be specified in case of quoting an `add` action: `ENNI_REFERENCE` and `UNI_REFERENCE`. The reference to an operator UNI product might use another Quote item or an existing product from the Seller's inventory. This example assumes that the UNI product is another item of the request with a unique identifier `item-002`. This Access E-Line product references to an existing ENNI product which is uniquely identified with id `00000000-0000-000a-0000-000000000001` in the Seller's inventory.

The place is not provided as Access E-Line product specification does not allow for a place description to be part of the request. Values for some of the available product attributes are provided under 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 MEF standardization. It is published as a dedicated MEF 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.

7.2.6.2 Quote 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 `cbs` (Committed Burst Size) values for `ENNI` and `UNI` bandwidth profiles are introduced. The Access E-Line product exists in Seller's inventory and is identified as `01494079-6c79-4a25-83f7-48284196d44d`.

[R45] The modify request **MUST** specify a reference (provide `product.id`) to an existing product which is a subject of this order and provide the desired `product.productConfiguration` (MEF 80 R28 [11]).

[R46] The modify request **MUST** repeat the same values (specified or empty) of `product.productOffering`, `product.productRelationship`, and `product.place` as they are for available in the inventory for a given product instance. These values cannot be updated nor deleted (MEF 80 R29 [11]).

Note: The above requirement stands with literal contradiction to MEF 80 R29 [11] yet implements its intention of not updating the `product.productOffering`. The pattern of repeating the whole `product` is applied consistently across POQ, Quote, and Order APIs.

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

Please also note, that in the `add` case, a reference to the UNI product used the `quoteItemRelationship` pointing to another `quoteItem` in the same Quote Request. This is because the UNI was not existing 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` equal to `00000000-0000-000a-0000-000000000098`.

```
{
  <<Quote attributes...>>
  "quoteItem": [
    {
      "id": "item-001",
      "action": "modify",
      ...
      "product": {
        "id" : "01494079-6c79-4a25-83f7-48284196d44d",
        "@type" : "MEFProductRefOrValue",
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:quote",
          "enniEp": {
            "ingressBandwidthProfilePerClassOfServiceName": [
                {
                    "classOfServiceName": "silver",
                    "bwpFlow": [
                        {
                            "envelopeRank": 1,
                            "couplingFlag": false,
                            "envelopeName": "defaultENNI",
                            "tokenRequestedOffset": 0,
```

```
                    "colorMode": "COLOR_BLIND",
                    "cir": {
                        "irValue": 40,
                        "irUnits": "MBPS"
                    },
                    "cbs": {
                        "dataSizeValue": 100,
                        "dataSizeUnits": "KBYTES"

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

                    },
                    "cirMax": {
                        "irValue": 40,
                        "irUnits": "MBPS"
                    },
                    "eirMax": {
                        "irValue": 0,
                        "irUnits": "BPS"
                    },
                }
            ]
        }
    ]
},
"maximumFrameSize": 1522,
"uniEp": {
    "ingressBandwidthProfilePerClassOfServiceName": [
        {
            "classOfServiceName": "silver",
            "bwpFlow": [
                {
                    "envelopeRank": 1,
                    "couplingFlag": false,
                    "envelopeName": "defaultUNI",
                    "tokenRequestedOffset": 0,
                    "colorMode": "COLOR_BLIND",
                    "cir": {
                        "irValue": 40,
                        "irUnits": "MBPS"
                    },
```

```
                        "cbs": {
                            "dataSizeValue": 100,
                            "dataSizeUnits": "KBYTES"

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

                        },
                        "cirMax": {
                            "irValue": 40,
                            "irUnits": "MBPS"
                        },
                        "eirMax": {
                            "irValue": 0,
                            "irUnits": "BPS"
                        },
                    }
                ]
            }
        ]
      }
    },
    "productRelationship": [
      {
        "relationshipType": "ENNI_REFERENCE",
        "id": "00000000-0000-000a-0000-000000000001"
      },
      {
        "relationshipType": "UNI_REFERENCE",
        "id": "00000000-0000-000a-0000-000000000098"
      }
    ]
  },
  "relatedContactInformation": [
    {
      "number": "1-234-567-890",
      "emailAddress": "john@example.com",
      "role": "buyerContactInformation",
      "name": "John Example"
    }
  ]
}
```

```
  ]
}
```

Note: The Buyer can update a Buyer-related contact by providing a full list of existing `relatedContactInformation` items, and updating the value of those with Buyer-related `roles`.

Note: The Buyer cannot update a Buyer-related note. New notes can only be appended to the existing list of `note` items.

7.2.6.3 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 (type `urn:mef:lso:spec:sonata:AccessElineOvc:1.0.0:quote`).

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

[R47] `product.id` is mandatory for the `quoteItem` in the `delete` request (MEF 80 R30 [11]).

[R48] `MEFProductConfiguration` **MUST NOT** be provided in the payload if an item `action` is set to `delete` (MEF 80 R31 [11]).

[R49] In the `delete` request the only allowed attributes are `product.id` and `@type` (MEF 80 R31 [11]).

7.2.7 Specifying Place Details

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

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

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

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

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

Examples of different place specification formats are provided below.

7.2.7.1 Fielded Address

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

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

7.2.7.2 Formatted Address

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

```
}
```

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

7.2.7.3 Geographic Point

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

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

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

7.2.7.4 Geographic Address Label

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

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

7.2.7.5 Geographic Site Reference

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

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

7.2.7.6 Geographic Address Reference

```
{
  "@type": "GeographicAddressRef",
```

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

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

7.3 Use Case 2: Retrieve Quote List

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

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

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

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

```
https://serverRoot/mefApi/sonata/quoteManagement/v8/quote?state=approved.orderable&quoteLevel=firm
```

The example above shows a Buyer's request to get all `Quotes` that are in the `approved.orderable` state and with the `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.

[R50] The Seller must put the following attributes into the `Quote_Find` object in the response (MEF 80 R77 [11]):

- `id`
- `effectiveQuoteCompletionDate`
- `expectedQuoteCompletionDate`
- `externalId`
- `projectId`
- `quoteDate`
- `quoteLevel`

- `requestedQuoteCompletionDate`
- `state`

If no items are matching the criteria an empty list is returned.

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"
  }
]
```


7.4 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 5 and Table 6 (MEF 80 R81 [11]).

Please note that for readability purposes the following Tables (5,6,7, and 8) 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	unableToProvite
id	R	R	R	R	R	R	R	R	R	R	R
state	R	R	R	R	R	R	R	R	R	R	R
quoteDate	R	R	R	R	R	R	R	R	R	R	R
stateChange	R	O	O	R	R	R	R	O	O	O	O
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	§	§	§	§	§	§	§	§	§	§	§
relatedContactInformation (role=sellerContactInformation)	R	E	R	R	R	R	R	R	R	E	R

§: E – Buyer / PR – Seller

Table 5 – Seller Response to Query by ID, FIRM Quote Level, Quote Attributes

	accepted	acknowledged	cancelled	declined	expired	inProgress	inProgress.draft	approved.orderable	approved.orderableAlternate	rejected	unableToProvite
subjectToFeasibilityCheck	R	PR	PR	R	R		R	R	R		
note	§	§	§	§	§	§	§	§	§	§	§
state	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

§: E – Buyer / PR – Seller

Table 6 – 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 7 and Table 8 (MEF 80 R82 [11]).

	answered	acknowledged	cancelled	expired	inProgress	rejected	unableToProvite
id	R	R	R	R	R	R	R
state	R	R	R	R	R	R	R
quoteDate	R	R	R	R	R	R	R
stateChange	O		O	R	R		O
expectedQuoteCompletionDate	O		O	O	R		O
validFor	R						
effectiveQuoteCompletionDate	R		R	R			R
quoteLevel	R	R	R	R	R	R	R
note	§	§	§	§	§	§	§
relatedContactInformation (role=sellerContactInformation)	R	R	R	R	R	R	R

§: E – Buyer / PR – Seller

Table 7 – Seller Response to Query by ID, BUDGETARY Quote Level, Quote Attributes

	answered	acknowledged	cancelled	expired	inProgress	rejected	unableToProvite
subjectToFeasibilityCheck	NA	NA	NA	NA	NA	NA	NA
note	§	§	§	§	§	§	
state	R	R	R	R	R	R	R
price	R			R			
quoteItemTerm	R			R			
quoteItemInstallationInterval	R			R			
terminationError						PR	PR

§: E – Buyer / PR – Seller

Table 8 – Seller Response to Query by ID, BUDGETARY Quote Level, QuoteItem Attributes

[R53] The `Quote` **MUST NOT** be in `approved.orderable` state unless all its `QuoteItems` are in the `approved.orderable` state as well (MEF 80 R70 [11]).

[D1] When moving the Quote to `rejected` or `unableToProvide` state, the Seller **SHOULD** use the `quoteItem.terminationError` to describe the reason for item processing failure.

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

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@example.com",
      "name": "John Example",
      "number": "12-345-6789",
      "role": "buyerContactInformation"
    },
```

```
    {
      "emailAddress": "kate.example@example.com",
      "name": "Kate Example",
      "number": "12-345-67890",
      "role": "sellerContactInformation"
    }
  ]
}
```

7.5 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 (MEF 80 R1 [11]).

The message body contains only two attributes:

- `quoteId` – mandatory attribute to point to which `Quote` must be cancelled
- `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."
}
```

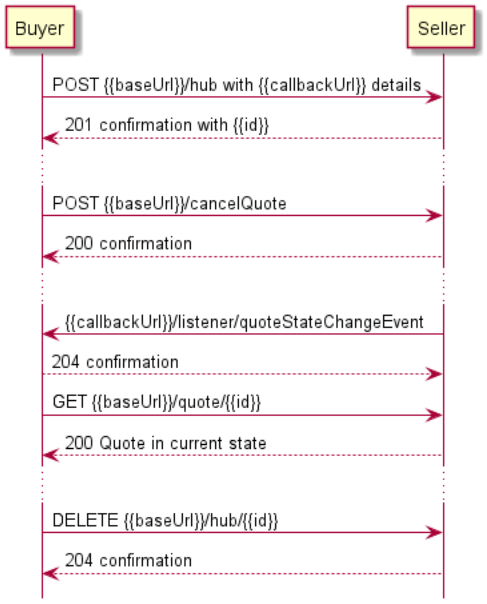

Figure 16 – Use Case 4, 5: Cancel or Decline Quote by Quote Identifier

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 consistent with the Ordering API, where operations like `amendOrder` or `cancelOrder` are long-lasting processes that can be tracked by the Buyer and might not necessarily end up with success.

Figure 16 presents an example of the Cancel use case flow.

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`).

7.6 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 (MEF 80 R2 [11]).

The Decline Request has the same body and the 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 timeout.

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

7.7 Use Case 6: Register for Quote Notifications

The Seller communicates with the Buyer with Notifications provided that:

- both Seller and Buyer support the `Quote` notification mechanism, and
- 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 event.

The usage of a combination of these attributes fulfills the MEF 80 R84, R85, and R86 requirements [11].

By using a simple request:

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

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

- `quoteStateChangeEvent`
- `quoteItemStateChangeEvent`

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

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

If the Buyer wishes to subscribe to 2 different types of events, there are 2 possible syntax variants [13]:

```
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.

The `query` formatting complies to RCF 3986 [3]. According to it, every attribute defined in the Event model (from the notification API) can be used in the `query`. However, this standard requires only the `eventType` attribute to be supported.

[O3] The Seller **MAY** support the ability for the Buyer to Register for Quote Notifications (MEF 80 O19 [11]).

[CR1]<[O3] If the Seller supports the ability for the Buyer to Register for Quote Notifications, they **MUST** support sending Quote Notifications (MEF 80 CR3<O19 [11]).

[R56] `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 the must be further used for unsubscribing.

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

7.8 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 sake of readability, all previous flow diagrams presented only cases of using only the `quoteStateChangeEvent`. Figure 17 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.

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 Create Quote request was provided. Thus any `Quote` or `QuoteItem` state change history until that moment is not a subject for notification. This applies in both deferred and immediate responses. Especially in the latter one when the `Quote` provided in the response is already in the completion state, so all of its `QuoteItems` must have had at least one state transition.

[O4] The Seller **MAY** support sending Quote Notifications (MEF 80 O20 [11]).

[R57] The Seller **MUST** support sending Quote Notifications if a deferred response is supported.

[CR2]<[O4] The Seller **MUST** be able to send Quote Notifications to Buyers for both Immediate and Deferred Quote Responses (MEF 80 CR4<O20 [11]).

[CR3]<[O4] 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`) (MEF 80 CR5<O20 [11]).

[CR4]<[O4] The Seller **MUST** send Quote Notifications to Buyers who have registered for Quote Notifications (MEF 80 CR7<O20 [11]).

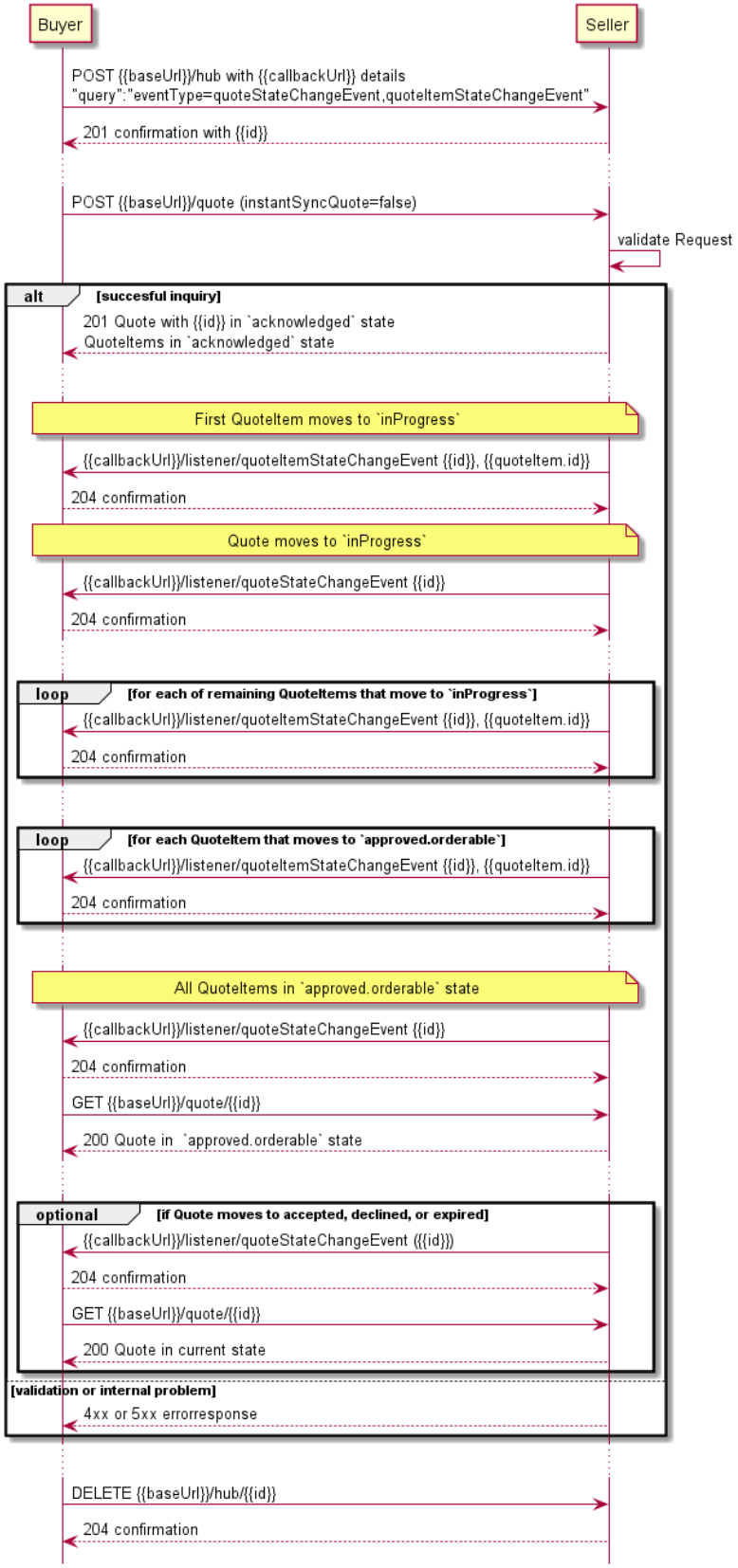

Figure 17 – Use Case 1b: Deferred Quote Response Requested and Provided with QuoteItem Notifications

[CR5]<[O4] The Seller **MUST NOT** send Quote Notifications to Buyers who have not registered for Quote Notifications (MEF 80 CR6<O20 [11]).

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

```
{
  "eventId": "event-001",
  "eventType": "quoteStateChangeEvent",
  "eventTime": "2020-08-10T16:40:37.422Z0",
  "event": {
    "id": "00000000-0000-0000-0000-000000000123"
  }
}
```

[R58] An event triggered by the state change of the `QuoteItem` **MUST** additionally contain the relative `quoteItemId`:

```
{
  "eventId": "event-002",
  "eventType": "quoteItemStateChangeEvent",
  "eventTime": "2020-08-10T16:45:37.422Z0",
  "event": {
    "id": "00000000-0000-0000-0000-000000000123",
    "quoteItemId": "item-002"
  }
}
```

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

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

The flow of this use case is presented in Figure 18. Notifications are sent to:

Sonata:

- `https://buyer.com/listenerEndpoint/mefApi/sonata/quoteNotification/v8/listener/quoteStateChangeEvent` in case of `quoteStateChangeEvent`
- `https://buyer.com/listenerEndpoint/mefApi/sonata/quoteNotification/v8/listener/quoteItemStateChangeEvent` in case of `quoteItemStateChangeEvent`

Cantata:

- `https://buyer.com/listenerEndpoint/mefApi/cantata/quoteNotification/v2/listener/quoteStateChangeEvent` in case of `quoteStateChangeEvent`
- `https://buyer.com/listenerEndpoint/mefApi/cantata/quoteNotification/v2/listener/quoteItemStateChangeEvent` in case of `quoteItemStateChangeEvent`

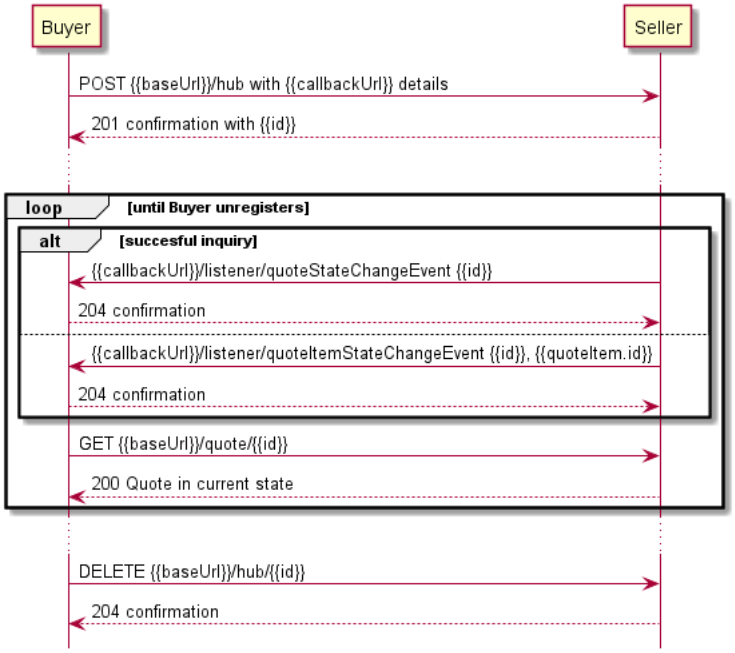

Figure 18 – Use Case 6, 7: Register and Send Notifications

8 API Details

8.1 API Patterns

8.1.1 Indicating Errors

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

Implementations can use HTTP error codes not specified in this standard in compliance with rules defined in RFC 7231 [5]. In such a case the error message body structure might be aligned with the `Error`.

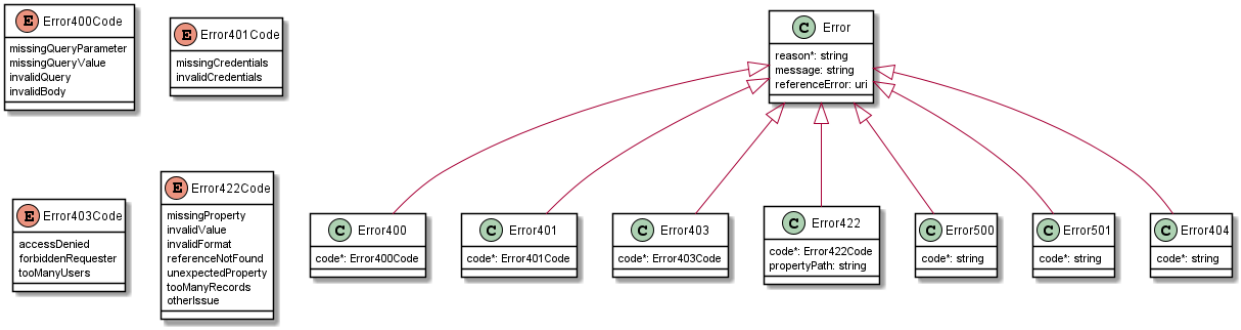

Figure 19 – Data Model Types to Represent an Erroneous Response

8.1.1.1 Type Error

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

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

Table 9 – Type Error

8.1.1.2 Type Error400

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

Inherits from:

- Error

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

Table 10 – Type Error400

8.1.1.3 Type Error401

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

Inherits from:

- Error

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

Table 11 – Type Error401

8.1.1.4 Type Error403

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

Inherits from:

- Error

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

Table 12 – Type Error403

8.1.1.5 Type Error404

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

Inherits from:

- Error

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

Table 13 – Type Error404

8.1.1.6 Type Error422

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

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

Inherits from:

- Error

Name	Type	Description
code*	string	One of the following error codes: - missingProperty: The property the Seller has expected is not present in the payload. - invalidValue: The property has an incorrect value. - invalidFormat: The property value does not comply with the expected value format. - referenceNotFound: The object referenced by the property cannot be identified in the Seller system. - unexpectedProperty: Additional property, not expected by the Seller has been provided. - tooManyRecords: the number of records to be provided in the response exceeds the Seller's threshold. - otherIssue: Other problem was identified (detailed information provided in a reason).
propertyPath	string	A pointer to a particular property of the payload that caused the validation issue. It is highly recommended that this property should be used. Defined using JavaScript Object Notation (JSON) Pointer [1].

Table 14 – Type Error422

8.1.1.7 Type Error500

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

Inherits from:

- Error

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

Table 15 – Type Error500

8.1.1.8 Type Error501

Description: Not Implemented. (https://datatracker.ietf.org/doc/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.

Table 16 – Type Error501

8.1.2 Response Pagination

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

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

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

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

[R59] Seller **MUST** use either `X-Total-Count` or `X-Pagination-Throttled` to indicate that the page was truncated and additional results are available.

8.2 Management API Data Model

Figure 20 presents the Quote Management data model. The data types, requirements related to them, and mapping to MEF 80 [11] specifications are discussed later in this section.

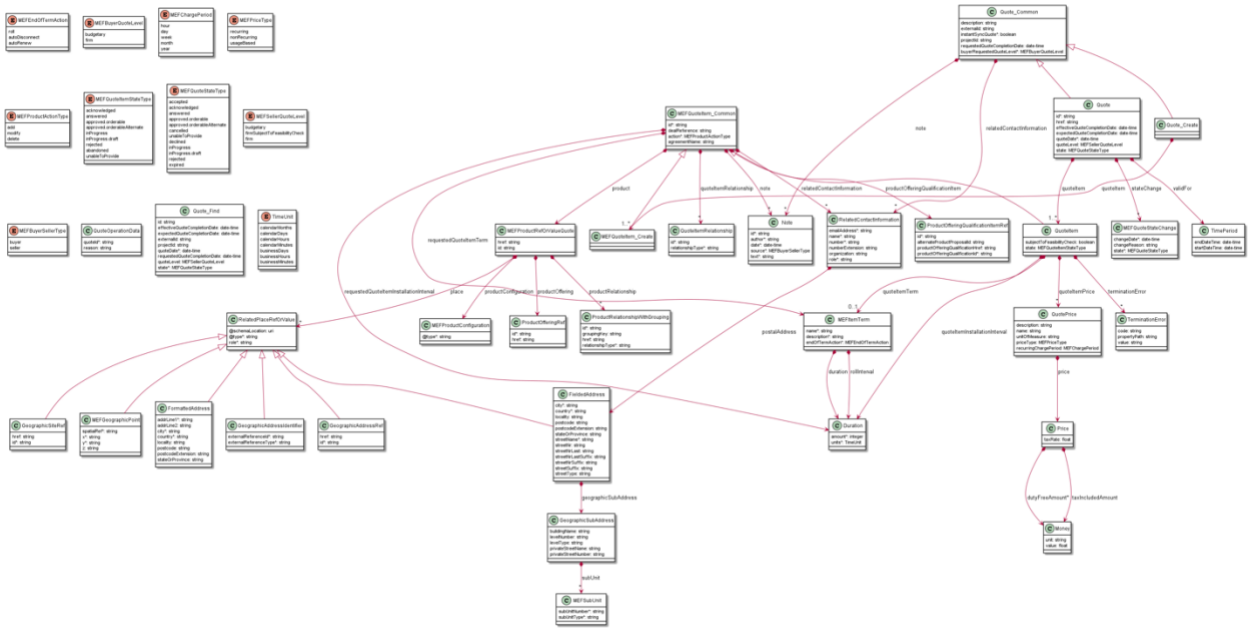

Figure 20 – Quote Management Data Model

8.2.1 Quote

8.2.1.1 Type Quote_Common

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

Name	Type	Description	MEF 80
`description`	string	Description of the quote.	Description
`externalId`	string	ID given by the consumer and only understandable by them (to facilitate their searches afterwards).	Buyer Quote Identifier
`instantSyncQuote*`	boolean	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
`projectId`	string	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
`requestedQuoteCompletionDate`	date-time	This is requested date – from quote requester – to get a complete response for this quote.	Requested Quote Completion Date
`buyerRequestedQuoteLevel*`	MEFBuyer QuoteLevel	An indication of whether the Buyer's Quote request is for a Quote of Budgetary or Firm level.	Buyer Requested Quote Level

note	Note	Free form text associated with the quote. Only useful in processes involving human interaction. Not applicable for the automated process.	Note
relatedContactInformation	Related Contact Information[]	Party playing a role for this quote. If instantSyncQuote equals false then the Buyer **MUST** specify Buyer Contact Information (role equals buyerContactInformation). The Seller **MUST** always specify Seller Contact Information (role equals sellerContactInformation).	Buyer Contact Information (role equals buyerContactInformation), Seller Contact Information (role equals sellerContactInformation)

Table 17 – Type Quote_Common

8.2.1.2 Type Quote_Create

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

This type MUST be used by the Buyer in the request.

Inherits from:

- Quote_Common

Name	Type	Description	MEF 80
quoteItem*	MEFQuoteItem_Create[]	An item of the quote – used to describe an operation on a product to be quoted.	Quote Item

Table 18 – Type Quote_Create

[R60] If instantSyncQuote equals false the relatedContactInformation list **MUST** contain an entry that represents Buyer Contact Information. The role for this entry **MUST** be buyerContactInformation (MEF 80 R16 [11]).

8.2.1.3 Type Quote

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

This type MUST be used by the Seller in the response.

Inherits from:

- Quote_Common

Name	Type	Description	MEF 80
id*	string	Unique identifier – attributed by quoting system.	Seller Quote Identifier

href	string	Hyperlink representing this Quote. Hyperlink **MAY** be used when providing a response by the Seller.	Not represented in MEF 80
effectiveQuoteCompletionDate	date-time	Date when the Quote State was set to one of the Completion States.	Quote Completion Date
expectedQuoteCompletionDate	date-time	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
quoteDate*	date-time	Date and time when the quote was created.	Quote Request Date
quoteItem*	QuoteItem	An item of the quote – it is used to describe an operation on a product to be quoted	Quote Item
quoteLevel	MEFSellerQuoteLevel	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
state	MEFQuoteStateType	The state of the Quote.	Quote State
stateChange	MEFQuoteStateChange[]	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
validFor	TimePeriod	Quote validity period. For use in the context of this attribute, only the endDateTime attribute must be used.	Valid Until Date

Table 19 – Type Quote

[R61] Each item in quoteItem list **MUST** correspond to one and only one Quote Item in the Buyer's Create Quote request (MEF 80 R36, R45 [11]).

[R62] Seller's response **MUST** contain an entry that represents *Seller Contact Information*. The role for this entry **MUST** be sellerContactInformation (MEF 80 R43 [11]).

8.2.1.4 enum *MEFQuoteStateType*

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

Value	MEF 80	Description
`accepted`	ACCEPTED	The `accepted` state is set by the Seller. It is triggered by a Product Order. All Quote Items must be in an `approved.orderable` or `approved.orderableAlternate` state. In that case, the Quote moves from the `approved.orderableAlternate` state to the `accepted` state.
`acknowledged`	ACKNOWLEDGED	A Create Quote request has been received by the Seller and has passed basic validation. Quote `id` is assigned in the acknowledged state. The Quote remains in this state until all validations of Quote and Quote Item(s) attributes' as applicable are completed. If the Quote and Quote Item attributes are validated the Quote moves to the `inProgress` state for a Deferred Quote Response. If all Quote and Quote Item(s) attributes are not validated, the Quote moves to the `rejected` state.
`answered`	ANSWERED	The `answered` state is applied when a Buyer Requested Quote Level is `BUDGETARY`, and the Seller has provided a Quote to answer the Buyer's Quote request. When the Quote request is in the `BUDGETARY` state, all Quote Items must be in a terminal state.
`approved.orderable`	ORDERABLE	The `approved.orderable` state is where the quote has been internally approved by the Seller and is ready for the Buyer to review. All Quote Items must also be in an `approved.orderable` state. In case of an Immediate Quote Response for a Quote, the Quote State can directly move to `approved.orderable` state without going through the `inProgress` state.
`approved.orderableAlternate`	ORDERABLE_ALTERNATE	This state is set by the Seller when the Seller's Quote contains a Quote Item Term that is > the Buyer's Requested Quote Item Term and/or a Quote Item Installation Interval that is > the Buyer's Requested Quote Item Installation Interval and Requested Quote Item Installation Interval Value. Quote items are in the `approved.orderableAlternate` or `approved.orderable` state. From `approved.orderable` or `approved.orderableAlternate` the quote can move to the `declined`, `expired`, or `accepted` states.
`cancelled`	CANCELLED	A Quote can only be `cancelled` when it is in an `inProgress` state or an `inProgress.draft` state. The `cancelled` state is when the quote process is stopped by a Buyer through an explicit Cancel Quote by Quote Identifier Request. Quote items might be in an `inProgress`, `inProgress.draft`, `approved.orderable`, or `approved.orderableAlternate` state. Any Quote Items in an `inProgress` or `inProgress.draft` state are moved to an `abandoned` state. The Seller moves the Quote to the `cancelled` state upon the receipt of a Cancel Quote by Quote Identifier request from the Buyer.
`declined`	DECLINED	The `declined` state is used when the Buyer does not wish to progress with the quotation. The Buyer requests to change the state of the Quote to `declined` through a Decline Request. In order for a Buyer to decline the Quote, the Quote must be in an `approved.orderable` or `approved.orderableAlternate` state. The Seller moves the Quote to `declined` state. The state of any associated Quote Items does not change.

expired	EXPIRED	This state is set by the Seller from an approved.orderable, approved.orderableAlternate, or answered state to indicate that the validFor date for the Quote has passed and Quote will no longer be honored. All the Quote Items must be in an approved.orderable, approved.orderableAlternate, or answered state.
inProgress	IN_PROGRESS	The Quote is in the inProgress state when the Seller is collecting all information for building the Quote according to the Buyer's requirements. The Seller may update pricing and expectedCompletionDate in this state. At least one Quote Item is in an inProgress state. No Quote Items may be in an unableToProvide state.
inProgress.draft	IN_PROGRESS_DRAFT	The inProgress.draft state is when the Seller has populated some information in the Quote in answer to the Buyer's request. The Seller is continuing to work on the Quote and the Quote cannot be referenced in a Product Order in this state. At least one Quote Item is in an inProgress.draft state. No Quote Items are in an abandoned, rejected, or unableToProvide state.
unableToProvide	UNABLE_TO_PROVIDE	This state is set when the Seller is unable to provide a Quote in the timeframe required by the Buyer (e.g. if an immediate response or a response date is set but cannot be met by the Seller) or cannot provide a price as requested by the Buyer. The Quote request moves to unableToProvide when one Quote Item is in an unableToProvide state. Any Quote Items in an inProgress state are moved to an abandoned state. When a Quote is in the unableToProvide state, some Quote Items might be in the approved.orderable or approved.orderableAlternate state. In case of an Immediate Quote Response for a Quote, the Quote State can directly move to the unableToProvide state without going through the inProgress state.
rejected	REJECTED	A Create Quote request was submitted, and it has failed at least one of the validation checks the Seller performs after it reached the acknowledged state.

Table 20 – enum MEFQuoteStateType

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.

If a Quote request does not pass an initial validation the appropriate error response is returned to the Buyer. In case a Quote request failed business rules validation the HTTP response code is 422 and a list of validation problems is returned. Otherwise, the Quote is processed and given an id.

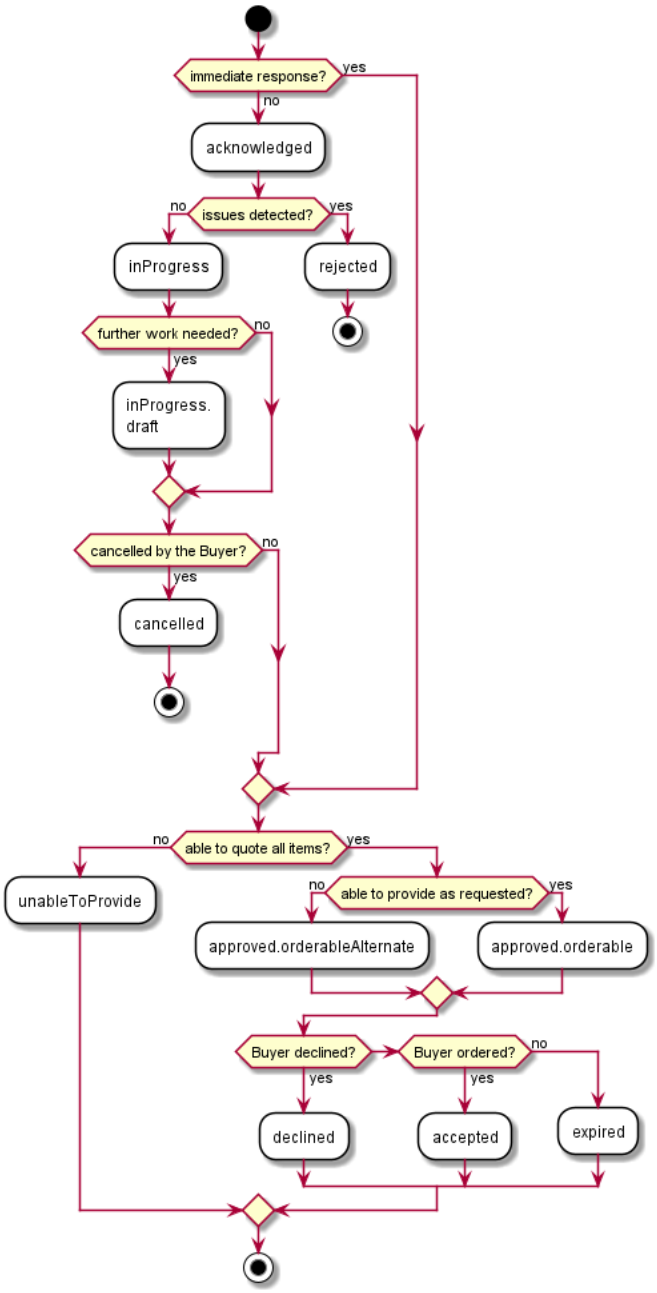

Figure 21 – Quote Firm Flow Activity Diagram

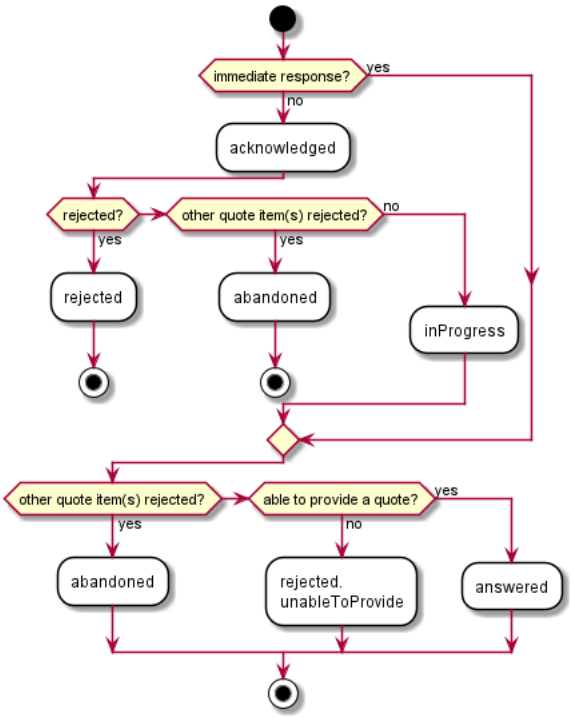

Figure 22 – Quote Budgetary Flow Activity Diagram

The following tables present the possible combinations of Quote and QuoteItem states, as described in MEF 80 [11]:

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

Table 21 – Allowable Quote Item States per Quote State for FIRM Quote Level

Quote State	Quote Items State	abandoned	answered	inProgress	rejected.insufficient InformationProvided	unableToProvide
acknowledged						
answered			X			
cancelled		X	X			
expired			X			
inProgress			X	X		
rejected						
unableToProvide		X	X			X

Table 22 – Allowable Quote Item States per Quote State for BUDGETARY Quote Level

8.2.1.5 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	Description
budgetary	BUDGETARY	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. Any charges specified are subject to change.
FIRM	FIRM	A quote provided to the Buyer based on a complete pre-order analysis. All 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.

Table 23 – enum MEFBuyerQuoteLevel

8.2.1.6 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	Description
budgetary	BUDGETARY	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. Any charges specified are subject to change.

firmSubjectToFeasibilityCheck	FIRM_SUBJECT_TO_FEASIBILITY_CHECK	A quote that is provided to the Buyer based on some, but not a complete, pre-order analysis. At this stage, the Seller may not be willing to perform any further work on the quote and requests that the Buyer use the Firm – Subject to Feasibility Check Quote to proceed to the Order process. Ordering is possible based on the Firm Subject to Feasibility Check Quote with some stipulations as to how 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.
FIRM	FIRM	A quote provided to the Buyer based on a complete pre-order analysis. All 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.

Table 24 – enum MEFSellerQuoteLevel

8.2.1.7 Type MEFQuoteStateChange

This list works as a history for a Quote instance.

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

Name	Type	Description	MEF 80
changeDate*	date-time	The date on when the state was reached.	Not represented in MEF 80
changeReason	string	Additional comment related to state change.	Not represented in MEF 80
state*	MEFQuoteStateType	A state reached at the change date.	Not represented in MEF 80

Table 25 – Type MEFQuoteStateChange

8.2.2 Quote Item

8.2.2.1 Type MEFQuoteItem_Common

Description: A quote item describes an action to be performed on a productOffering or a product in order to get pricing elements and condition.

Name	Type	Description	MEF 80
id*	string	Identifier of the quote item (generally it is a sequence number 01, 02, 03, ...)	Quote Item Reference Number
dealReference	string	A pre-agreed pricing modifier reference that the Seller is offering to the Buyer which will impact the price.	Quote Item Deal Reference
action*	MEFProduct ActionType	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

agreementName	string	Name of the agreement. The name is unique between the Buyer and the Seller.	Agreement
note	Note	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 RefOrValue	The Buyer's existing Product for which the quote is being requested.	Product Identifier, Product Specific Attributes
productOfferingQualificationItem	ProductOffering Qualification ItemRef	A reference to a previously done POQ with item specified.	POQ
quoteItemRelationship	QuoteItem Relationship[]	A relationship from item within a quote.	Quote Item Relationship
relatedContactInformation	RelatedContact Information[]	Contact information of an individual or organization playing a role for this Quote. If instantSyncQuote equals false then Quote Item Technical Contact (role: quoteItemTechnicalContact) MUST be specified. If instantSyncQuote equals false and the Quote Item requires a location, the Buyer MUST provide the Quote Item Location Contact Information (role: quoteItemLocationContact).	Quote Item Location Contact (role: quoteItemLocationContact), Quote Item Technical Contact (role: quoteItemTechnicalContact)
requestedQuoteItemInstallationInterval	Duration	The installation interval requested by the Buyer.	Requested Quote Item Installation Interval
requestedQuoteItemTerm	MEFItemTerm	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

Table 26 – Type MEFQuoteItem_Common

8.2.2.2 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

Repeated: **[R23]** if `instantSyncQuote` equals `false` the `relatedContactInformation[]` with an item of `role` equal to `quoteItemTechnicalContact` to specify the required Quote Item Technical Contact Information (MEF 80 R15 [11]).

Repeated: **[R24]** if `instantSyncQuote` equals `false` and the Quote Item requires a location, `relatedContactInformation[]` with an item of `role` equal to `quoteItemLocationContact` to specify the required Quote Item Location Contact Information (MEF 80 R24 [11]).

8.2.2.3 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.

Name	Type	Description	MEF 80
`quoteItemInstallationInterval`	Duration	Quote Item Installation Interval as proposed by the Seller for the Quote.	Quote Item Installation Interval
`quoteItemPrice`	QuotePrice[]	Price for this quote item.	Quote Item Price
`quoteItemTerm`	MEFItemTerm[]	Quote Item Term as defined by the Seller and part of the Quote for the Quote Item.	Quote Item Term
`subjectToFeasibilityCheck`	boolean	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
`state`	MEFQuoteItemStateType	The state of the Quote Item.	Quote Item State
`terminationError`	TerminationError[]	When the Seller cannot process the Quote Item Request, the Seller returns a text-based list of reasons here.	Quote Item Termination Error

Table 27 – Type QuoteItem

8.2.2.4 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.

Mapping to MEF 80:

Value	MEF 80
`add`	INSTALL
`modify`	CHANGE
`delete`	DISCONNECT

Table 28 – enum MEFProductActionType

8.2.2.5 enum MEFQuoteItemStateType

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

Value	MEF 80	Description
abandoned	ABANDONED	The abandoned state is applied to Quote Items that are in an inProgress state when the quote is moved to a terminal state (cancelled, unableToProvide).
answered	ANSWERED	The answered state is applied when a Buyer Requested Quote Level is budgetary, and the Seller has provided a Quote in answer to the Buyer's Create Quote request.
acknowledged	ACKNOWLEDGED	A Create Quote request has been received by the Seller and has passed basic validation. The Quote Item remains in the acknowledged state until all validations of Quote Item attributes as applicable are completed. If the Quote Item attributes are validated the Quote Item moves to the inProgress state for a Deferred Quote Response. If not all Quote Item attributes are validated, the Quote Item moves to the rejected state.
approved.orderable	ORDERABLE	The approved.orderable state is where the Quote Item has been internally approved by the Seller. In case of an Immediate Quote Response for a Quote, a Quote Item can directly move to approved.orderable state without going through the inProgress state.
approved.orderableAlternate	ORDERABLE_ALTERNATE	This state is set by the Seller when they have provided inProgress.draft Quote response for this Quote Item and have either decided that they will not perform any further work on the Quote or the Seller's response Quote contains a quoteItemInstallationInterval greater than the requestedQuoteItemInstallationInterval.
inProgress	IN_PROGRESS	The inProgress state is applied when the Quote Item is currently in the hands of the Seller to build it regarding Buyer requirements. The price for the Quote Item is under construction.
inProgress.draft	IN_PROGRESS_DRAFT	The inProgress.draft state is applied to a Buyer Requested Quote Level of firm and the Seller is providing a partial response for this Quote Item in answer to the Buyer's request. The information for this Quote Item may be subject to revision.
unableToProvide	UNABLE_TO_PROVIDE	The unableToProvide state is set on a Quote Item when the Seller cannot provide pricing for this item. In case of an Immediate Quote Response for a Quote, a Quote Item can directly move to the unableToProvide state without going through the inProgress state.
rejected	REJECTED	A Quote Item has failed at least one of the validation checks the Seller performs after it reached the acknowledged state if a Deferred Response is provided.

Table 29 – enum MEFQuoteItemStateType

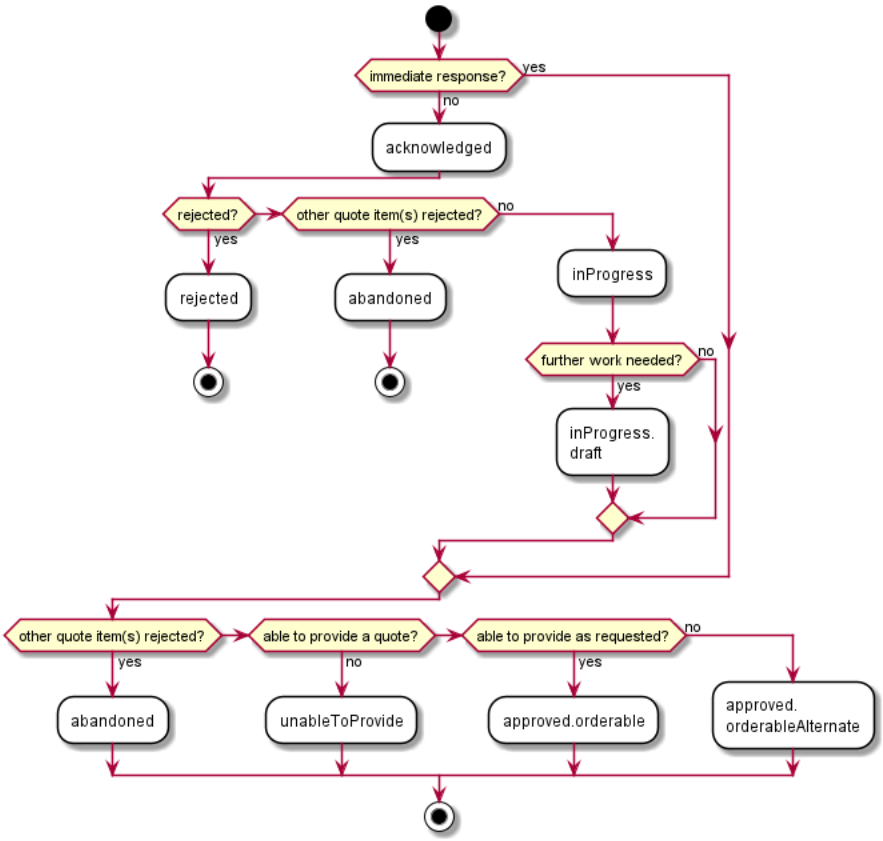

Figure 23 – Quote Item Firm Quote Flow Activity

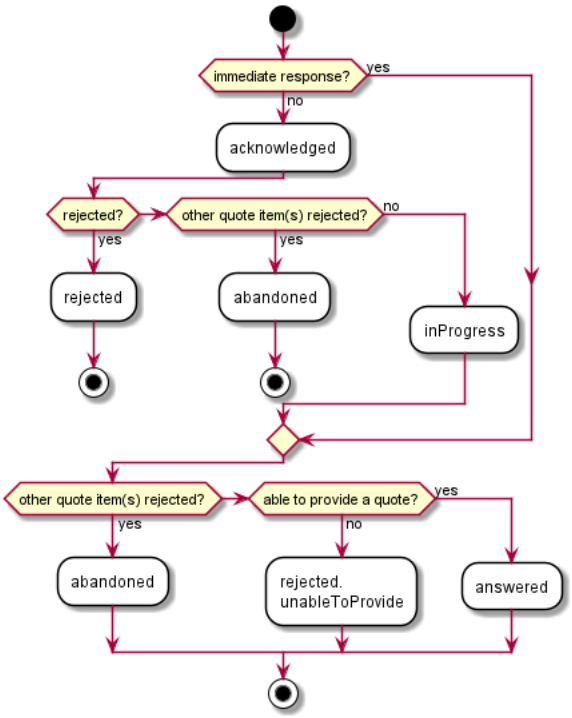

Figure 24 – Quote Item Budgetary Quote Flow Activity

8.2.2.6 Type ProductOfferingQualificationItemRef

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

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

Table 30 – Type ProductOfferingQualificationItemRef

8.2.2.7 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	Description	MEF 80
id*	string	Unique identifier of the Product Offering.	Quote.Product Offering Identifier
href	string	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

Table 31 – Type ProductOfferingRef

8.2.2.8 Type QuoteItemRelationship

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

Name	Type	Description	MEF 80
id*	string	ID of the related quote item (must be in the same quote).	Quote Item Relationship Identifier

relationshipType*	string	Relationship type as relies on, bundles, etc… MEF: 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

Table 32 – Type QuoteItemRelationship

8.2.2.9 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	Description	MEF 80
name*	string	Name of the term.	Quote Item Term Name
description	string	Description of the term.	Quote Item Term Description
duration*	Duration	Duration of the term.	Quote Item Term Duration
rollInterval	Duration	The recurring period that the Buyer is willing to pay to the end of upon disconnecting the Product after the original term has expired.	Roll Interval
endOfTermAction*	MEFEndOfTermAction	The action that needs to be taken by the Seller once the term expires.	Seller End of Term Action

Table 33 – Type MEFItemTerm

8.2.2.10 enum MEFEndOfTermAction

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

Value	MEF 80	Description
roll	ROLL	Roll indicates that the Product's contract will continue on a rolling basis for the duration of the Roll Interval at the end of the Quote Item Term.
autoDisconnect	AUTO_DISCONNECT	Auto-disconnect indicates that the Product will be disconnected at the end of the Quote Item Term.
value	AUTO_RENEW	Auto-renew indicates that the Product's contract will be automatically renewed for the Quote Item Term Duration at the end of the Quote Item Term.

Table 34 – enum MEFEndOfTermAction

8.2.2.11 Type QuotePrice

Table 35 shows the combination of attributes that **MUST** be provided for each Price Type (MEF 80 R55, R56 [11]).

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

Table 35 – Price Type Required Information

Description: Description of price and discount awarded.

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

Table 36 – Type QuotePrice

8.2.2.12 Type Price

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

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

Table 37 – Type Price

8.2.2.13 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

Table 38 – enum MEFPriceType

8.2.2.14 enum MEFChargePeriod

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

Value	MEF 80
hour	HOUR
day	DAY
week	WEEK
month	MONTH
year	YEAR

Table 39 – enum MEFChargePeriod

8.2.3 Product Representation

8.2.3.1 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	Description	MEF 80
href	string	Hyperlink to the product in Seller's inventory that is the quotation's subject. Hyperlink **MAY** be used by the Seller in response. 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	The unique identifier of an in-service Product that is the quotation's subject. This attribute **MUST** be populated if an item action is either modify or delete. This attribute **MUST** NOT be populated if an item action is add.	Product Identifier
place	RelatedPlaceRefOrValue[]	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	Technical attributes for the Product that would be delivered to fulfill the Quote Item..	Product Specific Attributes
productOffering	ProductOfferingRef	A particular Product Offering defines the technical and commercial attributes and behaviors of a Product.	Product Offering Identifier
productRelationship	ProductRelationship WithGrouping[]	A list of references to existing products that are related to the Product that would be delivered to fulfill the Quote Item.	Product Relationships

Table 40 – Type MEFProductRefOrValueQuote

8.2.3.2 Type MEFProductConfiguration

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

Name	Type	Description	MEF 80
`@type*`	string	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 MEF product, this is the URN provided in the Product Specification.	Not represented in MEF 80

Table 41 – Type MEFProductConfiguration

8.2.3.3 Type ProductRelationshipWithGrouping

Description: A relationship to existing Product. The requirements for usage for a given Product are described in the Product Specification. The "WithGrouping" flavor of the Product Relationship allows for providing a list of related product identifiers within a single Product Relationship. This can be later used while processing the request as defined in the Product Specification. The `groupingKey` attribute is used to achieve this behavior in the API by marking the list of `ProductRelationshipWithGroupings` within a `product` with a common key.

Name	Type	Description	MEF 80
`id*`	string	A unique identifier of a Product that is referenced.	Product Relationship Identifier
`groupingKey`	string	MEF 80 Section 8.1.8 [11] introduces a list of related ids for the ProductRelationship. For sake of TMF compliance, a `groupingKey` is introduced to retain `id` as a simple attribute. Ids from relationships having the same `groupingKey` and `relationshipType` **MUST** be treated as a list of identifiers.	Not represented in MEF 80
`href`	string	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
`relationshipType*`	string	Specifies the type (nature) of the relationship to the related Product. The nature of required relationships vary for Products of different types. For example, a UNI or ENNI Product may not have any relationships, but an Access E-Line may have two mandatory relationships (related to the UNI on one end and the ENNI on the other). More complex Products such as multipoint IP or Firewall Products may have more complex relationships. As a result, the allowed and mandatory `relationshipType` values are defined in the Product Specification.	Relationship Nature

Table 42 – Type ProductRelationshipWithGrouping

MEF 80 [11] allows for providing a list of related product identifiers within a single Product Relationship. This can be later used while processing the request as defined in the Product

Specification. The `groupingKey` attribute is used to achieve this behavior in the API by marking the list of `ProductRelationshipWithGroupings` within a `product` with a common key.

[R63] The Product Identifier from relationships having the same `groupingKey` and `relationshipType` **MUST** be treated as a list of identifiers.

Note: The Buyer may submit a Quote request for a Product that requires an interconnection between Buyer and Seller. The Quote request may contain the Product identifier for one or more interconnections. The Seller returns the best result for the Quote request over any of the related product identifiers. The Buyer includes one of the product `ids` in the Order request if they place an order based on this Quote. If none of the related product identifiers from the Quote are included in the Order request, the Seller may reject the Order request.

8.2.4 Place Representation

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

[R64] `GeographicAddressRef` or `GeographicSiteRef` **MUST** be used to provide place information by reference. This method is referred to as the "Known Address ID method" in MEF 79 Section 8.9.3.1 [10].

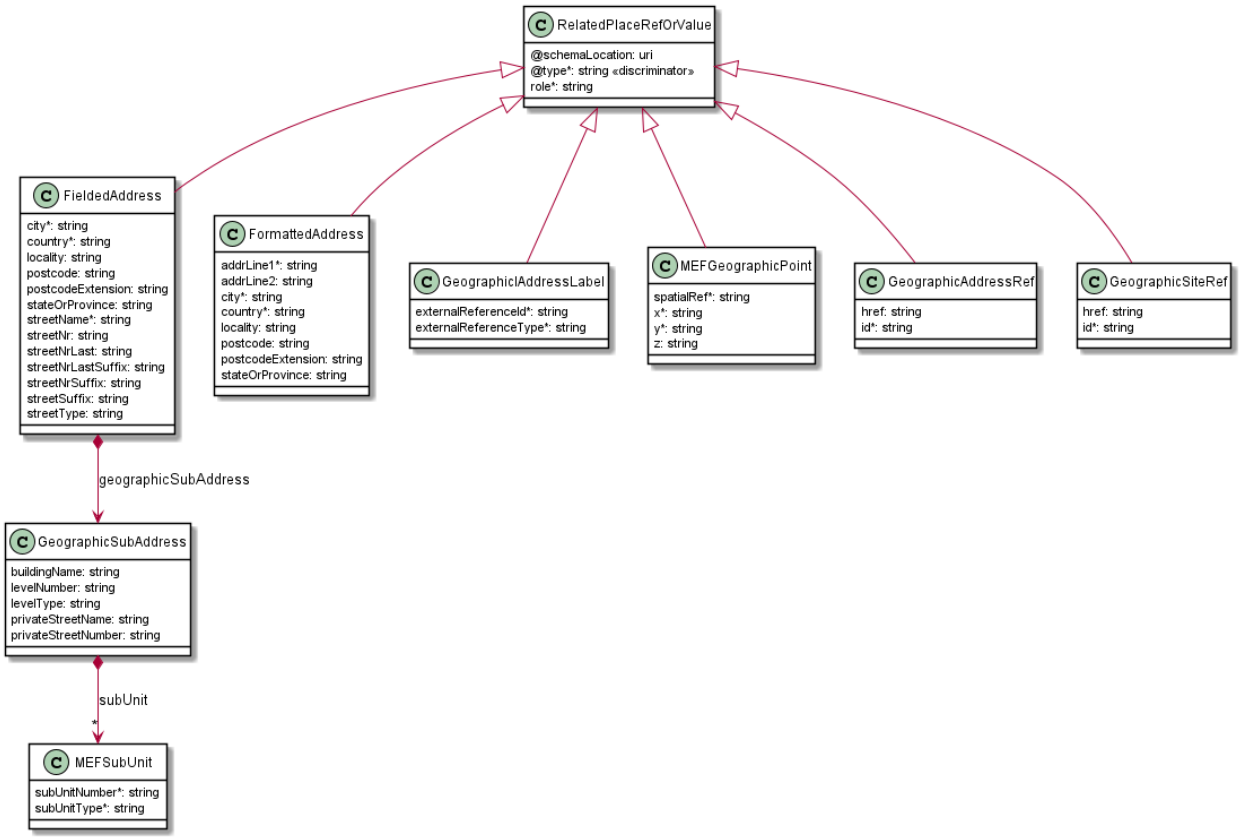

Figure 25 – Data Model Types Representing a Place

8.2.4.1 Type RelatedPlaceRefOrValue

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

Name	Type	Description	MEF 80
`@schemaLocation`	uri	A URL to a JSON-Schema file that defines additional attributes and relationships. May be used to define additional related place types. Usage of this attribute must be agreed between Buyer and Seller.	Not represented in MEF 80
`@type*`	string	This field is used as discriminator and is used between different place representations. This type might discriminate for additional related place as defined in `@schemaLocation`.	Not represented in MEF 80
`role*`	string	Role of this place.	RelatedPlaceRefOrValue

Table 43 – Type RelatedPlaceRefOrValue

8.2.4.2 Type FieldedAddress

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

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	Description	MEF 80
`city*`	string	The city that the address is in.	City
`country*`	string	The country that the address is in.	Country
`geographicSubAddress`	Geographic SubAddress	Additional fields used to specify an address, as detailed as possible.	Not represented in MEF 80
`locality`	string	The locality that the address is in.	Locality
`postcode`	string	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	An extension of a postal code. E.g. the part following the dash in a US urban property address.	Postal Code Extension
`stateOrProvince`	string	The State or Province that the address is in.	State Or Province
`streetName*`	string	Name of the street or other street type.	Street Name
`streetNr`	string	Number identifying a specific property on a public street. It may be combined with `streetNrLast` for ranged addresses.	Street Number
`streetNrLast`	string	Last number in a range of street numbers allocated to a property.	Street Number Last
`streetNrLastSuffix`	string	Last street number suffix for a ranged address.	Street Number Suffix Last
`streetNrSuffix`	string	The first street number suffix.	Street Number Suffix
`streetSuffix`	string	A modifier denoting a relative direction.	Street Suffix
`streetType`	string	The type of street (e.g., alley, avenue, boulevard, brae, crescent, drive, highway, lane, terrace, parade, place, tarn, way, wharf).	Street Type

Table 44 – Type FieldedAddress

8.2.4.3 Type FormattedAddress

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

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	Description	MEF 80
addrLine1*	string	The first address line in a formatted address.	Address Line 1
addrLine2	string	The second address line in a formatted address.	Address Line 2
city*	string	The city that the address is in.	City
country*	string	The country that the address is in.	Country
locality	string	An area of defined or undefined boundaries within a local authority or other legislatively defined area, usually rural or semi-rural in nature.	Locality
postcode	string	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	An extension of a postal code. E.g. the part following the dash in a US urban property address.	Postal Code Extension
stateOrProvince	string	The State or Province that the address is in.	State Or Province

Table 45 – Type FormattedAddress

8.2.4.4 Type MEFGeographicPoint

Description: A MEFGeographicPoint defines a geographic point through coordinates. Reference: MEF 79 Section 8.9.5 [10].

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	Description	MEF 80
spatialRef*	string	The spatial reference system used to determine the coordinates (e.g. "WGS84"). The system used and the value of this field are to be agreed during the onboarding process.	Spatial Reference
x*	string	The latitude expressed in the format specified by the spacialRef.	Latitude
y*	string	The longitude expressed in the format specified by the spacialRef.	Longitude
z	string	The elevation expressed in the format specified by the spacialRef.	Elevation

Table 46 – Type MEFGeographicPoint

[R65] The spatialRef value that can be used **MUST** be agreed between Buyer and Seller.

8.2.4.5 Type GeographicSubAddress

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

Name	Type	Description	MEF 80
buildingName	string	Allows for identification of places that require building name as part of addressing information.	Building Name
levelNumber	string	Used where a level type may be repeated e.g. BASEMENT 1, BASEMENT 2.	Level Number
levelType	string	Describes level types within a building.	Level Type
privateStreetName	string	Private streets internal to a property (e.g. a university) may have internal names that are not recorded by the land title office.	Private Street Name
privateStreetNumber	string	Private streets numbers internal to a private street.	Private Street Number
subUnit	MEFSubUnit[]	Representation of a MEFSubUnit. It is used for describing subunit within a subAddress e.g. BERTH, FLAT, PIER, SUITE, SHOP, TOWER, UNIT, WHARF.	Sub Unit List

Table 47 – Type GeographicSubAddress

8.2.4.6 Type GeographicAddressRef

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

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	Description	MEF 80
href	string	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	Identifier of the referenced Geographic Address. This identifier is assigned during a successful address validation request (Sonata Geographic Address Management API).	Fielded | Formatted | Geographic Address Label | Geographic Point Identifier

Table 48 – GeographicAddressRef

8.2.4.7 Type GeographicSiteRef

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

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	Description	MEF 80
href	string	Hyperlink to the referenced GeographicSite. Hyperlink **MAY** be used by the Seller in responses. Hyperlink **MUST** be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in MEF 80
id*	string	Identifier of the referenced Geographic Site.	Site Identifier

Table 49 – GeographicSiteRef

8.2.4.8 Type GeographicAddressLabel

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

Inherits from:

- RelatedPlaceRefOrValue

Name	Type	Description	MEF 80
externalReferenceId*	string	The unique reference to an Address as provided by the Administrative Authority.	Administrative Authority Address Label
externalReferenceType*	string	The organization or standard from the organization that administers this Geographic Address Label ensuring it is unique within the Administrative Authority. The value(s) to be used are to be agreed during the onboarding. For North American providers this would normally be CLLI (Common Language Location Identifier) code.	Administrative Authority

Table 50 – GeographicAddressLabel

8.2.4.9 Type MEFSubUnit

Description: Allows for subunit identification.

Name	Type	Description	MEF 80
subUnitNumber*	string	The discriminator used for the subunit, often just a simple number but may also be a range.	Sub Unit Name
subUnitType*	string	The type of subunit e.g. BERTH, FLAT, PIER, SUITE, SHOP, TOWER, UNIT, WHARF.	Sub Unit Type

Table 51 – Type MEFSubUnit

8.2.5 Notification Registration

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

8.2.5.1 Type EventSubscriptionInput

The `query` attribute is used to constrain the notification types that the Buyer is willing to receive to the callback endpoint. The `query` formatting complies to RCF 3986 [3] and TMF630 [13]. Every attribute defined in the Event model (from notification API) can be used in the `query`. Example:

```
"query":"eventType=quoteStateChangeEvent"
```

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

- `eventType=quoteStateChangeEvent,quoteItemStateChangeEvent` or
- `eventType=quoteStateChangeEvent&eventType=quoteItemStateChangeEvent`

Description: This class is used to register for Notifications.

Name	Type	Description	MEF 80
`callback*`	string	This callback value must be set to *host* property from the Buyer Notification API (`quoteNotification.api.yaml`).This property is appended with base path and notification resource path specified in that API to construct an URL to which the notification is sent. E.g. for "callback": `https://buyer.co/listenerEndpoint`, the create event notification will be sent to: `https://buyer.co/listenerEndpoint/mefApi/sonata/quoteNotification/v8/listener/quoteStateEvent`	Notification Target Information
`query`	string	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

Table 52 – Type EventSubscriptionInput

8.2.5.2 Type EventSubscription

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

Name	Type	Description	MEF 80
`id*`	string	An identifier of this Event Subscription assigned by the Seller when resource is created.	Not represented in MEF 80
`callback*`	string	The value provided by the Buyer in `EventSubscriptionInput` during notification registration.	Notification Target Information
`query`	string	This attribute is used to define notification registration constraints.	List of Notification Types

Table 53 – Type EventSubscription

8.2.6 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	Description	MEF 80
`quoteId*`	string	Unique (within the Seller quoting domain) identifier for the quote, as attributed by the Seller.	Seller Quote Identifier
`reason`	string	Allows the Buyer to specify a reason for the Cancel or Decline Quote request.	Reason

Table 54 – Type QuoteOperationData

8.2.7 Common

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

8.2.7.1 Type Duration

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

Name	Type	Description	MEF 80
amount*	integer	Duration (number of seconds, minutes, hours, etc.)	Duration Value
units*	TimeUnit	Time unit type	Duration Unit

Table 55 – Type Duration

8.2.7.2 Type Money

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

Name	Type	Description	MEF 80
unit	string	Currency (ISO 4217 [7] uses 3 letters to define the currency)	Currency
value	float	A positive floating-point number	Value

Table 56 – Type Money

8.2.7.3 Type Note

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

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

Table 57 – Type Note

8.2.7.4 enum MEFBuyerSellerType

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

Value	MEF 80
buyer	BUYER
seller	SELLER

Table 58 – Type MEFBuyerSellerType

8.2.7.5 Type RelatedContactInformation

Description: Contact data for a person or organization that is involved in a given context. It is specified by the Seller (e.g. Seller Contact Information) or by the Buyer.

Name	Type	Description	MEF 80
emailAddress*	string	Email address.	Contact Email Address
name*	string	Name of the contact.	Contact Name
number*	string	Phone number.	Contact Phone Number
numberExtension	string	Phone number extension.	Contact Phone Number Extension
organization	string	The organization or company that the contact belongs to.	Contact Organization
role*	string	A role of the particular contact in the request.	Not represented in MEF 80
postalAddress	FieldedAddress	Identifies the postal address of the person or office to be contacted.	Contact Postal Address

Table 59 – Type RelatedContactInformation

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=sellerContactInformation`
- Buyer Contact Information: `role=buyerContactInformation`
- Quote Item Location Contact: `role=quoteItemLocationContact`
- Quote Item Technical Contact: `role=quoteItemTechnicalContact`

8.2.7.6 Type TerminationError

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

Name	Type	Description
code	string	One of the following error codes: - `missingProperty`: The property the Seller has expected is not present in the payload - `invalidValue`: The property has an incorrect value - `invalidFormat`: The property value does not comply with the expected value format - `referenceNotFound`: The object referenced by the property cannot be identified in the Seller system - `unexpectedProperty`: Additional property, not expected by the Seller has been provided - `tooManyRecords`: the number of records to be provided in the response exceeds the Seller's threshold. - `otherIssue`: Other problem was identified (detailed information provided in a reason)
propertyPath	string	A pointer to a particular property of the payload that caused the validation issue. It is highly recommended that this property should be used. Defined using JavaScript Object Notation (JSON) Pointer [1].
value	string	Text to describe the reason of the termination.

Table 60 – Type TerminationError

8.2.7.7 *Type TimePeriod*

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

Name	Type	Description	MEF 80
`endDateTime`	date-time	End of the time period, using IETF RFC 3339 [3] format	Quote.Valid Until Date
`startDateTime`	date-time	Start of the time period, using IETF RFC 3339 [3] format. If you define a start, you must also define an end.	Not represented in MEF 80

Table 61 – Type TimePeriod

8.2.7.8 `enum` *TimeUnit*

Description: Represents a unit of time. Reference: MEF 79 Sections 8.4.3.1 and 8.4.3.2 [10].

Value	MEF 79
`calendarMonths`	CALENDAR_MONTHS
`calendarDays`	CALENDAR_DAYS
`calendarHours`	CALENDAR_HOURS
`calendarMinutes`	CALENDAR_MINUTES
`businessDays`	BUSINESS_DAYS
`businessHours`	BUSINESS_HOURS
`businessMinutes`	BUSINESS_MINUTES

Table 62 – Type TimeUnit

[R66] The clarification of what Business days, hours, and minutes mean **MUST** be done between the Buyer and the Seller during the onboarding process.

8.3 Notification API Data Model

Figure 26 presents the Quote Notification data model. The data types, requirements related to them and mapping to MEF 80 are discussed later in this section.

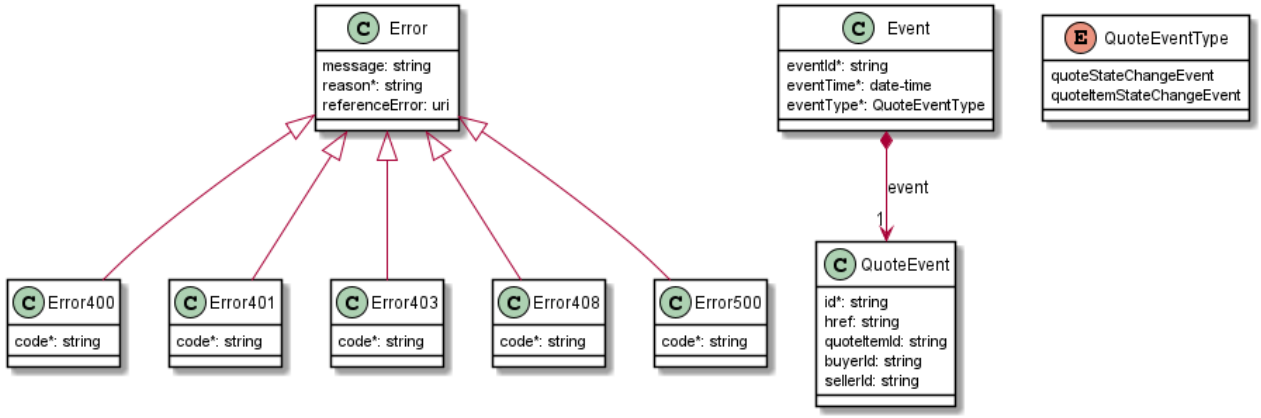

Figure 26 – Quote Notification Data Model

The Quote Management data model is used to construct requests and responses of the API endpoints described in Section 6.2.2.

8.3.1 Type Event

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

Name	Type	Description	MEF 80
eventId*	string	Id of the event.	Not represented in MEF 80
eventTime*	date-time	Datetime when the event occurred.	Not represented in MEF 80
eventType*	QuoteEventType	Indicates the type of the POQ event.	Notification Type
event*	QuoteEvent	Event data.	Not represented in MEF 80

Table 63 – Type Event

8.3.2 Type QuoteEvent

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

Name	Type	Description	MEF 80
id*	string	Id of the Quote attributed by quoting system.	Seller Quote Identifier
href	string	Hyperlink to access the Quote.	Not represented in MEF 80
quoteItemId	string	Id of the Quote Item (within the Quote) which state change triggered the event.	Quote Item Reference Number

Table 64 – Type QuoteEvent

8.3.3 enum QuoteEventType

Description: Type of the Event.

Value	MEF 80
quoteStateChangeEvent	QUOTE_STATE_CHANGE
quoteItemStateChangeEvent	QUOTE_ITEM_STATE_CHANGE

Table 65 – Type QuoteEventType

9 References

[1] IETF JSON Schema draft 7, *JSON Schema: A Media Type for Describing JSON Documents* and associated documents, by Austin Wright and Henry Andrews, March 2018. Copyright © 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

[2] IETF RFC 2119, *Key words for use in RFCs to Indicate Requirement Levels*, March 1997

[3] IETF RFC 3339, *Date and Time on the Internet: Timestamps*, July 2002, Copyright © The Internet Society (2002). All Rights Reserved.

[4] IETF RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax*, by Tim Berners-Lee and Roy T. Fielding and Larry M Masinter, January 2005. Copyright © The Internet Society (2005).

[5] IETF RFC 7231, *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*, by Roy T. Fielding and Julian Reschke, June 2014. Copyright © 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

[6] IETF RFC 8174, *Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words*, May 2017

[7] ISO 4217:2015, *Currency Codes*, August 2018

[8] Fielding, Roy Thomas, *Architectural Styles and the Design of Network-based Software Architectures (Ph.D),* 2000

[9] MEF 55.1, *Lifecycle Service Orchestration (LSO): Reference Architecture and Framework*, February 2021

[10] MEF 79, *Address, Service Site, and Product Ordering Qualification Management – Requirements and Use Cases*, November 2019

[11] MEF 80, *Quote Management Requirements and Use Cases*, July 2021

[12] OpenAPI Initiative, *OpenAPI Specification (OAS) v3.0.3*, February 2020

[13] TMF630 TM Forum, *TMF630 API Design Guidelines 4.0.1*, July 2020

[14] TMF648 TM Forum, *TMF648 Quote Management API REST Specification R19.0.1*, November 2019