Mplify Standard

Mplify 137.1

LSO Cantata and LSO Sonata Appointment and Work Order Management API - Developer Guide

November 2025

Disclaimer

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

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

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

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

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

Copyright

Table of Contents

List of Contributing Members

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

Member
Amartus
Colt Technology Services
Proximus

Table 1. Contributing Members

1. Abstract

This standard is intended to assist implementation of the Appointment functionality defined for the LSO Cantata and LSO Sonata Interface Reference Points (IRPs), for which requirements and use cases are defined in MEF 113 *Trouble Ticketing Requirements and Use Cases* [MEF 113]. This standard consists of this document and complementary API definitions for:

- Appointment Management and Appointment Notification.
- WorkOrder Management and WorkOrder Notification.

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

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

commit id: aaa03d484f98664a5a14f4f54f47b675d7efb3b8

- `productApi/workforce/appointment/appointmentManagement.api.yaml`
- `productApi/workforce/appointment/appointmentNotification.api.yaml`
- `productApi/workforce/workorder/workorderManagement.api.yaml`
- `productApi/workforce/workorder/workorderNotification.api.yaml`

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

commit id: 83d6edd0c70386058a9af6e677c069b498671da7

- `productApi/workforce/appointment/appointmentManagement.api.yaml`
- `productApi/workforce/appointment/appointmentNotification.api.yaml`
- `productApi/workforce/workorder/workorderManagement.api.yaml`
- `productApi/workforce/workorder/workorderNotification.api.yaml`

2. Terminology and Abbreviations

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

In addition, terms defined in the standards referenced below are included in this document by reference and are not repeated in the Table 2:

- MEF 55.1
- MEF 55.1.1
- MEF 113
- Mplify 150

Term	Description	Reference
Application Program Interface (API)	In the context of LSO, API describes one of the Management Interface Reference Points based on the requirements specified in an Interface Profile, along with a data model, the protocol that defines operations on the data and the encoding format used to encode data according to the data model. In this document, API is used synonymously with REST API	[MEF 55.1]
Appointment	In the context of this document denotes an arrangement between the Buyer and Seller for a Seller Technician to meet with the Buyer at a specified time and place.	[TMF646]
Buyer	In the context of this document, denotes the organization or individual acting as the customer in a transaction over a Cantata (Customer <-> Service Provider) or Sonata (Service Provider <-> Partner) Interface.	This document; adapted from [MEF 55.1.1]
Incident	An entry within a Seller's tracking system created by the context of this document, denotes a situation that is not part of normal operationSeller, which contains information about a Situation in the Seller's network that has a possible negative impact on the operability of the network on a Product for one or more Buyers	[MEF 113]
Issue	In the context of this document denotes a problem with a Product as experienced by the Buyer that is not part of normal operation.	[MEF 113]
Notification	A message sent from the Seller to the Buyer to inform about an event that has occurred regarding to a specific instance of a Ticket or an Incident	[MEF 113]
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.	[Mplify 150]
Responding Entity	The business organization that is acting on behalf of one or more Sellers. In the most common case, the Responding Entity	[Mplify 150]

	represents only one Seller and these terms are then synonymous.	
REST API	Representational State Transfer. REST provides a set of architectural constraints that, when applied as a whole, emphasizes scalability of component interactions, the generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems.	[REST]
Seller	In the context of this document, denotes the organization acting as the supplier in a transaction over a Cantata (Customer <-> Service Provider) or Sonata (Service Provider <-> Partner) Interface.	This document; adapted from [MEF 55.1.1]
Search Time Slot	This task resource is used to retrieve available time slots. One of these available time slots is after used to create or reschedule an appointment.	[TMF646]
Situation	In the context of this document denotes a problem that is not part of normal operation in the Seller's network that has a possible negative impact on the operability of a Product for one or more Buyers	[MEF 113]
Ticket	An entry within a Seller's tracking system created by the Buyer (or a third party on behalf of the Buyer), which contains information about an Issue impacting the normal operation of a Product, along with support interventions made by technical support staff, or third parties	[MEF 113]
WorkOrder	In the context of this document denotes a set of tasks to be confirmed and performed under the responsibility of a Technician at a given location	[MEF 113]

Table 2. Terminology

Term	Description	Reference
API	Application Program Interface	[MEF 55.1]
REST API	Representational State Transfer API	[REST]

Table 3. Abbreviations

3. Compliance Levels

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

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

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

4. Introduction

The Appointment and WorkOrder API allow the Buyer to create, retrieve, and update Appointment and WorkOrder as well as receive notifications and updates. This allows to manage issues and situations that are not part of the normal operations of the Product provided by the Seller.

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

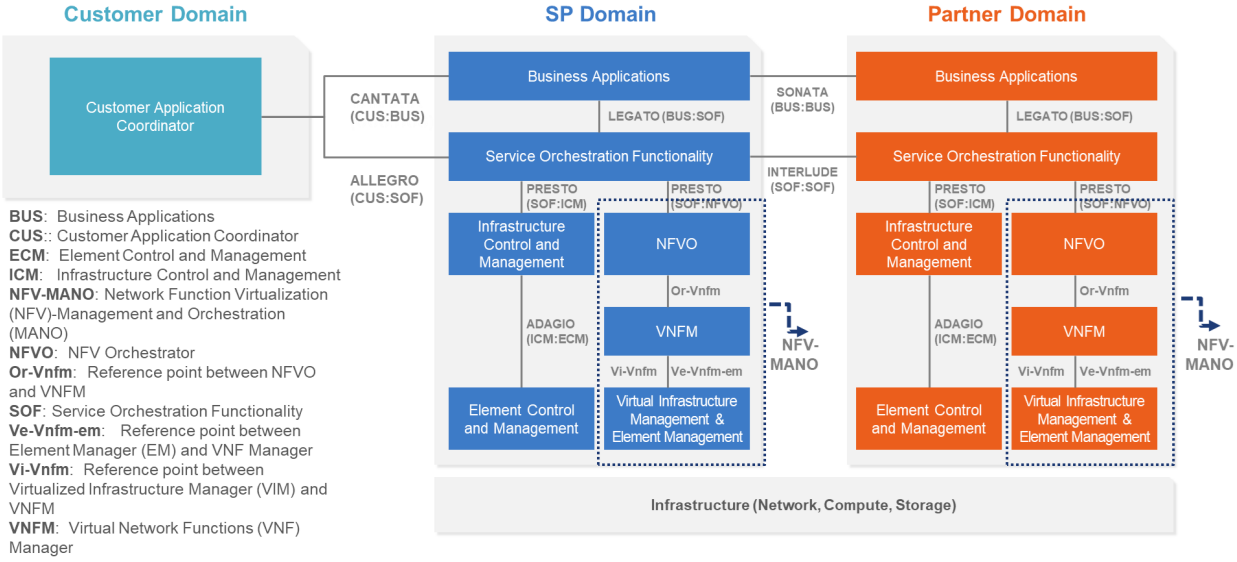


Figure 1. The LSO Reference Architecture

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

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

The business requirements and use cases for Appointment and WorkOrder are defined in MEF 113 *Trouble Ticketing Requirements and Use Cases* [MEF 113]. MEF 113 defines use cases that cover Trouble Ticket, Incident, Appointment and WorkOrder. This API and Developer Guide covers the Appointment and WorkOrder related use cases. The Trouble Ticket and Incident use cases are covered by MEF 124 LSO Cantata and LSO Sonata Trouble Ticket Management API - Developer Guide.

This document is structured as follows:

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

4.1. Conventions in the Document

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

4.2. Relation to Other Documents

This API implements the Appointment and WorkOrder related requirements and use cases that are defined in [MEF 113]. The API definition builds on *TMF646 Appointment Management API REST Specification R19.0.1* [TMF646] and *TMF697 Work Order Management API* [TMF697]. It is also related to *Cantata and LSO Sonata Trouble Ticket Management API - Developer Guide* [MEF 124], which covers outstanding Trouble Ticket use cases are described in MEF 113.

4.3. 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 (Mplify product specification data model)

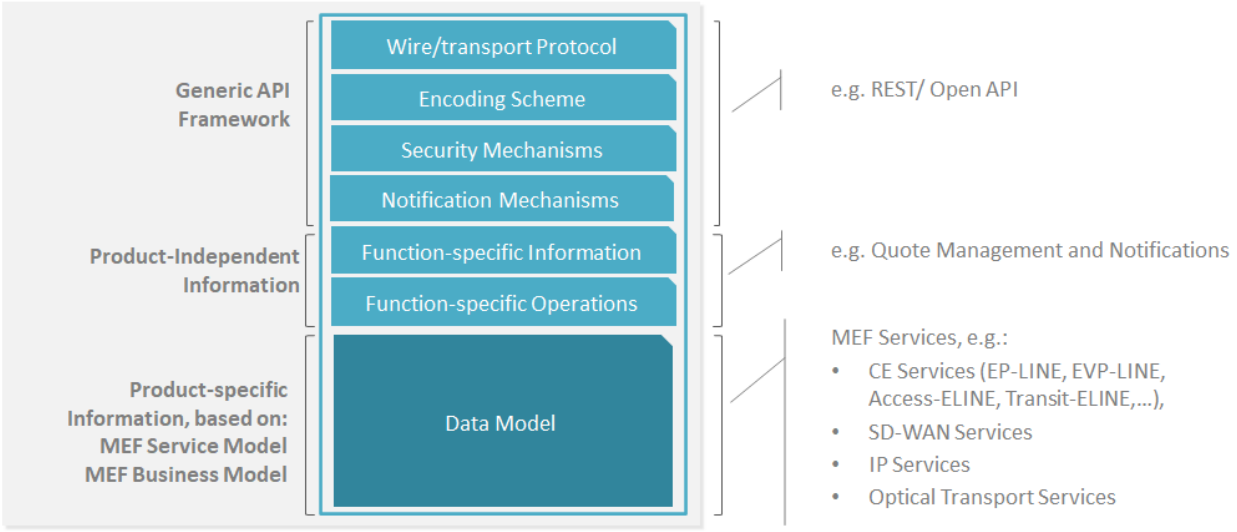


Figure 2. Cantata and Sonata API framework

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.
Finally, the product-specific information part of the framework focuses on Mplify product specifications that define business-relevant attributes and requirements for trading Mplify subscriber and Mplify operator services.

The Appointment and WorkOrder are product-agnostic in their nature and are not intended to carry any product-specific payloads. They only reference product from the inventory by `id`. It operates using the Generic API Framework and the Function-specific Information and Operations.

4.4. High-Level Flow

The Appointment and WorkOrder are 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 the Appointment and WorkOrder position within it.

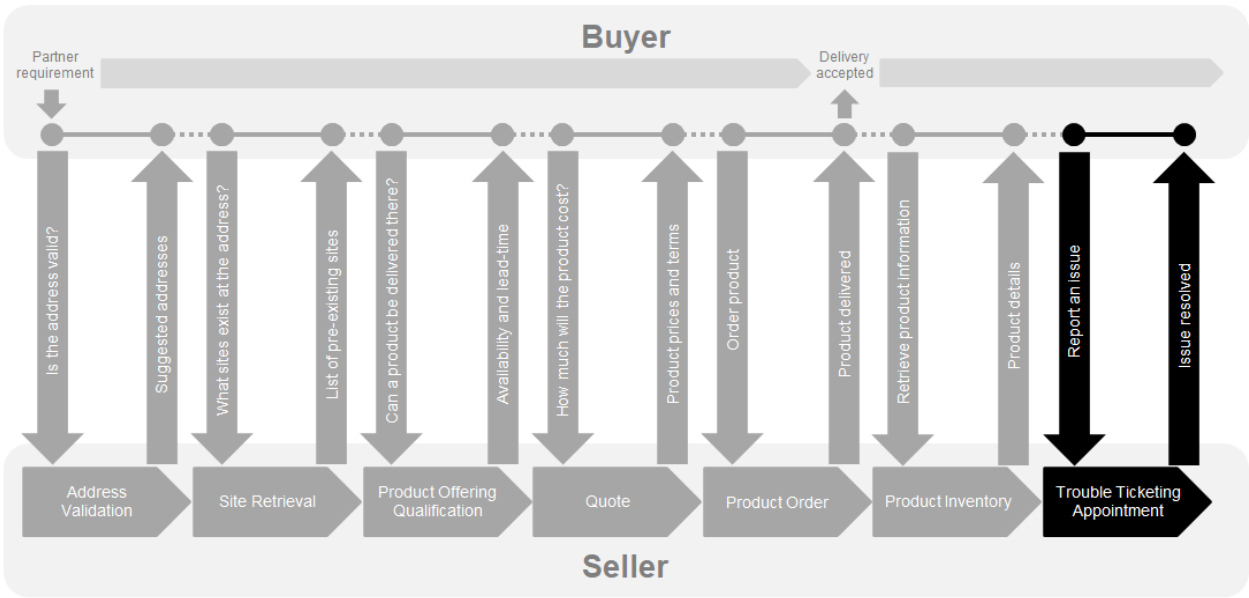


Figure 3. Cantata and Sonata End-to-End Function Flow

- 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 Geographic Site information including exact formats for Geographic Sites known to the Seller.
- Product Offering Qualification (POQ):
  - Allows the Buyer to check whether the Seller can deliver a product or set of products from among their product offerings at the geographic address or a Geographic Site specified by the Buyer; or modify a previously purchased product.
- Quote:
  - Allows the Buyer to submit a request to find out how much the installation of an instance of a Product Offering, an update to an existing Product, or a disconnect of an existing Product will cost.
- Product Order:
  - Allows the Buyer to request the Seller to initiate and complete the fulfillment process of an installation of a Product Offering, an update to an existing Product, or a disconnect of an existing Product at the address defined by the Buyer.
- Product Inventory:
  - Allows the Buyer to retrieve information about existing Product instances from Seller's Product Inventory.
- The Appointment and WorkOrder:
  - Allows the Buyer to create, retrieve, and update Appointment and WorkOrder as well as receive notifications about Appointment and WorkOrder' updates. This allows for managing issues and situations that are not part of normal operations of the Product provided by the Seller. It also allows for WorkOrder and appointment management.

5. API Description

This section presents the API structure and design patterns. It starts with the high-level use cases diagram. Then it describes the REST endpoints with use case mapping. Next, it gives an overview of the API resource model.

5.1. High-level use cases

Figure 4 presents a high-level use case diagram as specified in MEF 113 [MEF 113] in section 7. This picture aims to help understand endpoint mapping. Use cases are described extensively in chapter 6.

Note: As stated earlier, the scope of this API does not cover the Trouble Ticket related use cases. The diagram below lists only use cases that are part of the scope of this document. For easier requirements matching this document keeps the original MEF 113 numbering. The remaining use cases are covered by *LSO Cantata and LSO Sonata Trouble Ticket API Developer Guide* [MEF 124].

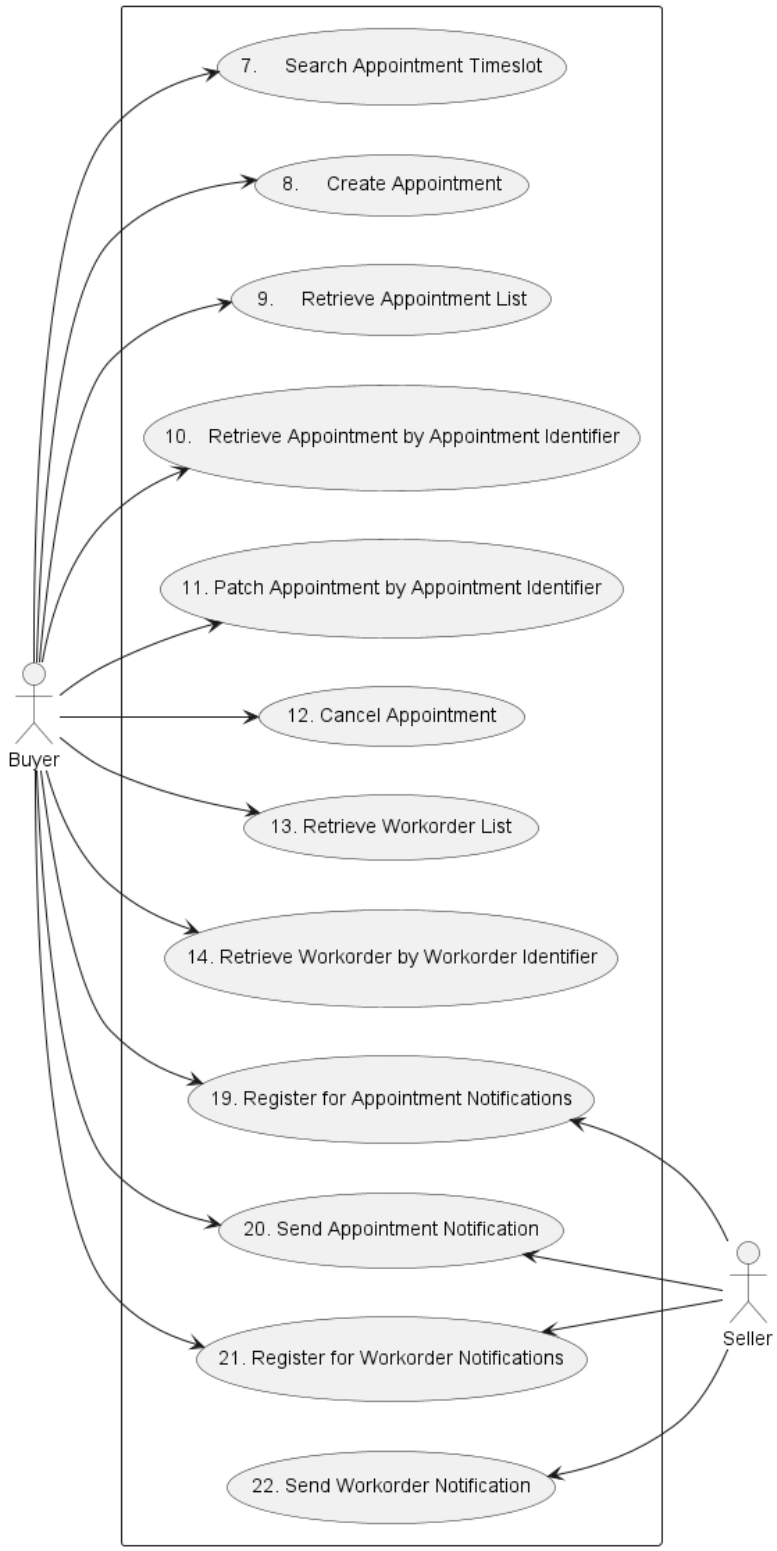


Figure 4. Use Cases

5.2. API Endpoint and Operation Description

5.2.1. Appointment API Endpoints

5.2.1.1. Seller side Appointment API Endpoints

Base URL for Cantata:

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

Base URL for Sonata:

`https://{{serverBase}}:{{port}}{{?/seller_prefix}}/mefApi/sonata/appointment/v2/`

The following API endpoints are implemented by the Seller and allow the Buyer to create, retrieve, modify an Appointment and register for Notifications. The endpoints and corresponding data model are defined in:

`/productApi/workforce/appointment/appointmentManagement.api.yaml`.

Appointment Use cases:

API endpoint	Description	MEF 113 Use Case mapping
`POST /searchTimeSlot`	A request by the Buyer to find a set of available time slots for scheduling or rescheduling an Appointment for a WorkOrder with a Seller Technician.	UC 7: Search Appointment Timeslot
`POST /appointment`	A request by the Buyer to create an Appointment for a WorkOrder with a Seller Technician.	UC 8: Create an Appointment
`GET /appointment`	The Buyer requests a list of Appointments from the Seller based on a set of specified filter criteria. The Seller returns a summarized list of Appointments.	UC 9: Retrieve na Appointment List
`GET /appointment/{{id}}`	The Buyer requests detailed information about a single Appointment based on an Appointment Identifier.	UC 10: Retrieve Appointment by Appointment Identifier
`PATCH /appointment/{{id}}`	A request by the Buyer to patch or reschedule an Appointment for a WorkOrder with a Seller Technician.	UC 11: Patch Appointment by Appointment Identifier
`POST /appointment/{{id}}/cancel`	A request by the Buyer to cancel an Appointment for a WorkOrder with a Seller Technician.	UC 12: Cancel Appointment by Appointment Identifier
`POST /hub`	The Buyer requests to subscribe to notifications.	UC 19: Register for Appointment Notifications
`GET /hub/{{id}}`	A request initiated by the Buyer to retrieve the details of the notification subscription.	UC 19: Register for Appointment Notifications

API endpoint	Description	MEF 113 Use Case mapping
`DELETE /hub/{{id}}`	A request initiated by the Buyer to instruct the Seller to stop sending notifications.	UC 19. Register for Appointment Notifications

Table 4. Seller side mandatory Appointment API endpoints

[R1] The Seller **MUST** support Appointment API endpoints listed in Table 4. [MEF113 R3]

5.2.1.2. Buyer side Appointment API Endpoints

Base URL for Cantata:

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

Base URL for Sonata:

```
https://{{serverBase}}:{{port}}
{{?/buyer_prefix}}/mefApi/sonata/appointmentNotification/v2/
```

The following API Endpoints are used by the Seller to post notifications to registered listeners. The endpoints and corresponding data model are defined in

`/productApi/workforce/appointment/appointmentNotification.api.yaml`.

Appointment Use cases

API endpoint	Description	MEF 113 Use Case mapping
`POST /listener /appointmentStatusChangeEvent`	The Seller sends a notification regarding an Appointment to the Buyer indicating one of the following Appointment Notification Types have occurred: `AppointmentStatusChangeEvent`	UC 20. Send Appointment Notification
`POST /listener /appointmentAttributeValueChangeEvent`	The Seller sends a notification regarding an Appointment to the Buyer indicating one of the following Appointment Notification Types have occurred: `AppointmentAttribute ValueChangeEvent`	UC 20. Send Appointment Notification

Table 5. Buyer side mandatory Appointment API endpoints

[R2] The Buyer **MUST** support Appointment API endpoints listed in Table 5. [MEF113 R3]

5.2.2. WorkOrder API Endpoints

5.2.2.1. Seller side WorkOrder API Endpoints

Base URL for Cantata:

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

Base URL for Sonata:

```
https://{{serverBase}}:{{port}}
{{?/seller_prefix}}/mefApi/sonata/workOrderManagement/v2/
```

The following API endpoints are implemented by the Seller and allow the Buyer to retrieve WorkOrders and register for Notifications. The endpoints and corresponding data model are defined in:

`/productApi/workforce/workorder/workorderManagement.api.yaml`.

WorkOrder API Use cases:

API endpoint	Description	MEF 113 Use Case mapping
`GET /workOrder`	The Buyer requests a list of WorkOrders from the Seller based on a set of specified filter criteria. The Seller returns a summarized list of WorkOrders.	UC 13. Retrieve WorkOrder List
`GET /workOrder/{{id}}`	The Buyer requests detailed information about a WorkOrder based on a WorkOrder Identifier.	UC 14. Retrieve WorkOrder by WorkOrder Identifier
`POST /hub`	The Buyer requests to subscribe to notifications.	UC 21: Register for WorkOrder Notifications
`GET /hub/{{id}}`	A request initiated by the Buyer to retrieve the details of the notification subscription.	UC 21: Register for WorkOrder Notifications
`DELETE /hub/{{id}}`	A request initiated by the Buyer to instruct the Seller to stop sending notifications.	UC 21. Register for WorkOrder Notifications

Table 6. Seller side mandatory WorkOrder API endpoints

[R3] The Seller **MUST** support WorkOrder API endpoints listed in Table 6. [MEF113 R4]

5.2.2.2. Buyer side WorkOrder API Endpoints

Base URL for Cantata:

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

Base URL for Sonata:

```
https://{{serverBase}}:{{port}}
{{?/buyer_prefix}}/mefApi/sonata/workOrderNotification/v2/
```

The following API endpoints are implemented by the Seller and allow the Buyer to register and send WorkOrder Notifications. The endpoints and corresponding data model are defined in:

`/productApi/workforce/appointment/workOrderNotification.api.yaml`.

WorkOrder API Use cases

API endpoint	Description	MEF 113 Use Case mapping
`POST /listener /workOrderCreateEvent`	The Seller sends a notification regarding an Appointment to the Buyer indicating one of the following WorkOrder Notification Types has occurred: `workOrderCreateEvent`	UC 22. Send WorkOrder Notification
`POST /listener /workOrderAttributeValueChangeEvent`	The Seller sends a notification regarding an Appointment to the Buyer indicating one of the following WorkOrder Notification Types has occurred: `workOrder StateChangeEvent`	UC 22. Send WorkOrder Notification
`POST /listener /workOrderAppointmentRequiredEvent`	The Seller sends a notification regarding an Appointment to the Buyer indicating one of the following WorkOrder Notification Types has occurred: `workOrder AppointmentRequiredEvent`	UC 22. Send WorkOrder Notification

Table 7. Buyer side mandatory WorkOrder API endpoints

[R4] The Buyer **MUST** support WorkOrder API endpoints listed in Table 7. [MEF113 R4]

5.3. Specifying the Buyer ID and the Seller ID

A business entity willing to represent multiple Buyers or multiple Sellers must follow requirements of [Mplify 150] chapter 8.8, which states:

> For requests of all types, there is a business entity that is initiating an Operation (called a Requesting Entity) and a business entity that is responding to this request (called the Responding Entity). In the simplest case, the Requesting Entity is the Buyer, and the Responding Entity is the Seller. However, in some cases, the

Requesting Entity may represent more than one Buyer and similarly, the Responding Entity may represent more than one Seller.

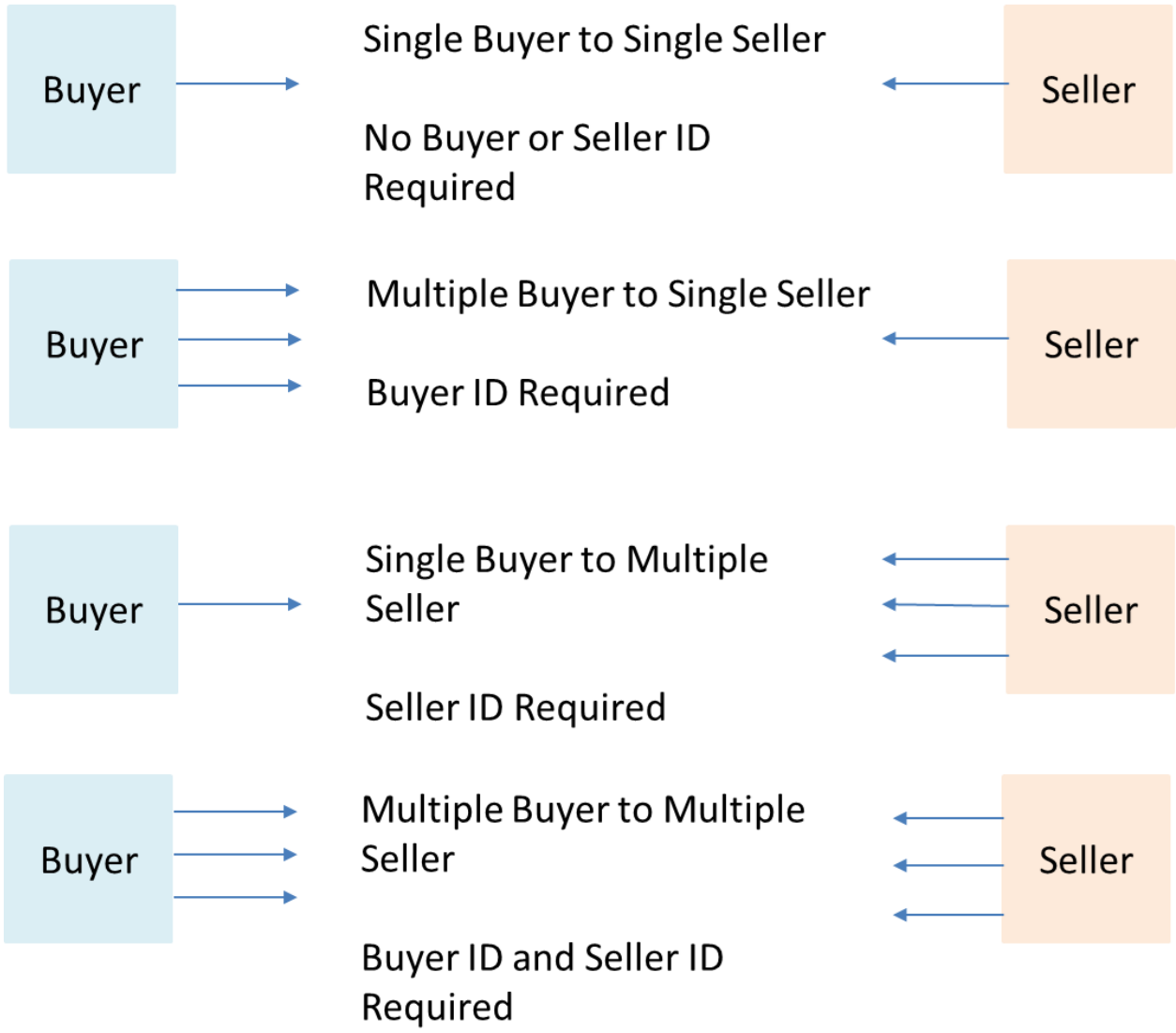


Figure 5. Buyer ID and Seller ID Examples

As shown in Figure 5, if a Requesting Entity representing a single Buyer is doing business with a Responding Entity representing a single Seller, Buyer and Seller IDs are not required to be passed between the two entities. If a Requesting Entity representing more than one Buyer is doing business with a Responding Entity representing a single Seller, the Buyer ID is required to be passed between the two entities. If a Requesting Entity representing a single Buyer is doing business with a Responding entity representing multiple Sellers, the Seller ID is required to be passed between the two entities. If a Requesting Entity representing multiple Buyers is doing business with a Responding Entity representing multiple Sellers, both the Buyer ID and the Seller ID are required to be passed between the entities.

While it is outside the scope of this specification, it is assumed that the Requesting Entity and the Responding Entity are aware of each other and can authenticate requests initiated by the other party. It is further assumed that the Requesting Entity knows:

- the list of Buyers the Requesting Entity represents when interacting with this Responding Entity; and

> - the list of Sellers that this Responding Entity represents to this Requesting Entity.
>
> It is also assumed that the Responding Entity knows:
>
> - the list of Sellers that this Responding Entity represents to this Requesting Entity and
> - the list of Buyers the Requesting Entity represents when interacting with this Responding Entity.

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

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

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

5.4. Providing the place information

When required the place information can be provided with the `RelatedPlaceRefOrQuery` type, which is presented in Figure 6.

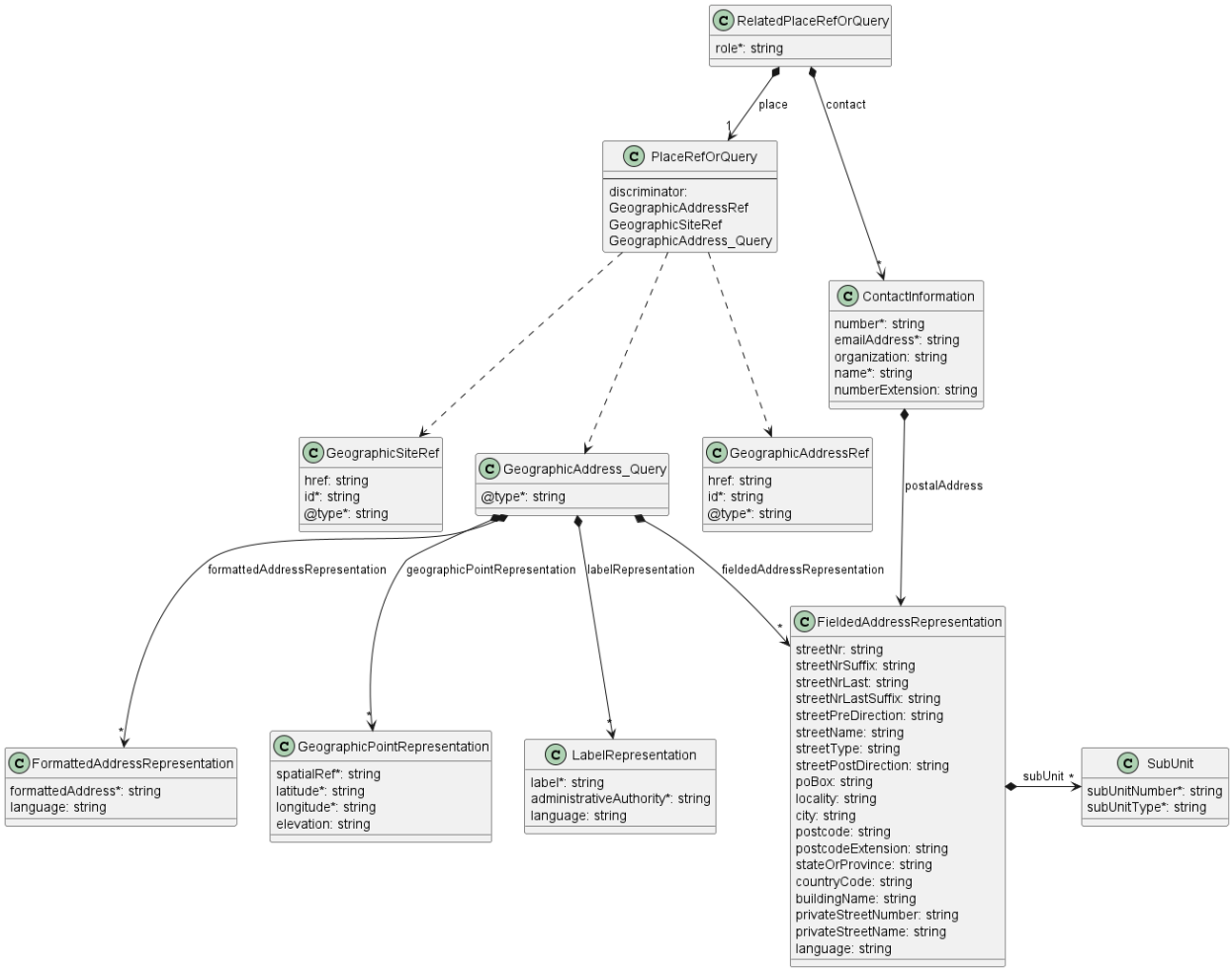


Figure 6. Data model - referring to a place

The `role` defines the function that the place plays for given context.

`contact` provides additional information about the person to contact to get access to this place in case such access is required.

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

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

One or more of these representations can be used to describe a single place.

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

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

5.5. Model Structural Validation

The structure of the HTTP payloads exchanged via Trouble Ticket API endpoints is defined using OpenAPI version 3.0.

[R7] Implementations **MUST** use payloads that conform to these definitions.

[R8] The Buyer and the Seller **MUST NOT** use any operation, entity or attribute that is not explicitly defined or allowed by this standard.

5.6. Security Considerations

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

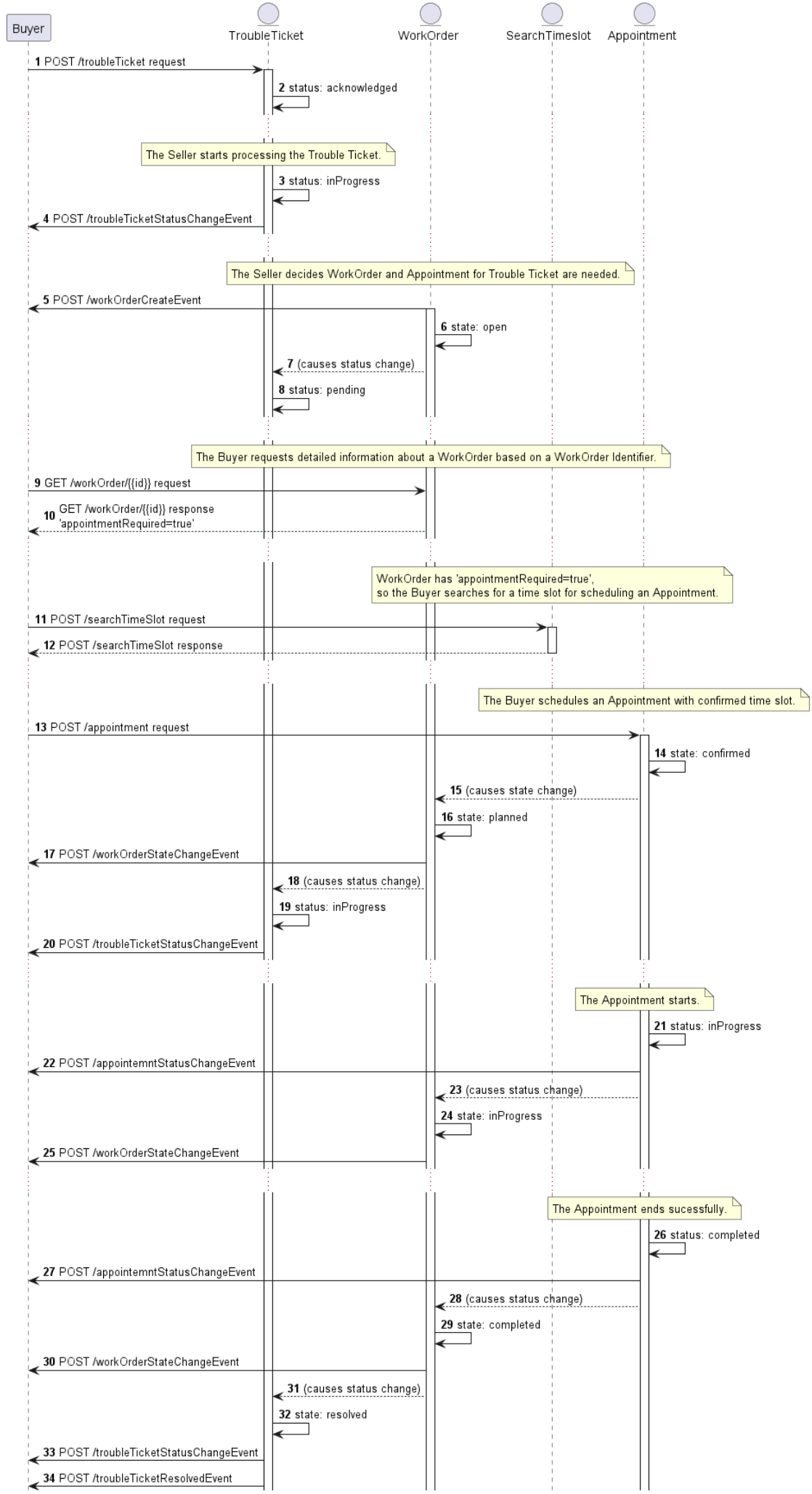
6. API Interactions and Flows

This section provides a detailed insight into the API functionality, use cases, and flows. It starts with Figure 7 and Table 8, presenting a list and short description of all business use cases then presents the variants of end-to-end interaction flows, and in the following subchapters describes the API usage flow and examples for each of the use cases.

MEF 113 defines use cases related to three domains:

- Trouble Ticket
- WorkOrder
- Appointment

Figure 7 presents an example of an end-to-end flow that shows dependencies between all the domains:


Buyer
TroubleTicket
WorkOrder
SearchTimeslot
Appointment
1 POST /troubleTicket request
2 status: acknowledged
The Seller starts processing the Trouble Ticket.
3 status: inProgress
4 POST /troubleTicketStatusChangeEvent
The Seller decides WorkOrder and Appointment for Trouble Ticket are needed.
5 POST /workOrderCreateEvent
6 state: open
7 (causes status change)
8 status: pending
The Buyer requests detailed information about a WorkOrder based on a WorkOrder Identifier.
9 GET /workOrder/{{id}} request
10 GET /workOrder/{{id}} response 'appointmentRequired=true'
WorkOrder has 'appointmentRequired=true', so the Buyer searches for a time slot for scheduling an Appointment.
11 POST /searchTimeSlot request
12 POST /searchTimeSlot response
The Buyer schedules an Appointment with confirmed time slot.
13 POST /appointment request
14 state: confirmed
15 (causes state change)
16 state: planned
17 POST /workOrderStateChangeEvent
18 (causes status change)
19 status: inProgress
20 POST /troubleTicketStatusChangeEvent
The Appointment starts.
21 status: inProgress
22 POST /appointemntStatusChangeEvent
23 (causes status change)
24 state: inProgress
25 POST /workOrderStateChangeEvent
The Appointment ends sucessfully.
26 status: completed
27 POST /appointemntStatusChangeEvent
28 (causes status change)
29 state: completed
30 POST /workOrderStateChangeEvent
31 (causes status change)
32 state: resolved
33 POST /troubleTicketStatusChangeEvent
34 POST /troubleTicketResolvedEvent

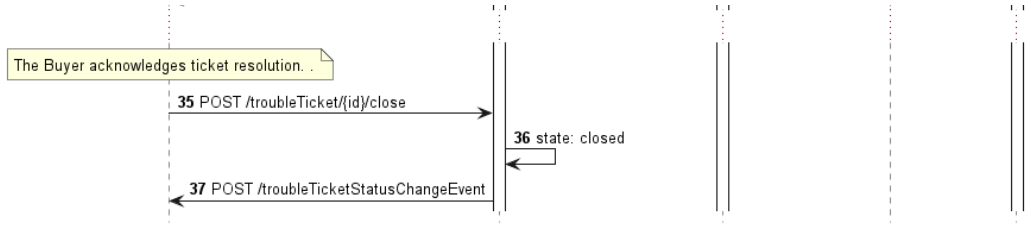


Figure 7. End-to-End API Flows

- (1) The Buyer experiences the issue in the network and creates the Trouble Ticket.
- (2) The Seller creates the Trouble Ticket and sets the status: `acknowledged`.
- (3) Trouble Ticket status changes to `inProgress`
- (4) The Buyer gets information about Trouble Ticket status change.
- (5-6)The Seller decides that a WorkOrder with an Appointment is needed to resolve the issue. The Seller creates a WorkOrder in state `open` and sends a `workOrderCreateEvent`.
- (7-8)Workorder state changes if an Appointment is needed (WorkOrder has `appointmentRequired=true`) causing the Trouble Ticket status to change to `pending`
- (9) The Buyer requests detailed information about the WorkOrder.
- (10) The Buyer proposes time slots for scheduling an Appointment if the WorkOrder requires an Appointment (if the parameter is set to `appointmentRequired=true`)
- (11) The Buyer requests detailed information about the available time slot.
- (12) The Seller responds with the list of available time slots.
- (13) The Buyer schedules an Appointment with the agreed time slot.
- (14) The Seller sets the Appointment status to `confirmed`.
- (15-16) Appointment creation causes the WorkOrder state change to `planned`
- (17) The Buyer gets information about WorkOrder state change.
- (18-19) WorkOrder state change causes the Trouble Ticket status change to `inProgress`
- (20) The Buyer gets information about Trouble Ticket status change.
- (21) Appointment status changes to `inProgress`
- (22) The Buyer gets information about Appointment status change.
- (23-24) Appointment status change causes the WorkOrder state change back to `inProgress`
- (25) The Buyer gets information about WorkOrder state change.
- (26) Appointment status goes to `completed`.
- (27) The Buyer gets information about Appointment status change.
- (28-29) Appointment status change causes the WorkOrder state change to `completed`.
- (30) The Buyer gets information about WorkOrder state change.
- (31-32) If there are no additional open WorkOrders for a Trouble Ticket, then the WorkOrder state change causes the Trouble Ticket status change back to `resolved`.
- (33) The Buyer gets information about Trouble Ticket status change.
- (34) The Buyer gets information about resolving Trouble Ticket.
- (35) The Buyer acknowledges ticket resolution.
- (36) Trouble Ticket status changes to `closed`
- (37) The Buyer gets information about the Trouble Ticket status change.

Note: (...) This symbol on the diagram means that the next operation isn't automated.

Use Case #	Use Case Name	Use Case Description

Use Case #	Use Case Name	Use Case Description
7	Search Appointment Timeslot	A request by the Buyer to find a set of available time slots for scheduling or rescheduling an Appointment for a WorkOrder with a Seller Technician.
8	Create Appointment List	A request by the Buyer to create an Appointment for a WorkOrder with a Seller Technician.
9	Retrieve Appointment List	The Buyer requests a list of Appointments from the Seller based on a set of specified filter criteria. The Seller returns a summarized list of Appointments.
10	Retrieve Appointment by Appointment Identifier	The Buyer requests detailed information about a single Appointment based on an Appointment Identifier.
11	Patch Appointment by Appointment Identifier	A request by the Buyer to patch or reschedule an Appointment for a WorkOrder with a Seller Technician.
12	Cancel Appointment by Appointment Identifier	A request by the Buyer to cancel an Appointment for a WorkOrder with a Seller Technician.
13	Retrieve WorkOrder List	The Buyer requests a list of Work orders from the Seller based on a set of specified filter criteria. The Seller returns a summarized list of Work orders.
14	Retrieve WorkOrder by WorkOrder Identifier	The Buyer requests detailed information about a WorkOrder based on a WorkOrder Identifier.
19	Register for Appointment Notifications	The Buyer requests to subscribe to Appointment Notifications.
20	Send Appointment Notification	The Seller sends a notification regarding an Appointment to the Buyer indicating one of the following Appointment Notification Types has occurred: `appointmentAttributeValueChangeEvent`, `appointmentStatusChangeEvent`
21	Register for WorkOrder Notifications	The Buyer requests to subscribe to WorkOrder Notifications.

Use Case #	Use Case Name	Use Case Description
22	Send WorkOrder Notification Identifier	The Seller sends a notification regarding an Appointment to the Buyer indicating one of the following WorkOrder Notification Types has occurred: `workOrderCreateEvent`, `workOrderStateChangeEvent`, `workOrderAppointmentRequiredEvent`

Table 8. Use Cases description

The detailed business requirements of each of the use cases are described in sections 7 and 8 of MEF 113 [MEF 113].

6.1. Use Case 7: Search Appointment Timeslot

This is the initial step for Appointment processing.

6.1.1. Interaction flow

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

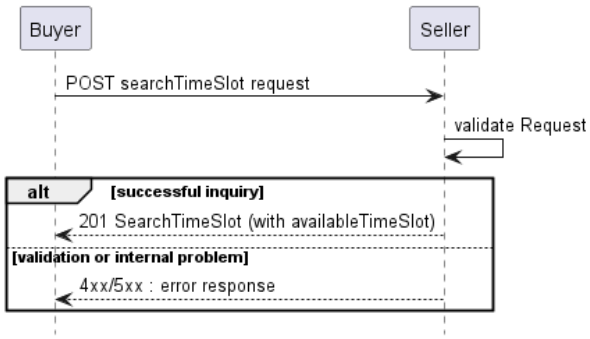


Figure 8. Use Case 7: Appointment Timeslot create request flow

The Buyer sends a request with a `SearchTimeSlot_Create` type in the body.

6.1.2. Create Appointment Timeslot - Request

Figure 9 presents the data model of the Appointment Timeslot. The model of the request message (`SearchTimeSlot_Create`) is a subset of the `SearchTimeSlot` 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.

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

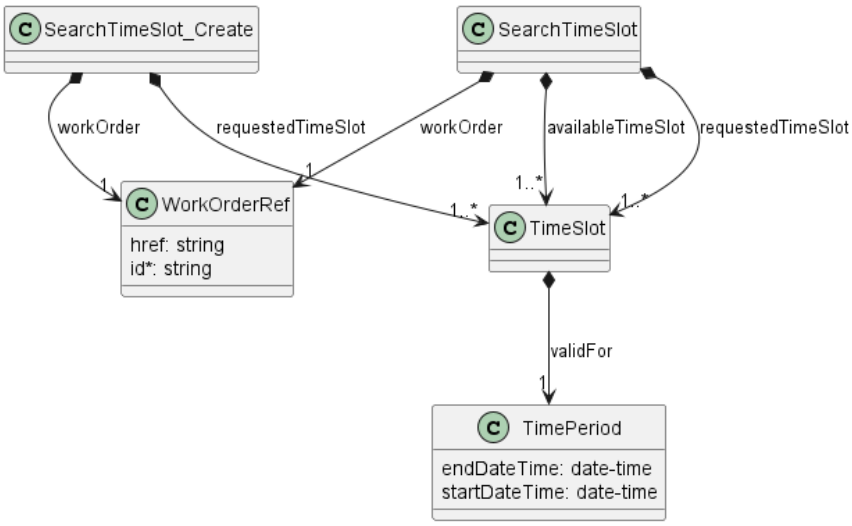


Figure 9. Create Appointment Timeslot Model

The snippet below presents an example of the Create Appointment Timeslot Request:

`SearchTimeSlot` **Create**

```
{
  "requestedTimeSlot": [
    {
      "validFor": {
        "startDateTime": "2022-05-19T08:00:00.170Z",
        "endDateTime": "2022-05-21T16:00:00.170Z"
      }
    }
  ],
  "workOrder": {
    "id": "01494079-6c79-4a25-83f7-48284196d44d"
  }
}
```

[R9] The Buyer's request **MUST** include the following attributes: [MEF113 R71]

- `workOrder`
- `requestedTimeSlot`

6.1.3. Create Appointment Timeslot - Response

The Seller responds with a `SearchTimeSlot` type, which adds some attributes to the `SearchTimeSlot_Create` that was used in the Buyer's request.

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

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

```
{
  "availableTimeSlot": [
    {
      "validFor": {
        "startDateTime": "2022-05-21T08:00:00.125Z",
        "endDateTime": "2022-05-21T16:00:00.125Z"
      }
```

```
    }
  ],
  "requestedTimeSlot": [
    {
    << as provided by the Buyer >>
      "validFor": {
        "startDateTime": "2022-05-19T08:00:00.170Z",
        "endDateTime": "2022-05-21T16:00:00.170Z"
      }
    }
  ] ,
  "workOrder": {
    << as provided by the Buyer >>
      "id": "01494081-6c79-4a25-42f7-48284196d55d"
    }
}
```

[R10] The Seller **MUST** return an error `Error422` if any of the included attributes in the Buyer's request are invalid. [MEF113 R73]

[R11] The Seller's response **MUST** return an empty list of `availableTimeSlot` if no Seller resources are available for an appointment that falls within any of the `requestedTimeSlot`. [MEF113 R74]

[R12] The Seller's response **MUST** include one or more `availableTimeSlot`, if Seller resources are available for an appointment that falls within the `requestedTimeSlot`. [MEF113 R77]

6.2. Use Case 8: Create Appointment

This is the initial step for Appointment processing.

6.2.1. Interaction flow

The flow of this use case is very simple and is described in Figure 10.

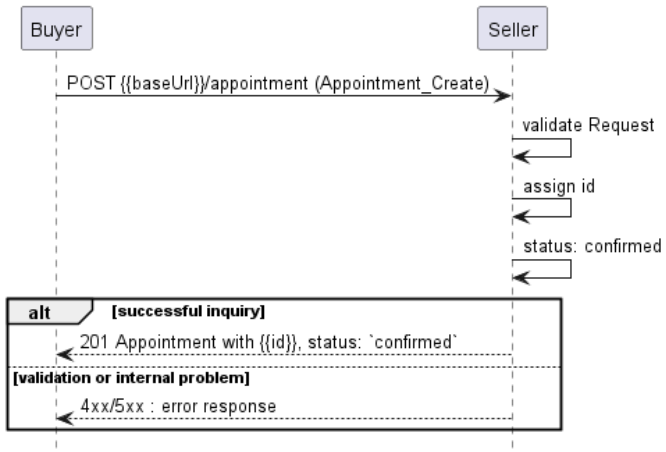


Figure 10. Use Case 8: Appointment create request flow

The Buyer sends a request with an `Appointment_Create` type in the body. The Seller performs request validation, assigns an `id`, and returns an `Appointment` type in the response body, with a `status` set to `confirmed`. From this point, the Appointment is ready for further processing. The Buyer must track the progress of the process by subscribing for notifications (see chapter 6.9). The flow example with the use of Notifications is presented in Figure 11.

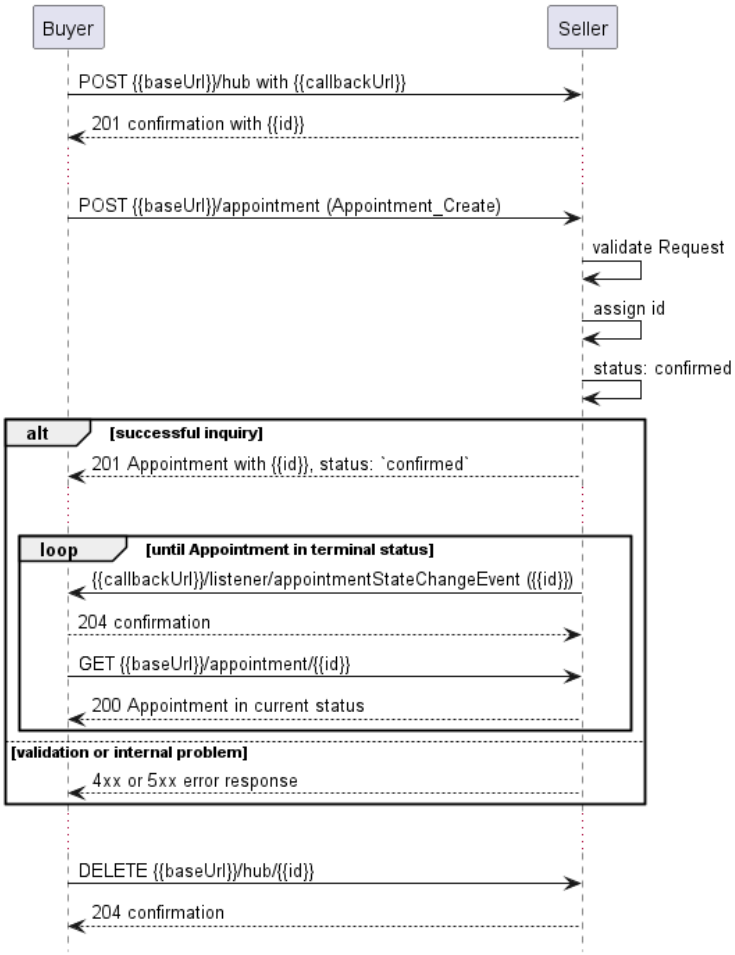


Figure 11. Appointment progress tracking - Notifications

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

6.2.2. Create Appointment - Request

Figure 12 presents the data model of the Appointment. The model of the request message (`Appointment_Create`) is a subset of the `Appointment` 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.

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

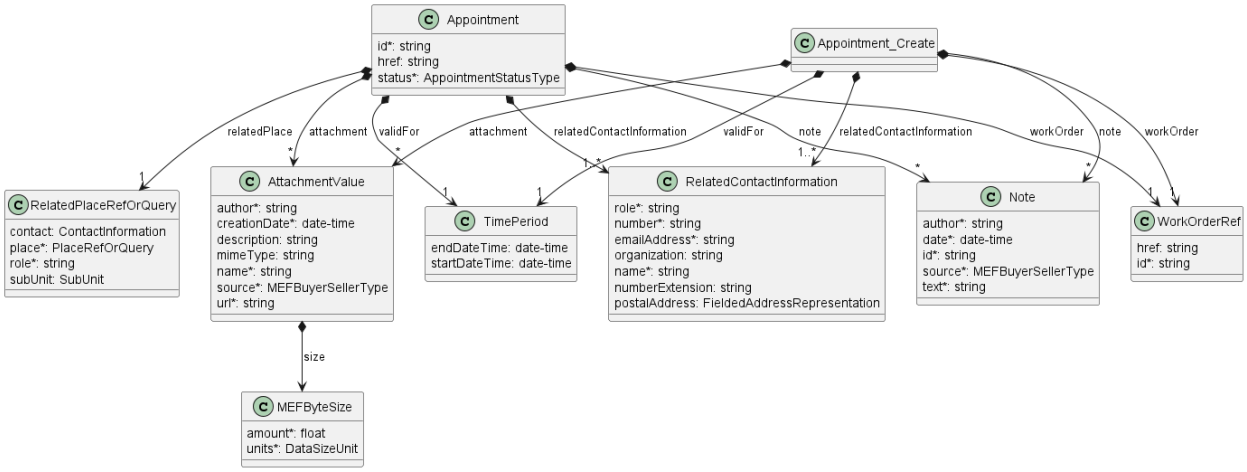


Figure 12. Create Appointment Model

The snippet below presents an example of the Create Appointment Request:

`Appointment` Create

```
{
  "attachment": [
    {
      "author": "John Example",
      "creationDate": "2022-05-19T10:04:18.812Z",
      "description": "Print screen from the assurance system",
      "mimeType": "image/jpeg",
      "name": "Alarm",
      "size": {
        "amount": 5.3,
        "units": "MBYTES"
      },
      "source": "buyer",
      "url": "https://buyer.mef.com/documents/00000000-0000-1111-2222-000000001111"
    }
  ],
  "relatedContactInformation": [
    {
      "emailAddress": "john.example@buyer.mef.com",
      "name": "John Example",
      "number": "+12-345-678-90",
      "organization": "Buyer Example Co.",
      "role": "buyerAppointmentContact"
    }
  ],
  "validFor": [
    {<< as provided in availableTimeSlot  >>
      "startDateTime": "2022-05-21T08:00:00.125Z",
      "endDateTime": "2022-05-21T16:00:00.125Z"
    }
  ],
  "workOrder": {
    "id": "01494081-6c79-4a25-42f7-48284196d55d"
  },
  "note": [
    {
      "id": "note-1",
      "author": "John Example",
      "date": "2021-06-02T14:25:11.090Z",
      "source": "buyer",
      "text": "Couldn't reach the support on phone."
    }
  ]
}
```

[R13] The Buyer's request **MUST** include the following attributes: [MEF113 R78]

- `relatedContactInformation` - item with `role=buyerAppointmentContact`
- `relatedContactInformation` - item with `role=appointmentPlaceContact`
- `workOrder`
- `validfor`

[O1] The Buyer's request **MAY** include the `note` and `attachment` attributes: [MEF113 O14]

6.2.3. Create Appointment - Response

The Seller responds with an `Appointment` type, which adds some attributes to the `Appointment_Create` that was used in the Buyer's request.

```
{
  "id": "00000000-4444-5555-6666-000000000987",
  "status": "confirmed",
  "attachment": [
    {<< as provided by the Buyer >>
      "author": "John Example",
      "creationDate": "2022-05-19T10:04:18.812Z",
      "description": "Print screen from the assurance system",
      "mimeType": "image/jpeg",
      "name": "Alarm",
      "size": {
        "amount": 5.3,
        "units": "MBYTES"
      },
      "source": "buyer",
      "url": "https://buyer.mef.com/documents/00000000-0000-1111-2222-000000001111"
    }
  ],
  "note": [
    {<< as provided by the Buyer >>
      "id": "note-1",
      "author": "John Example",
      "date": "2021-06-02T14:25:11.090Z",
      "source": "buyer",
      "text": "Couldn't reach the support on phone."
    }
  ],
  "workOrder": {<< as provided by the Buyer >>
    "id": "01494081-6c79-4a25-42f7-48284196d55d"
  },
  "relatedPlace": {
    "place": {
        "@type": "GeographicAddressRef",
        "id": "00000000-0000-0030-0305-873500002010"
    },
    "role": "APPOINTMENT_LOCATION"
  },
  "validFor": {
    "startDateTime": "2022-05-21T08:00:00.125Z",
    "endDateTime": "2022-05-21T16:00:00.125Z"
  },
  "relatedContactInformation": [
    {<< as provided by the Buyer >>
      "emailAddress": "john.example@buyer.mef.com",
      "name": "John Example",
      "number": "+12-345-678-90",
      "organization": "Buyer Example Co.",
      "role": "buyerAppointmentContact"
    }
  ]
}
```

The response to the create request does not contain all possible attributes.

[R14] The Seller **MUST** return an error `Error422` if the `validFor` is not valid. [MEF113 R81]

[R15] The Seller **MUST** return an error `Error422` if a Seller technician is not available for an appointment at the `relatedPlace` for the specified `requestedTimeSlot`. [MEF113 R82]

[R16] The Seller's response **MUST** include the Buyer following attributes. [MEF113 R83-R84]

- `id`
- `relatedContactInformation` - item with `role=buyerAppointmentContact`
- `relatedContactInformation` - item with `role=appointmentPlaceContact`
- `relatedContactInformation` - item with `role=sellerAppointmentContact`
- `relatedPlace`
- `status`
- `workOrder`
- `validFOr`

[O2] The Seller's response **MAY** add the `note` and `attachment` attributes. [MEF113 O15]

[R17] The Seller **MUST** set the appointment `status` to `confirmed`. [MEF113 R85]

[R18] The Seller **MUST** return an error `Error422` if the `workOrder` references a workOrder that contains a reference to an appointment with `status` of `confirmed` or `inProgress`. [MEF113 R86]

[R19] The Seller **MUST** return an error `Error422` if the related workOrder `state` is not `open`. [MEF113 R87]

[R20] After creating the Appointment instance, the Seller **MUST** add the `appointment.id` to the `appointment` attribute of the related WorkOrder. [MEF113 R88]

[R21] The Seller **MUST** set the `appointmentRequired` attribute to `false` in the related WorkOrder. [MEF113 R89]

[R22] The Seller **MUST** set the related WorkOrder `state` to `planned`. [MEF113 R90]

[R23] The Seller **MUST** update the `status` of the Trouble Ticket, for which this Appointment was created to `inProgress`. [MEF113 R91]

6.2.4. Appointment - Lifecycle

Figure 13 presents the Appointment state machine:

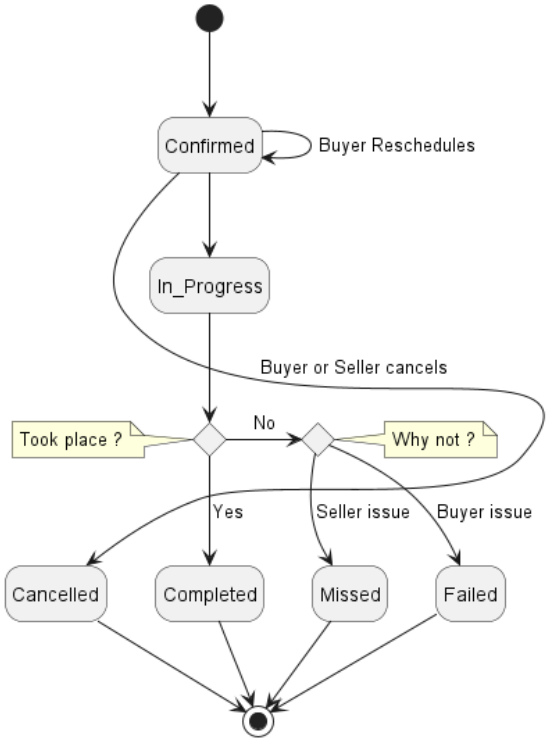


Figure 13. Appointment State Machine

After receiving the request, the Seller performs a validation of the message. If any problem is found an Error response is provided. If the validation passes a response is provided with `Appointment` in `confirmed` status. When `Appointment` has `inProgress` status, it can no longer be cancelled (point of no return), this is up to the Seller's discretion. Then the Seller starts working on resolving the issue and moves the Appointment to `completed` state. The Appointment changes a status to `missed` when it didn't take place because of the Seller issue or to `failed` if caused the Buyer issue. Buyer or Seller can `cancel` the Appointment.

Table 9 presents the mapping between the API `status` names (aligned with TMF) and the MEF 113 naming, together with statuses' description.

status	MEF 113 name	Description
`confirmed`	SCHEDULED	The Buyer has negotiated and confirmed the Appointment with the Seller (scheduled).
`inProgress`	IN_PROGRESS	The Appointment can no longer be cancelled (point of no return), this is up to the Seller's discretion.
`cancelled`	CANCELLED	The Appointment was cancelled by the Buyer or the Seller determined that an Appointment was not required. This is a terminal state.
`missed`	MISSED	The Appointment did not take place, because of a Seller related issue. For example, no Seller Technician was available on the date of the appointment. .
`failed`	FAILED	The Appointment did not take place, because of an issue with the Buyer. For example, Seller Technician was unable to get to the Appointment due to an incorrect location or unable to get access to the Buyer's site. This is a terminal state.

status	MEF 113 name	Description
`completed`	COMPLETED	The Appointment took place as confirmed. This is a terminal state.

Table 9. Appointment statuses

[R24] The Seller **MUST** support all Appointment `status` and associated transitions. [MEF113 R173]

[R25] If the Seller wants to reschedule an `Appointment`, the **MUST** update the Appointment `status` to `cancelled` and request the Buyer to schedule a new `Appointment` by setting `validFor` to TRUE in the related `WorkOrder`. [MEF113 R174]

[R26] The Seller **MUST** send an `AppointmentStatusChangeEvent` whenever an Appointment `status` change has occurred. [MEF113 R175]

[R27] The Seller **MUST** send an `AppointmentAttributeValueChangeEvent` whenever the Seller has updated any of the following attributes: [MEF113 R176]

- `attachment`
- `note`
- `relatedContactInformation` - item with `role=sellerAppointmentContact`

6.3. Use Case 9: Retrieve Appointment List

[O3] The Buyer **MAY** use `GET /appointment` operation with the following filter criteria : [MEF113 O16]

- `geographicSiteId`
- `geographicAddressId`
- `workOrder`
- `status`
- `validFor.gt`
- `validFor.lt`

The Buyer may also ask for pagination with the use of the `offset` and `limit` parameters. The filtering and pagination attributes must be specified in URI query format RFC3986.

```
https://serverRoot/mefApi/sonata/appointment/v2/appointment?status=confirmed&limit=10&offset=0
```

The example above shows a Buyer's request to get all Appointments that are in the `confirmed` status. Additionally, the Buyer asks only for a first (`offset=0`) pack of 10 results (`limit=10`) to be returned. The correct response (HTTP code `200`) in the response body contains a list of `Appointment_Find` objects matching the criteria. To get more details (e.g. the item level information), the Buyer has to query a specific `Appointment` by `id`.

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

- `X-Total-Count` header attribute with the total number of available results

- `X-Pagination-Throttled` header set to `true`

[D1] The Seller **SHOULD** support the pagination mechanism.

[CR1]<[D1] Seller **MUST** use either `X-Total-Count` or `X-Pagination-Throttled` to indicate that the page was truncated and additional results are available.

[R28] The Seller **MUST** return the following attributes: [MEF113 R93]

- `id`
- `href`
- `workOrder`
- `relatedPlace`
- `status`
- `validFor`

[R29] The Seller response **MUST** include all Appointments with an `validFor` that partially or completely overlaps with the `validFor.gt` and `validFor.lt` filter criteria. [MEF113 R94]

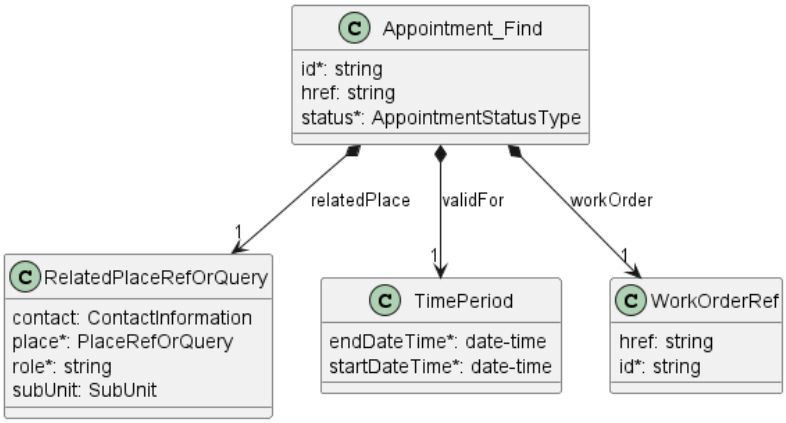


Figure 14. Use Case 9: Retrieve Appointment List - Model

6.4. Use Case 10: Retrieve Appointment by Appointment Identifier

The Buyer can get detailed information about the Appointment from the Seller by using a `GET /appointment/{{id}}` operation.

[R30] The Buyer **MUST** include the `id`. [MEF113 R105]

[R31] In case `id` does not allow to find a `Appointment` instance in Seller's system, an error `Error404` **MUST** be returned. [MEF113 R96]

[R32] The Seller's response **MUST** include the following attributes: [MEF113 R98]

- `id`
- `href`
- `relatedPlace`
- `status`
- `validfor`
- `workOrder`
- `relatedContactInformation`:
  - item with `role=appointmentPlaceContact`
  - item with `role=buyerAppointmentContact`

- item with `role=sellerAppointmentContact`

[R33] The Seller's response **MUST** include all of the following optional attributes, if they were set by the Buyer or the Seller: [MEF113 R99]

- `attachment`
- `note`

6.5. Use Case 11: Patch Appointment by Appointment Identifier

The update operation is realized with the use of the REST PATCH operation. For that purpose, a specialized type `Appointment_Update` is provided. It consists of attributes limited to a subset that includes only the Buyer updatable attributes.

The PATCH usage recommendation follows TMF 621 json/merge (https://tools.ietf.org/html/rfc7386).

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

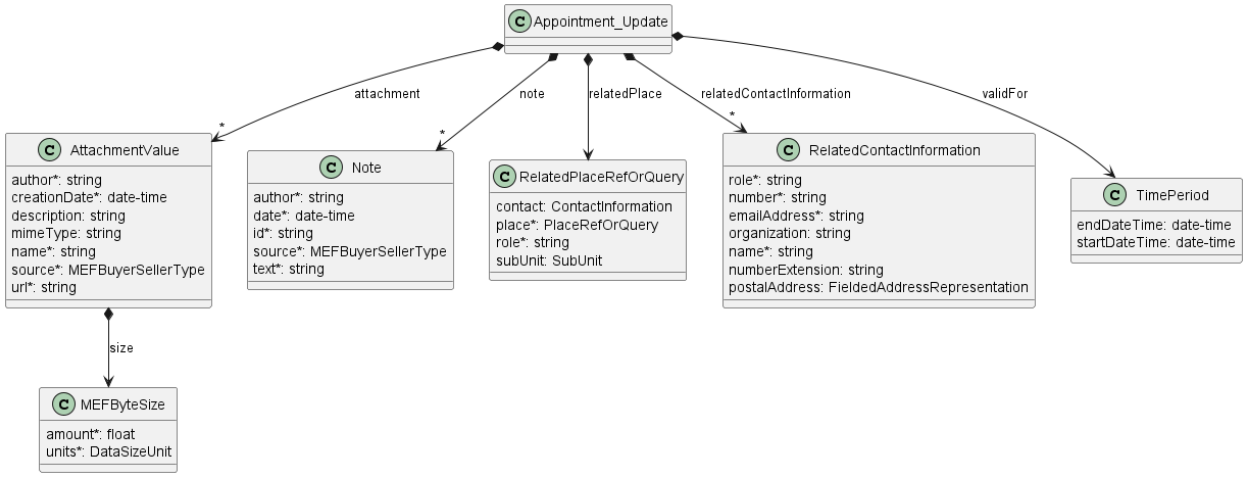


Figure 15. Patch request Model

[R34] The Buyer **MUST** include at least one of the following attributes: [MEF113 R101]

- `attachment`
- `note`
- `relatedContactInformation` - item with `role=buyerAppointmentContact`
- `relatedContactInformation` - item with `role=appointmentPlaceContact`
- `validfor`

[R35] The Buyer **MUST NOT** change any of the following attributes: [MEF113 R102]

- `attachment`
- `note`
- `relatedContactInformation` - item with `role=buyerAppointmentContact`
- `relatedContactInformation` - item with `role=appointmentPlaceContact`
- `validfor`

[R36] The Buyer **MUST NOT** change the `relatedContactInformation` attribute. [MEF113 R103]

[R37] The Seller **MUST** return an error `Error404` if the Appointment with the `id` is not found. [MEF113 R104]

[R38] The Seller **MUST** return an error `Error422` if the `status` of the Appointment to be updated is not `confirmed`. [MEF113 R105]

[R39] If `validfor` was included by the Buyer, the Seller **MUST** return an error `Error422` if the `validfor` is not valid. [MEF113 R106]

[R40] If an `validfor` was included by the Buyer, the Seller **MUST** return an error `Error422` if a Seller technician is not available for an appointment at the `relatedPlace` for the specified `availableTimeSlot`. [MEF113 R107]

[R41] To add a new `note` the Buyer **MUST** append it to the existing `note` list. [MEF113 R108]

[R42] To add a new `attachment` the Buyer **MUST** append it to the existing `attachment` list. [MEF113 R109]

The example below shows a request to add a new `note` (an existing cannot be modified or deleted):

```
{
  "note": [
    {<<previously existing>>
      "id": "note-1",
      "author": "John Example",
      "date": "2021-06-02T14:25:11.090Z",
      "source": "buyer",
      "text": "Couldn't reach the support on phone."
    },
    {<<added new note>>
      "id": "note-2",
      "author": "Kate Example",
      "date": "2021-06-02T19:25:11.090Z",
      "source": "buyer",
      "text": "Support reached after 5 hours"
    }
  ]
}
```

6.6. Use Case 12: Cancel Appointment by Appointment Identifier

The Buyer may request to Cancel an Appointment by using `POST /appointment/{{id}}/cancel` endpoint. This operation only requires providing the `id` in the path and has an empty `204` confirmation response.

The sequence diagram below presents this use case in detail.

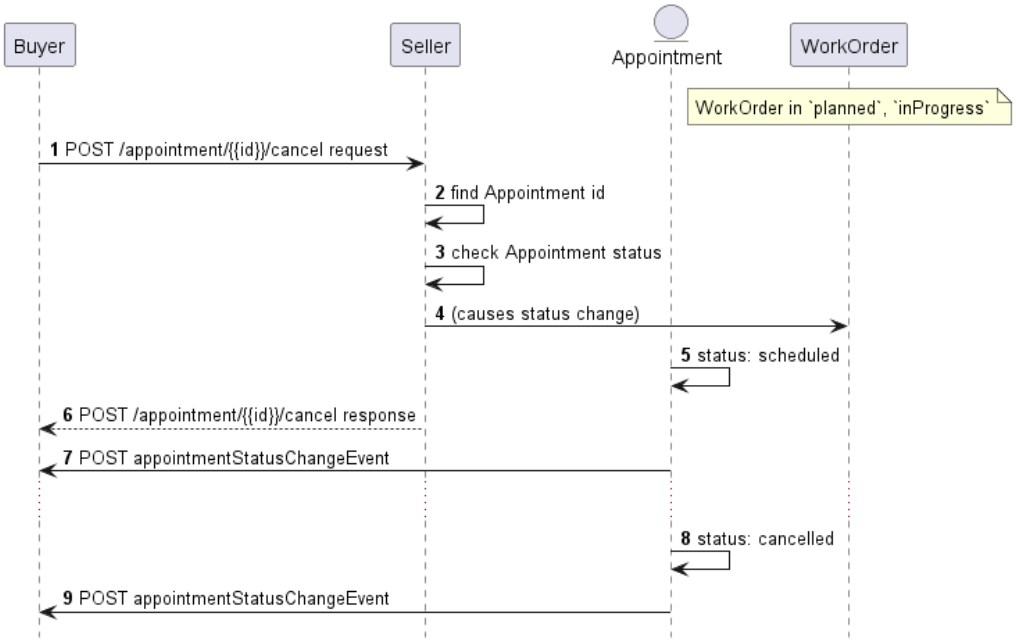


Figure 16. Cancel Appointment Flow

The Seller verifies the request, then searches for an Appointment to be cancelled by the given `id`. If found, the status is verified and `confirmed`. If everything is verified correctly, the Seller moves the appointment to the `assessingCancellation` status, sends a successful response to a cancellation request, and starts assessing the cancellation process for the appointment.

[R43] The Seller **MUST** return an error `Error404` if the Appointment `id` is not found. [MEF113 R112]

[R44] The Seller **MUST** return an error `Error422` if the Appointment `status` is not `confirmed`. [MEF113 R113]

[R45] The Seller **MUST** return an error `Error422` if the related WorkOrder `state` is not `planned`. [MEF113 R114]

[R46] The Seller **MUST** set the `appointmentRequired` attribute to `true` in the related WorkOrder. [MEF113 R115]

[R47] The Seller **MUST** set the related WorkOrder `state` to `open`. [MEF113 R116]

[R48] The Seller **MUST** set the Appointment `status` to `cancelled`. [MEF113 R117]

6.7. Use Case 13: Retrieve WorkOrder List

[O4] The Buyer **MAY** use `GET /workOrder` operation with desired filtering criteria: [MEF113 O18]

- `appointmentRequired`
- `geographicalSiteId`
- `geographicalAddressId`
- `relatedEntity`
- `state`

The Buyer may also ask for the pagination with the use of the `offset` and `limit` parameters. The filtering and the pagination attributes must be specified in URI query format RFC3986.

```
https://serverRoot/mefApi/sonata/appointment/v2/appointment?
status=inProgress&priority=critical&limit=10&offset=0
```

The example above shows a Buyer's request to get all WorkOrder that are in the `inProgress`. Additionally, the Buyer asks only for a first (`offset=0`) pack of 10 results (`limit=10`) to be returned. The correct response (HTTP code `200`) in the response body contains a list of `WorkOrder_Find` objects matching the criteria. To get more details (e.g. the item level information), the Buyer has to query a specific `WorkOrder` by `id`.

[R49] The Seller **MUST** respond to a `WorkOrder_Find` containing the workOrder attributes that match the Buyer's filter criteria: [MEF113 R119]

- `appointmentRequired`
- `id`
- `place`
- `relatedEntity`
- `state`

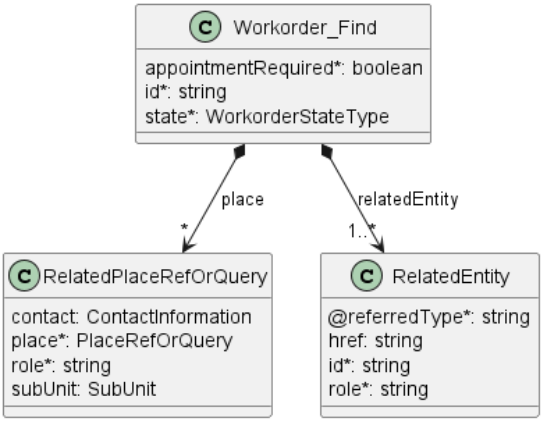


Figure 17. Use Case 13: Retrieve WorkOrder List - Model

6.8. Use Case 14: Retrieve WorkOrder by WorkOrder Identifier

The Buyer can retrieve detailed information about the WorkOrder from the Seller by using a `GET /workOrder/{{id}}` operation.

[R50] In case `id` does not allow to finding a `WorkOrder` instance, an error `Error404` **MUST** be returned. [MEF113 R121]

[R51] The Seller's response **MUST** include the following WorkOrder attributes. [MEF113 R123]

- `appointmentRequired`
- `id`
- `place`
- `relatedContactInformation` - item with `role=technician`
- `relatedEntity`
- `state`
- `task`

[R52] The Seller **MUST** include all of the optional WorkOrder attributes if they were set by the Buyer or Seller: [MEF113 R124]

- `appointment`
- `note`
- `plannedExecutionDate`
- `relatedContactInformation` - item with `role=technicalContact`

Figure 18 presents the model of the WorkOrder.

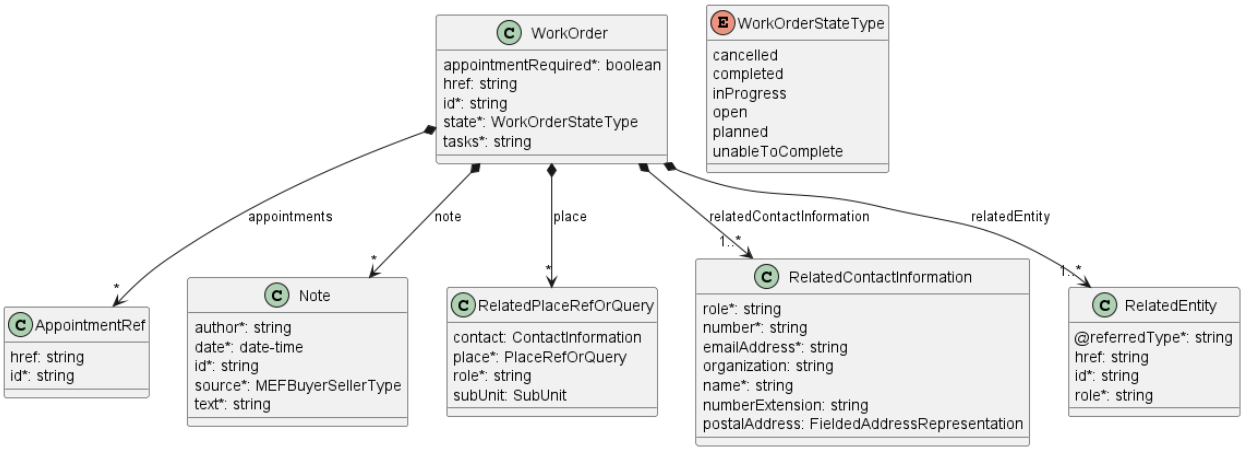


Figure 18. Use Case 14: WorkOrder - Model

Table 10 presents the mapping between the API `state` names and the MEF 113 naming, together with their description.

state	MEF 113 name	Description
`completed`	COMPLETED	Technician responsible for the WorkOrder has successfully completed all the assigned Tasks.
`cancelled`	CANCELLED	The WorkOrder has been cancelled by the Seller or due to the Buyer requesting to cancel the Ticket.
`inProgress`	IN_PROGRESS	Technician responsible for the WorkOrder has been assigned and started one or more of the assigned Tasks.
`open`	OPEN	A workOrder was initiated by the Seller to be assigned to a Technician responsible for resolving the Ticket.
`planned`	PLANNED	The WorkOrder has been given an execution date for resolving one or more Tasks.
`unableToComplete`	UNABLE_TO_COMPLETE	The Technician responsible for the workOrder was unable to complete one or more of the assigned Tasks, because of additional skills or information is required. Additional tasks are required to resolve the ticket and a new workOrder needs to be opened.

Table 10. WorkOrder states

Figure 19 presents the WorkOrder state machine:

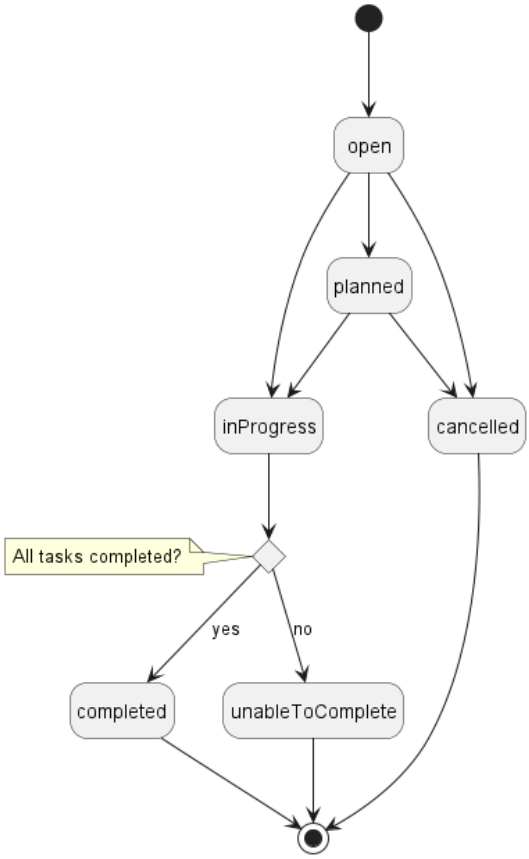


Figure 19. WorkOrder State Machine

[R53] The Seller **MUST** support all `state` values and their associated transitions. [MEF113 R162]

Below you can find a snippet with a WorkOrder example:


```
{
  "id": "98765432-9876-5432-0000-000000000055",
  "appointmentRequired": false,
  "relatedContactInformation": [
    {
      "emailAddress": "technical.contact@seller.mef.com",
      "name": "Technical Contact",
      "number": "+12-345-678-91",
      "organization": "Seller Example Co.",
      "role": "technicalContact"
    }
  ],
  "relatedEntity": [
    {
      "id": "01494064-6c79-4a32-83f7-53284196d35d",
      "role": "parentTroubleTicket",
      "@referredType": "TroubleTicket"
    }
  ],
  "state": "open",
  "plannedExecutionDate": "2022-05-20T10:04:18.812Z",
  "task": ["Replace the broken SFP", "Perform OTDR"]
}
```


[R54] The Seller **MUST** send a `workOrderCreateEvent` whenever a new WorkOrder has been created. [MEF113 R163]

[R55] The Seller **MUST** send a `workOrderStateChangeEvent` whenever a WorkOrder `state` change has occurred. [MEF113 R164]

[R56] The Seller **MUST** send a `workOrderAppointmentRequiredEvent` when the Seller sets the `appointmentRequired` attribute in the WorkOrder to `true`. [MEF113 R165]

[R57] If the `appointmentRequired` attribute in a WorkOrder is `true`, the Buyer **MUST** schedule an Appointment using a Search Appointment request followed by a Create Appointment request before the Seller can continue processing the associated Ticket. [MEF113 R166]

6.9. Use Case 19: Register for Appointment Notifications

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

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

The usage of a combination of these attributes fulfills the [ME113 R125], [ME113 R126], [ME113 R127] requirements.

By using a simple request:

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

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

- `appointmentAttributeValueChangeEvent`
- `appointmentStatusChangeEvent`

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

```
eventType=appointmentAttributeValueChangeEvent,appointmentStatusChangeEvent
```

or

```
eventType=appointmentAttributeValueChangeEvent&eventType=appointmentStatusChangeEvent
```

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

6.10. Use Case 20: Send Appointment Notification

Notifications are used to asynchronously inform the Buyer about the respective objects and attribute changes. The Seller's synchronous responses to an Appointment create requests are considered to act as a Create Notification so there is no explicit respective Create Notification type. The next notification must be sent when the state changes compared to the previously sent one.

[R58] The Seller **MUST NOT** send Appointment Notifications for an Appointment Notification Type to a Buyer who has not registered for the Appointment Notification Type. [MEF113 R146]

[R59] The Seller **MUST** send Appointment Notifications to a Buyer who has registered for the Appointment Notification Type. [MEF113 R147]

Figure 20 shows all entities involved in the Notification use cases.

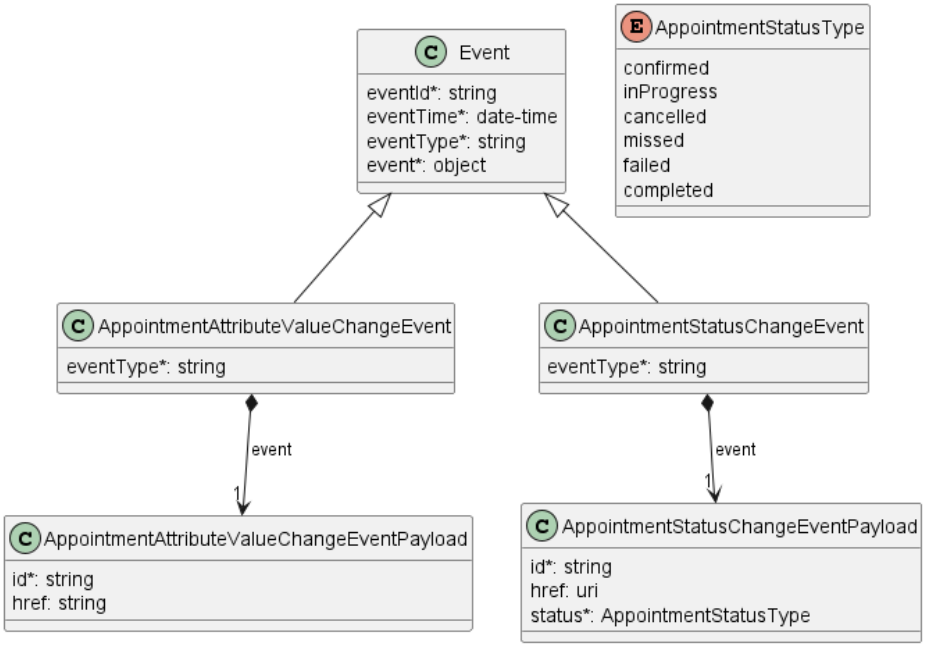


Figure 20. Use Case 20. Appointment Notification Data Model

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

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

Table 11 presents the mapping between the API Notification types' names and the ones in MEF 113. The inconsistencies are caused by using the TMF event types as the base for this API.

API name	MEF 113 name
`appointmentAttributeValueChangeEvent`	APPOINTMENT_UPDATE
`appointmentStatusChangeEvent`	APPOINTMENT_STATUS_CHANGE

Table 11. Appointment Notification types mapping

6.11. Use Case 21: Register for WorkOrder Notifications

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

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

The usage of a combination of these attributes fulfills the [ME113 R125], [ME113 R126], [ME113 R127] requirements.

By using a simple request:

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

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

- `workOrderCreateEvent`
- `workOrderStateChangeEvent`
- `workOrderAppointmentRequiredEvent`

If the Buyer wishes to subscribe to 3 different types of events, there are 3 possible syntax variants [TMF630]:

```
eventType=workOrderCreateEvent,workOrderStateChangeEvent,workOrderAppointmentRequiredEvent
```

or

```
eventType=workOrderCreateEvent&eventType=workOrderStateChangeEvent&eventType=workOrderAppointmentRequiredEven
t
```

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

6.12. Use Case 22: Send WorkOrder Notification

Notifications are used to asynchronously inform the Buyer about the respective objects and attribute changes. The Seller's synchronous response to a WorkOrder creates requests are considered to act as a Create Notification so there is no explicit respective Create Notification type. The next notification must be sent when the state changes compared to the previously sent one.

[R60] The Seller **MUST NOT** send WorkOrder Notifications for a WorkOrder Notification Type to a Buyer who has not registered for the WorkOrder Notification Type. [MEF113 R152]

[R61] The Seller **MUST** send WorkOrder Notifications to a Buyer who has registered for the WorkOrder Notification Type. [MEF113 R153]

- `workOrderCreateEvent`
- `workOrderStateChangeEvent`
- `workOrderAppointmentRequiredEvent`

Figure 21 shows all entities involved in the Notification use cases.

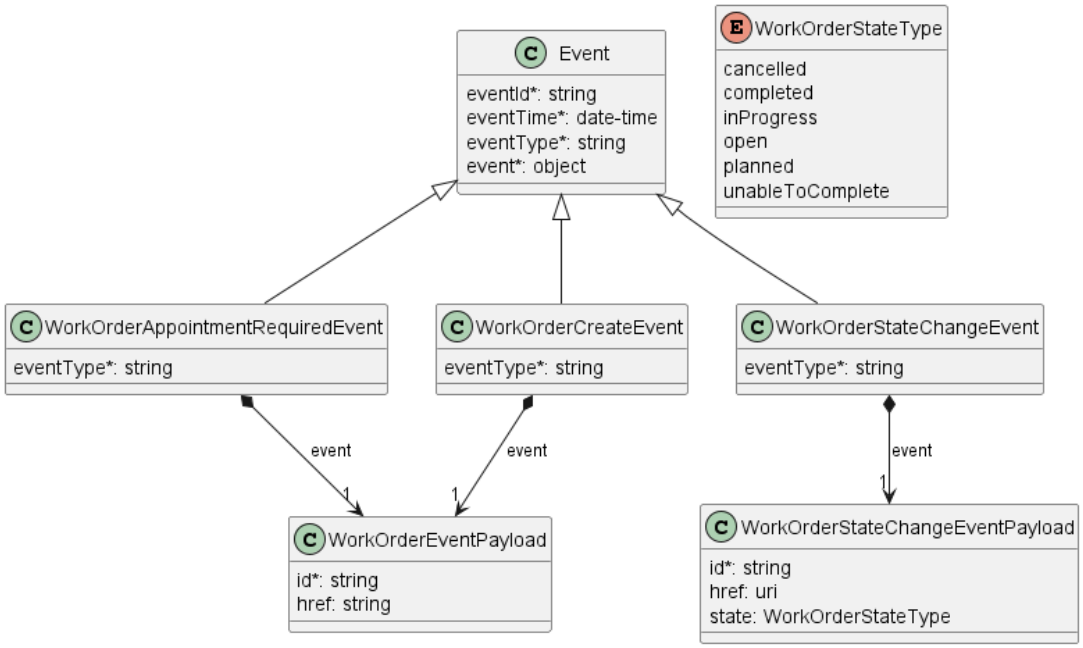


Figure 21. Use Case 22. WorkOrder Notification Data Model

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

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

Table 12 presents the mapping between the API Notification types' names and the ones in MEF 113. The inconsistencies are caused by using the TMF event types as the base for this API.

API name	MEF 113 name
`workOrderCreateEvent`	WORKORDER_CREATE
`workOrderStateChangeEvent`	WORKORDER_STATE_CHANGE
`workOrderAppointmentRequiredEvent`	WORKORDER_APPOINTMENT_REQUIRED

Table 12. WorkOrder Notification types mapping

7. API Details

7.1. API patterns

7.1.1. Indicating errors

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

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

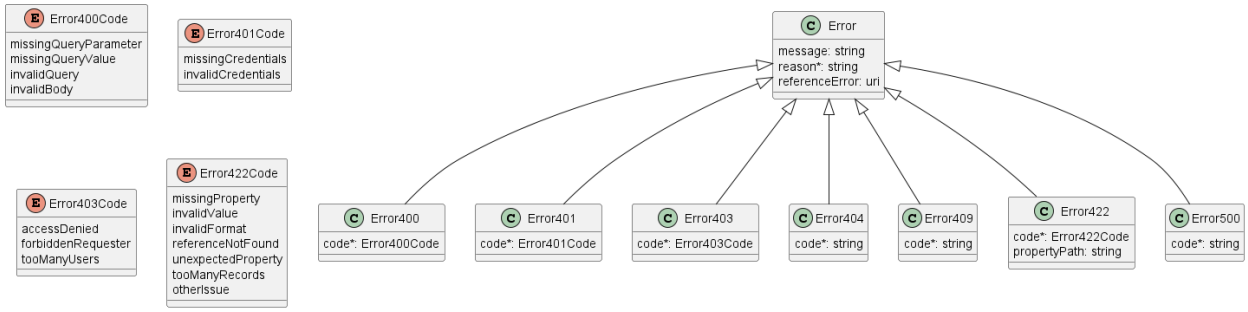


Figure 22. Data model types to represent an erroneous response

7.1.1.1. Type Error

Description: Standard Class used to describe API response error Not intended to be use directly only via specialization which defines a code

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

7.1.1.2. Type Error400

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

Inherits from:

- Error

Name	Type	Description

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

7.1.1.3. enum Error400Code

Description: One of the following error codes:

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

7.1.1.4. Type Error401

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

Inherits from:

- Error

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

7.1.1.5. enum Error401Code

Description: One of the following error codes:

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

7.1.1.6. Type Error403

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

Inherits from:

- Error

Name	Type	Description

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

7.1.1.7. `enum` Error403Code

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

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

7.1.1.8. Type Error404

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

Inherits from:

- Error

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

7.1.1.9. Type Error409

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

Inherits from:

- Error

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

7.1.1.10. Type Error422

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

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

Inherits from:

- Error

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

7.1.1.11. enum Error422Code

Description: One of the following error codes:

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

7.1.1.12. Type Error500

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

Inherits from:

- Error

Name	Type	Description

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

7.2. Management API Data model

7.2.1. Appointment

Figure 23 presents the whole Appointment Management data model the data types, requirements related to them and mapping to MEF 113 specifications are discussed later in this section.

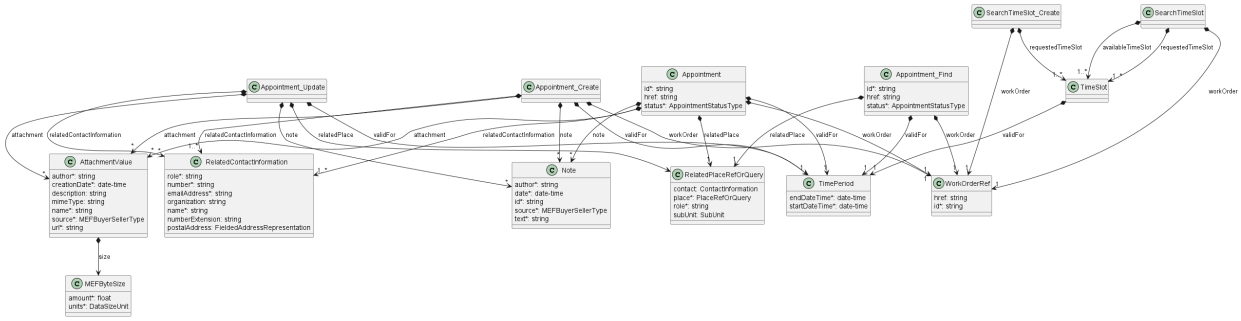


Figure 23. Appointment Management Data Model

7.2.1.1. Type Appointment

Description: In the context of MEF 113 document, denotes an arrangement between the Buyer and Seller for a Seller Technician to meet with the Buyer at a specified time and place.

Name	Type	M/O	Description	MEF 113
id	string	M	Unique identifier of the appointment	Appointment Identifier
href	string	O	Unique URI used to access to the appointment resource	Not represented in MEF 113
attachment	AttachmentValue[]	O	Attachments to the Appointment, such as a file, screenshot or embedded content.	Appointment Attachments
note	Note[]	O	Notes describing the purpose of and the results of the Appointment	Appointment Notes
relatedPlace	RelatedPlace-RefOrQuery	M	The location of the Appointment.	Appointment Place
status	Appointment-StatusType	M	The state of the Appointment.	Appointment State
validFor	TimePeriod	M	The Date and Time interval the Seller Technician is scheduled to arrive at the Appointment.	Appointment Timeslot

Name	Type	M/O	Description	MEF 113
relatedContact-Information	RelatedContact-Information[] *minItems = 1*	M	Party playing a role in this Appointment. The 'role' is to specify the type of contact as specified in MEF 113: Appointment Place Contact (role=appointmentPlaceContact) - The site contact(s) that the Seller Technician may need to contact in order to get access to the Appointment Place during the Appointment. This could be an end-user, security personnel or any authorized person Buyer Appointment Contact (role=buyerAppointmentContact)- The Buyer contact(s) assigned to and responsible for the Appointment. Seller Appointment Contact (role=sellerAppointmentContact) - The Seller contact(s) assigned to and responsible for the Appointment.	Buyer Appointment Contacts, Seller Appointment Contact
workOrder	WorkOrderRef	M	A reference to a WorkOrder for which the Appointment is created.	Not represented in MEF 113

7.2.1.2. Type Appointment_Create

Description: In the context of MEF 113 document, denotes an arrangement between the Buyer and Seller for a Seller Technician to meet with the Buyer at a specified time and place.'

Name	Type	M/O	Description	MEF 113
attachment	AttachmentValue[]	O	Attachments to the Appointment, such as a file, screen shot or embedded content.	Appointment Attachments
note	Note[]	O	Notes describing the purpose of and the results of the Appointment	Appointment Notes

Name	Type	M/O	Description	MEF 113
relatedContact-Information	RelatedContact-Information[] *minItems = 1*	M	Party playing a role in this Appointment. The 'role' is to specify the type of contact as specified in MEF 113: Appointment Place Contact (role=appointmentPlaceContact) - The site contact(s) that the Seller Technician may need to contact in order to get access to the Appointment Place during the Appointment. This could be an end-user, security personnel or any authorized person Buyer Appointment Contact (role=buyerAppointmentContact)- The Buyer contact(s) assigned to and responsible for the Appointment. Seller Appointment Contact (role=sellerAppointmentContact) - The Seller contact(s) assigned to and responsible for the Appointment.	Buyer Appointment Contacts
validFor	TimePeriod	M	The Date and Time interval the Seller Technician is scheduled to arrive at the Appointment.	Appointment Timeslot
workOrder	WorkOrderRef	M	A reference to a WorkOrder for which the Appointment is created.	Not represented in MEF 113

7.2.1.3. Type Appointment_Find

Description: In the context of MEF 113 document, denotes an arrangement between the Buyer and Seller for a Seller Technician to meet with the Buyer at a specified time and place.

Name	Type	M/O	Description	MEF 113
id	string	M	Unique identifier of the appointment	Appointment Identifier
href	string	O	Unique URI used to access to the appointment resource	Not represented in MEF 113
relatedPlace	RelatedPlace-RefOrQuery	M	Related place defines (by reference or value) the place where the appointment will take place.	Appointment Place
status	Appointment-StatusType	M	The state of the Appointment.	Appointment State

Name	Type	M/O	Description	MEF 113
validFor	TimePeriod	M	The Date and Time interval the Seller Technician is scheduled to arrive at the Appointment.	Appointment Timeslot
workOrder	WorkOrderRef	M	A reference to a WorkOrder for which the Appointment is created.	Not represented in MEF 113

7.2.1.4. Type Appointment_Update

Description: In the context of MEF 113 document, denotes an arrangement between the Buyer and Seller for a Seller Technician to meet with the Buyer at a specified time and place.

Name	Type	M/O	Description	MEF 113
attachment	AttachmentValue[]	O	Attachments to the Appointment, such as a file, screen shot or embedded content.	Appointment Attachments
note	Note[]	O	Notes describing the purpose of and the results of the Appointment.	Appointment Notes
relatedPlace	RelatedPlace-RefOrQuery	O	Related place defines (by reference or value) the place where the appointment will take place.	Appointment Place
relatedContact-Information	RelatedContact-Information[] *minItems = 1*	O	Party playing a role in this Appointment. The 'role' is to specify the type of contact as specified in MEF 113: Appointment Place Contact (role=appointmentPlaceContact) - The site contact(s) that the Seller Technician may need to contact in order to get access to the Appointment Place during the Appointment. This could be an end-user, security personnel or any authorized person Buyer Appointment Contact (role=buyerAppointmentContact)- The Buyer contact(s) assigned to and responsible for the Appointment. Seller Appointment Contact (role=sellerAppointmentContact) - The Seller contact(s) assigned to and responsible for the Appointment.	Buyer Appointment Contacts

Name	Type	M/O	Description	MEF 113
validFor	TimePeriod	O	The Date and Time interval the Seller Technician is scheduled to arrive at the Appointment.	Appointment Timeslot

7.2.1.5. `enum` AppointmentStatusType

Description: Valid values for the lifecycle status of the appointment.

status	MEF 113 name	Description
`confirmed`	SCHEDULED	The Buyer has negotiated and scheduled the Appointment with the Seller.
`inProgress`	IN_PROGRESS	The Appointment can no longer be cancelled (point of no return), this is up to the Seller's discretion.
`cancelled`	CANCELLED	The Appointment was cancelled by the Buyer or the Seller determined that an Appointment was not required. This is a terminal state.
`missed`	MISSED	The Appointment did not take place, because of a Seller related issue. For example, no Seller Technician was available on the date of the appointment
`failed`	FAILED	The Appointment did not take place, because of an issue with the Buy-er. For example, Seller Technician was unable to get to the Appointment due to an incorrect location or unable to get access to the Buyer`s site. This is a terminal state
`completed`	COMPLETED	The Appointment took place as scheduled. This is a terminal state.

7.2.1.6. Type SearchTimeSlot

Description: This task resource is used to retrieve available time slots. One of this available time slot is after used to create or reschedule an appointment

Name	Type	M/O	Description	MEF 113
availableTimeSlot	TimeSlot[]	M	A set of time slots with the availability of a Seller Technician returned by the Seller, which the Buyer may select for creating or rescheduling an Appointment.	Available Timeslots

Name	Type	M/O	Description	MEF 113
requestedTimeSlot	TimeSlot[]	M	A set of preferred time slots the Buyer is requesting the Seller to verify for availability by a Sellers Technician at the Place referenced in the Appointment Related Entity. For example Monday thru Friday, or a set of specific time slots.	Requested Timeslots
workOrder	WorkOrderRef	M	A reference to a WorkOrder for which the Appointment is created.	Not represented in MEF 113

7.2.1.7. Type SearchTimeSlot_Create

Description: This task resource is used to retrieve available time slots. One of this available time slot is after used to create or reschedule an appointment

Name	Type	M/O	Description	MEF 113
requestedTimeSlot	TimeSlot[]	M	A set of preferred time slots the Buyer is requesting the Seller to verify for availability by a Sellers Technician at the Place referenced in the Appointment Related Entity. For example Monday thru Friday, or a set of specific time slots.	Requested Timeslots
workOrder	WorkOrderRef	M	A reference to a WorkOrder for which the Appointment is created.	Not represented in MEF 113

7.2.1.8. Type WorkOrderRef

Description: A reference to an WorkOrder resource.

Name	Type	M/O	Description	MEF 113
href	string	O	Hyperlink to the referenced WorkOrder.	Not represented in MEF 113
id	string	M	Identifier of the referenced WorkOrder.	WorkOrder Identifier

7.2.2. WorkOrder

Figure 24 presents the whole WorkOrder Management data model the data types, requirements related to them and mapping to MEF 113 specifications are discussed later in this section.

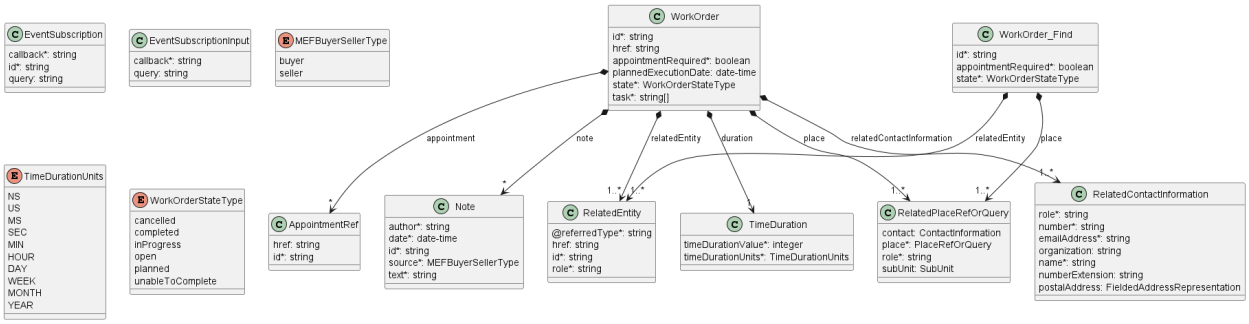


Figure 24. WorkOrder Management Data Model

7.2.2.1. Type WorkOrder

Description: A set of tasks to be scheduled and performed under the responsibility of a Seller Technician(s)

Name	Type	M/O	Description	MEF 113
id	string	M	Unique (within the Seller domain) identifier for the WorkOrder.	WorkOrder Identifier
href	string	O	Hyperlink, a reference to the WorkOrder entity	Not represented in MEF 113
appointmentRequired	boolean	M	The Seller requires the Buyer to schedule an Appointment. If set to true, the Seller is Requesting the Buyer to schedule an Appointment.	Appointment Required
appointment	AppointmentRef[]	O	A reference to a set of Appointments for the WorkOrder. A WorkOrder may contain only one open Appointment at a time (e.g. with the state of 'scheduled').	WorkOrder Appointments
duration	TimeDuration	M	The current state of the WorkOrder	WorkOrder Duration
note	Note[]	O	A set of unstructured comments or information associated to the WorkOrder	WorkOrder Notes

Name	Type	M/O	Description	MEF 113
place	RelatedPlace-RefOrQuery[]	M	The location where the WorkOrder Tasks are to be performed. If an appointment is needed, this will also be the location where the Appointment takes place and includes the site contact which the Seller technician may need to get access to the Buyer's site during the Appointment. This could be an end-user, security personnel or any authorized person	WorkOrder Place
plannedExecutionDate	date-time *format = date-time*	O	The date provided by the Seller to indicate when the Workorder is expected to be started.	Planned Execution Date
relatedContact-Information	RelatedContact-Information[] *minItems = 1*	M	Party playing a role in this WorkOrder. The 'role' is to specify the type of contact as specified in MEF 113: Technical Contact ('role=technicalContact') - REQUIRED to be set by the Seller. The Seller technical contact responsible for the WorkOrder. Technician ('role=technician') - The Seller technician assigned to the WorkOrder and responsible for performing a set of tasks. In certain instances this could be a Buyer technician that works on behalf of the Seller.	Workorder Contact, Technician
relatedEntity	RelatedEntity[] *minItems = 1*	M	An entity that is related to the WorkOrder such as a service, a product, etc. The entity which the WorkOrder is associated with.	WorkOrder Related Entity

Name	Type	M/O	Description	MEF 113
state	WorkOrder-StateType	M	The current state of the WorkOrder	WorkOrder State
task	string[] *minItems = 1*	M	A set of tasks to be performed under the responsibility of the Technician to fulfil the WorkOrder. Each item is a description of a specific task to be performed under the responsibility of the Technician.	Tasks

7.2.2.2. Type WorkOrder_Find

Description: A set of tasks to be scheduled and performed under the responsibility of a Seller Technician(s)

Name	Type	M/O	Description	MEF 113
id	string	M	Unique (within the Seller domain) identifier for the WorkOrder.	WorkOrder Identifier
appointmentRequired	boolean	M	The Seller requires the Buyer to schedule an Appointment. If set to true, the Seller is Requesting the Buyer to schedule an Appointment.	Appointment Required
place	RelatedPlace-RefOrQuery[]	M	The location where the WorkOrder Tasks are to be performed. If an appointment is needed, this will also be the location where the Appointment takes place and includes the site contact which the Seller technician may need to get access to the Buyer's site during the Appointment. This could be an end-user, security personnel or any authorized person	WorkOrder Place

Name	Type	M/O	Description	MEF 113
relatedEntity	RelatedEntity[] *minItems = 1*	M	An entity that is related to the WorkOrder such as a service, a product, etc. The entity which the WorkOrder is associated with.	WorkOrder Related Entity
state	WorkOrder-StateType	M	The current state of the WorkOrder	WorkOrder State

7.2.2.3. `enum` WorkOrderStateType

Description: Possible values for the state of the WorkOrder

state	MEF 113 name	Description
`completed`	COMPLETED	The Seller Technician responsible for the WorkOrder has successfully completed all the assigned Tasks.
`cancelled`	CANCELLED	The WorkOrder has been cancelled by the Seller or due to the Buyer requesting to cancel the WorkOrder.
`inProgress`	IN_PROGRESS	The Seller Technician responsible for the WorkOrder has been assigned and started one or more of the assigned Tasks.
`open`	OPEN	A WorkOrder was initiated by the Seller to be assigned to a Technician responsible for resolving the WorkOrder.
`planned`	PLANNED	The WorkOrder has been given an execution date for resolving one or more Tasks.
`unableToComplete`	UNABLE_TO_COMPLETE	The Seller Technician responsible for the WorkOrder was unable to complete one or more of the assigned Tasks, because additional skills or information is required. Additional tasks are required to resolve the WorkOrder and a new WorkOrder needs to be opened.

7.2.2.4. Type AppointmentRef

Description: A reference to an Appointment resource available through Appointment API.

Name	Type	M/O	Description	MEF 113

Name	Type	M/O	Description	MEF 113
href	string	O	Hyperlink to the referenced Appointment. 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 113
id	string	M	Identifier of the referenced Appointment.	Appointment Identifier

7.2.3. Common

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

7.2.3.1. Type AttachmentValue

Description: Complements the description of an element (for instance a product) through video, pictures...

Name	Type	M/O	Description	MEF 113
author	string	M	The name of the person or organization who added the Attachment.	Attachment Author
creationDate	date-time *format = date-time*	M	The date the Attachment was added.	Attachment Date
description	string	O	A narrative text describing the content of the attachment	Description
mimeType	string	O	Attachment mime type such as extension file for video, picture and document	Mime Type
name	string	M	The name of the attachment	Attachment Name
size	MEFByteSize	O	The size of the attachment.	Size
source	MEFBuyerSellerType	M	Indicates if the attachment was added by the Buyer or the Seller.	Attachment Source
url	string	M	URL where the attachment is located.	URL

7.2.3.2. enum DataSizeUnit

Description: The unit of measure in the data size.

Value
BYTES
KBYTES
MBYTES

Value
GBYTES
TBYTES
PBYTES
EBYTES
ZBYTES
YBYTES

7.2.3.3. enum MEFBuyerSellerType

Description: An enumeration with buyer and seller values.

Value
buyer
seller

7.2.3.4. Type MEFByteSize

Description: A size represented by value and Byte units

Name	Type	M/O	Description	MEF 113
amount	float *format = float*	M	Numeric value in a given unit	Value
units	DataSizeUnit	M	Byte Unit	Unit

7.2.3.5. Type Note

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

Name	Type	M/O	Description	MEF 113
author	string	M	Author of the note	Note Author
date	date-time *format = date-time*	M	Date of the note	Note Date
id	string	M	Identifier of the note within its containing entity (may or may not be globally unique, depending on provider implementation)	Not represented in MEF 57.2
source	MEFBuyerSellerType	M	Indicates if this Note was added by the Buyer or Seller.	Note source
text	string	M	Text of the note	Note Text

7.2.3.6. Type RelatedContactInformation

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

Name	Type	M/O	Description	MEF 113
role	string	M	The role of the particular contact in the request	Not represented in Mplify 79.1
number	string	M	Phone number	Contract Phone Number
emailAddress	string	M	Email address	Contact email Address
postalAddress	FieldedAddressRepresentation	O	Identifies the postal address of the person or office to be contacted.	Contact Postal Address
organization	string	O	The organization or company that the contact belongs to	Contact Organization
name	string	M	Name of the contact	Contact Name
numberExtension	string	O	Phone number extension	Contract Phone Number Extension

7.2.3.7. Type RelatedEntity

Description: A reference to an entity, where the type of the entity is not known in advance.

Name	Type	M/O	Description	MEF 113
@referredType	string *default = Product*	M	The actual type of the target instance when needed for disambiguation.	Entity Type
href	string	O	Reference of the related entity.	Not represented in MEF 113
id	string	M	Unique identifier of a related entity.	Entity Identifier

Name	Type	M/O	Description	MEF 113
role	string	M	The role of an entity.	Not represented in MEF 113

7.2.3.8. Type TimeDuration

Description: This class is used to describe durations expressed as a 2-tuple, (value, units). The units from nanoseconds to years.

Name	Type	M/O	Description	MEF 113
timeDurationValue	integer	M	The value of the duration. For example, if the duration is 20 ms, this element is 20.	Not represented in MEF 113
timeDurationUnits	TimeDurationUnits	M	The unit of measure in the duration. For example, if an interval is 2ms, this element is MS.	Not represented in MEF 113

7.2.3.9. enum TimeDurationUnits

Description: The unit of measure in the duration. For example, if an interval is 2ms, this element is MS.

Value
NS
US
MS
SEC
MIN
HOUR
DAY
WEEK
MONTH
YEAR

7.2.3.10. Type TimePeriod

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

Name	Type	M/O	Description	MEF 113

Name	Type	M/O	Description	MEF 113
endDateTime	date-time *format = date-time*	M	End of the time period, using IETC-RFC-3339 format	End Time
startDateTime	date-time *format = date-time*	M	Start of the time period, using IETC-RFC-3339 format. If you define a start, you must also define an end	Start Time

7.2.3.11. Type TimeSlot

Description:

Name	Type	M/O	Description	MEF 113
validFor	TimePeriod	M	Start date and end date of the timeSlot	Start Time, End Time

7.2.4. Place representation

7.2.4.1. Type RelatedPlaceRefOrQuery

Description: Allows pointing to a place by referring to a GeographicAddress, GeographicSite, or providing GeographicAddress by value. It also provides additional information like the `role` the place plays for given Product and `contact` needed access to this place.

Name	Type	M/O	Description	MEF 113
place	PlaceRefOrQuery	M		Place
role	string	M	Role of this place. The values that can be specified here are described by Product Specification (e.g. "INSTALL_LOCATION").	Role
contact	ContactInformation[]	O	The person to call to get access to this place in case such access is required to complete the evaluation of this POQ Item.	Appointment Contact Information

7.2.4.2. Type PlaceRefOrQuery

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

7.2.4.3. Type ContactInformation

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

Name	Type	M/O	Description	MEF 113
number	string	M	Phone number	Contact Phone Number
emailAddress	string	M	Email address	Contact Email Address
postalAddress	FieldedAddressRepresentation	O	Identifies the postal address of the person or office to be contacted.	Contact Postal Address
organization	string	O	The organization or company that the contact belongs to	Contact Organization
name	string	M	Name of the contact	Contact Name
numberExtension	string	O	Phone number extension	Contact Phone Number Extension

7.2.4.4. Type GeographicAddress_Query

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

Name	Type	M/O	Description	MEF 113
fieldedAddress-Representation	FieldedAddress-Representation[]	O	A list of Fielded Address representations	Installation Place Representations
formattedAddress-Representation	FormattedAddress-Representation[]	O	A list of Formatted Address representations	Installation Place Representations
geographicPoint-Representation	GeographicPoint-Representation[]	O	A list of Geographic Point Address representations	Installation Place Representations
label-Representation	Label-Representation[]	O	A list of Label Address representations	Installation Place Representations
@type	string	M	Used to unambiguously designate the class type when using `oneOf`	Not represented in Mplify 150

7.2.4.5. Type FieldedAddressRepresentation

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

Name	Type	M/O	Description	MEF 113
streetNr	string	O	Number identifying a specific property on a public street. It may be combined with streetNrLast for ranged addresses.	Street Number
streetNrSuffix	string	O	The first street number suffix (in a street number range) or the suffix for the street number if there is no range	Street Number Suffix
streetNrLast	string	O	Last number in a range of street numbers allocated to an Address	Street Number Last
streetNrLastSuffix	string	O	Last street number suffix for a ranged Address	Street Number Last Suffix
streetPreDirection	string	O	The direction of the street that appears before the Street Name	Street Pre-Direction
streetName	string	O	Name of the street or other street type	Street Name
streetType	string	O	The type of street (e.g., alley, avenue, boulevard, brae, crescent, drive, highway, lane, terrace, parade, place, tarn, way, wharf)	Street Type
streetPostDirection	string	O	A modifier denoting a relative direction that appears after the Street Name.	Street Post-Direction
poBox	string	O	Number identifying a specific location in a post office.	PO Box Number
locality	string	O	An area of defined or undefined boundaries within a local authority or other legislatively defined area.	Locality
city	string	O	City in which the Address is located.	City
postcode	string	O	A descriptor for a postal delivery area used to speed and simplify the delivery of mail (also known as zip code)	Postal Code

Name	Type	M/O	Description	MEF 113
postcodeExtension	string	O	The extension used on a postal code. Note: there are different use codes for this attribute depending upon the country.	Postal Code Extension
stateOrProvince	string	O	The State or Province in which the Address is located.	State or Province
countryCode	string *minLength = 2* *maxLength = 2*	O	Country in which the Address is located, defined using two characters as defined in ISO 3166	Country
subUnit	SubUnit[]	O	The Sub Unit represented as a list. This is a list to allow complex sub-unit information such as SUITE 42 ROOM A	Sub Units
buildingName	string	O	The well-known name of a building that is located at this Address (e.g., where there is one Address for a campus).	Building Name
privateStreetNumber	string	O	Street number on a private street within the Address.	Private Street Number
privateStreetName	string	O	Private streets internal to a property (e.g., a university) may have internal names that are not recorded by the land title office.	Private Street Name
language	string *minLength = 2* *maxLength = 2*	O	The language in which the address is expressed. It MUST use the ISO 639:2023 two letter code 639:2023	Language

7.2.4.6. Type FormattedAddressRepresentation

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

Name	Type	M/O	Description	MEF 113
formattedAddress	string	M	A formatted Address Representation that contains a non-fielded address.	Formatted Address
language	string *minLength = 2* *maxLength = 2*	O	The language in which the address is expressed. Based on ISO 639:2023	Language

7.2.4.7. Type GeographicPointRepresentation

Description: A GeographicPointRepresentation defines a geographic point through coordinates.

Name	Type	M/O	Description	MEF 113

Name	Type	M/O	Description	MEF 113
spatialRef	string	M	The spatial reference system used to determine the coordinates. The system used and the value of this field are to be agreed during the onboarding process.	Spatial Reference
latitude	string	M	The latitude expressed in the format specified by the `spacialRef`	Latitude
longitude	string	M	The longitude expressed in the format specified by the `spacialRef`	Longitude
elevation	string	O	The elevation expressed in the format specified by the `spacialRef`	Elevation

7.2.4.8. Type LabelRepresentation

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

Name	Type	M/O	Description	MEF 113
label	string	M	The unique reference to a Geographic Address assigned by the Administrative Authority.	Installation Place Label
administrativeAuthority	string	M	The organization or standard from the organization that administers this Geographic Address Label ensuring it is unique within the Administrative Authority.	Administrative Authority
language	string *minLength = 2* *maxLength = 2*	O	The language in which the label is expressed. Based on ISO 639:2023	Language

7.2.4.9. Type GeographicAddressRef

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

Name	Type	M/O	Description	MEF 113
href	string	O	Hyperlink to the referenced Address. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in Mplify 79.1

Name	Type	M/O	Description	MEF 113
id	string	M	Identifier of the referenced Geographic Address. This identifier is assigned during a successful address validation request (Geographic Address Management API)	Installation Place Identifier
@type	string	M	Used to unambiguously designate the class type when using `oneOf`	Not represented in Mplify 79.1

7.2.4.10. Type GeographicSiteRef

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

Name	Type	M/O	Description	MEF 113
href	string	O	Hyperlink to the referenced Site. Hyperlink MAY be used by the Seller in responses. Hyperlink MUST be ignored by the Seller in case it is provided by the Buyer in a request.	Not represented in Mplify 79.1
id	string	M	Identifier of the referenced Geographic Site.	Site Identifier
@type	string	M	Used to unambiguously designate the class type when using `oneOf`	Not represented in Mplify 79.1

7.2.4.11. Type SubUnit

Description: Allows for sub unit identification

Name	Type	M/O	Description	MEF 113
subUnitNumber	string	M	The discriminator used for the subunit, often just a simple number but may also be a range.	Sub Unit Name
subUnitType	string	M	The type of subunit e.g. BERTH, FLAT, PIER, SUITE, SHOP, TOWER, UNIT, WHARF.	Sub Unit Type

7.2.5. Notification registration

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

7.2.5.1. EventSubscription

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

Name	Type	Description	MEF 113
callback*	string	The value provided by the Buyer in `EventSubscriptionInput` during notification registration	Notification Target Information
id*	string	An identifier of the event subscription assigned by the Seller when a resource is created.	Not represented in MEF 113
query	string	This attribute is used to define notification registration constraints.	List of Notification Event Types, Action

7.2.5.2. EventSubscriptionInput

Description: This class is used to register for Notifications.

Name	Type	Description	MEF 113
callback*	string	This callback value must be set to *host* property from Buyer Notification API (appointmentNotification.api.yaml). This property is appended with the base path and notification resource path specified in that API to construct an URL to which notification is sent. E.g. for "callback": "http://buyer.com/listenerEndpoint", the status change event notification will be sent to: `http://buyer.com/listenerEndpoint /mefApi/sonata/appointmentNotification /v2/listener/appointmentStatusChangeEvent`	Notification Target Information
query	string	This attribute is used to define to which type of events to register to. Example: "query":"eventType = AppointmentStatusChangeEvent". To subscribe for more than one event type, put the values separated by comma: `eventType=AppointmentStatusChangeEvent, AppointmentResolvedEvent`. The possible values are enumerated by the 'AppointmentEventType' in AppointmentNotification.api.yaml. An empty query is treated as specifying no filters - ending in subscription for all event types.	List of Notification Event Types, Action

7.3. Notification API Data Model

7.3.1. Common Notification

7.3.1.1. Event

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

Name	Type	Description	MEF 113

Name	Type	Description	MEF 113
eventId*	string	Id of the event	Not represented in MEF 113
eventTime*	date-time	Date time when the event occurred	Not represented in MEF 113

7.3.2. Appointment Notification.

Figure 25 presents the whole WorkOrder Management data model the data types, requirements related to them and mapping to MEF 113 specifications are discussed later in this section.

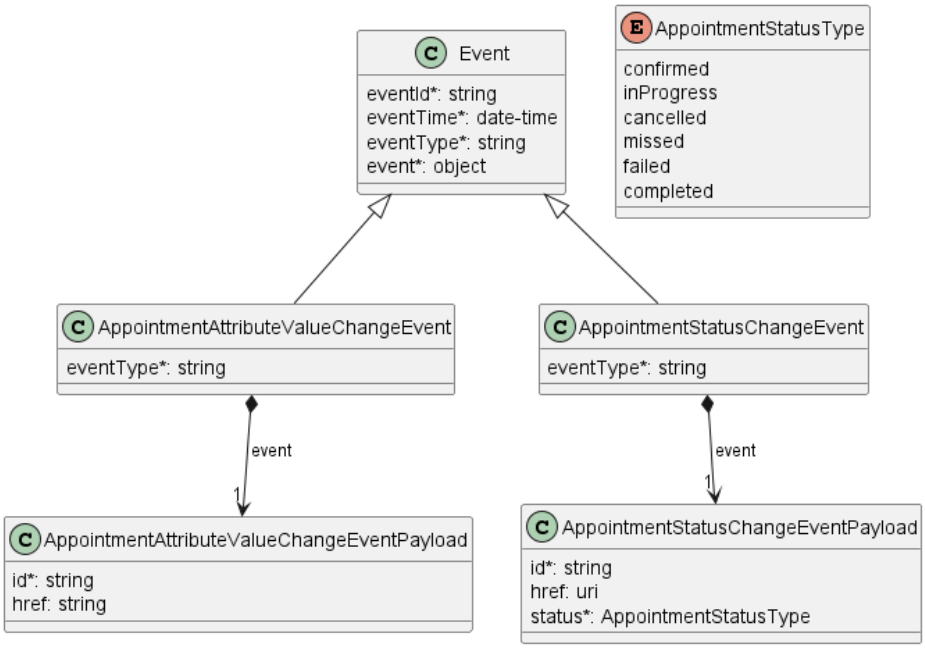


Figure 25. Appointment Notification Data Model

7.3.2.1. Type AppointmentAttributeValueChangeEvent

Description:

Inherits from:

- Event

Name	Type	M/O	Description	MEF 113
eventType	string	M	Indicates the type of the event.	Appointment Notification Type
event	AppointmentAttributeValueChangeEventPayload	M	A reference to the object that is the source of the notification.	Appointment Identifier

7.3.2.2. Type AppointmentAttributeValueChangeEventPayload

Description: The identifier of the Appointment is subject of this event.

Name	Type	M/O	Description	MEF 113
id	string	M	ID of the Appointment attributed by quoting system	Appointment Identifier
href	string	O	Hyperlink to access the Appointment	Not represented in MEF 113

7.3.2.3. Type AppointmentStatusChangeEvent

Description:

Inherits from:

- Event

Name	Type	M/O	Description	MEF 113
eventType	string	M	Indicates the type of the event.	Appointment Notification Type
event	AppointmentStatusChangeEventPayload	M	A reference to the object that is the source of the notification.	Appointment Identifier

7.3.2.4. Type AppointmentStatusChangeEventPayload

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

Name	Type	M/O	Description	MEF 113
id	string	M	ID of the Product Offering attributed by the Seller	Appointment Identifier
href	uri *format = uri*	O	Hyperlink to access the Product Offering	Not represented in MEF 113
status	AppointmentStatusType	M	The current lifecycle status of the Product Offering.	Appointment Status

7.3.3. WorkOrder Notification.

Figure 26 presents the whole WorkOrder Management data model the data types, requirements related to them and mapping to MEF 113 specifications are discussed later in this section.

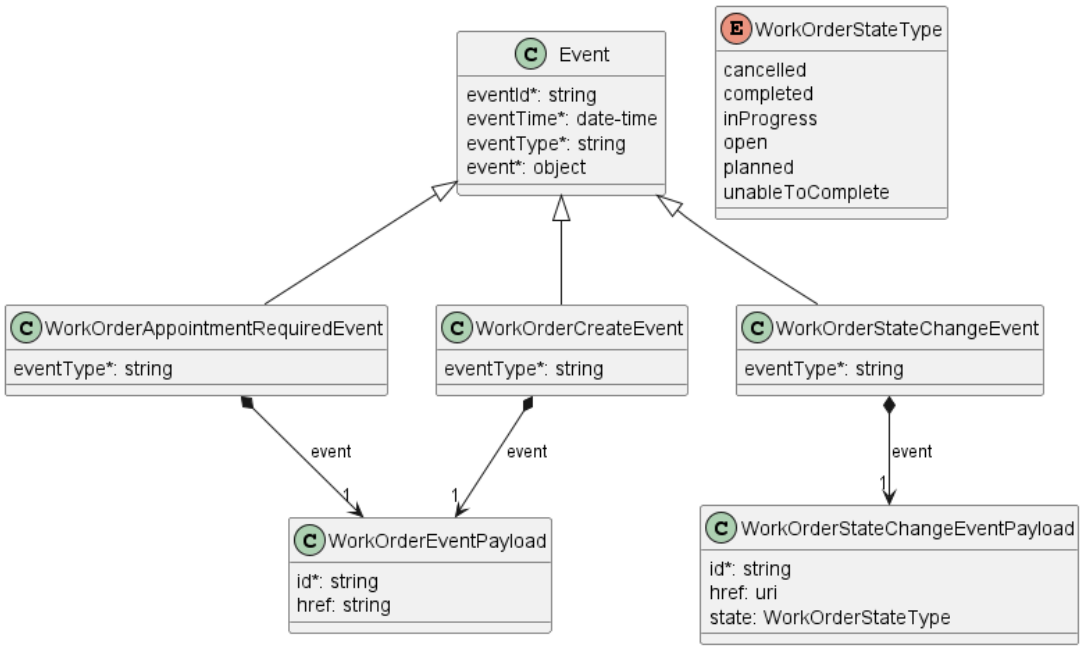


Figure 26. WorkOrder Notification Data Model

7.3.2.1. Type WorkOrderAppointmentRequiredEvent

Description:

Inherits from:

- Event

Name	Type	M/O	Description	MEF 113
eventType	string	M	Indicates the type of the event.	Workorder Notification Type
event	WorkOrderEventPayload	M	A reference to the object that is the source of the notification.	Workorder Identifier

7.3.2.2. Type WorkOrderCreateEvent

Description:

Inherits from:

- Event

Name	Type	M/O	Description	MEF 113
eventType	string	M	Indicates the type of the event.	Workorder Notification Type
event	WorkOrderEventPayload	M	A reference to the object that is the source of the notification.	Workorder Identifier

7.3.2.3. Type WorkOrderEventPayload

Description: The identifier of the WorkOrder is subject of this event.

Name	Type	M/O	Description	MEF 113
id	string	M	ID of the WorkOrder attributed by quoting system	Workorder Identifier
href	string	O	Hyperlink to access the WorkOrder	Not represented in MEF 113

7.3.2.4. Type WorkOrderStateChangeEvent

Description:

Inherits from:

- Event

Name	Type	M/O	Description	MEF 113
eventType	string	M	Indicates the type of the event.	Workorder Notification Type
event	WorkOrderStateChangeEventPayload	M	A reference to the object that is the source of the notification.	Workorder Identifier

7.3.2.5. Type WorkOrderStateChangeEventPayload

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

Name	Type	M/O	Description	MEF 113
id	string	M	ID of the Product Offering attributed by the Seller	Workorder Identifier
href	uri *format = uri*	O	Hyperlink to access the Product Offering	Not represented in MEF 113
state	WorkOrderStateType	O	The current state of the WorkOrder	WorkOrder State

8. References

- JSON, JSON Schema: A Media Type for Describing JSON Documents and associated documents, by Austin Wright and Henry Andrews, March 2018. Copyright © 2018 IETF Trust and the persons identified as the document authors. All rights reserved.
- MEF 55.1, Lifecycle Service Orchestration (LSO): Reference Architecture and Framework, February 2021
- MEF 55.1.1, Amendment to MEF 55.1: Reference Architecture and Framework - Terminology, June 2023
- MEF 113, Trouble Ticketing Business Requirements and Use Cases, July 2022
- Mplify 124.1, LSO Cantata and LSO Sonata Trouble Ticket and Incident Management API - Developer Guide, July 2025
- MEF 128.1, LSO API Security Profile, April 2024
- Mplify 150, Installation Place and Service Site Management Business Requirements and Use Cases, June 2025
- OAS-V3, February 2020
- REST, Fielding, Roy Thomas, Architectural Styles and the Design of Network-based Software Architectures (Ph.D.).
- RFC 2119, Key words for use in RFCs to Indicate Requirement Levels, March 1997
- RFC 3986 Uniform Resource Identifier (URI): Generic Syntax, January 2005
- RFC 7231, Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content, June 2014
- RFC 8174, Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words, May 2017
- TMF630, TMF630 API Design Guidelines 4.2.0
- TMF646, TMF646 Appointment Management API REST Specification R19.0.1
- TMF697, TMF697 Work Order Management API REST Specification v4.0.0

Appendix A Acknowledgments

Mike **BENCHECK**

Tomasz **CHMAL**

Pankaj **BODADE**

Michał **ŁĄCZYŃSKI**

Jack **PUGACZEWSKI**

Patrick **ROOSEN**

Fahim **SABIR**