MEF

Working Draft
MEF W148 v0.1

LSO Allegro, LSO Interlude and LSO Legato Fault Management API - Developer Guide

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

December 2024

Table of Contents

List of Contributing Members

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

Member

Table 1. Contributing Members

1. Abstract

This standard is intended to assist the implementation of the Application Programming Interfaces (APIs) for the Fault Management functionality of the Service Orchestration Function at the LSO Allegro, LSO Interlude and LSO Legato Interface Reference Points (IRPs), for which requirements and use cases are defined in MEF W133.1 [MEF133.1]. The requirements and use cases are the same for all IRPs. This standard consists of this document and complementary API definitions for Fault Management and Fault Management Notification.

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

MEF-LSO-Allegro-SDK

- `serviceApi/fm/faultManagement.api.yaml`
- `serviceApi/fm/faultNotification.api.yaml`

MEF-LSO-Interlude-SDK

- `serviceApi/fm/faultManagement.api.yaml`
- `serviceApi/fm/faultNotification.api.yaml`

MEF-LSO-Legato-SDK

- `serviceApi/fm/faultManagement.api.yaml`
- `serviceApi/fm/faultNotification.api.yaml`

The Fault Management API is defined using OpenAPI 3.0 [OAS-V3]

2. Terminology and Abbreviations

This section aims to clarify the terminology used throughout this document. In many cases, the authoritative definitions of terms can be found in separate documents. To ensure accuracy and consistency, the third column of this document serves to provide the appropriate references from MEF or external sources that govern these definitions.

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

- MEF W133.1 *Allegro, Interlude and Legato Fault Management and Performance Monitoring BR&UC* February 2023 [MEF 133.1]
- MEF 55.1, *Lifecycle Service Orchestration (LSO): Reference Architecture and Framework* February 2021 [MEF 55.1]

Term	Definition	Source
API Endpoint	The endpoint of a communication channel (the complete URL of an API Resource) to which the HTTP-REST requests are addressed to operate on the *API Resource*.	rapidapi.com This document
API Resource	A REST Resource. In REST, the primary data representation is called Resource. In this document, *API Resource* is defined as an OAS *SchemaObject* with specified *API Endpoints*.	restfulapi.net This document
Notification	A notification is a representation of an event that is exchanged between interested parties. An event is a significant occurrence or change in system state that is important from the perspective of system administration.	MEF W133.1
On-Demand	Fault Management Job actions that are initiated for a limited time to carry out the Fault Management measurements.	MEF W133.1
OpenAPI	The OpenAPI 3.0 Specification, formerly known as the Swagger specification is an API description format for REST APIs.	spec.openapis.org
Operation	An interaction between the Server and Client, potentially involving multiple back-and-forth transactions.	This document
Passive	Fault Management Job action to support the collection and reporting of network and service faults.	MEF W133.1
Proactive	Fault Management Job actions that are carried on continuously to permit timely reporting of fault status.	MEF W133.1
REST API	Representational State Transfer. REST provides a set of architectural constraints that, when applied as a whole, emphasizes scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems.	REST API
SchemaObject	The construct that allows the definition of input and output data types. These types can represent object classes, as well as primitives and array specifications.	spec.openapis.org

Table 2. Terminology

Term	Definition	Source
API	Application Programming Interface. In this document, API is used synonymously with REST API.	This document
BUS	Business Applications	MEF 55.1
CUS	Customer Application Coordinator	MEF 55.1
FM	Fault Management	MEF W133.1
IRP	Interface Reference Point	MEF 55.1
OAS	OpenAPI Specification	openapis.org
SOF	Service Orchestration Functionality	MEF 55.1

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 [RFC2119], RFC 8174 [RFC8174]) when, and only when, they appear in all capitals, as shown here. All key words must be in bold text.

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

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

Fault Management API allows to manage Fault Management Jobs, collect Fault Management Reports, and receive notifications related to these entities. This provides the ability for the Buyer/Client to request that the Seller/Server performs FM Job on a Service, and to retrieve the results of the Job. Examples of the Fault Management Job are Link Trace or Loopback.

This standard specification document describes the Application Programming Interface (API) for Fault Management functionality of the LSO Allegro Interface Reference Point (IRP), LSO Interlude Interface Reference Point (IRP) and LSO Sonata IRP as defined in the *MEF 55.1 Lifecycle Service Orchestration (LSO): Reference Architecture and Framework* [MEF55.1]. The LSO Reference Architecture is shown in Figure 1 with the three IRPs highlighted.
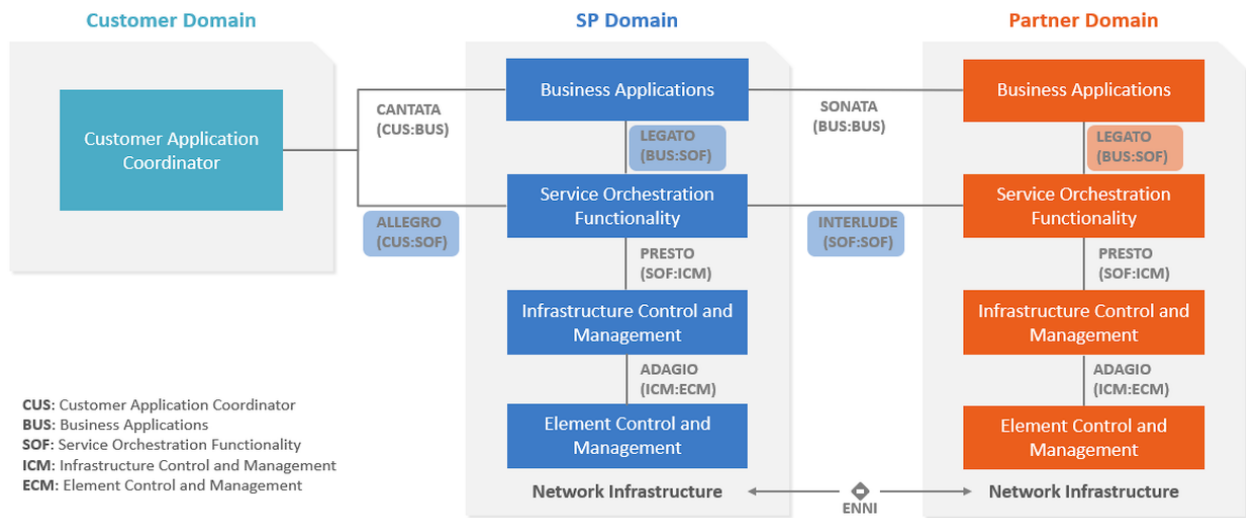
Figure 1. The LSO Reference Architecture

Note: The use cases and business requirements in this document assume a two-actor relationship based on the set of actors in the LSO architecture. The names of the relationships are specific to the Interface Reference Point. For both Allegro and Interlude there is a Buyer and Seller. For Allegro the Buyer is the Customer and the Seller is the Service Provider. In Interlude the Buyer is the Service Provider and the Seller is the Partner. In the case of the Legato IRP, given this is within a single Service Provider or Partner, the relationship is between Client and Server, where the Business Application (BA) is the Client, and the Service Orchestration Functionality (SOF) is the Server. Considering this duality, actors in the document are referred to as Buyer/Client and Seller/Server.

4.1. Description

This standard is scoped to cover APIs for following Service Orchestration Functionalities:

- Fault Management
    - Includes management of Fault Management Jobs and collecting Fault Reports
- Fault Management Notification
    - Includes Event Subscription/Hub and Listener notification functions

This document supports interactions over the Legato interface within a single operator as well as interaction with Partner Domain and Customer Domain through Interlude and Allegro interfaces respectively.

Business Applications (BUS), Customer Application Coordinator (CUS) and Service Orchestration Functionality (SOF) systems use the information contained within this document.

This standard is intended to support the design of API implementations that enable interoperable SOF operations (in the scope of this standard) across the Allegro IRP, Interlude IRP, and Legato IRP.

The Fault Management API allows the Buyer (CUS/SOF) or Client (BUS) to provision fault management job in the Server (intra-operator SOF) or in the Seller (inter-operator SOF) and collect fault reports from Server/Seller.

4.2. Conventions in the Document

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

4.3. Relation to Other Documents

This API implements the Fault Management related requirements and use cases that are defined in MEF W133.1 [MEF133.1]. Fault Management Use Cases must support the use of MEF service fault indicators specifications as payload.

4.4. Approach

As presented in Figure 2. the Allegro, Interlude, and Legato API frameworks consist of three structural components:

- Generic API framework
- Service-independent information (Function-specific information and Function-specific operations)
- Service-specific information (MEF service specification data model)
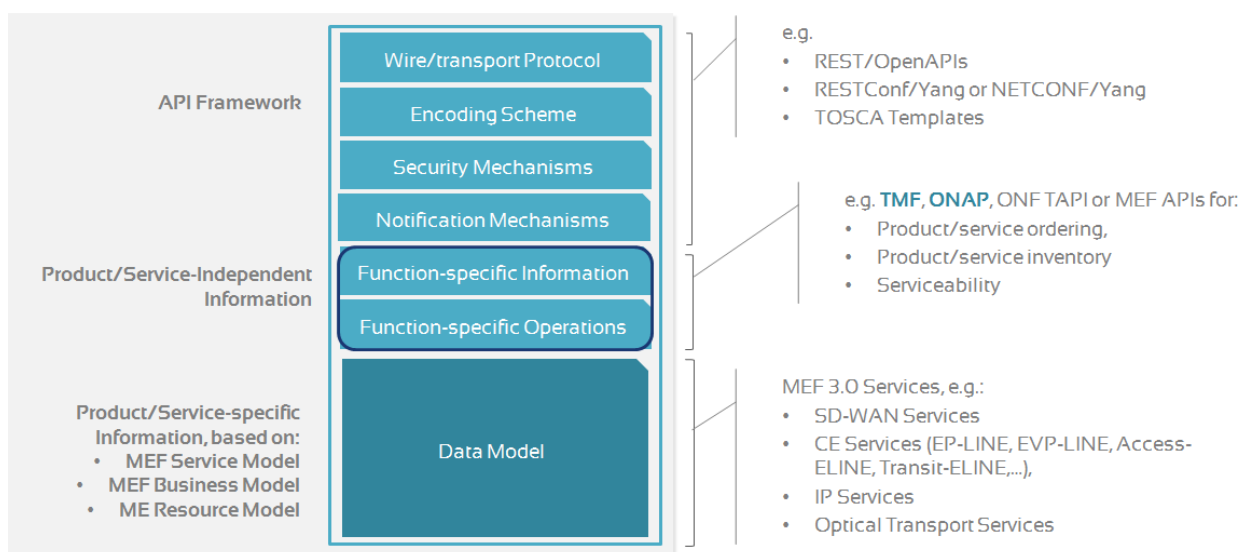
Figure 2. Allegro, Interlude and Legato API Structure

The essential concept behind the framework is to decouple the common structure, information, and operations from the specific service information content. Firstly, the Generic API Framework defines a set of design rules and patterns that are applied across all Allegro, Interlude, and Legato APIs. Secondly, the service-independent information of the framework focuses on a model of a particular Allegro, Interlude, or Legato functionality and is agnostic to any of the service specifications. For example, this standard is describing the Fault Management model and operations that allow provisioning of the fault indicators for any types of service. Finally, the service-specific information part of the framework focuses on attributes related to fault management configuration and requirements for provisioning intra-provider or inter-provider FM objectives.

This Developer Guide does not define MEF service fault management specifications but can be used in combination with any specifications defined by or compliant with MEF. MEF service fault management schemas are defined by:

- MEF 152: Carrier Ethernet Payload Schema/Guide for SOAM [MEF152]
- MEF 153: IP/IPVPN Schema/Guide for SOAM [MEF153]
- MEF 154: SD-WAN Schema/Guide for SOAM [MEF154]

Figure 3 presents the relationship between the Fault Mamagement API entities and the fault management specification model. The `ServiceSpecificPayloadAttribute` serves as an extension point for configuring service-specific fault parameters. On the other hand, the `ResultPayload` acts as an extension point for capturing and representing the fault management results.
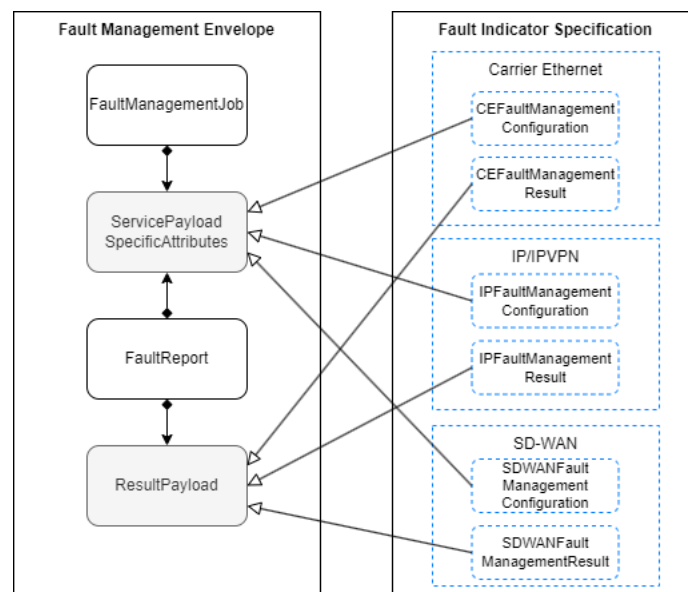
Figure 3. Fault Indicators specification for Allegro, Interlude, Legato

4.5. High-Level Flow

The Fault Management API in essence allows the Buyer/Client to request SOF to provision intervals, schedules, and fault indicators between ordered pairs. An ordered pair is an association between two endpoints. The Fault Notifications API provides means to exchange information about significant changes in the system state between interested parties. Figure 4 presents an exemplary high-level flow of Fault Management Job provisioning.
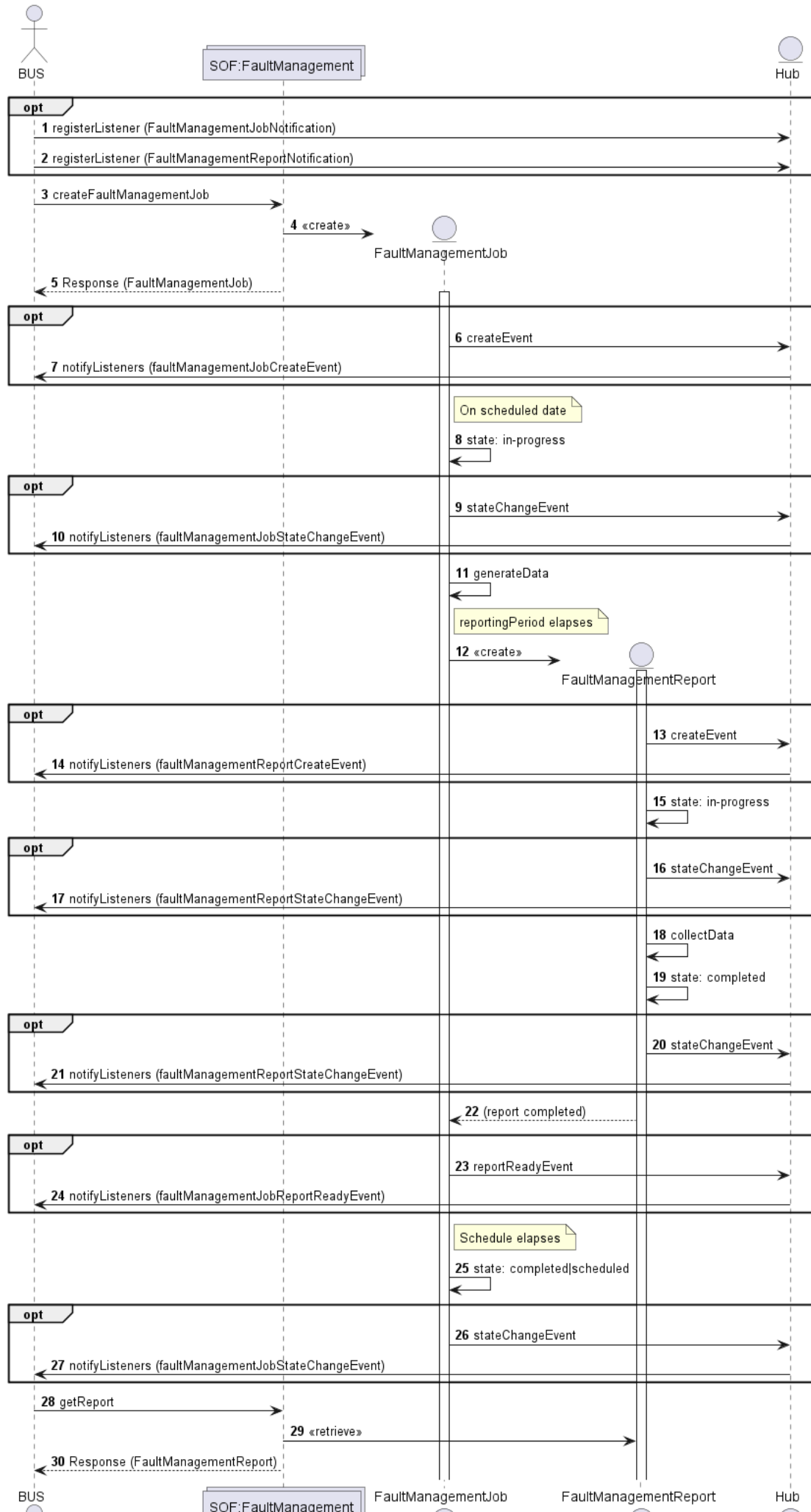
Figure 4. High-Level Flow of Fault Management Job Provisioning

The following steps describe the high-level flow:

- (optional) The BUS system registers for notifications.
  Note1: Fault Notifications are optional and do not impact end-to-end flow
- The BUS system triggers provisioning of Fault Management Job.
  - The SOF provisions Fault Management by creating a `FaultManagementJob` which contains the configuration of fault indicators and related service.
  - `FaultManagementJob` also carries a configuration including granularity, reporting period, schedule definition, and output format.
  - The `FaultManagementJob` is processed by the SOF as per the state transition rules described in 6.1.4.
  - (optional) The SOF reports the `FaultManagementJob` state changes.
  - On a scheduled date according to schedule definition, fault job is executed and data generation is started.
  - When the configured reporting period elapses, a `FaultManagementReport` entity is created to collect the fault management results.
  - `FaultManagementReport` is processed as per the state transition rules described in 6.13.1.
  - (optional) The SOF reports the `FaultManagementReport` state change.
  - The BUS system can collect `FaultManagementReport` through *Fault Management API*

The same *Fault Management API* is used by the BUS to create **new** `FaultManagementJob` instances, as well as update **existing** ones or trigger state transitions (e.g. cancel **existing** `FaultManagementJob` instance)

Figure 5 presents relations between entities that are managed through *Fault Management API*. The diagram is simplified and does not contain all types of operations.
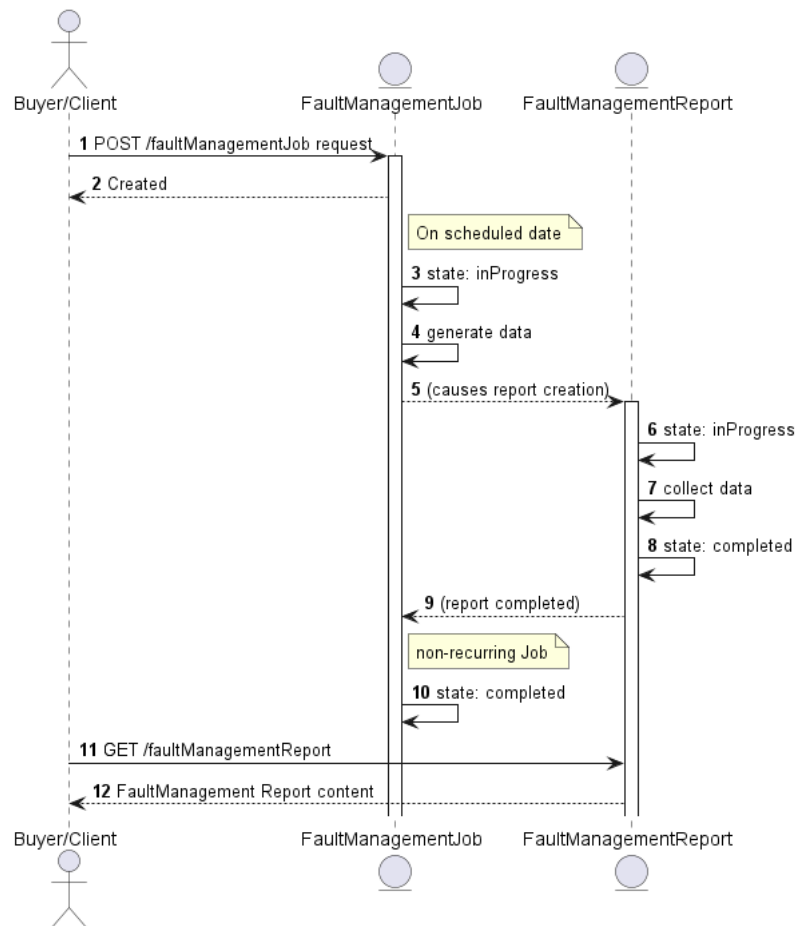
Figure 5. The flow between API endpoints

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 explains the design pattern that is used to combine service-agnostic and service-specific parts of API payloads. Finally, payload validation and API security aspects are discussed.

5.1. High-level use cases

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

Buyer/Client

1. Create Fault Management Job
2. Retrieve Fault Management Job List
3. Retrieve Fault Management Job by Job Identifier
4. Modify Fault Management Job
5. Retrieve Modify Fault Management Job List
6. Retrieve Modify Fault Management Job by Identifier
7. Cancel Fault Management Job
8. Retrieve Cancel Fault Management Job List
9. Retrieve Cancel Fault Management Job by Identifier
10. Suspend Fault Management Job
11. Resume Fault Management Job
12. Create Fault Management Job Complex Query
13. Retrieve Fault Management Report List
14. Retrieve Fault Management Report by Report Identifier
15. Create Fault Management Report Complex Query

Figure 6. Use cases

5.2. API Endpoint and Operation Description

5.2.1. Seller/Server (SOF) side Fault Management API Endpoints

Base URL for Allegro:

`https://{{serverBase}}:{{port}}{{?/sof_prefix}}/mefApi/allegro/faultManagement/v1/`

Base URL for Interlude:

`https://{{serverBase}}:{{port}}{{?/sof_prefix}}/mefApi/interlude/faultManagement/v1/`

Base URL for Legato:

`https://{{serverBase}}:{{port}}{{?/sof_prefix}}/mefApi/legato/faultManagement/v1/`

The following API endpoints are implemented by the Seller/Server (SOF) and allow the Buyer/Client (SOF/CUS/BUS) to create, retrieve and modify `FaultManagementJob` and `FaultManagementReport` instances. The endpoints and corresponding data model are defined in `serviceApi/fm/faultManagement.api.yaml`.

API Endpoint	Description	MEF W133.1 Use Case Mapping
POST /faultManagementJob	A request initiated by the Buyer/Client to create a Fault Management Job in the Seller/Server system.	1
GET /faultManagementJob	The Buyer/Client requests a list of Fault Management Jobs based on a set of filter criteria.	N/A
GET /faultManagementJob/{{id}}	The Buyer/Client requests detailed information about a single Fault Management Job.	N/A

API Endpoint	Description	MEF W133.1 Use Case Mapping
POST /modifyFaultManagementJob	A request initiated by the Buyer/Client to modify a Fault Management Job in the Seller/Server system.	2
GET /modifyFaultManagementJob	The Buyer/Client requests a list of Modify Fault Management Job based on a set of filter criteria.	2
GET /modifyFaultManagementJob/{{id}}	The Buyer/Client requests detailed information about a single Modify Fault Management Job.	2
POST /cancelFaultManagementJob	A request initiated by the Buyer/Client to cancel a Fault Management Job in the Seller/Server system.	3
GET /cancelFaultManagementJob	The Buyer/Client requests a list of Cancel Fault Management Job based on a set of filter criteria.	3
GET /cancelFaultManagementJob/{{id}}	The Buyer/Client requests detailed information about a single Cancel Fault Management Job.	3
POST /faultManagementJob/{{id}}/suspend	A request initiated by the Buyer/Client to suspend a Fault Management Job in the Seller/Server system.	4
POST /faultManagementJob/{{id}}/resume	A request initiated by the Buyer/Client to resume a Fault Management Job in the Seller/Server system.	5
POST /faultManagementJobComplexQuery	A request initiated by the Buyer/Client to create a Fault Management Job Complex Query in the Seller/Server system.	N/A
GET /faultManagementReport	The Buyer/Client requests a list of Fault Management Reports based on a set of filter criteria.	9
GET /faultManagementReport/{{id}}	The Buyer/Client requests detailed information about a single Fault Management Report, including the content of the report.	10
POST /faultManagementReportComplexQuery	A request initiated by the Buyer/Client to create a Fault Management Report Complex Query in the Seller/Server system.	9
GET /trackingRecord	The Buyer/Client requests a list of Tracking Records based on a set of filter criteria.	N/A
GET /trackingRecord/{{id}}	The Buyer/Client requests detailed information about a single Tracking Record.	N/A

Table 4. Seller/Server (SOF) Fault Management mandatory API endpoints

[R1] Seller/Server (SOF) **MUST** support all API endpoints listed in Table 4.

[R2] The Seller/Server **MUST** support the retrieval of a List of Fault Management Reports Use Case. [MEF133.1 R33]

[R3] Buyer/Client **MUST** support the retrieval of a List of Fault Management Reports Use Case. [MEF133.1 R34]

API endpoints listed in Table 5 are optional and may be exposed by the SOF.

API Endpoint	Description	MEF W133.1 Use Case Mapping
POST /hub	The Buyer/Client requests to subscribe to the Fault Management Job or Fault Management Report Notifications.	6
GET /hub/{{id}}	The Buyer/Client retrieves a specific `EventSubscription` from the SOF, that matches the `id` value provided as `path` parameter.	N/A
DELETE /hub/{{id}}	The Buyer/Client requests to unsubscribe from the Fault Management Job or Fault Management Report Notifications.	8

Table 5. Seller/Server (SOF) Fault Management optional API endpoints

5.2.2. Buyer/Client (CUS, BUS, SOF) side Fault Management API Endpoints

Base URL for Allegro:

```
https://{{serverBase}}:{{port}}
{{?/bus_cus_sof_prefix}}/mefApi/allegro/faultNotification/v1/
```

Base URL for Interlude:

```
https://{{serverBase}}:{{port}}
{{?/bus_cus_sof_prefix}}/mefApi/interlude/faultNotification/v1/
```

Base URL for Legato:

```
https://{{serverBase}}:{{port}}
{{?/bus_cus_sof_prefix}}/mefApi/legato/faultNotification/v1/
```

The following API Endpoints are used by SOF to post notifications to registered CUS, BUS, or SOF listeners. The endpoints and corresponding data model are defined in `serviceApi/fm/faultNotification.api.yaml`

API Endpoint	Description	MEF W133.1 Use Case Mapping

API Endpoint	Description	MEF W133.1 Use Case Mapping
POST /listener/faultManagementJobCreateEvent	A request initiated by the Seller/Server to notify Buyer/Client on `FaultManagementJob` instance creation.	7
POST /listener/faultManagementJobStateChangeEvent	A request initiated by the Seller/Server to notify Buyer/Client on the `FaultManagementJob` instance state change.	7
POST /listener/faultManagementJobAttributeValueChangeEvent	A request initiated by the Seller/Server to notify Buyer/Client on the `FaultManagementJob` instance attribute value change.	7
POST /listener/faultManagementJobReportReadyEvent	A request initiated by the Seller/Server to notify Buyer/Client that `FaultManagementReport` was generated for the `FaultManagementJob` instance.	7
POST /listener/faultManagementJobReportPreparationErrorEvent	A request initiated by the Seller/Server to notify Buyer/Client that `FaultManagementReport` was not generated for the `FaultManagementJob` instance due to an error.	7
POST /listener/cancelFaultManagementJobStateChangeEvent	A request initiated by the Seller/Server to notify Buyer/Client on the `CancelFaultManagementJob` instance state change.	7
POST /listener/modifyFaultManagementJobStateChangeEvent	A request initiated by the Seller/Server to notify Buyer/Client on the `ModifyFaultManagementJob` instance state change.	7
POST /listener/faultManagementReportCreateEvent	A request initiated by the Seller/Server to notify Buyer/Client on `FaultManagementReport` instance creation.	7

API Endpoint	Description	MEF W133.1 Use Case Mapping
POST /listener/faultManagementReportStateChangeEvent	A request initiated by the Seller/Server to notify Buyer/Client on the `FaultManagementReport` instance state change.	7

Table 6. Buyer/Client (CUS, BUS, SOF) Fault Management API endpoints

[O1] The Buyer/Client (CUS, BUS, SOF) **MAY** support API endpoints listed in Table 6.

[O2] The Buyer/Client (CUS, BUS, SOF) **MAY** register to receive Fault Management notifications.

[R4] The Seller/Server FM Notifications MUST be sent to Buyer/Clients who have subscribed to FM Notifications. [MEF133.1 R30]

[R5] The Seller/Server FM Notifications MUST Not be sent to Buyer/Clients who have not subscribed to FM Notifications. [MEF133.1 R31]

5.3. Integration of Service Fault Management Specification into Fault Management API

Fault Management API discussed in this document is a generic envelope that allows for the lifecycle management of relevant Fault Management objects. The API itself does not provide explicit definitions for configuring fault
management or prescribing the structure of output data. However, it offers flexible extensibility to accommodate the configuration of service-specific fault indicators and results. This allows for customization and adaptation to various fault management requirements and desired data formats. This fault configuration and result schemas are defined using JsonSchema (draft 7) format JSON Schema draft 7 and can be integrated into the `FaultManagementJob` and `FaultManagementReport` using the TMF extension pattern.

The extension hosting types in the API data model are:

- `ServicePayloadSpecificAttributes` - this type is extended with Fault Management configuration schema
- `ResultPayload` - this type is extended with Fault Management result schema

The `@type` attribute of those extension hosting types must be set to a value that uniquely identifies the fault management schmeas. A unique identifier for MEF standard service schemas is in URN format and is assigned by MEF. This identifier is provided as root schema `$id`. Use of non-MEF standard fault management configuration is allowed. In such a case the schema identifier must be agreed upon between the Buyer/Client and the Seller/Server.

The example below shows a header of a schema, which describes the IP service Fault Management configuration, where `"$id": urn:mef:lso:spec:legato:ip-fault-management-configuration:v0.0.1:all` is the above-mentioned URN:

```
'$schema': http://json-schema.org/draft-07/schema#
'$id': urn:mef:lso:spec:legato:ip-fault-management-configuration:v0.0.1:all
title: MEF LSO Legato - IP Fault Management Configuration
```

Fault configuration payload is introduced in multiple FM API entities through a `servicePayloadSpecificAttributes` attribute of type `ServicePayloadSpecificAttributes` which is used as an extension point for configuration attributes.

In terms of fault results, the appropriate payload is introduced via `ReportContent`. This entity has a `measurementDataPoints` array of items of type `ResultPayload` which is used as an extension point for service-specific output content.

Implementations might choose to integrate selected Fault Management specifications to data model during development. In such a case an integrated data model is built, and fault configurations are in an inheritance relationship accordingly with either `ServicePayloadSpecificAttributes` or `ResultPayload` as described in the OAS specification. This pattern is called **Static Binding**. The snippets below present an example of a static binding of the envelope API with exemplary MEF fault management configuration, for both extension points.

```
ServicePayloadSpecificAttributes:
  type: object
  description: ServicePayloadSpecificAttributes is used as an extension point
    for MEF-specific service fault management configuration. The `@type`
    attribute is used as a discriminator.
  discriminator:
    mapping:
                                        urn:mef:lso:spec:legato:ip-fault-management-configuration:v0.0.1:all:
'#/components/schemas/IpFaultManagementConfiguration'
    propertyName: '@type'
  properties:
    '@type':
      type: string
      description:
        The name that uniquely identifies type of fault management job
        configuration. In the case of MEF services, this is the URN provided in
        the fault management configuration specification. The named type must
        be a subclass of ServicePayloadSpecificAttributes.
```

```
IpFaultManagementConfiguration:
  allOf:
    - $ref: '#/components/schemas/ServicePayloadSpecificAttributes'
    - type: object
      description: IP Fault Management Configuration Schema.
```

```
ResultPayload:
  type: object
  description:
    ResultPayload is used as an extension point for MEF-specific fault
    management job results. The `@type` attribute is used as a discriminator.
  discriminator:
    mapping:
                                        urn:mef:lso:spec:legato:ip-fault-management-results:v0.0.1:all:
'#/components/schemas/IpFaultManagementResults'
    propertyName: '@type'
  properties:
    '@type':
      type: string
      description:
        The name that uniquely identifies the type of results that are
        returned by the Fault Management Report. In the case of MEF
        services, this is the URN provided in the fault management results
        specification. The named type must be a subclass of ResultPayload.
```

```
IpFaultManagementResults:
  allOf:
    - $ref: '#/components/schemas/ResultPayload'
    - type: object
      description: IP Fault Management Results Schema.
```

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

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

[R6] `ServicePayloadSpecificAttributes` and `ResultPayload` types are extension points that **MUST** be used to integrate service fault management properties into a request/response payload.

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

[R8] Attributes specified in the payload must conform to the fault management configuration specified in the `@type` property.
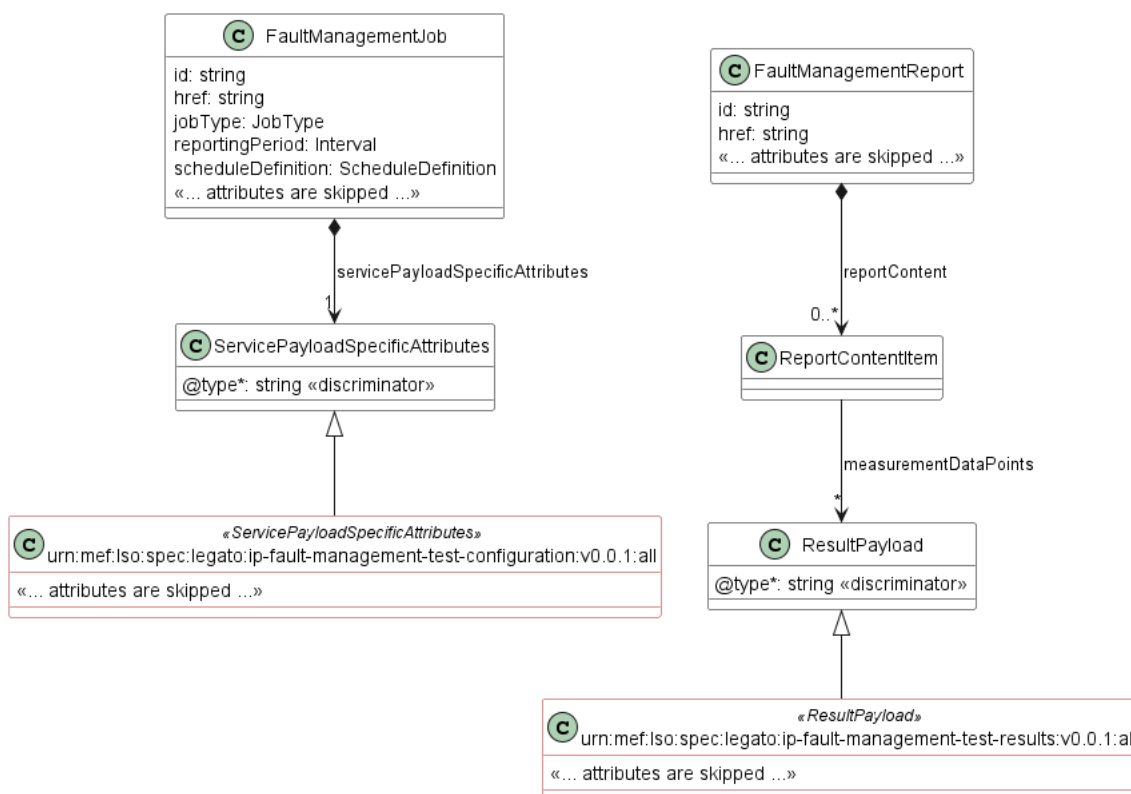
Figure 7. The Extension Pattern with Sample Service-Specific Extension

Figure 7 presents two MEF Fault Management schemas that represent configuration and result classes for IP services. When these schemas are used, the `@type` of `ServicePayloadSpecificAttributes` takes `"urn:mef:lso:spec:legato:ip-fault-management-configuration:v0.0.1:all"` value to indicate which fault management specification should be used to interpret a set of service-specific attributes included in the payload. Similarly, for `ResultPayload`, the `@type` attribute takes `"urn:mef:lso:spec:legato:ip-fault-management-results:v0.0.1:all"` value which indicates how the resulting fault collection should be interpreted.

5.4. Model structure and validation

The structure of the payloads exchanged via Allegro, Interlude, and Legato Fault Management API endpoints is defined using:

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

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

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

5.5. Security Considerations

Although the Legato IRP is internal to a Service Provider/Operator business boundary, it is expected that some minimal security mechanisms are in place for any communication over this IRP. There must also be authorization mechanisms in place to control what a particular Buyer/Client or SOF is allowed to do and what information may be obtained. For Allegro and Interlude IRPs, security should follow rules for external communication. The definition of the exact security mechanism and configuration is outside the scope of this document. The LSO Security mechanisms are defined by MEF 128 *LSO API Security Profiles* [MEF128].

6. API Interactions and Flows

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

Use Case #	Use Case Name	Use Case Description
1	Create Fault Management Job	A request initiated by the Buyer/Client to create a Fault Management Job in the Seller/Server system to indicate Fault Management Job configuration.
2	Retrieve Fault Management Job List	The Buyer/Client requests a list of Fault Management Job based on a set of filter criteria. The Seller/Server returns a summarized list of FM Jobs.
3	Retrieve Fault Management Job by Job Identifier	The Buyer/Client requests detailed information about a single Fault Management Job based on the Fault Management Job Identifier.
4	Modify Fault Management Job	A request initiated by the Buyer/Client to modify a Fault Management Job in the Seller/Server system.
5	Retrieve Modify Fault Management Job List	The Buyer/Client requests a list of Modify Fault Management Job based on a set of filter criteria.
6	Retrieve Modify Fault Management Job by Identifier	The Buyer/Client requests detailed information about a single Modify Fault Management Job based on the Modify Fault Management Job Identifier.
7	Cancel Fault Management Job	A request initiated by the Buyer/Client to cancel a Fault Management Job in the Seller/Server system.
8	Retrieve Cancel Fault Management Job List	The Buyer/Client requests a list of Cancel Fault Management Job based on a set of filter criteria.
9	Retrieve Cancel Fault Management Job by Identifier	The Buyer/Client requests detailed information about a single Cancel Fault Management Job based on the Cancel Fault Management Job Identifier.
10	Suspend Fault Management Job	A request initiated by the Buyer/Client to suspend a Fault Management Job in the Seller/Server system.
11	Resume Fault Management Job	A request initiated by the Buyer/Client to resume a Fault Management Job in the Seller/Server system.
12	Create Fault Management Job Complex Query	A request initiated by the Buyer/Client to create a Fault Management Job Complex Query in the Seller/Server system.
13	Retrieve Fault Management Report List	The Buyer/Client requests a list of Fault Management Reports based on a set of filter criteria. The Seller/Server returns a summarized list of FM Reports.

Use Case #	Use Case Name	Use Case Description
14	Retrieve Fault Management Report by Report Identifier	The Buyer/Client requests detailed information, including generated content, about a single Fault Management Report based on the Fault Management Report Identifier.
15	Create Fault Management Report Complex Query	A request initiated by the Buyer/Client to create a Fault Management Report Complex Query in the Seller/Server system.
16	Retrieve Tracking Record List	The Buyer/Client requests a list of Tracking Records based on a set of filter criteria. The Seller/Server returns a summarized list of Tracking Records.
17	Retrieve Tracking Record List by Identifier	The Buyer/Client requests detailed information about a single Tracking Record based on the Tracking Record Identifier.
18	Register for Event Notifications	The Buyer/Client requests to subscribe to Fault Management Job or Fault Management Report Notifications.
19	Send Event Notification	A request initiated by the Seller/Server to notify the Buyer/Client.

Table 7. Use cases description

6.1. Use Case 1: Create a Fault Management Job

A Fault Management Job entity is used by the Buyer/Client to specify the fault management job configuration specific to a service instance. This provides the ability for the Buyer/Client to request that the Seller/Server performs FM activities on a service, and consequently to retrieve the fault results. Examples of fault management job are Link Trace and Loopback. FM Job is typically run as part of a troubleshooting or diagnostic process of a service. Besides, Fault Management Job entity provides configuration, including measurement intervals and schedules of FM job execution.

The Fault Management Jobs should result in Fault Management Collections (Reports) that will provide the Buyer/Client with fault indicator results.

There are three types of Fault Management Job:

- Proactive - carried on continuously to permit timely reporting of fault status. Typically, it runs indefinitely.
- On-Demand - initiated for a limited time, typically a single run or non-continual run, to carry out the FM job and support troubleshooting during service assurance.
- Passive - supports the collection and reporting of network and service faults.

6.1.1. Interaction flow

The flow of this use case is shown in Figure 8.
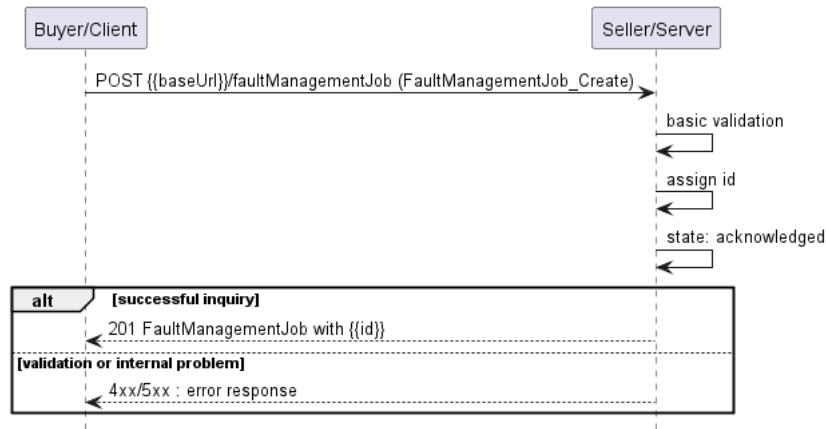
Figure 8. Use Case 1 - Fault Management Job create request flow

The Buyer/Client sends a request with a `FaultManagementJob_Create` type in the body. The Seller/Server performs request validation, assigns an `id`, and returns the `FaultManagementJob` type in the response body, with a `state` set to `acknowledged`. From this point, the Fault Management Job is ready for further processing. The Buyer/Client can track the progress of the process either by subscribing for notifications or by periodically polling the `FaultManagementJob`. The two patterns are presented in the following diagrams.
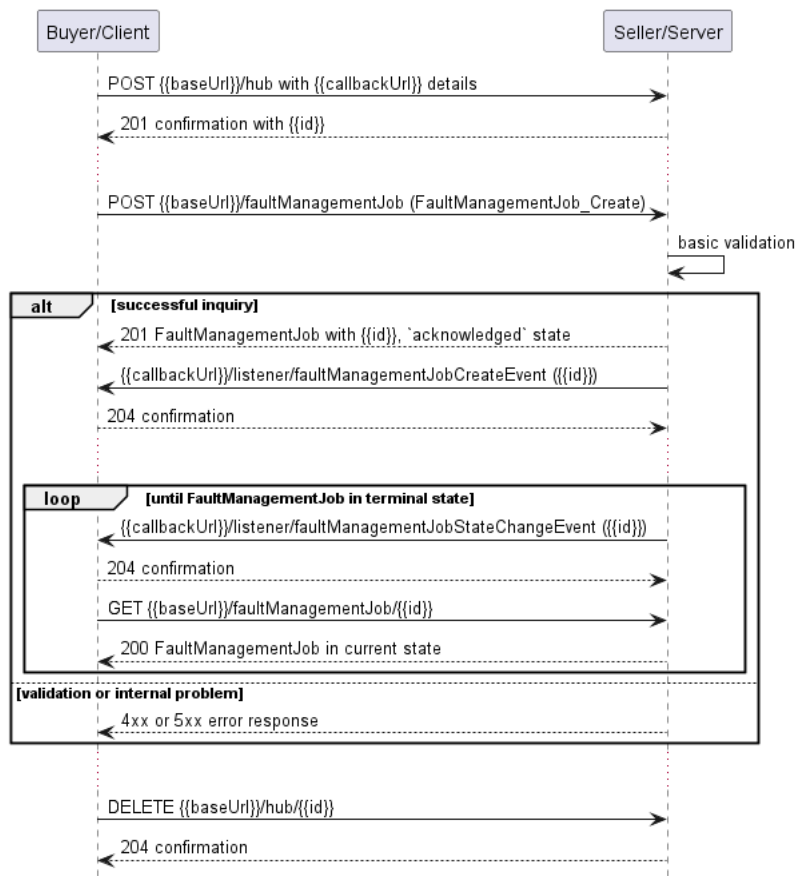
Figure 9. Fault Management Job progress tracking - Notifications
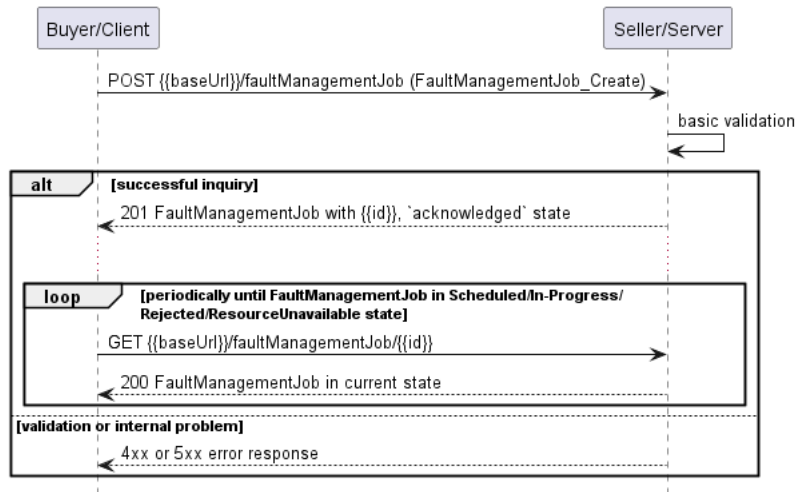
Figure 10. Fault Management Job progress tracking - Polling

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

6.1.2. Create Fault Management Job Request

Figure 11 presents the most important part of the data model used during the Create Fault Management Job request (`POST /faultManagementJob`) and response. The model of the request message - `FaultManagementJob_Create` is a subset of the `FaultManagementJob` model and contains only attributes that can (or must) be set by the Buyer/Client. The Seller/Server (SOF) then enriches the entity in the response with additional information.

Note: `FaultManagementJob_Create` is an entity used by the Buyer/Client to make a request. `FaultManagementJob` is an entity used by the Seller/Server to provide a response. The request entity has a subset of attributes of the response entity. Thus for visibility of these shared attributes `FaultManagementJob_Common` has been introduced (this class is not supposed to be used directly in the exchange).

A `FaultManagementJob_Create` defines measurement intervals, schedules, and service-specific configuration of FM job (in `servicePayloadSpecificAttributes` section).

Section `servicePayloadSpecificAttributes` of the Create Fault Management Job request allows for the introduction of service-specific properties as the API payload. The extension mechanism is described in detail in Section 5.3.

The full list of attributes is available in Section 7 and in the API specification which is an integral part of this standard.
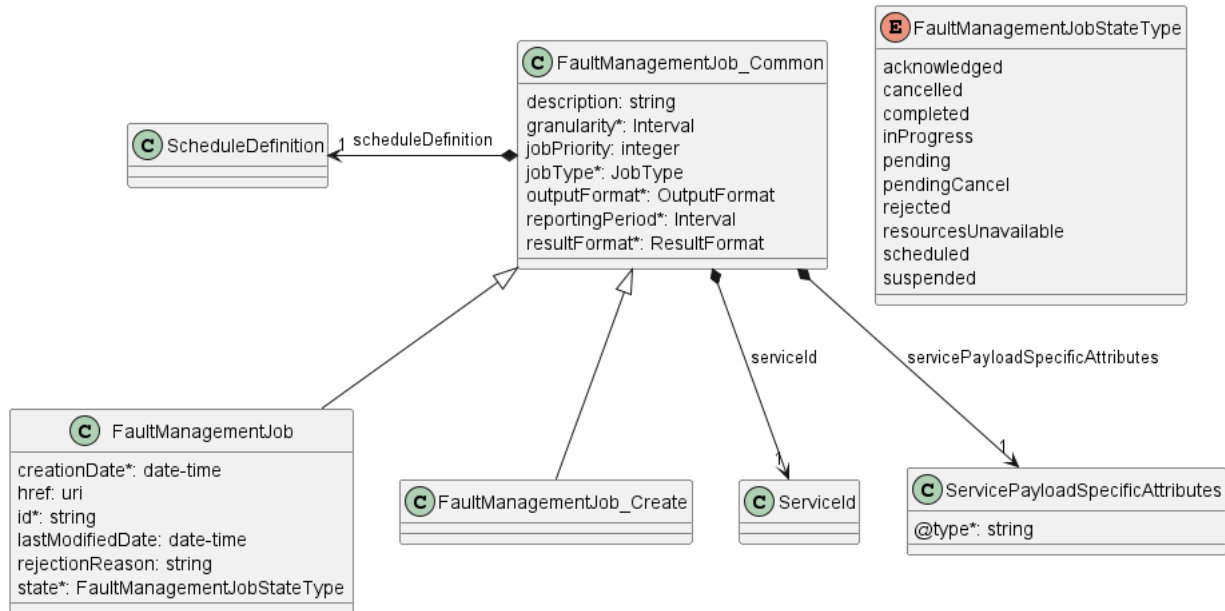
Figure 11. Fault Management Job Key Entities

To send a create Fault Management Job request the Buyer/Client uses the `createFaultManagementJob` operation from the API: `POST /faultManagementJob`. Some of the create Fault Management Job payload's attributes might be omitted to improve examples' readability.

Fault Management Job Create Request

```
{
  "description": "Exemplary Create Fault Management Job request",
  "granularity": "15 minutes",
  "jobPriority": 5,
  "jobType": "proactive",
  "outputFormat": "json",
  "reportingPeriod": "1 hour",
  "resultFormat": "payload",
  "scheduleDefinition": {
    "scheduleDefinitionStartTime": "2025-01-01T00:00:00.000Z",
    "scheduleDefinitionEndTime": "2026-01-01T00:00:00.000Z",
    "recurringSchedule": {
      "second": "0",
      "minute": "0",
      "hour": "*/2",
      "dayOfMonth": "*",
      "month": "*",
      "dayOfWeek": "*"
    },
    "executionDuration": "1 hour"
  },
  "serviceId": {
    "serviceIdFrom": "905d9f87-6478-4153-a5de-fcc70257f03c",
    "serviceIdTo": "1d7dd934-7aa3-4997-9b6a-906a3e35a08e"
  },
  "servicePayloadSpecificAttributes": {
    "@type": "urn:mef:lso:spec:legato:ip-fault-management-configuration:v0.0.1:all"
  }
}
```

[R11] The Buyer/Client's Create FM Job **MUST** contain the following attributes: [MEF133.1 R1]

- Job Type
- Output Format
- Granularity

- Reporting Period
- Result Format
- Schedule Definition
- Service Specific Payload
- Service ID

[O3] The Buyer/Client's Create FM Job **MAY** contain the following attributes: [MEF133.1 O1]

- Description
- FM Job Priority

6.1.3. Create Fault Management Job Response

Entities used for providing a response to Create Fault Management Job requests are presented in Figure 11. The Seller/Server responds with a `FaultManagementJob` type, which adds some attributes (like `id` or `state`) to the `FaultManagementJob_Create` that was used in the Buyer/Client request.

Note: The term "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/Server response. It has the same structure as in the retrieve by identifier operation.

`Fault Management Job` **Create Response**

```json
{
  "description": "Exemplary Create Fault Management Job request",
  "granularity": "15 minutes",
  "jobPriority": 5,
  "jobType": "proactive",
  "outputFormat": "json",
  "reportingPeriod": "1 hour",
  "resultFormat": "payload",
  "scheduleDefinition": {
    "scheduleDefinitionStartTime": "2025-01-01T00:00:00.000Z",
    "scheduleDefinitionEndTime": "2026-01-01T00:00:00.000Z",
    "recurringSchedule": {
      "second": "0",
      "minute": "0",
      "hour": "*/2",
      "dayOfMonth": "*",
      "month": "*",
      "dayOfWeek": "*"
    },
    "executionDuration": "1 hour"
  },
  "serviceId": {
    "serviceIdFrom": "905d9f87-6478-4153-a5de-fcc70257f03c",
    "serviceIdTo": "1d7dd934-7aa3-4997-9b6a-906a3e35a08e"
  },
  "servicePayloadSpecificAttributes": {
    "@type": "urn:mef:lso:spec:legato:ip-fault-management-configuration:v0.0.1:all"
  },
  "creationDate": "2025-01-01T00:00:00.000Z", << added by SOF >>
  "href": "{{baseUrl}}/faultManagement/v1/755e55e2-72b0-4e3b-af00-693e3beac691", << added by SOF >>
  "id": "755e55e2-72b0-4e3b-af00-693e3beac691", << added by SOF >>
  "lastModifiedDate": "2025-01-01T00:00:00.000Z", << added by SOF >>
  "state": "acknowledged" << added by SOF >>
}
```

Attributes that are set by the Seller/Server in the response are marked with the `<< added by SOF >>` tag.

[R12] The Seller/Server sets the Creation Time and Job Identifier attribute. [MEF133.1 R2]

[R13] The Seller's/Server's response **MUST** echo back all Buyer/Client provided attributes. [MEF133.1 R3]

[R14] The FM Job Identifier supplied by the Seller/Server **MUST** be unique within the Seller/Server's network. [MEF133.1 R4]

[R15] The Seller/Server **MUST** specify the following attributes in a response:

- `id`
- `state`
- `creationDate`

[R16] The `id` **MUST** remain the same value for the life of the Fault Management Job.

6.1.4. Fault Management Job State Machine

Figure 12 presents the Fault Management Job state machine:
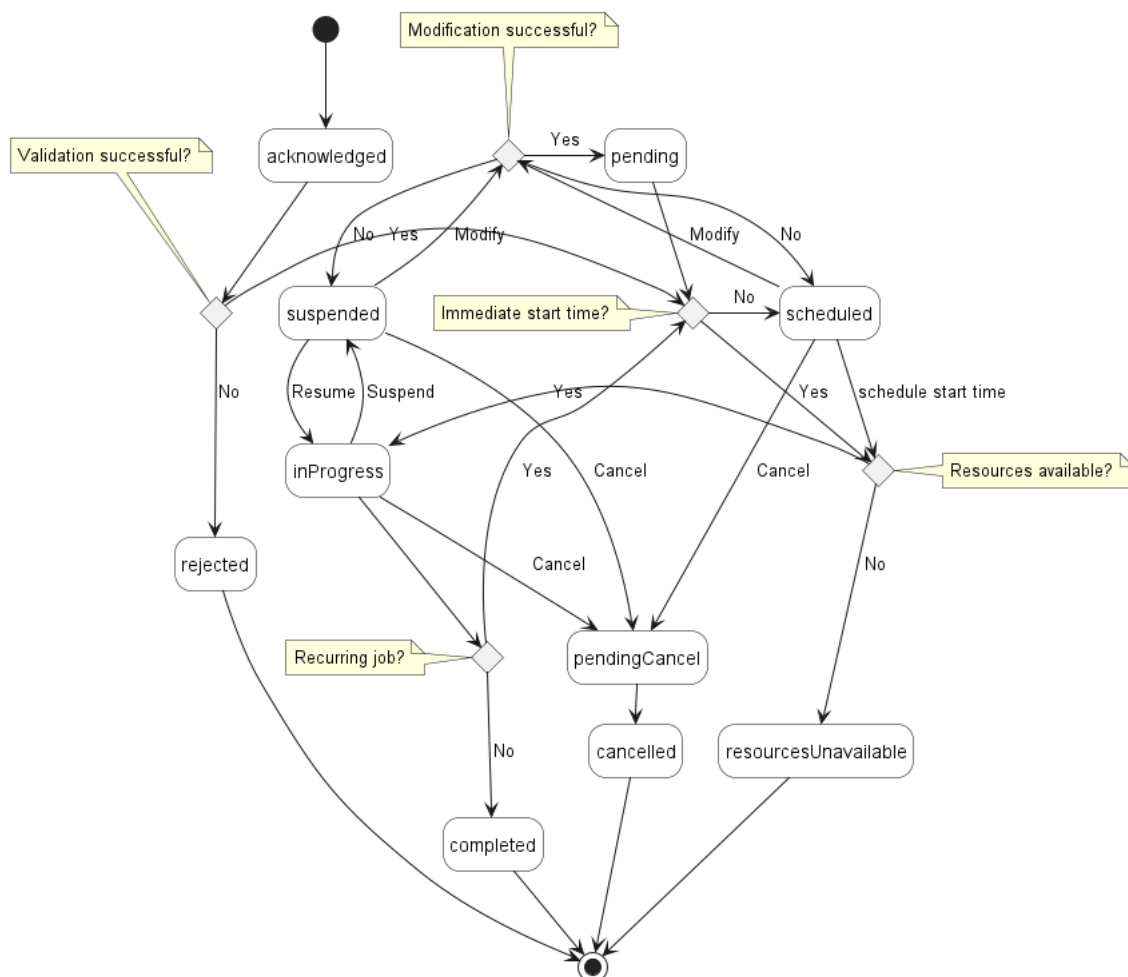
Figure 12. Fault Management Job State Machine

After receiving the request, the Seller/Server (SOF) performs basic checks of the message. If any problem is found an Error response is provided. If the validation passes a response is provided with `FaultManagementJob` in `acknowledged` status. Next, the Seller/Server performs all the remaining business and time-consuming validations. At this point, an Error response cannot be provided anymore, so the job moves to a `rejected` state if some issues are found. The `FaultManagementJob.rejectionReason` acts as a placeholder to provide a detailed description of what caused the problem. `FaultManagementJob` moves to either the `scheduled` or `inProgress` state depending on the assigned schedule. `FaultManagementJob` remains `scheduled` state until the

scheduled start time is reached. `FaultManagementJob` that is starting needs appropriate resources on Seller/Server side. If required resources cannot be assigned, `FaultManagementJob` moves to `resourceUnavailable` state. After completion, the Seller/Server verifies if `FaultManagementJob` is recurring. If yes, `FaultManagementJob` moves to either `scheduled` or `inProgress` state depending on the schedule definition. Otherwise, it moves to a `completed` state. `FaultManagementJob` can be cancelled when in `scheduled`, `inProgress` or `suspended`. When cancellation is successful, `FaultManagementJob` moves to `cancelled` state. Cancellation includes an intermediary `pendingCancel` state. `FaultManagementJob` can be modified only in the `scheduled` or `suspended` state. Modification includes an intermediary `pending` step.

Table 8 presents the mapping between the API `status` names and the MEF W133.1 naming, together with the statuses' description.

state	MEF W133.1 name	Description
`acknowledged`	Acknowledged	A Create Fault Management Job request has been received by the Seller/Server and has passed basic validation. Fault Management Job Identifier is assigned in the Acknowledged state. The request remains in the Acknowledged state until all validations as applicable are completed. If the attributes are validated the Seller/Server determines if the start time is immediate or scheduled. If immediate, the Fault Management Job moves to the In-progress state. Otherwise, the Fault Management Job moves to the Scheduled state. If not all attributes are validated, the request moves to the Rejected state.
`cancelled`	Cancelled	A Fault Management Job that was In-Progress, Suspended, or Scheduled is cancelled.
`completed`	Completed	A non-recurring Fault Management Job finished execution.
`inProgress`	In-Progress	A Fault Management Job is running. Upon completion of the Job, a determination if the Fault Management Job is a one-time Job or is recurring is performed. If the Fault Management Job is a one-time Job, the state of the Fault Management Job moves to the Completed state. If the Fault Management Job is recurring, the Fault Management Job circles back to determine if it has an immediate start time or a scheduled start time. If the FM job has an immediate start time it moves back to In-Progress state, otherwise it moves to Scheduled. In case a Suspend Fault Management Job request is completed, the Job moves to the Suspended state. If a Cancel Fault Management Job request is accepted, the Job moves to the Pending Cancel state.
`pending`	Pending	A Modify Fault Management Job request has been accepted by the Seller/Server. The Fault Management Job remains in the Pending state while updates to the Job are completed. Once updates are complete, the Job returns to the Scheduled or In-Progress status depending on the schedule definition.

state	MEF W133.1 name	Description
pendingCancel	Pending Cancel	A Cancel Fault Management Job request has been accepted by the Seller/Server. The Fault Management Job remains Pending Cancel while resources used by the Job are being released. Once updates are complete, the Job moves to the Cancelled status.
rejected	Rejected	A Create Fault Management Job request fails validation and is rejected with error indications by the Seller/Server.
resourcesUnavailable	Resource Unavailable	A Fault Management Job cannot be allocated with necessary resources when moving to execution (In-Progress state).
scheduled	Scheduled	A Fault Management Job is created that does not have an immediate start time. The Fault Management Job stays in the Scheduled state until the start time is reached. The Fault Management Job then moves to In-Progress. If the Cancel Fault Management Job request is accepted, the Job moves to the Pending Cancel state. If the Modify Fault Management Job request is accepted, the Job moves to the Pending state.
suspended	Suspended	A Suspend Fault Management Job request is completed by the Seller/Server. The Job remains in the Suspended state until a Resume Fault Management Job request is completed by the Seller/Server at which time the Job returns to the In-Progress state. If the Cancel Fault Management Job request is accepted, the Job moves to the Pending Cancel state. If the Modify Fault Management Job request is accepted, the Job moves to the Pending state.

Table 8. Fault Management Job State Machine states

[R17] The Seller/Server **MUST** support all Fault Management Job statuses and their associated transitions as described in Figure 12 and Table 8.

6.2. Use Case 2: Retrieve List of Fault Management Job

The Buyer/Client can retrieve a list of `FaultManagementJob` by using a `GET /faultManagementJob` operation with desired filtering criteria.

[O4] The Buyer/Client Retrieve List of Fault Management Jobs request **MAY** contain none or more of the following attributes as filter criteria:

- serviceIdFrom
- serviceIdTo
- state
- creationDate.gt
- creationDate.lt
- jobType
- granularity

- reportingPeriod
- jobPriority

```
https://serverRoot/mefApi/legato/faultManagement/v1/faultManagementJob?state=suspended&limit=10&offset=0
```

The example above shows a Buyer/Client's request to get all Fault Management Job objects that are in the `suspended` state. Additionally, the Buyer/Client 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 `FaultManagementJob_Find` objects matching the criteria. To get all the details, the Buyer/Client has to query a specific `FaultManagementJob` by its `id`. Details related to pagination are described in section 7.1.2

If the quantity of the records requested to be returned exceeds a Seller/Server policy, the Seller/Server must choose to respond with either:

- An empty list and message that indicates the result set is too large or
- A response that indicates the result is too large and includes a subset of the matching FM Jobs.

[R18] The Seller/Server's response to the Buyer's/Client's Retrieve List of Fault Management Jobs **MUST** include the following attributes as applicable:

- creationDate
- description
- granularity
- id
- jobPriority
- jobType
- outputFormat
- reportingPeriod
- scheduleDefinition
- serviceId
- state

[R19] If the Seller/Server validates the Buyer's/Client's request but finds no matching Fault Management Jobs, the Seller/Server **MUST** return an empty list.

Figure 13 presents entities related to the use case.
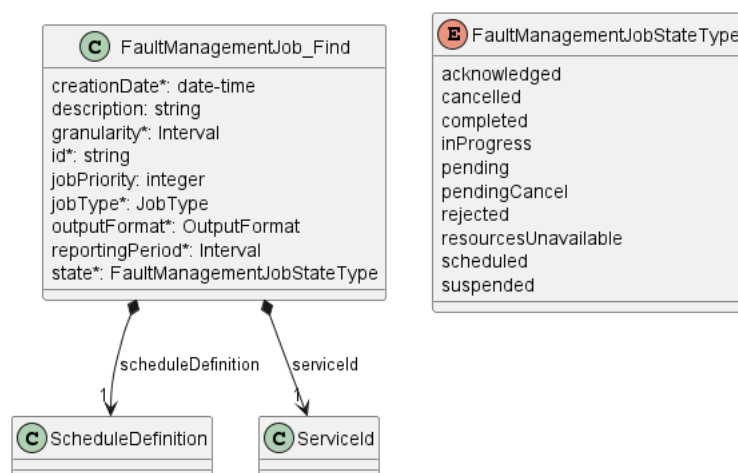
Figure 13. Use Case 2: Retrieve Fault Management Job List - Model

6.3. Use Case 3: Retrieve Fault Management Job by Job Identifier

The Buyer/Client can get detailed information about the Fault Management Job from the Seller/Server by using a `GET /faultManagementJob/{{id}}` operation. The payload returned in the response is a full representation of the Fault Management Job and includes all attributes the Buyer/Client has provided while sending a Fault Management Job create request, together with additional attributes set by Seller/Server.

Get List and Get by Identifier operations return different representations of Fault Management Job. Get List returns the `FaultManagementJob_Find` object which is a subset of `FaultManagementJob` returned by Get by Identifier operation. A response to a Get by ID for a `FaultManagementJob` with `id=755e55e2-72b0-4e3b-af00-693e3beac691` would return exactly the same response as presented in section 6.1.3.

[R20] The Buyer/Client's Retrieve Fault Management Job by Job Identifier request **MUST** contain the Fault Management Job Identifier.

[R21] In case `id` does not allow finding a `FaultManagementJob` in Seller/Server's system, an error response `Error404` **MUST** be returned.

6.4. Use Case 4: Modify Fault Management Job

Due to the need for provisioning and resource reservation on the SOF side, the modification operation associated with the Fault Management Job may exhibit a prolonged duration. Consequently, this operation is implemented through a separate lifecycle process.

6.4.1. Interaction flow

The flow of this use case is shown in Figure 14.
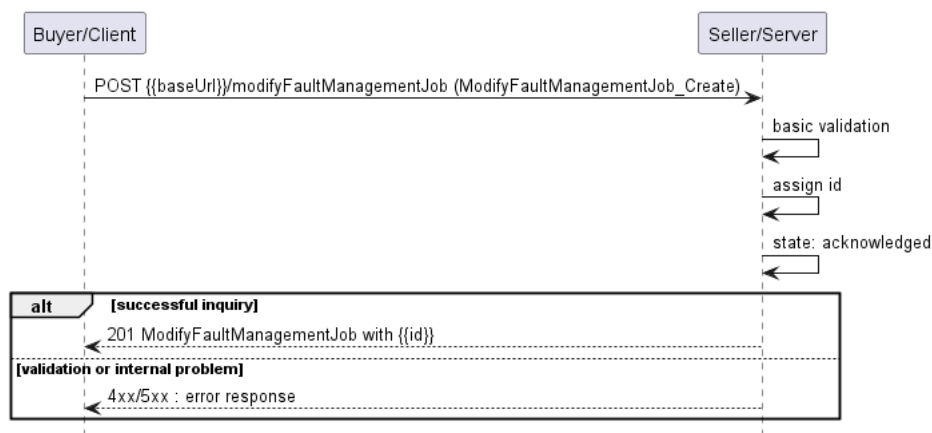
Figure 14. Use Case 4 - Modify Fault Management Job create request flow

The Buyer/Client sends a request with a `ModifyFaultManagementJob_Create` type in the body. The Seller/Server performs request validation, assigns an `id`, and returns the `ModifyFaultManagementJob` type in the response body, with a `state` set to `acknowledged`. Further processing is performed by Seller/Server which will in case of success update the Fault Management Job. The Buyer/Client can track the progress of the process either by subscribing for notifications or by periodically polling the `ModifyFaultManagementJob`. The two patterns are presented in the following diagrams.
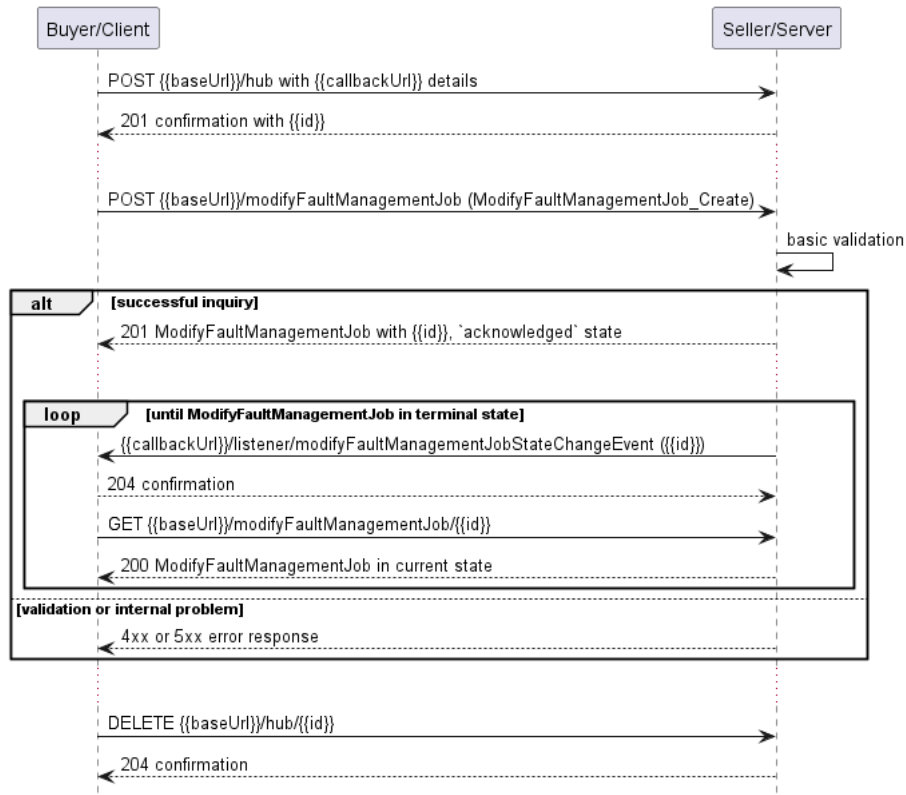
Figure 15. Modify Fault Management Job progress tracking - Notifications
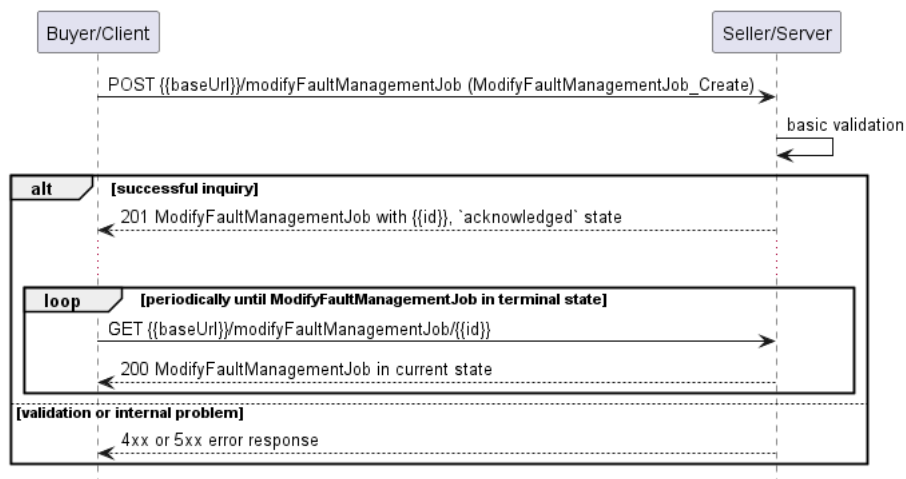
Figure 16. Modify Fault Management Job progress tracking - Polling

Note: The Modify Fault Management Job process is altering the state of the FM job itself. It is important to note that notifications resulting from changes in the state of the Fault Management Job are not represented in Figures 15 and 16.

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.

6.4.2. Modify Fault Management Job Request

Figure 17 presents the most important part of the data model used during the Modify Fault Management Job request (`POST /modifyFaultManagementJob`) and response. The model of the request message - `ModifyFaultManagementJob_Create` is a subset of the `ModifyFaultManagementJob` model and contains only attributes that can (or must) be set by the

Buyer/Client. The Seller/Server (SOF) then enriches the entity in the response with additional information.

Note: `ModifyFaultManagementJob_Create` is an entity used by the Buyer/Client to make a request. `ModifyFaultManagementJob` is an entity used by the Seller/Server to provide a response. The request entity has a subset of attributes of the response entity. Thus for visibility of these shared attributes `ModifyFaultManagementJob_Common` has been introduced (this class is not supposed to be used directly in the exchange).

A `ModifyFaultManagementJob_Create` is a subset that includes only the updateable attributes. The `FaultManagementJobRef` section of `ModifyFaultManagementJob_Create` is used to specify which Fault Management Job object is a subject of the modification process (relationship by reference using `id` of the Job).

Note: Only those attributes of the Fault Management Job that should be modified, should be included in the Modify Fault Management Job Request.

Section `servicePayloadSpecificAttributes` of the Modify Fault Management Job request allows for the introduction of service-specific properties of Fault Management Job as the API payload. The extension mechanism is described in detail in Section 5.3.

The full list of attributes is available in Section 7 and in the API specification which is an integral part of this standard.
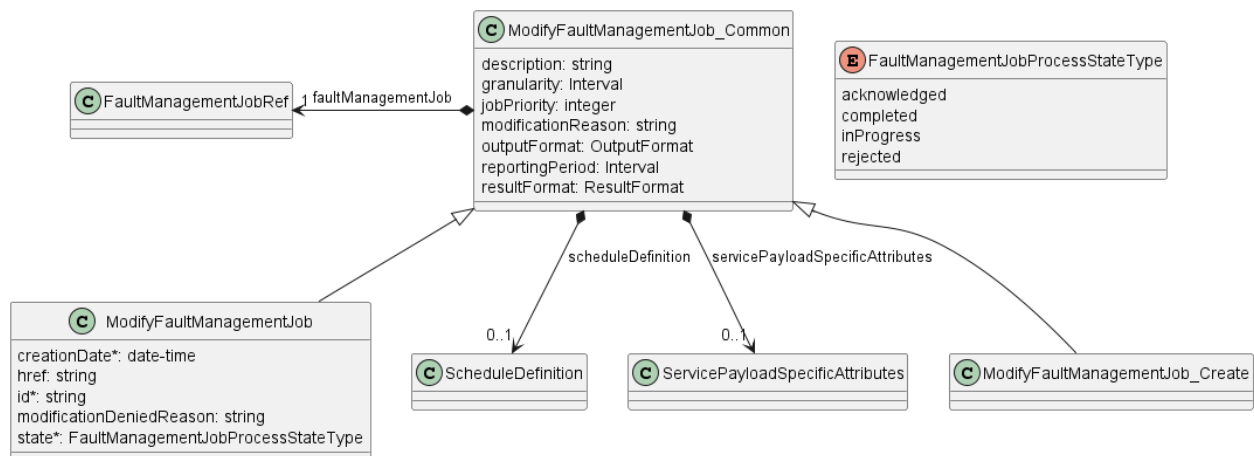
Figure 17. Modify Fault Management Job Key Entities

To send a Modify Fault Management Job request the Buyer/Client uses the `createModifyFaultManagementJob` operation from the API: `POST /modifyFaultManagementJob`. Some of the payload's attributes might be omitted to improve examples' readability.

The example below shows a request to create a modification process for `FaultManagementJob` that was created in section 6.1.2.

The request below aims to:

- change `description` of the Fault Management Job
- modify `granularity`

```
{
  "description": "Fault Management Job after modification",
  "granularity": "1 minute",
  "modificationReason": "Modify Fault Management Job sample",
  "faultManagementJob": {
    "faultManagementJobId": "755e55e2-72b0-4e3b-af00-693e3beac691"
```

```
      }
   }
```

[R22] The Buyer's/Client's Modify FM Job request **MUST** include the FM Job Identifier. [MEF133.1 R6]

[R23] The Buyer's/Client's Modify FM Job request **MUST** contain one or more of the following attributes: [MEF133.1 R7]

- Output Format
- Granularity
- Description
- FM Job Priority
- Reporting Period
- Result Format
- Schedule Definition
- Service Payload Specific Attributes

6.4.3. Modify Fault Management Job Response

Entities used for providing a response to Modify Fault Management Job requests are presented in Figure 17. The Seller/Server responds with a `ModifyFaultManagementJob` type, which adds some attributes (like `id` or `state`) to the `ModifyFaultManagementJob_Create` that was used in the Buyer/Client request.

Note: The term "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/Server response. It has the same structure as in the retrieve by identifier operation.

```
{
  "description": "Fault Management Job after modification",
  "granularity": "1 minute",
  "modificationReason": "Modify Fault Management Job sample",
  "faultManagementJob": {
    "faultManagementJobId": "755e55e2-72b0-4e3b-af00-693e3beac691"
  },
  "creationDate": "2025-01-02T00:00:00.000Z", << added by SOF >>
  "href": "{{baseUrl}}/faultManagement/v1/9c51d971-185d-403e-952f-2110f33a9628", << added by SOF >>
  "id": "9c51d971-185d-403e-952f-2110f33a9628", << added by SOF >>
  "state": "acknowledged" << added by SOF >>
}
```

Attributes that are set by the Seller/Server in the response are marked with the `<< added by SOF >>` tag.

[R24] The Seller's/Server's response to the Buyer's/Client's Modify FM Job request **MUST** echo back the attributes in the Client's request. [MEF133.1 R8]

[R25] The Seller's/Server's response to the Buyer's/Client's Modify FM Job request **MUST** indicate if the request has been accepted or rejected. [MEF133.1 R9]

[R26] The Seller/Server **MUST** specify the following attributes in a response:

- `id`
- `state`
- `creationDate`

[R27] The `id` **MUST** remain the same value for the life of the Modify Fault Management Job.

In case Seller/Server cannot successfully validate the request, Modify Fault Management Job process fails, which results in setting the state to `rejected` with a proper explanation in `modificationDeniedReason`. This includes situation when:

- `id` does not allow to find a `FaultManagementJob` that is to be updated in Seller/Server's system
- requested attributes cannot be modified
- Fault Management Job is in a state that does not allow for modification.

6.4.4. Modify Fault Management Job State Machine

Figure 18 presents the Modify Fault Management Job state machine:
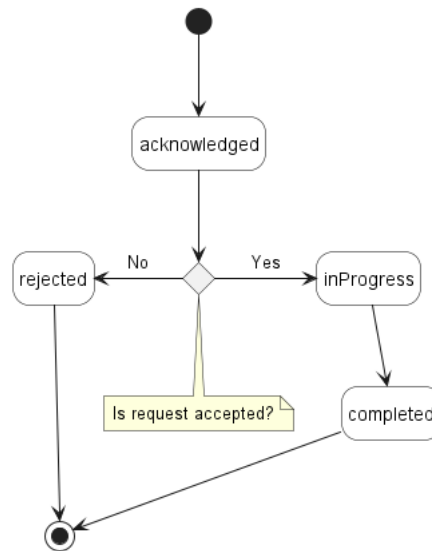
Figure 18. Modify Fault Management Job State Machine

After receiving the request, the Seller/Server (SOF) performs basic checks of the message. If any problem is found an Error response is provided. If the validation passes, a response is provided with `ModifyFaultManagementJob` in `acknowledged` status. Next, the Seller/Server performs all the remaining business and time-consuming validations. At this point, an Error response cannot be provided anymore, so the process moves to a `rejected` state if some issues are found. The `ModifyFaultManagementJob.modificationDeniedReason` acts as a placeholder to provide a detailed description of what caused the problem. If validation is successful, `ModifyFaultManagementJob` moves to the `inProgress` state. At this point, the related `FaultManagementJob` moves to a `pending` state, and the Seller/Server starts all necessary arrangements to provision modification request. `FaultManagementJob` remains in the `pending` state until the Modify Fault Management Job process is finished and moved to the `completed` state. This causes the `FaultManagementJob` state to change to `scheduled` or `inProgress` depending on the `ScheduleDefinition`.

Table 9 presents the mapping between the API `status` names and the MEF W133.1 naming, together with the statuses' description. The list of statuses is the same for all processes related to Fault Management Job (cancel/modify).

state	MEF W 133.1 name	Description

state	MEF W 133.1 name	Description
acknowledged	Acknowledged	The Cancel/Modify Fault Management Job request has been received by the Seller/Server and has passed basic validation. Fault Management Job Process Identifier is assigned in the Acknowledged state. The request remains Acknowledged until all validations as applicable are completed. If the attributes are validated, the request moves to the In-Progress state. If not all attributes are validated, the request moves to the Rejected state.
completed	Completed	The Cancel/Modify Fault Management Job request has been completed by the Seller/Server.
inProgress	In-Progress	The Cancel/Modify Fault Management Job request has been validated and accepted by the Seller/Server and is in-progress.
rejected	Rejected	The Cancel/Modify Fault Management Job request has failed validation and has been declined by the Seller/Server.

Table 9. Fault Management Job Process State Machine states

[R28] The Seller/Server **MUST** support all Modify Fault Management Job statuses and their associated transitions as described in Figure 18 and Table 9.

6.5. Use Case 5: Retrieve Modify Fault Management Job List

The Buyer/Client can retrieve a list of Modify Fault Management Job objects by using a GET /modifyFaultManagementJob operation with desired filtering criteria.

[O5] The Buyer/Client Retrieve List of Modify Fault Management Jobs request **MAY** contain none or more of the following attributes:

- faultManagementJobId
- state
- creationDate.gt
- creationDate.lt

```
https://serverRoot/mefApi/legato/faultManagement/v1/modifyFaultManagementJob?state=acknowledged&limit=10&offset=0
```

The example above shows a Buyer's/Client's request to get all Modify Fault Management Job objects that are in the acknowledged state. Additionally, the Buyer/Client 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 ModifyFaultManagementJob_Find objects matching the criteria. Details related to pagination are described in section 7.1.2.

[R29] The Seller **MUST** include following attributes (if set) in the ModifyFaultManagementJob_Find object in the response:

- creationDate
- id
- faultManagementJobId
- state

[R30] In case no items matching the criteria are found, the Seller/Server **MUST** return a valid response with an empty list.

Figure 19 presents entities related to the use case.
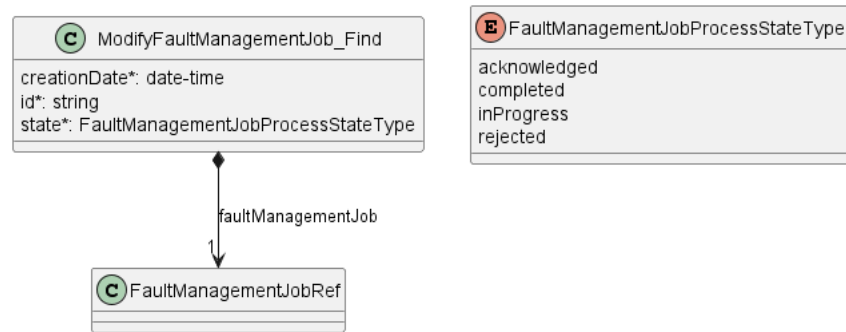
Figure 19. Use Case 5: Retrieve Modify Fault Management Job List - Model

6.6. Use Case 6: Retrieve Modify Fault Management Job by Identifier

The Buyer/Client can get detailed information about the Modify Fault Management Job from the Seller/Server by using a `GET /modifyFaultManagementJob/{{id}}` operation. The payload returned in the response is a full representation of Modify Fault Management Job and includes all attributes the Buyer/Client has provided while sending a Modify Fault Management Job create request, together with additional attributes set by Seller/Server.

Get List and Get by Identifier operations return different representations of Modify Fault Management Job. Get List returns the `ModifyFaultManagementJob_Find` object which is a subset of the `ModifyFaultManagementJob` returned by the Get by Identifier operation. A response to a Get by Identifier for a `ModifyFaultManagementJob` with `id=9c51d971-185d-403e-952f-2110f33a9628` would return exactly the same response as presented in .

[R31] In case `id` does not allow finding a `ModifyFaultManagementJob` in Seller/Server's system, an error response `Error404` **MUST** be returned.

[R32] The Seller/Server **MUST** include following attributes in the `ModifyFaultManagementJob` object in the response:

- `creationDate`
- `id`
- `faultManagementJob`
- `state`

[R33] The Seller **MUST** provide all remaining optional attributes if they were previously set by the Buyer or the Seller.

6.7. Use Case 7: Cancel Fault Management Job

Due to the need for deprovisioning of the Fault Management Job on the SOF side, the cancel operation associated with the Fault Management Job may exhibit a prolonged duration. Consequently, this operation is implemented through a separate lifecycle process.

6.7.1. Interaction flow

The flow of this use case is shown in Figure 20.
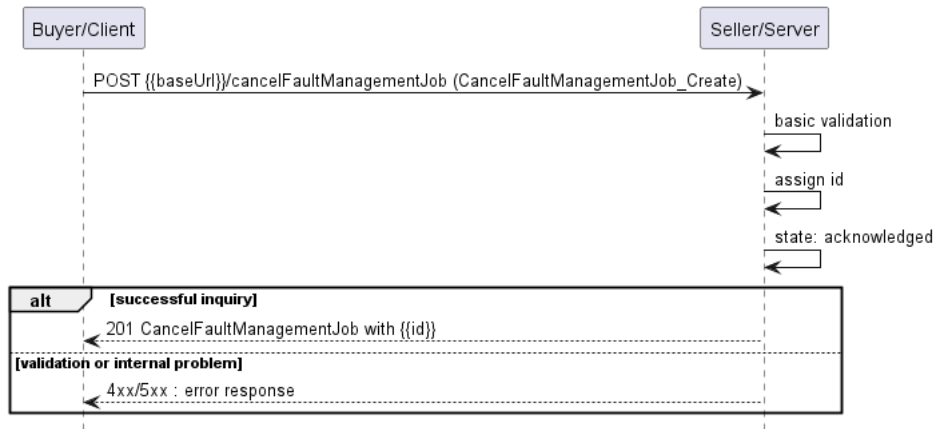
Figure 20. Use Case 7 - Cancel Fault Management Job create request flow

The Buyer/Client sends a request with a `CancelFaultManagementJob_Create` type in the body. The Seller/Server performs request validation, assigns an `id`, and returns the `CancelFaultManagementJob` type in the response body, with a `state` set to `acknowledged`. Further processing is performed by Seller/Server which will in case of success update the Fault Management Job. The Buyer/Client can track the progress of the process either by subscribing for notifications or by periodically polling the `CancelFaultManagementJob`. The two patterns are presented in the following diagrams.
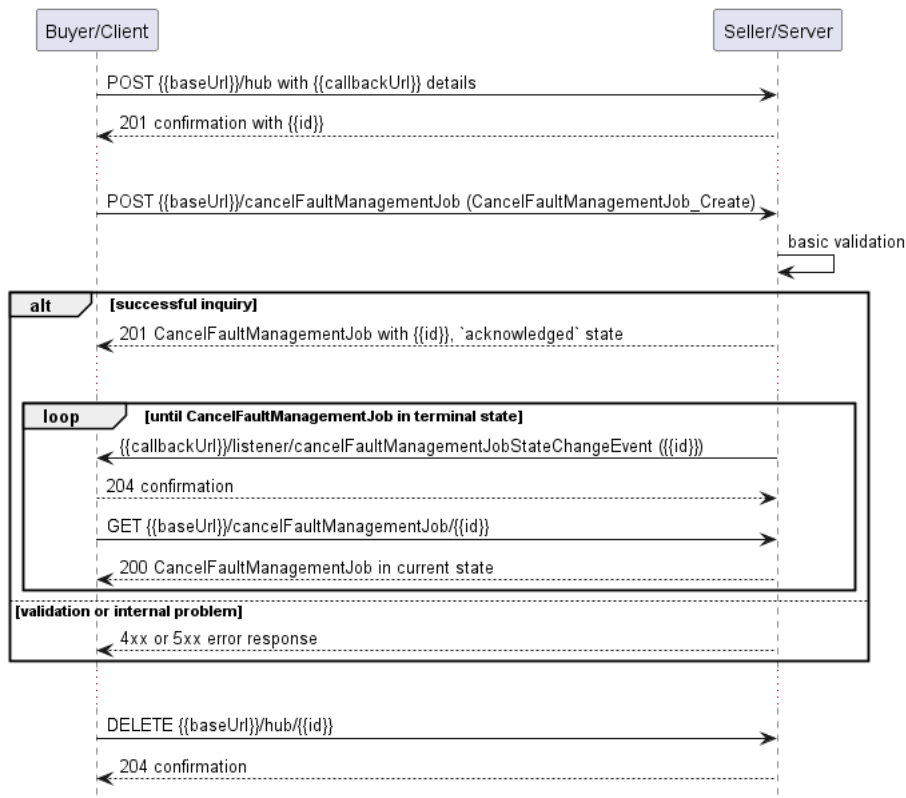
Figure 21. Cancel Fault Management Job progress tracking - Notifications
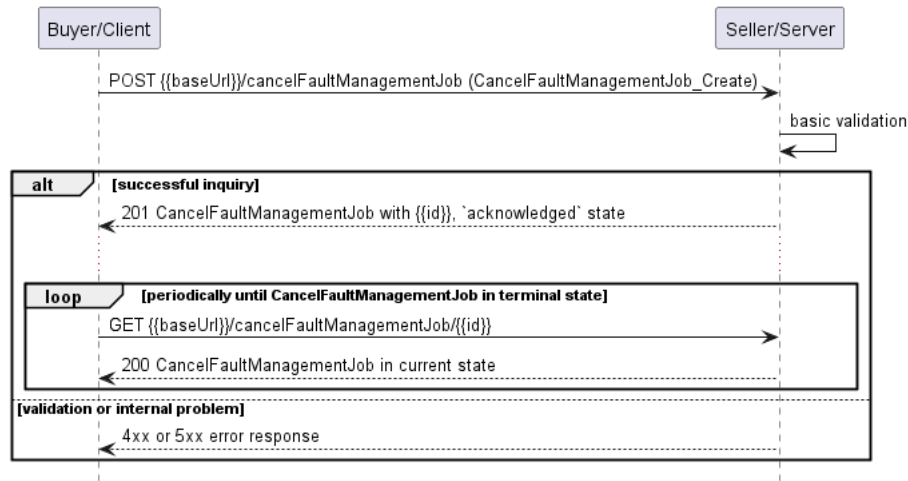
Figure 22. Cancel Fault Management Job progress tracking - Polling

Note: The Cancel Fault Management Job process is altering the state of the job itself. It is important to note that notifications resulting from changes in the state of the Fault Management Job are not represented in Figures 21 and 22.

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.

6.7.2. Cancel Fault Management Job Request

Figure 23 presents the most important part of the data model used during the Cancel Fault Management Job request (`POST /cancelFaultManagementJob`) and response. The model of the request message - `CancelFaultManagementJob_Create` is a subset of the `CancelFaultManagementJob` model and contains only attributes that can (or must) be set by the Buyer/Client. The Seller/Server (SOF) then enriches the entity in the response with additional information.

Note: `CancelFaultManagementJob_Create` is an entity used by the Buyer/Client to make a request. `CancelFaultManagementJob` is an entity used by the Seller/Server to provide a response. The request entity has a subset of attributes of the response entity. Thus for visibility of these shared attributes `CancelFaultManagementJob_Common` has been introduced (this class is not supposed to be used directly in the exchange).

The `FaultManagementJobRef` section of `CancelFaultManagementJob_Create` is used to specify which Fault Management Job object is a subject of the cancellation process (relationship by reference using `id` of the Job).

The full list of attributes is available in Section 7 and in the API specification which is an integral part of this standard.
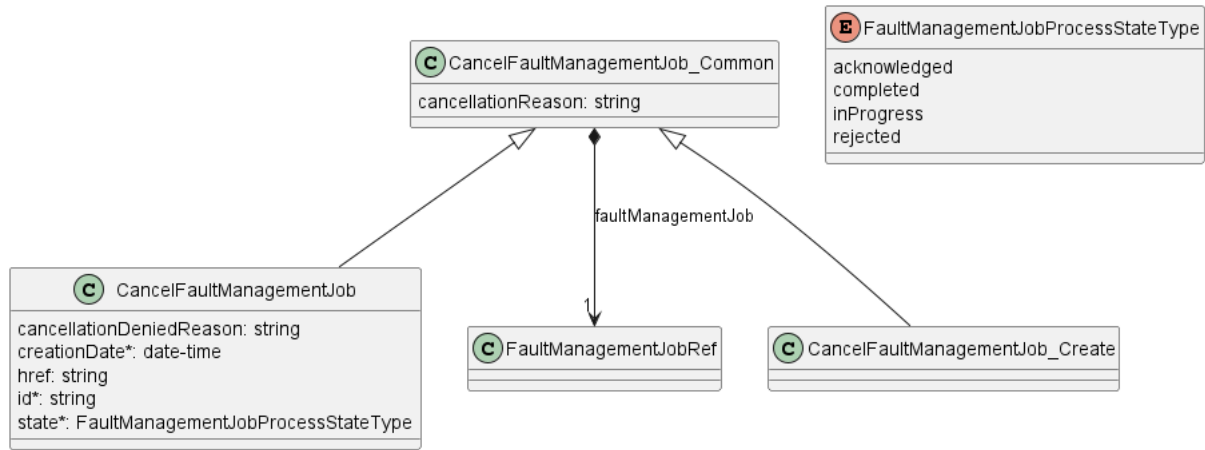
Figure 23. Cancel Fault Management Job Key Entities

To send a Cancel Fault Management Job request the Buyer/Client uses the `createCancelFaultManagementJob` operation from the API: `POST /cancelFaultManagementJob`.

The example below shows a request to create a cancellation process for `FaultManagementJob` that was created in section 6.1.2.

```
{
  "cancellationReason": "Cancel Fault Management Job sample",
  "faultManagementJob": {
    "faultManagementJobId": "755e55e2-72b0-4e3b-af00-693e3beac691"
  }
}
```

[R34] The Buyer's/Client's Cancel FM Job request **MUST** include the FM Job Identifier. [MEF133.1 R10]

6.7.3. Cancel Fault Management Job Response

Entities used for providing a response to Cancel Fault Management Job requests are presented in Figure 23. The Seller/Server responds with a `CancelFaultManagementJob` type, which adds some attributes (like `id` or `state`) to the `CancelFaultManagementJob_Create` that was used in the Buyer/Client request.

Note: The term "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/Server response. It has the same structure as in the retrieve by identifier operation.

```
{
  "cancellationReason": "Cancel Fault Management Job sample",
  "faultManagementJob": {
    "faultManagementJobId": "755e55e2-72b0-4e3b-af00-693e3beac691"
  },
  "creationDate": "2025-01-02T00:00:00.000Z", << added by SOF >>
  "href": "{{baseUrl}}/faultManagement/v1/aea2769a-23f3-4ddc-b095-542a63b12481", << added by SOF >>
  "id": "aea2769a-23f3-4ddc-b095-542a63b12481", << added by SOF >>
  "state": "acknowledged" << added by SOF >>
}
```

Attributes that are set by the Seller/Server in the response are marked with the `<< added by SOF >>` tag.

[R35] The Seller/Server's response **MUST** include all and unchanged attributes' values as provided by the Buyer/Client in the request.

[R36] The Seller/Server **MUST** specify the following attributes in a response:

- `id`
- `state`
- `creationDate`

[R37] The `id` **MUST** remain the same value for the life of the Cancel Fault Management Job.

[R38] The Seller's/Server's response to the Buyer's/Client's Cancel FM Job request **MUST** indicate if the request is Accepted or Declined. [MEF133.1 R11]

[R39] If the Seller/Server accepts the Buyer's/Client's Cancel FM Job request, the Job **MUST** stop. [MEF133.1 R12]

[R40] If the Seller/Server declines the Buyer's/Client's Cancel FM Job request, the Job **MUST NOT** stop. [MEF133.1 R13]

[R41] If the Seller/Server declines the Client's Cancel FM Job request, they **MUST** provide a reason the request was declined. [MEF133.1 R14]

In case Seller/Server cannot successfully validate the request, Cancel Fault Management Job process fails, which results in setting the state to `rejected` with a proper explanation in `cancellationDeniedReason`. This includes situation when:

- `id` does not allow to find a `FaultManagementJob` that is to be cancelled in Seller/Server's system
- Fault Management Job is in a state that does not allow for cancellation.

6.7.4. Cancel Fault Management Job State Machine

Figure 24 presents the Cancel Fault Management Job state machine:
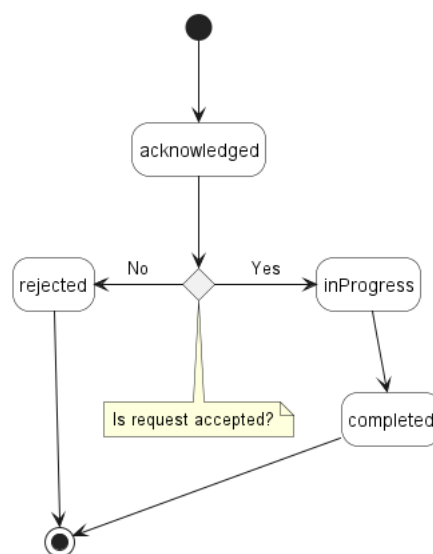
Figure 24. Cancel Fault Management Job State Machine

After receiving the request, the Seller/Server (SOF) performs basic checks of the message. If any problem is found an Error response is provided. If the validation passes a response is provided with `CancelFaultManagementJob` in `acknowledged` status. Next, the Seller/Server performs all the remaining business and time-consuming validations. At this point, an Error response cannot be

provided anymore, so the process moves to a `rejected` state if some issues are found. The `CancelFaultManagementJob.cancellationDeniedReason` acts as a placeholder to provide a detailed description of what caused the problem. If validation is successful, `CancelFaultManagementJob` moves to the `inProgress` state. At this point, the related `FaultManagementJob` moves to a `pendingCancel` state, and the Seller/Server starts all necessary arrangements to provision cancellation request. `FaultManagementJob` remains in the `pendingCancel` state until the Cancel Fault Management Job process is finished and moved to the `completed` state. This causes the `FaultManagementJob` state to change to `cancelled`.

Description and mapping of the Cancel Fault Management Job States are the same as in table 9.

6.8. Use Case 8: Retrieve Cancel Fault Management Job List

The Buyer/Client can retrieve a list of Cancel Fault Management Job objects by using a `GET /cancelFaultManagementJob` operation with desired filtering criteria.

[O6] The Buyer/Client Retrieve List of Cancel Fault Management Jobs request **MAY** contain none or more of the following attributes:

- `faultManagementJobId`
- `state`
- `creationDate.gt`
- `creationDate.lt`

```
https://serverRoot/mefApi/legato/faultManagement/v1/cancelFaultManagementJob?state=acknowledged&limit=10&offset=0
```

The example above shows a Buyer/Client's request to get all Cancel Fault Management Job objects that are in the `acknowledged` state. Additionally, the Buyer/Client 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 `CancelFaultManagementJob_Find` objects matching the criteria. Details related to pagination are described in section 7.1.2.

[R42] The Seller **MUST** include following attributes in the `CancelFaultManagementJob_Find` object in the response:

- `creationDate`
- `id`
- `faultManagementJobId`
- `state`

[R43] In case no items matching the criteria are found, the Seller/Server **MUST** return a valid response with an empty list.

Figure 25 presents entities related to the use case.
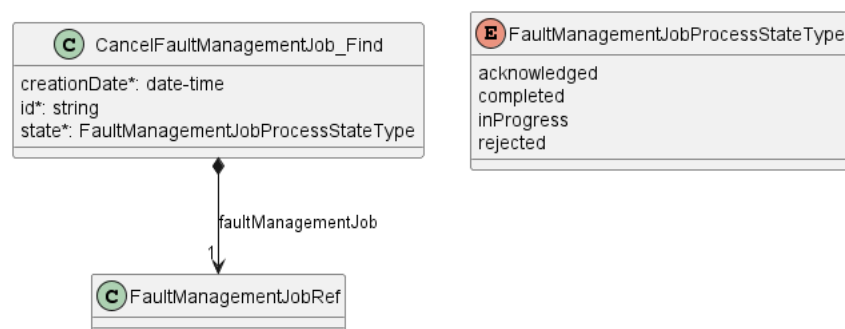
Figure 25. Use Case 8: Retrieve Cancel Fault Management Job List - Model

6.9. Use Case 9: Retrieve Cancel Fault Management Job by Identifier

The Buyer/Client can get detailed information about the Cancel Fault Management Job from the Seller/Server by using a `GET /cancelFaultManagementJob/{{id}}` operation. The payload returned in the response is a full representation of the Cancel Fault Management Job and includes all attributes the Buyer/Client has provided while sending a Cancel Fault Management Job create request, together with additional attributes set by Seller/Server.

Get List and Get by Identifier operations return different representations of Cancel Fault Management Job. Get List returns the `CancelFaultManagementJob_Find` object which is a subset of the `CancelFaultManagementJob` returned by the Get by Identifier operation. A response to a Get by Id for a `CancelFaultManagementJob` with `id=aea2769a-23f3-4ddc-b095-542a63b12481` would return exactly the same response as presented in section 6.7.3.

[R44] In case `id` does not allow finding a `CancelFaultManagementJob` in Seller/Server's system, an error response `Error404` **MUST** be returned.

[R45] The Seller/Server **MUST** include following attributes in the `CancelFaultManagementJob` object in the response:

- `creationDate`
- `id`
- `faultManagementJob`
- `state`

[R46] The Seller **MUST** provide all remaining optional attributes if they were previously set by the Buyer or the Seller.

6.10. Use Case 10: Suspend Fault Management Job

The Buyer/Client may request to suspend a Fault Management Job by using `POST /faultManagementJob/{{id}}/suspend` endpoint. This operation only requires providing the `id` of the Fault Management Job in the path and has an empty `204` confirmation response.

[R47] The Buyer/Client's Suspend FM Job request **MUST** include the FM Job Identifier. [MEF133.1 15]

The sequence diagram in Figure 26 presents this use case in detail.
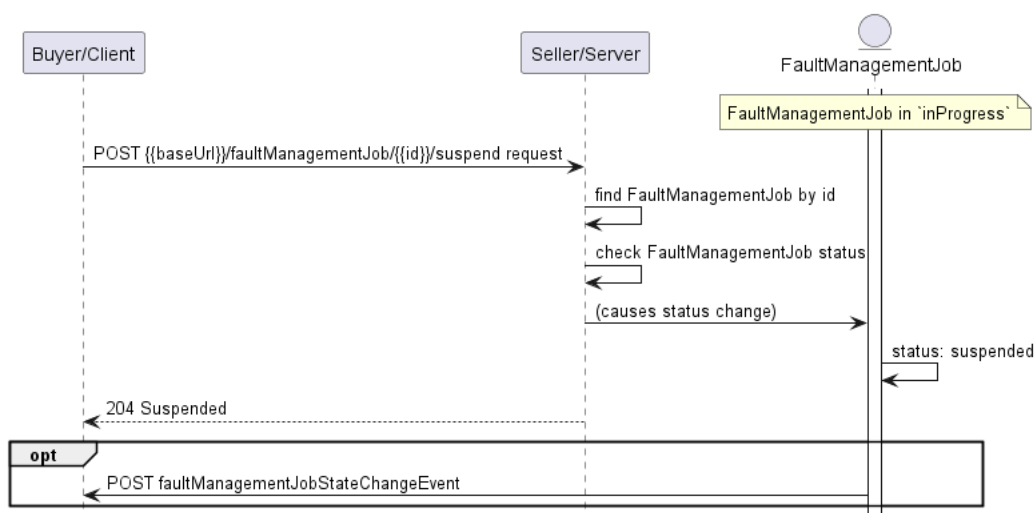
Figure 26. Use Case 10 - Suspend Fault Management Job Flow

The Buyer/Client sends a request specifying `id` of the Fault Management Job to be suspended. The Seller/Server performs request validation, then searches for Fault Management Job. If found, the status is verified (`inProgress`). If everything is verified correctly, the Seller/Server moves the Fault Management Job to the `suspended` status, and sends a successful response to a suspension request optionally followed by `faultManagementJobStateChangeEvent`.

When the Fault Management Job is suspended, it does not generate Fault Management Reports.

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

[R48] In case `id` does not allow to find a `FaultManagementJob` that is to be suspended, an error response `Error404` **MUST** be returned.

[R49] The FM Job **MUST** be in the In-Progress state to be Suspended. [MEF133.1 16]

[R50] The Seller/Server's response to the Client's Suspend FM Job request **MUST** indicate if the request is Accepted or Declined. [MEF133.1 R17]

[R51] If the Seller/Server accepts the Client's Suspend FM Job request, the Job **MUST** be suspended. [MEF133.1 R18]

[R52] If the Seller/Server declines the Buyer/Client's suspend FM Job request, the Job **MUST NOT** be suspended. [MEF133.1 R19]

[R53] If the Seller/Server declines the Buyer/Client's suspend FM Job request, they **MUST** provide a reason why the request was declined. [MEF133.1 R20]

6.11. Use Case 11: Resume Fault Management Job

The Buyer/Client may request to resume a Fault Management Job by using `POST /faultManagementJob/{{id}}/resume` endpoint. This operation only requires providing the `id` of the Fault Management Job in the path and has an empty `204` confirmation response.

[R54] The Client's Resume FM Job request **MUST** include the Job Identifier. [MEF133.1 R21]

The sequence diagram in Figure 27 presents this use case in detail.
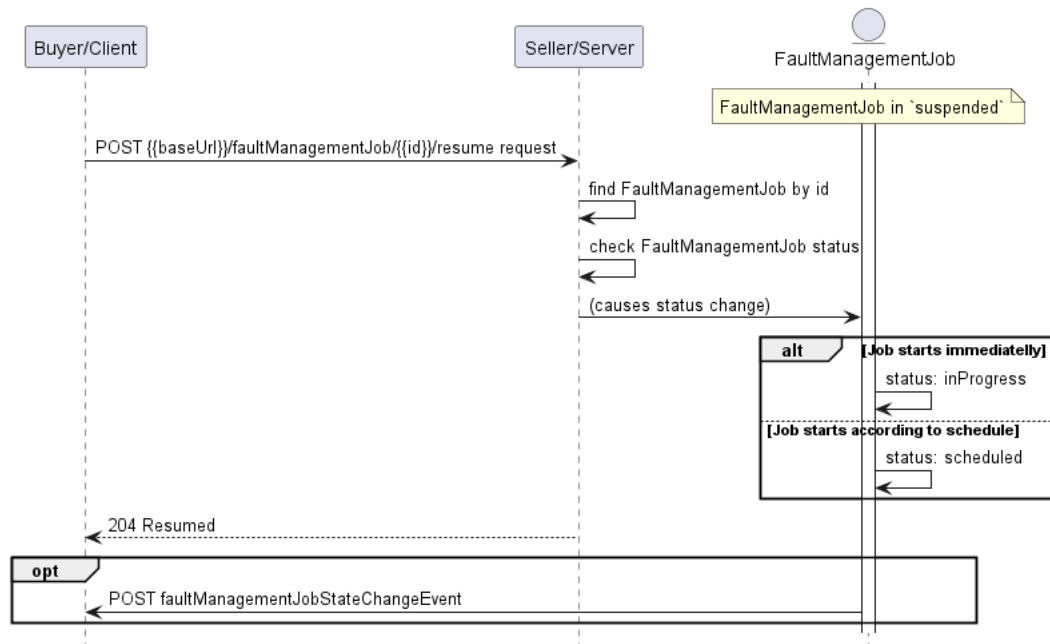
Figure 27. Use Case 11 - Resume Fault Management Job Flow

The Buyer/Client sends a request specifying `id` of the Fault Management Job to be resumed. The Seller/Server performs request validation, then searches for Fault Management Job. If found, the status is verified (`suspended`). If everything is verified correctly, the Seller/Server moves the Fault Management Job to `scheduled` or `inProgress` status depending on the schedule, and sends a successful response to a resumption request optionally followed by `faultManagementJobStateChangeEvent`.

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

[R55] In case `id` does not allow to find a `FaultManagementJob` that is to be resumed, an error response `Error404` **MUST** be returned.

[R56] The FM Job **MUST** be in the Suspended state. [MEF133.1 R22]

[R57] The Seller/Server's response to the Buyer/Client's Resume FM Job request **MUST** indicate if the request is Accepted or Declined. [MEF133.1 R23]

[R58] If the Seller/Server accepts the Client's Resume FM Job request, the Job **MUST** be resumed and return to the In-Progress or Scheduled state. [MEF133.1 R24]

[R59] If the Seller/Server declines the Buyer/Client's Resume FM Job request, the FM Job **MUST NOT** be resumed. [MEF133.1 R25]

[R60] If the Seller/Server declines the Buyer/Client's Resume FM Job request, they **MUST** provide a reason the request was declined. [MEF133.1 26]

6.12. Use Case 12: Create Fault Management Job Complex Query

The `FaultManagementJob` defines complex structures with multiple levels of nesting, such as `scheduleDefinition`. To facilitate filtering based on these structures, the API provides an additional endpoint `POST /faultManagementJobComplexQuery`. This endpoint allows filtering by values defined by the `FaultManagementJob` and returns a list of `FaultManagementJob` objects that match the specified filters.

6.12.1. Create Fault Management Job Complex Query Request

Figure 28 depicts the key components of the data model utilized in the Create Fault Management Job Complex Query request (`POST /faultManagementJobComplexQuery`) and its corresponding response. The request message model, `FaultManagementJobComplexQuery_Create`, is a subset of the `FaultManagementJobComplexQuery` model and includes only attributes that can be specified by the Buyer/Client, representing filtering options. In response, the Seller/Server provides a list of `FaultManagementJobComplexQuery` entities that contain the matched FM Job objects.

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

Figure 28. Fault Management Job Complex Query Key Entities

To send a request the Buyer/Client uses the `createFaultManagementJobComplexQuery` operation from the API. The snippet below presents an example of a Create Fault Management Job Complex Query request. It filters for `FaultManagementJob` objects that:

- have granularity set to 1 minute
- run on a schedule with the recurring frequency set to every full hour
- are in a `scheduled` state

`Fault Management Job Complex Query` Create Request

```
{
  "granularity": "1 minute",
  "scheduleDefinition": {
    "recurringSchedule": {
```

```
      "second": "0",
      "minute": "0",
      "hour": "*/1",
      "dayOfMonth": "*",
      "month": "*",
      "dayOfWeek": "*"
    }
  },
  "state": "scheduled"
}
```

6.12.2. Create Fault Management Job Complex Query Response

Entities used for providing a response to Create Fault Management Job Complex Query request are presented in Figure 28. The Seller/Server responds with a list of `FaultManagementJobComplexQuery` objects, which represent matched Fault Management Jobs.

Note: The term "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/Server response.

Fault Management Job Complex Query Create Response

```
[
  {
    "creationDate": "2025-01-01T00:00:00.000Z",
    "description": "Exemplary Create Fault Management Job request",
    "faultManagementJob": {
      "faultManagementJobId": "755e55e2-72b0-4e3b-af00-693e3beac691"
    },
    "granularity": "1 minute",
    "jobPriority": 5,
    "jobType": "proactive",
    "lastModifiedDate": "2025-01-01T00:00:00.000Z",
    "outputFormat": "json",
    "reportingPeriod": "1 hour",
    "resultFormat": "payload",
    "scheduleDefinition": {
      "scheduleDefinitionStartTime": "2025-01-01T00:00:00.000Z",
      "scheduleDefinitionEndTime": "2026-01-01T00:00:00.000Z",
      "recurringSchedule": {
        "second": "0",
        "minute": "0",
        "hour": "*/1",
        "dayOfMonth": "*",
        "month": "*",
        "dayOfWeek": "*"
      },
      "executionDuration": "1 hour"
    },
    "serviceId": {
      "serviceIdFrom": "905d9f87-6478-4153-a5de-fcc70257f03c",
      "serviceIdTo": "1d7dd934-7aa3-4997-9b6a-906a3e35a08e"
    },
    "servicePayloadSpecificAttributes": {
      "@type": "urn:mef:lso:spec:legato:ip-fault-management-configuration:v0.0.1:all"
    },
    "state": "scheduled"
  }
]
```

6.13. Use Case 13: Retrieve Fault Management Report List

The Buyer/Client can retrieve a list of `FaultManagementReport` objects by using a `GET /faultManagementReport` operation with desired filtering criteria.

[O7] The Buyer's/Client's Retrieve List of Fault Management Reports request **MAY** contain none or more of the following attributes as filter criteria:

- `faultManagementJobId`
- `serviceIdFrom`
- `serviceIdTo`
- `state`
- `creationDate.gt`
- `creationDate.lt`
- `reportingTimeframe.startDate.gt`
- `reportingTimeframe.startDate.lt`
- `reportingTimeframe.endDate.gt`
- `reportingTimeframe.endDate.lt`
- `granularity`
- `jobType`
- `outputFormat`
- `resultFormat`

```
https://serverRoot/mefApi/legato/faultManagement/v1/faultManagementReport?state=completed&limit=10&offset=0
```

The example above shows a Buyer/Client's request to get all Fault Management Report objects that are in the `completed` state. Additionally, the Buyer/Client asks only for a first (`offset=0`) pack of 10 results (`limit=10`) to be returned. Details related to pagination are described in section 7.1.2 The correct response (HTTP code `200`) in the response body contains a list of `FaultManagementReport_Find` objects matching the criteria. `FaultManagementReport_Find` object is a subset of all Fault Management Report attributes. In particular, it does not contain the collected measurements. To get all details, the Buyer/Client has to query a specific `FaultManagementReport` by its `id`.

[R61] The Seller/Server **MUST** support the retrieval of a List of Fault Management Reports Use Case. [MEF133.1 R33]

[R62] The Buyer/Client **MUST** support the retrieval of a List of Fault Management Reports Use Case. [MEF133.1 R34]

[R63] The Seller/Server's response to the Buyer's/Client's retrieve List of Fault Management Reports **MUST** include the following attributes as applicable: [MEF133.1 R35]

- Description
- Report ID

[R64] In case no items matching the criteria are found, the Seller/Server **MUST** return a valid response with an empty list.

Figure 29 presents entities related to the use case.
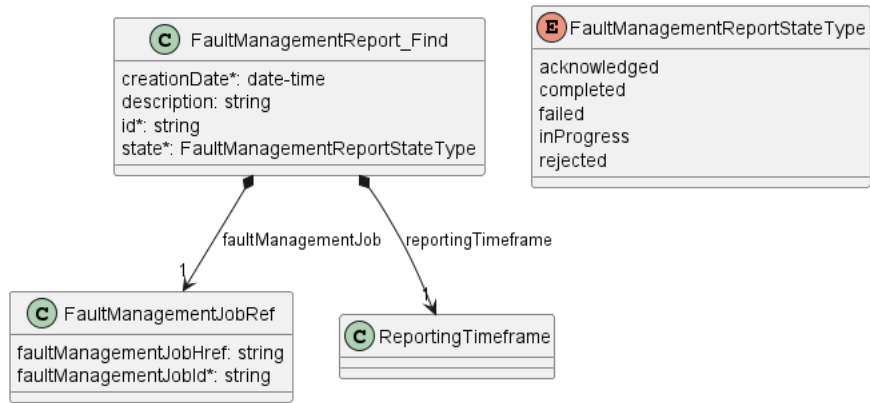
Figure 29. Use Case 13: Retrieve Fault Management Report List - Model

6.13.1. Fault Management Report State Machine

Figure 30 presents the Fault Management Report state machine:
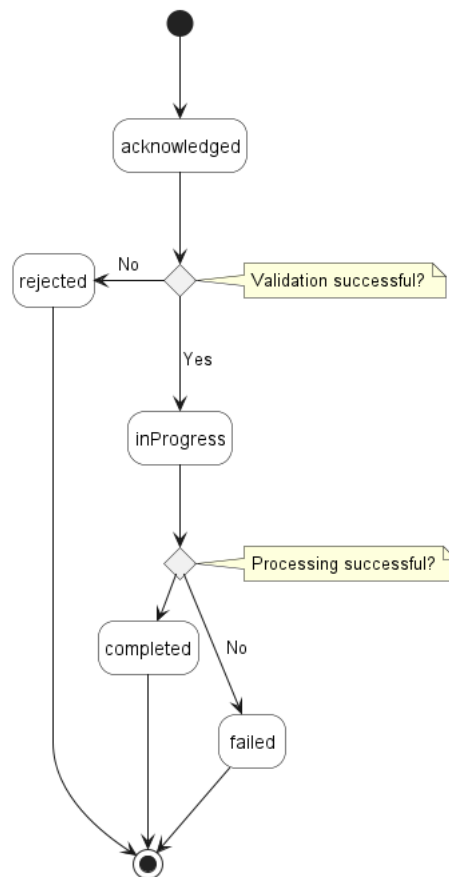
Figure 30. Fault Management Report State Machine

`FaultManagementReport` is created in `acknowledged` status. In case any issues are found in report validation, `FaultManagementReport` moves to a `rejected` state. `FaultManagementReport` moves to `inProgress` state during which report content is collected into the report. Depending on the outcome of the processing, `FaultManagementReport` moves to `completed` or `failed` state. The `FaultManagementReport.failureReason` acts as a placeholder to provide a detailed description of what caused the problem.

Table 10 presents the list of `status` names and their descriptions.

State	Description

State	Description
acknowledged	A Fault Management Report is created by the Seller/Server. Fault Management Report Identifier is assigned in the Acknowledged state. The report remains Acknowledged until all validations as applicable are completed. If the attributes are validated, the Fault Management Report moves to the In-Progress state. If not all attributes are validated, the report moves to the Rejected state.
completed	A Fault Management Report is completed and results are available.
failed	A Fault Management Report processing has failed.
inProgress	A Fault Management Report processing has started.
rejected	This state indicates that the request failed to meet validation rules for `FaultManagementReport` delivery (processing).

Table 10. Fault Management Report State Machine states

[R65] The Seller/Server **MUST** support all Fault Management Report statuses and their associated transitions as described in Figure 30 and Table 10.

6.14. Use Case 14: Retrieve Fault Management Report by Report Identifier

The Buyer/Client can get detailed information about a Fault Management Report from the Seller/Server by using a `GET /faultManagementReport/{{id}}` operation. The response payload provides a comprehensive representation of the Fault Management Report and encompass all attributes, including the results of fault indicators collection. Fault Management Report also contains
a reference to the Fault Management Job that generated the report.

Get List and Get by Identifier operations return different representations of Fault Management Report. Get List returns the `FaultManagementReport_Find` object which is a subset of `FaultManagementReport` returned by the Get by Identifier operation. Specifically, the object returned by the Get by Identifier operation contains a collection of fault indicators, either in the form of attached file or directly in the payload of returned `FaultManagementReport` object. FM Job results are not returned by the Get List operation. A response to Get by Identifier request is presented in the following snippet.

```
{
  "creationDate": "2025-01-01T01:00:00.000Z",
  "description": "Report generated by FM Job",
  "faultManagementJob": {
    "faultManagementJobId": "755e55e2-72b0-4e3b-af00-693e3beac691"
  },
  "id": "c7ad584f-506a-4db1-bbeb-acbd6672f0a7",
  "lastModifiedDate": "2025-01-01T01:00:00.000Z",
  "reportContent": [
    {
      "measurementTime": {
        "measurementStartDate": "2025-01-01T00:00:00.000Z",
        "measurementEndDate": "2025-01-01T01:00:00.000Z"
      },
      "measurementDataPoints": [
        {
          "@type": "urn:mef:lso:spec:legato:ip-fault-management-results:v0.0.1:all"
        }
      ]
    }
  ],
  "reportingTimeframe": {
    "reportingStartDate": "2025-01-01T00:00:00.000Z",
    "reportingEndDate": "2025-01-01T01:00:00.000Z"
  },
```

```
    "state": "completed"
  }
```

[R66] The Seller/Server **MUST** include following attributes in the `FaultManagementReport` object in the response:

- `creationDate`
- `id`

[R67] The Seller/Server **MUST** provide all remaining optional attributes if they were previously set by the Buyer or the Seller.

[R68] The results regardless of the format **MUST** contain the FM results as specified with FM Job request. [MEF133.1 R36]

[R69] In case `id` does not allow finding a `FaultManagementReport` in Seller/Server's system, an error response `Error404` **MUST** be returned.

6.15. Use Case 15: Create Fault Management Report Complex Query

The `FaultManagementReport` defines complex structures with multiple levels of nesting, such as `servicePayloadSpecificAttributes`. To facilitate filtering based on these structures, the API provides an additional endpoint `POST /faultManagementReportComplexQuery`. This endpoint allows filtering by values defined by the `FaultManagementReport` and `FaultManagementJob` types and returns a list of Fault Management Report objects that match the specified filters.

6.15.1. Create Fault Management Report Complex Query Request

Figure 31 depicts the key components of the data model utilized in the Create Fault Management Report Complex Query request (`POST /faultManagementReportComplexQuery`) and its corresponding response. The request message model, `FaultManagementReportComplexQuery_Create`, is a subset of the `FaultManagementReportComplexQuery` model and includes only attributes that can specified by the Buyer/Client, representing filtering options. In response, the Seller/Server provides a list of `FaultManagementReportComplexQuery` entities that contain the matched Fault Management Report objects.

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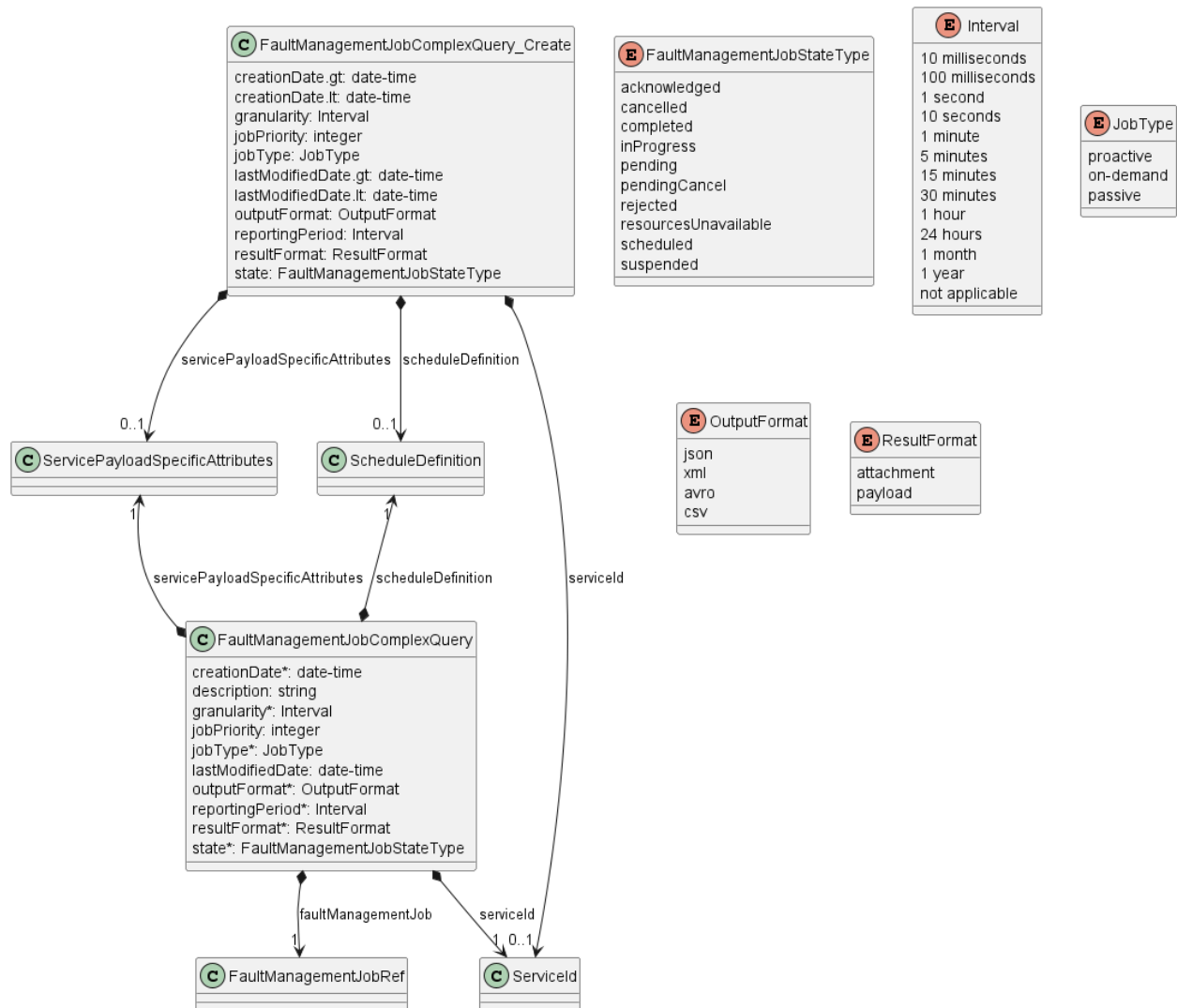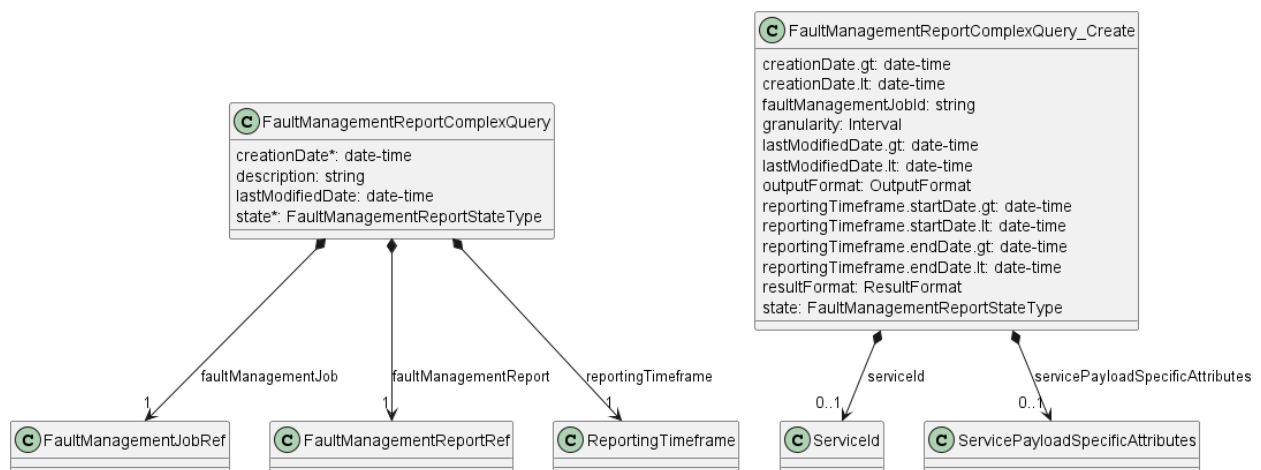

Figure 31. Fault Management Report Complex Query Key Entities

To send a request the Buyer/Client uses the `createFaultManagementReportComplexQuery` operation from the API. The snippet below presents an example of a Create Fault Management Report Complex Query request. It filters for Fault Management Report objects that:

- were created between 2025-01-01 00:00:00 and 2025-01-02 00:00:00
- Output format is JSON
- relate to specified Service Id From

Fault Management Report Complex Query Create Request

```
{
  "creationDate.gt": "2025-01-01T00:00:00.000Z",
  "creationDate.lt": "2025-01-02T00:00:00.000Z",
  "outputFormat": "json",
  "serviceId": {
    "serviceIdFrom": "905d9f87-6478-4153-a5de-fcc70257f03c"
  },
  "state": "completed"
}
```

6.15.2. Create Fault Management Report Complex Query Response

Entities used for providing a response to Create Fault Management Report Complex Query requests are presented in Figure 31. The Seller/Server responds with a list of `FaultManagementReportComplexQuery` objects, which represent matched Fault Management Reports.

Note: The term "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/Server response.

Fault Management Report Complex Query Create Response

```
[
  {
    "creationDate": "2025-01-01T01:00:00.000Z",
    "description": "Report generated by FM Job",
    "faultManagementJob": {
      "faultManagementJobId": "755e55e2-72b0-4e3b-af00-693e3beac691"
    },
    "faultManagementReport": {
      "faultManagementReportId": "c7ad584f-506a-4db1-bbeb-acbd6672f0a7"
    },
    "lastModifiedDate": "2025-01-01T01:00:00.000Z",
    "reportingTimeframe": {
      "reportingStartDate": "2025-01-01T00:00:00.000Z",
      "reportingEndDate": "2025-01-01T01:00:00.000Z"
    },
    "state": "completed"
  }
]
```

6.16. Use Case 16: Retrieve Tracking Record List

Tracking Records allow the tracking of actions performed on main entities described in this document:

- Fault Management Job
- Fault Management Report

Tracking Records store information regarding the timing and nature of actions performed on a specific object. The association with Fault Management entities can be established through the `relatedObjectId` attribute of the `TrackingRecord` type.

The Buyer/Client can retrieve a list of `TrackingRecord` by using a `GET /trackingRecord` operation with desired filtering criteria.

[O8] The Buyer/Client Retrieve List of Tracking Record request **MAY** contain none or more of the following attributes:

- `relatedObjectId`
- `creationDate.gt`
- `creationDate.lt`
- `user`

```
https://serverRoot/mefApi/legato/faultManagement/v1/trackingRecord?relatedObjectId=755e55e2-72b0-4e3b-af00-
693e3beac691&limit=10&offset=0
```

The example above shows a Buyer/Client's request to get all Tracking Record objects that are related to the object with `id=755e55e2-72b0-4e3b-af00-693e3beac691`. Additionally, the Buyer/Client 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 `TrackingRecord_Find` objects matching the criteria. To get all the details, the Buyer/Client has to query a specific `TrackingRecord` by its `id`. Details related to pagination are described in section 7.1.2

[R70] The Seller/Server **MUST** include following attributes (if set) in the `TrackingRecord_Find` object in the response:

- `creationDate`
- `id`
- `relatedObjectId`

[R71] Optionally The Seller/Server **MAY** return :

- `description`
- `user`

[R72] In case no items matching the criteria are found, the Seller/Server **MUST** return a valid response with an empty list.

Figure 32 presents the main Tracking Record entities.
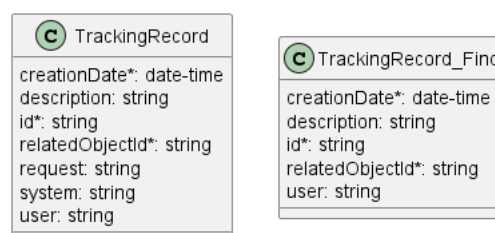
Figure 32. Tracking Record Model

6.17. Use Case 17: Retrieve Tracking Record by Identifier

The Buyer/Client can get detailed information about the Tracking Record from the Seller/Server by using a `GET /trackingRecord/{{id}}` operation. The payload returned in the response is a full

representation of the Tracking Record.

Get List and Get by Identifier operations return different representations of Tracking Record. Get List returns the `TrackingRecord_Find` object which is a subset of `TrackingRecord` returned by the Get by Identifier operation.

[R73] In case `id` does not allow finding a `TrackingRecord` in Seller/Server's system, an error response `Error404` **MUST** be returned.

[R74] The Seller/Server **MUST** include following attributes in the `TrackingRecord` object in the response:

- `creationDate`
- `id`
- `relatedObjectId`

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

6.18. Use Case 18: Register for Notifications

The Buyer/Client can track the lifecycle of the Fault Management objects by subscribing to notifications. An exemplary use case for exchanging notifications is presented in Figure 33.
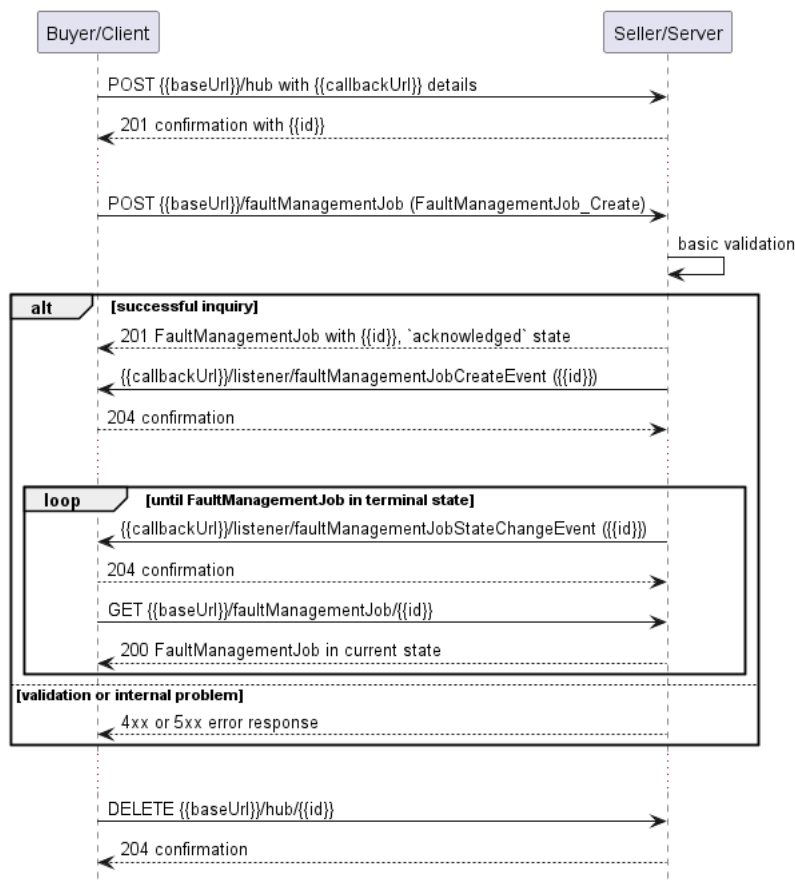
Figure 33. Fault Management Notification Example

The Seller/Server communicates with the Buyer/Client with Notifications provided that:

- Buyer/Client supports a notification mechanism
- Buyer/Client has registered to receive notifications from the Seller/Server

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

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

Figure 34 shows all entities involved in the Notification use cases.
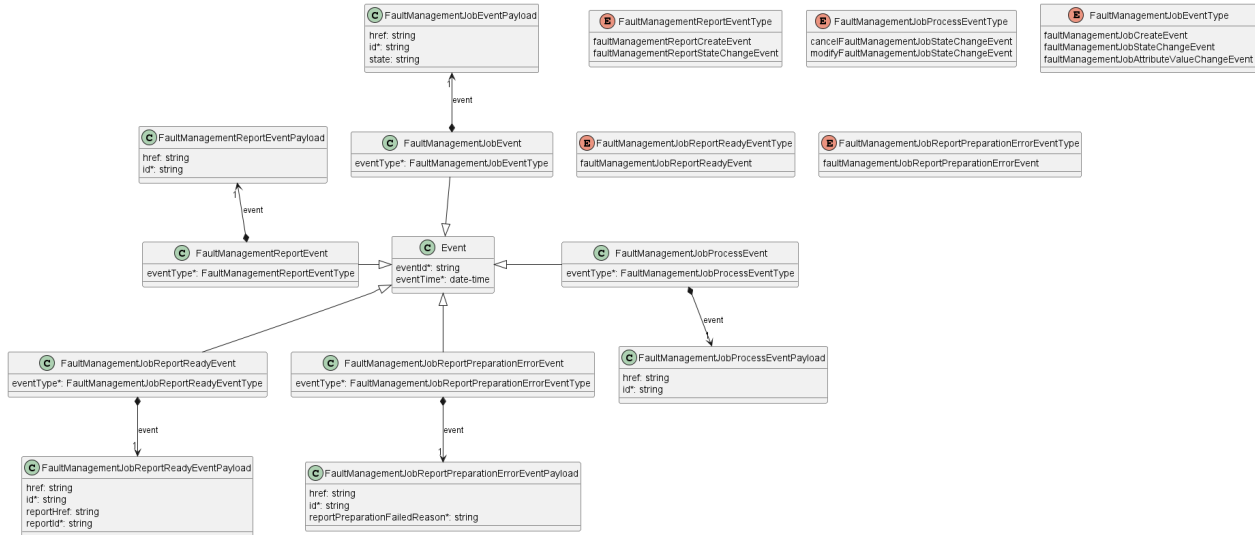
Figure 34. Fault Management Notification Data Model

By using a request in the following snippet, the Buyer/Client subscribes for notification of all types of events. Those are:

- `faultManagementJobCreateEvent`
- `faultManagementJobStateChangeEvent`
- `faultManagementJobAttributeValueChangeEvent`
- `faultManagementJobReportReadyEvent`
- `faultManagementJobReportPreparationErrorEvent`
- `cancelFaultManagementJobStateChangeEvent`
- `modifyFaultManagementJobStateChangeEvent`
- `faultManagementReportCreateEvent`
- `faultManagementReportStateChangeEvent`

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

[R75] The Client request **MUST** contain the following: [MEF133.1 R27]

- FM Job Notification Target Information
- List of Job Notification Types

[R76] The Seller/Server **MUST** respond to the Client's Register for FM Job Notifications request to indicate that the request was accepted or rejected. [MEF133.1 R28]

[R77] If the Seller/Server rejects the Client's Register for FM Job Notifications request, the response **MUST** include a reason for the rejection. [MEF133.1 R29]

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

```
{
  "callback": "https://bus.com/listenerEndpoint",
  "query": "eventType=faultManagementJobStateChangeEvent"
}
```

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

```
eventType=faultManagementJobStateChangeEvent,faultManagementJobReportReadyEvent
```

or

```
eventType=faultManagementJobStateChangeEvent&eventType=faultManagementJobReportReadyEvent
```

The query formatting complies with 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.

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

```
{
  "id": "ae34ccfd-14ad-43e4-9063-f99fe5bc8c4b",
  "callback": "https://bus.com/listenerEndpoint",
  "query": "eventType=faultManagementJobStateChangeEvent"
}
```

Example of a final address that the Notifications will be sent to (for faultManagementJobStateChangeEvent):

https://bus.com/listenerEndpoint/mefApi/legato/faultNotification/v1/listener/faultManagementJobStateChangeEvent

6.19. Use Case 19: Send Notification

Notifications are used to asynchronously inform the Buyer/Client about the respective objects and attributes changes.

Figure 35 presents notifications produced by Seller/Server for the whole lifecycle of FaultManagementJob assuming that Buyer/Client subscribed to all event types.

```
           ┌─────────────┐                                              ┌─────┐
           │ Buyer/Client │                                              │ SOF │
           └─────────────┘                                              └─────┘
                 │                                                          │
                 │  POST {{baseUrl}}/hub with {{callbackUrl}}               │
                 │─────────────────────────────────────────────────────────▶│
                 │                                                          │
                 │  201 confirmation with {{id}}                            │
                 │◀- - - - - - - - - - - - - - - - - - - - - - - - - - - - -│
                 │                                                          │
                 │                                                          │
                 │  POST {{baseUrl}}/faultManagementJob                     │
                 │─────────────────────────────────────────────────────────▶│
                 │                                                          │ basic validation
                 │                                                          │◀─┐
                 │                                                          │  │
                 │                                                          │ assign id
                 │                                                          │◀─┐
```

alt [successful inquiry]

201 FaultManagementJob with {{id}} in `acknowledged` state

{{callbackUrl}}/listener/faultManagementJobCreateEvent {{id}}

204 confirmation

> FaultManagementJob moves to `scheduled`

{{callbackUrl}}/listener/faultManagementJobStateChangeEvent {{id}}, state=`scheduled`

204 confirmation

> Schedule start time is reached
> FaultManagementJob moves to `inProgress`

{{callbackUrl}}/listener/faultManagementJobStateChangeEvent {{id}}, state=`inProgress`

204 confirmation

> FaultManagementReport is generated

{{callbackUrl}}/listener/faultManagementJobReportReadyEvent {{id}}, {{reportId}}

204 confirmation

> FaultManagementJob is suspended

{{callbackUrl}}/listener/faultManagementJobStateChangeEvent {{id}}, state=`suspended`

204 confirmation

> ModifyFaultManagementJob request is accepted

{{callbackUrl}}/listener/faultManagementJobStateChangeEvent {{id}}, state=`pending`

204 confirmation

> ModifyFaultManagementJob request is completed

{{callbackUrl}}/listener/faultManagementJobAttributeValueChangeEvent {{id}}

204 confirmation

{{callbackUrl}}/listener/faultManagementJobStateChangeEvent {{id}}, state=`inProgress`

204 confirmation

> FaultManagementReport generation failed

{{callbackUrl}}/listener/faultManagementJobReportPreparationErrorEvent {{id}}, failureReason

204 confirmation

> FaultManagementJob is not recurring

{{callbackUrl}}/listener/faultManagementJobStateChangeEvent {{id}}, state=`completed`

204 confirmation

[validation or internal problem]

4xx or 5xx error response

DELETE {{baseUrl}}/hub/{{id}}

204 confirmation

Figure 35. Fault Management Job lifecycle with all Notifications

After a successful Notification subscription, the Seller/Server sends a `FaultManagementJob` create request. The SOF responds with `FaultManagementJob` in an `acknowledged` state. Creation of `FaultManagementJob` is notified with a `faultManagementJobCreateEvent`. When the validation is successful and the Fault Management Job is not immediate, it moves to `scheduled` and a `faultManagementJobStateChangeEvent` is sent. When the scheduled start time is reached, `FaultManagementJob` moves to `inProgress` status and the `faultManagementJobStateChangeEvent` is sent. Fault Management Job periodically produces a Fault Management Report. This is when the `faultManagementJobReportReadyEvent` is sent. Additional actions, like suspension or modification trigger `faultManagementJobStateChangeEvent`. In addition, in the case of `FaultManagementJob` modification, Seller/Server produces `faultManagementJobAttributeValueChangeEvent` notification. When report generation fails, `faultManagementJobReportPreparationErrorEvent` is generated.

The following snippets present an example of `faultManagementJobCreateEvent` and `faultManagementJobReportReadyEvent`.

```json
{
  "eventId": "708b8a70-4246-4223-83b2-cddfa96ba3bc",
  "eventTime": "2025-01-01T00:00:00.000Z",
  "eventType": "faultManagementJobCreateEvent",
  "event": {
    "id": "755e55e2-72b0-4e3b-af00-693e3beac691"
  }
}
```

```json
{
  "eventId": "b8c35598-cc86-4902-99d2-36eef8e9786a",
  "eventTime": "2025-01-01T01:00:00.000Z",
  "eventType": "faultManagementJobReportReadyEvent",
  "event": {
    "id": "755e55e2-72b0-4e3b-af00-693e3beac691",
    "reportId": "c7ad584f-506a-4db1-bbeb-acbd6672f0a7"
  }
}
```

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

Note: The state change notification is sent only when the state attribute changes its value. There are no status change notifications sent upon Fault Management Job creation.

[R78] The Seller/Server FM Notifications **MUST** be sent to Buyer/Clients who have subscribed to FM Notifications. [MEF133.1 R30]

[R79] The Seller/Server FM Notifications **MUST NOT** be sent to Buyer/Clients who have not subscribed to FM Notifications. [MEF133.1 R31]

[R80] The Seller/Server's FM Notification **MUST** include the following attributes: [MEF133.1 R32]

- FM Notification Type
- FM Notification Identifier

[R81] If the Buyer/Client registered for FM Notifications, the Seller/Server **MUST** notify the Buyer/Client when FM Job results are available. [MEF133.1 R5]

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

7. API Details

7.1. API patterns

7.1.1. Indicating errors

Erroneous situations are indicated by appropriate HTTP responses. An error response is indicated by HTTP status 4xx (for client errors) or 5xx (for server errors) and the appropriate response payload. The Fault Management 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`.
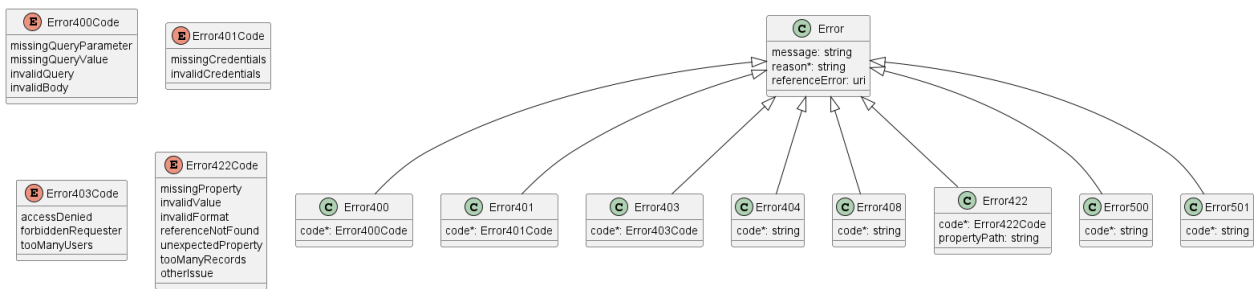
Figure 36. Data model types to represent an erroneous response

7.1.1.1. Type Error

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

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

7.1.1.2. Type Error400

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

Inherits from:

- Error

Name	Type	Description
code*	Error400Code	

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	

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 refused to authorize it. (https://tools.ietf.org/html/rfc7231#section-6.5.3)

Inherits from:

- Error

Name	Type	Description
code*	Error403Code	

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 Error408

Description: Request Time-out (https://tools.ietf.org/html/rfc7231#section-6.5.7)

Inherits from:

- Error

Name	Type	Description
code*	string	List of supported error codes: - timeOut: Request Time-out - indicates that the server did not receive a complete request message within the time that it was prepared to wait.

7.1.1.10. Type Error422

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

Inherits from:

- Error

Name	Type	Description
code*	Error422Code	
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 that was 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 target system
- unexpectedProperty: Additional, not expected property has been provided
- tooManyRecords: The number of records to be provided in the response exceeds the 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
code*	string	The following error code: - internalError: Internal server error - the server encountered an unexpected condition that prevented it from fulfilling the request.

7.1.1.13. Type Error501

Description: Not Implemented. Used in case Seller is not supporting an optional operation (https://tools.ietf.org/html/rfc7231#section-6.6.2)

Inherits from:

- Error

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

7.1.2. Response pagination

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

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

The filtering and pagination attributes must be specified in URI query format RFC3986.The Seller/Server returns a list of elements that comply with the requested `limit`. If the requested `limit` is higher than the supported list size the smaller list result is returned. In that case, the size of the result is returned in the header attribute `X-Result-Count`. The Seller can indicate that there are additional results available using:

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

[R82] Seller **MUST** use either `X-Total-Count` or `X-Pagination-Throttled` to indicate that the page was truncated and additional results are available.

7.2. Management API Data model

Figure 37 presents the full Fault Management data model. The data types, requirements related to them, and mapping to MEF W133.1 specification are discussed later in this section.

FaultManagementJob_Create

ServicePayloadSpecificAttributes
@type*: string

FaultManagementJobRef
faultManagementJobHref: string
faultManagementJobId*: string

AttachmentURL
retentionPeriod: date-time
url*: uri

Interval (E)
10 milliseconds
100 milliseconds
1 second
10 seconds
1 minute
5 minutes
15 minutes
30 minutes
1 hour
24 hours
1 month
1 year

JobType (E)
proactive
on-demand
passive

MeasurementTime
measurementStartDate: date-time
measurementEndDate: date-time
measurementInterval: interval

FaultManagementJob_Common
description: string
granularity*: Interval
jobPriority: integer
jobType*: JobType
outputFormat*: OutputFormat
reportingPeriod*: Interval
resultFormat*: ResultFormat

FaultManagementJobComplexQuery_Create
creationDate.gt: date-time
creationDate.lt: date-time
granularity: Interval
jobPriority: integer
jobType: JobType
lastModifiedDate.gt: date-time
lastModifiedDate.lt: date-time
outputFormat: OutputFormat
reportingPeriod: Interval
resultFormat: ResultFormat
state: FaultManagementJobStateType

FaultManagementJob_Find
creationDate*: date-time
description: string
granularity*: Interval
id*: string
jobPriority: integer
jobType*: JobType
outputFormat*: OutputFormat
reportingPeriod*: Interval
state*: FaultManagementJobStateType

FaultManagementJobComplexQuery
creationDate*: date-time
description: string
granularity*: Interval
jobPriority: integer
jobType*: JobType
lastModifiedDate: date-time
outputFormat*: OutputFormat
reportingPeriod*: Interval
resultFormat*: ResultFormat
state*: FaultManagementJobStateType

OutputFormat (E)
json
xml
avro
csv

FaultManagementJobStateType (E)
acknowledged
cancelled
completed
inProgress
pending
pendingCancel
rejected
resourcesUnavailable
scheduled
suspended

ResultFormat (E)
attachment
payload

ResultPayload
@type*: string

FaultManagementJob
creationDate*: date-time
href: uri
id*: string
lastModifiedDate: date-time
rejectionReason: string
state*: FaultManagementJobStateType

ScheduleDefinition
scheduleDefinitionStartTime: date-time
scheduleDefinitionEndTime: date-time
executionDuration: Interval

ServiceId
serviceIdFrom*: string
serviceIdTo*: string

TrackingRecord
creationDate*: date-time
description: string
id*: string
relatedObjectId*: string
request: string
system: string
user: string

TrackingRecord_Find
creationDate*: date-time
description: string
id*: string
relatedObjectId*: string
user: string

RecurringSchedule
second: string
minute: string
hour: string
dayOfMonth: string
month: string
dayOfWeek: string

HourRange
start: date-time
end: date-time

FaultManagementReportComplexQuery_Create
creationDate.gt: date-time
creationDate.lt: date-time
faultManagementJobId: string
granularity: Interval
lastModifiedDate.gt: date-time
lastModifiedDate.lt: date-time
outputFormat: OutputFormat
reportingTimeframe.startDate.gt: date-time
reportingTimeframe.startDate.lt: date-time
reportingTimeframe.endDate.gt: date-time
reportingTimeframe.endDate.lt: date-time
resultFormat: ResultFormat
state: FaultManagementReportStateType

FaultManagementReportStateType (E)
acknowledged
completed
failed
inProgress
rejected

ResultPayload

MeasurementTime

FaultManagementReportRef
faultManagementReportHref: string
faultManagementReportId*: string

ServiceId

ServicePayloadSpecificAttributes

ReportContentItem

AttachmentURL

FaultManagementReport
creationDate*: date-time
description: string
failureReason: string
href: uri
id*: string
lastModifiedDate: date-time
state*: FaultManagementReportStateType

FaultManagementReport_Find
creationDate*: date-time
description: string
id*: string
state*: FaultManagementReportStateType

FaultManagementReportComplexQuery
creationDate*: date-time
description: string
lastModifiedDate: date-time
state*: FaultManagementReportStateType

ReportingTimeframe
reportingStartDate: date-time
reportingEndDate: date-time

FaultManagementJobRef

ServicePayloadSpecificAttributes

ModifyFaultManagementJob_Common
description: string
granularity: Interval
jobPriority: integer
modificationReason: string
outputFormat: OutputFormat
reportingPeriod: Interval
resultFormat: ResultFormat

CancelFaultManagementJob_Common
cancellationReason: string

ModifyFaultManagementJob_Find
creationDate*: date-time
id*: string

CancelFaultManagementJob_Find
creationDate*: date-time
id*: string

ScheduleDefinition

CancelFaultManagementJob
cancellationDeniedReason: string
creationDate*: date-time
href: string
id*: string

CancelFaultManagementJob_Create

FaultManagementJobRef

ModifyFaultManagementJob
creationDate*: date-time
href: string
id*: string
modificationDeniedReason: string

ModifyFaultManagementJob_Create

FaultManagementJobProcessStateType (E)
acknowledged
completed
inProgress
rejected

Relationship labels: servicePayloadSpecificAttributes, faultManagementJob, scheduleDefinition, serviceId, recurringSchedule, hourRange, measurementDataPoints, measurementTime, reportContent, reportUrl, reportingTimeframe, faultManagementReport, state, faultManagementJob

Figure 37. Fault Management Data Model

7.2.1. FaultManagementJob

7.2.1.1. Type FaultManagementJob_Common

Description: A Fault Management Job specifies the configuration of FM Job to be performed on a Service by the Seller/Server.

Name	Type	Description
description	string	A free-text description of th Fault Management Job.
granularity*	Interval	Sampling rate of the collection production of Fault Managemer Job results.
jobPriority	integer	The priority of the Fau Management Job. The way th management application will us the Job priority to schedule Jc execution is application-specifi and out the scope.
jobType*	JobType	The type of FM Job.
outputFormat*	OutputFormat	List of possible output forma for the Fault Manageme Report.
reportingPeriod*	Interval	Defines the interval for the repo generation.
resultFormat*	ResultFormat	List of possible result forma that define how Seller/Server w deliver Fault Managemer Report to the Buyer/Client.
scheduleDefinition*	ScheduleDefinition	The schedule definition f running jobs.
serviceId*	ServiceId	Identifier of a Service.
servicePayloadSpecificAttributes*	ServicePayloadSpecificAttributes	ServicePayloadSpecificAttribute is used as an extension point f MEF-specific service fau management configuration.

7.2.1.2. Type FaultManagementJob_Create

Description: A Fault Management Job specifies the configuration of FM Job to be executed on a Service by the Seller/Server.

Inherits from:

- FaultManagementJob_Common

7.2.1.3. Type FaultManagementJob

Description: A Fault Management Job specifies the configuration of the job to be executed by Seller/Server as part of the troubleshooting or diagnostic process of a Service.

Inherits from:

- FaultManagementJob_Common

Name	Type	Description	MEF W133.1
creationDate*	date-time	Date when Fault Management Job was created.	Creation Date
href	uri	Hyperlink reference	Href
id*	string	Unique identifier	FM Job Identifier
lastModifiedDate	date-time	Date when the job was last modified.	Last Modified Date
rejectionReason	string	Reason in case creation request was rejected.	Not present
state*	FaultManagementJobStateType	The state of the Fault Management Job.	State

7.2.1.4. Type FaultManagementJob_Find

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

Name	Type	Description	MEF W133.1
creationDate*	date-time	Date when the job was created.	Creation Date
description	string	A free-text description of the Fault Management Job.	Description
granularity*	Interval	Sampling rate of the collection or production of Fault Management Job results.	Granularity
id*	string	Unique identifier	FM Job Identifier

Name	Type	Description	MEF W133.1
jobPriority	integer	The priority of the Fault Management Job. The way the management application will use the Job priority to schedule Job execution is application-specific and out the scope.	FM Job Priority
jobType*	JobType	The type of FM Job.	FM Job Type
outputFormat*	OutputFormat	List of possible output formats for the Fault Management Report.	Output Format
reportingPeriod*	Interval	Defines the interval for the report generation.	Reporting Period
scheduleDefinition*	ScheduleDefinition	The schedule definition for running jobs.	Schedule Definition
serviceId*	ServiceId	Identifier of a Service.	Service ID
state*	FaultManagementJobStateType	The state of the Fault Management Job.	State

7.2.1.5. Type CancelFaultManagementJob_Common

Description: Request for cancellation of an existing Fault Management Job.

Name	Type	Description	MEF W133.1
cancellationReason	string	An optional attribute that allows the Buyer/Client to provide additional details to the Seller/Server on the reason for cancelling Fault Management Job.	Not present
faultManagementJob*	FaultManagementJobRef	A reference to a Fault Management Job resource.	FM Job Identifier

7.2.1.6. Type CancelFaultManagementJob_Create

Description: Request for cancellation of an existing Fault Management Job.

Inherits from:

- CancelFaultManagementJob_Common

7.2.1.7. Type CancelFaultManagementJob

Description: Request for cancellation of an existing Fault Management Job

Inherits from:

- CancelFaultManagementJob_Common

Name	Type	Description	MEF W133.1
cancellationDeniedReason	string	If the Cancel Fault Management Job request is denied by the Seller/Server, the Seller/Server provides a reason to the Buyer/Client using this attribute.	Not present
creationDate*	date-time	Date when Cancel Fault Management Job was created.	Not present
href	string	Hyperlink to the Cancel Fault Management Job entity	Not present
id*	string	Unique identifier for the Cancel Fault Management Job that is generated by the Seller/Server when the Cancel Fault Management Job request `state` is set to `acknowledged`.	Not present
state*	FaultManagementJobProcessStateType	The state of the process related to the Fault Management Job.	Not present

7.2.1.8. Type CancelFaultManagementJob_Find

Description: This class represents a single list item for the response of `listCancelFaultManagementJob`

Name	Type	Description	MEF W133.1

Name	Type	Description	MEF W133.1
creationDate*	date-time	Date when Cancel Fault Management Job was created.	Not present
id*	string	Unique identifier for the Cancel Fault Management Job that is generated by the Seller/Server when the Cancel Fault Management Job request `state` is set to `acknowledged`.	Not present
faultManagementJob*	FaultManagementJobRef	A reference to a Fault Management Job resource.	FM Job Identifier
state*	FaultManagementJobProcessStateType	The state of the process related to the Fault Management Job.	Not present

7.2.1.9. Type ModifyFaultManagementJob_Common

Description: Request for modification of an existing Fault Management Job

Name	Type	Description
description	string	A free-text description of the Fault Management Job.
faultManagementJob*	FaultManagementJobRef	A reference to a Fault Management Job resource.
granularity	Interval	Sampling rate of the collection or production of Fault Management Job results.
jobPriority	integer	The priority of the Fault Management Job. The way the management application will use the Job priority to schedule Job execution is application-specific and out the scope.

Name	Type	Description
modificationReason	string	An optional attribute that allows the Buyer/Client to provide additional detail to the Seller/Server on the reason for modifying Fault Management Job.
outputFormat	OutputFormat	List of possible output formats for the Fault Management Report.
reportingPeriod	Interval	Defines the interval for the report generation.
resultFormat	ResultFormat	List of possible result formats that define how Seller/Server will deliver Fault Management Report to the Buyer/Client.
scheduleDefinition	ScheduleDefinition	The schedule definition for running jobs.
servicePayloadSpecificAttributes	ServicePayloadSpecificAttributes	ServicePayloadSpecificAttributes is used as an extension point for MEF-specific service fault management configuration.

7.2.1.10. Type ModifyFaultManagementJob_Create

Description: Request for modification of an existing Fault Management Job.

Inherits from:

- ModifyFaultManagementJob_Common

7.2.1.11. Type ModifyFaultManagementJob

Description: Request for modification of an existing Fault Management Job

Inherits from:

- ModifyFaultManagementJob_Common

Name	Type	Description	MEF W133.1
creationDate*	date-time	Date when Modify Fault Management Job was created.	Not present

Name	Type	Description	MEF W133.1
href	string	Hyperlink to the Modify Fault Management Job entity	Not present
id*	string	Unique identifier for the Modify Fault Management Job that is generated by the Seller/Server when the Modify Fault Management Job request `state` is set to `acknowledged`	Not present
modificationDeniedReason	string	If the Modify Fault Management Job request is denied by the Seller/Server, the Seller/Server provides a reason to the Buyer/Client using this attribute.	Not present
state*	FaultManagementJobProcessStateType	The state of the process related to the Fault Management Job.	Not present

7.2.1.12. Type ModifyFaultManagementJob_Find

Description: This class represents a single list item for the response of `listModifyFaultManagementJob`

Name	Type	Description	MEF W133.1
creationDate*	date-time	Date when Modify Fault Management Job was created.	Not present

Name	Type	Description	MEF W133.1
id*	string	Unique identifier for the Modify Fault Management Job that is generated by the Seller/Server when the Modify Fault Management Job request `state` is set to `acknowledged`.	Not present
faultManagementJob*	FaultManagementJobRef	A reference to a Fault Management Job resource.	FM Job Identifier
state*	FaultManagementJobProcessStateType	The state of the process related to the Fault Management Job.	Not present

7.2.1.13. Type FaultManagementJobComplexQuery_Create

Description: Fault Management Job Complex Query entity is used to perform searches on Fault Management Job entities, including clauses based on ScheduleDefinition and ServicePayloadSpecificAttributes.

Name	Type	Description
creationDate.gt	date-time	Date when the Fault Management Job was created greater than.
creationDate.lt	date-time	Date when the Fault Management Job was created lower than.
granularity	Interval	Sampling rate of the collection or production of Fault Management Job results.
jobPriority	integer	The priority of the Fault Management Job. The way the management application will use the Job priority to schedule Job execution is application-specific and out the scope.
jobType	JobType	The type of FM Job.

Name	Type	Description
lastModifiedDate.gt	date-time	Date when the Faul Management Job was last modified - greater than.
lastModifiedDate.lt	date-time	Date when the Faul Management Job was last modified - lower than.
outputFormat	OutputFormat	List of possible output formats for the Fault Managemen Report.
reportingPeriod	Interval	Defines the interval for the repor generation.
resultFormat	ResultFormat	List of possible result formats that define how Seller/Server wil deliver Fault Managemen Report to the Buyer/Client.
scheduleDefinition	ScheduleDefinition	The schedule definition fo running jobs.
serviceId	ServiceId	Identifier of a Service.
servicePayloadSpecificAttributes	ServicePayloadSpecificAttributes	ServicePayloadSpecificAttributes is used as an extension point fo MEF-specific service faul management configuration.
state	FaultManagementJobStateType	The state of the Faul Management Job.

7.2.1.14. Type FaultManagementJobComplexQuery

Description: Fault Management Job Complex Query entity is used to perform searches on Fault Management Job entities, including clauses based on ScheduleDefinition and ServicePayloadSpecificAttributes.

Name	Type	Description
creationDate*	date-time	Date when the Fau Management Job was created.
description	string	A free-text description of th Fault Management Job.
faultManagementJob*	FaultManagementJobRef	A reference to a Fau Management Job resource.
granularity*	Interval	Sampling rate of the collection production of Fault Manageme Job results.

Name	Type	Description
jobPriority	integer	The priority of the Fault Management Job. The way the management application will use the Job priority to schedule Job execution is application-specific and out the scope.
jobType*	JobType	The type of FM Job.
lastModifiedDate	date-time	Date when the Fault Management Job was last modified.
outputFormat*	OutputFormat	List of possible output formats for the Fault Management Report.
reportingPeriod*	Interval	Defines the interval for the report generation.
resultFormat*	ResultFormat	List of possible result formats that define how Seller/Server will deliver Fault Management Report to the Buyer/Client.
scheduleDefinition*	ScheduleDefinition	The schedule definition for running jobs.
serviceId*	ServiceId	Identifier of a Service.
servicePayloadSpecificAttributes*	ServicePayloadSpecificAttributes	ServicePayloadSpecificAttributes is used as an extension point for MEF-specific service fault management configuration.
state*	FaultManagementJobStateType	The state of the Fault Management Job.

7.2.1.15. enum FaultManagementJobProcessStateType

Description: The state of the process related to the Fault Management Job

state	MEF 133 name	Description
acknowledged	Acknowledged	The Cancel/Modify Fault Management Job request has been received by the Seller/Server and has passed basic validation. Fault Management Job Process Identifier is assigned in the Acknowledged state. The request remains Acknowledged until all validations as applicable are completed. If the attributes are validated, the request moves to the In-Progress state. If not all attributes are validated, the request moves to the Rejected state.
completed	Completed	The Cancel/Modify Fault Management Job request has been completed by the Seller/Server.

state	MEF 133 name	Description
inProgress	In-Progress	The Cancel/Modify Fault Management Job request has been validated and accepted by the Seller/Server and is in-progress.
rejected	Rejected	The Cancel/Modify Fault Management Job request has failed validation and has been declined by the Seller/Server.

7.2.1.16. Type FaultManagementJobRef

Description: A reference to a Fault Management Job resource.

Name	Type	Description	MEF 133.1
faultManagementJobHref	string	Hyperlink to the referenced Fault Management Job.	Href
faultManagementJobId*	string	Identifier of the referenced Fault Management Job.	FM Job Identifier

7.2.1.17. enum FaultManagementJobStateType

Description: The state of the Fault Management Job.

state	MEF 133 name	Description
acknowledged	Acknowledged	A Create Fault Management Job request has been received by the Seller/Server and has passed basic validation. Fault Management Job Identifier is assigned in the Acknowledged state. The request remains Acknowledged until all validations as applicable are completed. If the attributes are validated the request determines if the start time is immediate or scheduled. If immediate, the Fault Management Job moves to the In-progress state. If scheduled, the Fault Management Job moves to the Scheduled state. If not all attributes are validated, the request moves to the Rejected state.
cancelled	Cancelled	A Fault Management Job that is In-Progress, Suspended, or Scheduled is cancelled.
completed	Completed	A non-recurring Fault Management Job finished execution.

state	MEF 133 name	Description
inProgress	In-Progress	A Fault Management Job is running. Upon completion of the Job, a determination if the Fault Management Job is a one-time Job or is recurring is performed. If the Fault Management Job is a one-time Job, the state of the Fault Management Job moves to the Completed state. If the Fault Management Job is recurring, the Fault Management Job circles back to determine if it has an immediate start time or a scheduled start time. If a Suspend Fault Management Job request is accepted, the Job moves to the Suspended state. If a Cancel Fault Management Job request is accepted, the Job moves to the Cancelled state.
pending	Pending	A Modify Fault Management Job request has been accepted by the Seller/Server. The Fault Management Job remains Pending while updates to the Job are completed. Once updates are complete, the Job returns to the Scheduled or In-Progress status depending on the schedule definition.
pendingCancel	Pending Cancel	A Cancel Fault Management Job request has been accepted by the Seller/Server. The Fault Management Job remains Pending Cancel while resources used by the Job are being released. Once updates are complete, the Job moves to the Cancelled status.
rejected	Rejected	A Create Fault Management Job request fails validation and is rejected with error indications by the Seller/Server.
resourcesUnavailable	Resources Unavailable	A Fault Management Job cannot be allocated necessary resources when moving to execution (In-Progress state).
scheduled	Scheduled	A Fault Management Job is created that does not have an immediate start time. The Fault Management Job stays Scheduled until the start time is reached. The Fault Management Job then moves to In-Progress. If the Cancel Fault Management Job request is accepted, the Job moves to the Cancelled state. If the Modify Fault Management Job request is accepted, the Job moves to the Pending state.
suspended	Suspended	A Suspend Fault Management Job request is accepted by the Seller/Server. The Job remains Suspended until a Resume Fault Management Job request is accepted by the Seller/Server at which time the Job returns to the In-Progress state. If the Cancel Fault Management Job request is accepted, the Job moves to the Cancelled state. If the Modify Fault Management Job request is accepted, the Job moves to the Pending state.

7.2.2. FaultManagementReport

7.2.2.1. Type FaultManagementReport

Description: The execution of Fault Management Job creates FM Reports that provide Buyer/Client with the results of the Job.

Name	Type	Description	MEF W133.1
creationDate*	date-time	Date when Fault Management Report was created.	Not present
description	string	A free-text description of the Fault Management Report.	Not present
failureReason	string	Reason in case report generation failed.	Not present
faultManagementJob*	FaultManagementJobRef	A reference to a Fault Management Job resource.	FM Job
href	uri	Hyperlink reference	Not present
id*	string	Unique identifier	Report Identifier
lastModifiedDate	date-time	Date when the report was last modified.	Not present
reportContent	ReportContentItem[]		Not present
reportingTimeframe*	ReportingTimeframe	Specifies the date range between which data points will be included in the report.	Not present
reportUrl	AttachmentURL	The URL pointing to an Attachment for download.	File Transfer Data
state*	FaultManagementReportStateType	Possible values for the state of a Fault Management Report.	State

7.2.2.2. Type FaultManagementReport_Find

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

Name	Type	Description	MEF W133.1
creationDate*	date-time	Date when the report was created.	Not present

Name	Type	Description	MEF W133.1
description	string	A free-text description of the Fault Management Report	Not present
faultManagementJob*	FaultManagementJobRef	A reference to a Fault Management Job resource.	FM Job Identifier
id*	string	Unique identifier	Report Identifier
reportingTimeframe*	ReportingTimeframe	Specifies the date range between which data points will be included in the report.	Not present
state*	FaultManagementReportStateType	Possible values for the state of a Fault Management Report.	State

7.2.2.3. Type FaultManagementReportComplexQuery_Create

Description: Fault Management Report Complex Query entity is used to perform searches on Fault Management Report entities, including clauses based on ServicePayloadSpecificAttributes.

Name	Type	Description
creationDate.gt	date-time	Date when Fault Manageme Report was created - greater tha
creationDate.lt	date-time	Date when Fault Manageme Report was created - lower than
faultManagementJobId	string	Identifier of the referenced Fau Management Job.
granularity	Interval	Sampling rate of the collection production of Fault Manageme Job results.
lastModifiedDate.gt	date-time	Date when the report was la modified - greater than.
lastModifiedDate.lt	date-time	Date when the report was la modified - lower than.
outputFormat	OutputFormat	List of possible output forma for the Fault Manageme Report.
reportingTimeframe.startDate.gt	date-time	Start date of reporting timefran - greater than.
reportingTimeframe.startDate.lt	date-time	Start date of reporting timefran - lower than.
reportingTimeframe.endDate.gt	date-time	End date of reporting timeframe greater than.

Name	Type	Description
reportingTimeframe.endDate.lt	date-time	End date of reporting timeframe lower than.
resultFormat	ResultFormat	List of possible result forma that define how Seller/Server wi deliver Fault Managemen Report to the Buyer/Client.
serviceId	ServiceId	Identifier of a Service.
servicePayloadSpecificAttributes	ServicePayloadSpecificAttributes	ServicePayloadSpecificAttribute is used as an extension point f MEF-specific service fau management configuration.
state	FaultManagementReportStateType	Possible values for the state of Fault Management Report.

7.2.2.4. Type FaultManagementReportComplexQuery

Description: Fault Management Report Complex Query entity is used to perform searches on Fault Management Report entities, including clauses based on ServicePayloadSpecificAttributes.

Name	Type	Description	MEF W133.1
creationDate*	date-time	Date when Fault Management Report was created.	Not present
description	string	A free-text description of the Fault Management Report.	Not present
faultManagementJob*	FaultManagementJobRef	A reference to a Fault Management Job resource.	FM Job Identifier
faultManagementReport*	FaultManagementReportRef	A reference to a Fault Management Report resource.	Report Identifier
lastModifiedDate	date-time	Date when the report was last modified.	Not present
reportingTimeframe*	ReportingTimeframe	Specifies the date range between which data points will be included in the report.	Not present
state*	FaultManagementReportStateType		State

7.2.2.5. Type FaultManagementReportRef

Description: A reference to a Fault Management Report resource.

Name	Type	Description	MEF W133.1
faultManagementReportHref	string	Hyperlink to the referenced Fault Management Report.	Not present
faultManagementReportId*	string	Identifier of the referenced Fault Management Report.	Report Identifier

7.2.2.6. enum FaultManagementReportStateType

Description: Possible values for the state of a Fault Management Report.

State	Description
acknowledged	A Fault Management Report is created by the Seller/Server. Fault Management Report Identifier is assigned in the Acknowledged state. The report remains Acknowledged until all validations as applicable are completed. If the attributes are validated, the Fault Management Report moves to the In-Progress state. If not all attributes are validated, the report moves to the Rejected state.
completed	A Fault Management Report is completed and results are available.
failed	A Fault Management Report processing has failed.
inProgress	A Fault Management Report processing has started.
rejected	This state indicates that the request failed to meet validation rules for `FaultManagementReport` delivery (processing).

7.2.3. Common

Types described in this subsection are shared among two or more LSO APIs.

7.2.3.1. Type AttachmentURL

Description: The URL pointing to an Attachment for download.

Name	Type	Description	MEF 133.1
retentionPeriod	date-time	A date until which the file will be retained.	Retention Period
url*	uri	The URL pointing to an Attachment for download.	File Location

7.2.3.2. Type HourRange

Description: Defines start and end date,

Name	Type	Description	MEF W133.1
start	date-time	Start date	Hour Range
end	date-time	End date	Hour Range

7.2.3.3. enum Interval

Description: Enumeration of applicable time intervals.

Value	MEF 133.1
10 milliseconds	10 milliseconds
100 milliseconds	100 milliseconds
1 second	1 second
10 seconds	10 seconds
1 minute	1 minute
5 minutes	5 minutes
15 minutes	15 minutes
30 minutes	30 minutes
1 hour	1 hour
24 hours	24 hours
1 month	1 month
1 year	1 year

7.2.3.4. enum JobType

Description: The type of FM Job.

Value	MEF W133.1
proactive	Proactive
on-demand	On-demand
passive	Passive

7.2.3.5. Type MeasurementTime

Description: Timeframe boundary for collected data. Provide measurementStartDate and measurementEndDate or measurementStartDate and measurementInterval.

Name	Type	Description	MEF W133.1
measurementStartDate	date-time	Start date of the period to which collected data points belong.	Not present
measurementEndDate	date-time	End date of the period to which collected data points belong.	Not present
measurementInterval	Interval	Length of the measurement interval.	Not present

7.2.3.6. enum OutputFormat

Description: List of possible output formats for the Fault Management Report.

Value	MEF W133.1
json	JSON
xml	XML
avro	AVRO
csv	CSV

7.2.3.7. Type RecurringSchedule

Description: A definition of recurring schedule to run a job based on the Cron utility in Linux-like systems. It defines how the job should periodically run at specified times, dates, or intervals.

Name	Type	Description	MEF W133.1
second	string	A definition of time (seconds) to run a job. Allowed values: 0-59, and special characters: (,-*/), where: ``` - `*` -> any value - `,` -> value list seprator - `-` -> range of values - '/' -> step values For example: - */5 * * * * * -> run a job at every 5th second. - */30 */1 * * * * -> run a job at every 30 seconds past every minute. ```	Not present
minute	string	A definition of time (minutes) to run a job. Allowed values: 0-59, and special characters: (,-*/), where: ``` - `*` -> any value - `,` -> value list seprator - `-` -> range of values - '/' -> step values For example: - 0 */10 * * * * -> run a job at every 10th minute. - 0 */30 0 * * * -> run a job at every 30th minute past midnight. - 0 */30 8 * * 1 -> run a job at every 30th minute past hour 8 on Monday. ```	Not present

Name	Type	Description	MEF W133.1
hour	string	A definition of time (hours) to run a job. Allowed values: 0-23, and special characters: (,-*/), where: ``` - `*` -> any value - `,` -> value list seprator - `-` -> range of values - '/' -> step values For example: - 0 0 10 10 * * -> run a job 10 am on 10th day of every month. - 0 * 1,2 * * * -> run a job at every minute past hour 1 and 2. - 0 0 */2 * * * -> run a job at every 2nd hour. ```	Not present

dayOfMonth	string	A definition of time (day of month) to run a job. Allowed values: 1-31, and special characters: (,-*/), where:
		
		``
		 - `*` -> any value
		 - `,` -> value list seprator
		 - `-` -> range of values
		 - '/' -> step values
		For example:
		 - 0 0 0 1,5,10,15 * * -> run a
		job at midnight
		  on every 1st, 5th, 10th, 15th
		day of month.
		 - 0 0 */1 1-10 1-3 * -> run a
		job at every full hour between
		1st
		   and 10th day of month in
		January, February, and March.
		``
			Not present

month	string	A definition of time (month) to run a job. Allowed values: 1-12 or JAN-DEC, and special characters: (,-*/), where: ``` - `*` -> any value - `,` -> value list seprator - `-` -> range of values - '/' -> step values For example: - 0 5 0 * 8 * -> run a job at 00:05 on every day in August. - 0 0 0,12 1 */2 * -> run a job at midnight and noon on every 1st day of every 2nd month. ```	Not present

Name	Type	Description	MEF W133.1
dayOfWeek	string	A definition of time (day of week) to run a job. Allowed values: 0-6 or SUN-SAT, and special characters: (,-*/), where: ``` - `*` -> any value - `,` -> value list seprator - `-` -> range of values - '/' -> step values For example: - 0 0 22 * * 1-5 -> run a job at 22:00 on every day between Monday and Friday. - 0 5 4 * * sun -> run a job at 04:05 on Sunday. ```	Not present
hourRange	HourRange[]	A list of time ranges within a specific day that the schedule will be active on, for example, 08:00-12:00, 16:00-19:00.	Hour Range

7.2.3.8. Type ReportContentItem

Description: Single item of the Fault Management Job results in case result format was set to payload. Each item contains the timeframe of the collected data and a list of values measured in that timeframe.

Name	Type	Description	MEF W133.1
measurementTime*	MeasurementTime	Timeframe boundary for collected data.	Not present
measurementDataPoints	ResultPayload[]	List of Fault Management Job results measured in the related timeframe.	Not present

7.2.3.9. Type ReportingTimeframe

Description: Specifies the date range between which data points will be included in the report.

Name	Type	Description	MEF W133.1
reportingStartDate	date-time	Start date of reporting timeframe.	Not present
reportingEndDate	date-time	End date of reporting timeframe.	Not present

7.2.3.10. enum ResultFormat

Description: List of possible result formats that define how Seller/Server will deliver Fault Management Report to the Buyer/Client.

Value	MEF W133.1
attachment	Attachment
payload	Payload

7.2.3.11. Type ResultPayload

Description: ResultPayload is used as an extension point for MEF-specific fault management job results. The `@type` attribute is used as a discriminator.

Name	Type	Description	MEF W133.1
@type*	string	The name that uniquely identifies the type of results that are returned by the Fault Management Report. In the case of MEF services, this is the URN provided in the fault management results specification. The named type must be a subclass of ResultPayload.	Not present

7.2.3.12. Type ScheduleDefinition

Description: The schedule definition for running jobs.

Name	Type	Description	MEF W133.1
scheduleDefinitionStartTime	date-time	The start time of the Schedule Definition. If the attribute is empty the Schedule starts immediately after provisioning of the Job.	Start Time
scheduleDefinitionEndTime	date-time	The end time of the Schedule Definition. If the attribute is empty the Schedule runs forever, not having a time constraint.	End Time
recurringSchedule	RecurringSchedule	A recurring frequency to run a job within a timeframe defined by schedule definition, for example, every 5 minutes, 15 minutes, 1 hour, 1 day. If the attribute is empty, job runs non-stop.	Recurring Frequency
executionDuration	Interval	Total time for running one execution of a schedule. Depending on the reportingPeriod attribute, one execution of a schedule might produce multiple reports (e.g., when reporting period is 15 minutes and executionDuration is 1 hour, every execution of a schedule will produce 4 reports).	Not present

7.2.3.13. Type ServiceId

Description: Identifier of a Service.

Name	Type	Description	MEF 133.1
serviceIdFrom*	string	Identifier of a Service Endpoint.	Service ID From
serviceIdTo*	string	Identifier of a Service Endpoint.	Service ID To

7.2.3.14. Type ServicePayloadSpecificAttributes

Description: ServicePayloadSpecificAttributes is used as an extension point for MEF-specific service fault management configuration. The `@type` attribute is used as a discriminator.

Name	Type	Description	MEF W133.1
@type*	string	Uniquely identifies the type of fault management job configuration. In the case of MEF services, this is the URN provided in the fault management configuration specification. The named type must be a subclass of ServicePayloadSpecificAttributes.	Not present

7.2.3.15. Type TrackingRecord

Description: Tracking Records allow the tracking of modifications of Fault Management Job or Report.

Name	Type	Description	MEF W133.1
creationDate*	date-time	Date when the record was created.	Creation Date
description	string	Free-text field describing the action that created the Tracking Record and its details.	Description
id*	string	Identifier of the Tracking Record.	Identifier
relatedObjectId*	string	Identifier of Fault Management Job or Report.	Related Object Identifier
request	string	Request that created the Tracking Record.	Request
system	string	Describes the system from which the action was done.	System
user	string	Describes the user doing the action.	User

7.2.3.16. Type TrackingRecord_Find

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

Name	Type	Description	MEF W133.1
creationDate*	date-time	Date when record was created.	Creation Date
description	string	Free-text field describing the action that created the Tracking Record and its details.	Description
id*	string	Identifier of the Tracking Record.	Identifier
relatedObjectId*	string	Identifier of Fault Management Job or Report.	Related Object Identifier

Name	Type	Description	MEF W133.1
user	string	User that executed the action which created a Tracking Record.	User

7.2.4. Notification Registration

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

7.2.4.1. Type EventSubscriptionInput

Description: This class is used to register for Notifications.

Name	Type	Description
callback*	string	This callback value must be set to the *host* property from Fault (faultNotification.api.yaml). This property is appended with the base path and noti that API to construct a URL to which notification is sent. E.g. for 'callback': "http Fault Management Job state change event notification `https://buyer.co/listenerEndpoint/mefApi/legato/faultManagement/v1/listener/faultM
query	string	This attribute is used to define which type of events to register to. faultManagementReportStateChangeEvent'. To subscribe for more than one event ty comma: `eventType=faultManagementReportStateChangeEvent,faultManagement values are enumerated by Event type enums in faultNotification.api.yaml. An empty filters - ending in subscription for all event types.

7.2.4.2. Type EventSubscription

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

Name	Type	Description	MEF W133.1
callback*	string	The value provided by the `EventSubscriptionInput` during notification registration.	Notification Target Information
id*	string	An identifier of this Event Subscription assigned when a resource is created.	Register Notification Identifier
query	string	The value provided by the `EventSubscriptionInput` during notification registration.	List of Notification Types

7.3. Notification API Data model

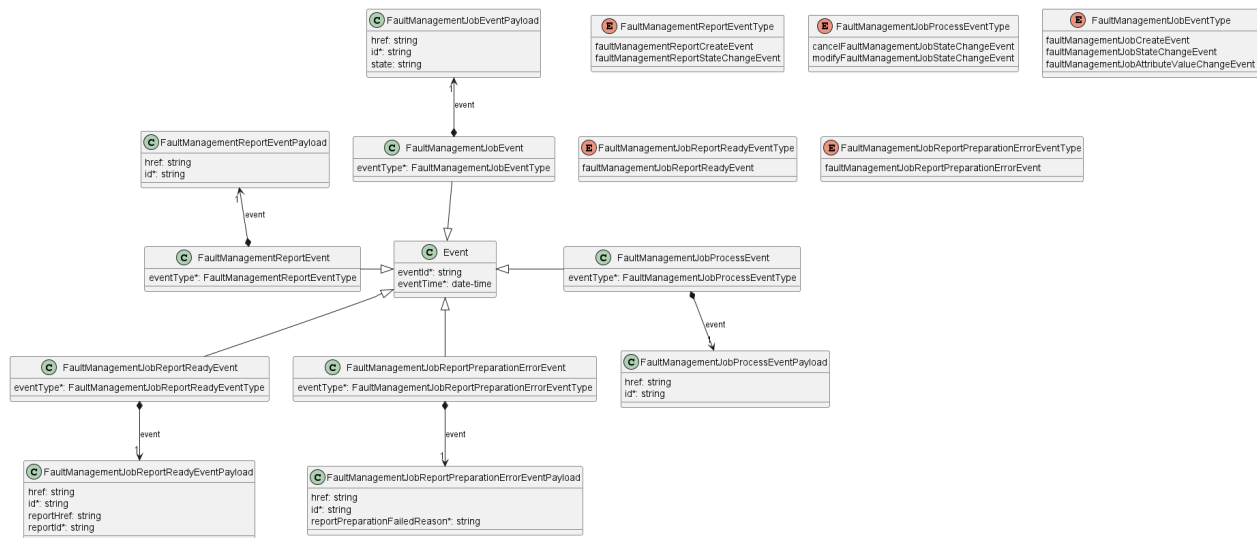Figure 38 presents the Fault Management Notification data model.

Figure 38. Fault Management Notification Data Model

This data model is used to construct requests and responses of the API endpoints described in 5.2.2. Buyer/Client (CUS, BUS, SOF) side Fault Management API Endpoints.

7.3.1. Type Event

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

Name	Type	Description	MEF W133.1
eventId*	string	Id of the event	Not present
eventTime*	date-time	Date-time when the event occurred	Not present

7.3.2. Type FaultManagementJobEvent

Description:

Inherits from:

- Event

Name	Type	Description	MEF 133.1
eventType*	FaultManagementJobEventType		Not present
event*	FaultManagementJobEventPayload		Not present

7.3.3. enum FaultManagementJobEventType

Description: Indicates the type of Fault Management Job event.

Value	MEF 133.1
faultManagementJobCreateEvent	FM Job createdT
faultManagementJobStateChangeEvent	FM Job state changed

Value	MEF 133.1
faultManagementJobAttributeValueChangeEvent	FM Job attribute value changed

7.3.4. Type FaultManagementJobEventPayload

Description: The identifier of the Fault Management Job being the subject to this event and its state.

Name	Type	Description	MEF 133.1
href	string	Hyperlink to access the Fault Management Job.	Href
id*	string	ID of the Fault Management Job.	FM Job Identifier
state	string	State of the Fault Management Job.	State

7.3.5. Type FaultManagementJobProcessEvent

Description:

Inherits from:

- Event

Name	Type	Description	MEF 133.1
eventType*	FaultManagementJobProcessEventType		Not present
event*	FaultManagementJobProcessEventPayload		Not present

7.3.6. enum FaultManagementJobProcessEventType

Description: Indicates the type of Fault Management Job Process event.

Value	MEF 133.1
cancelFaultManagementJobStateChangeEvent	Not present
modifyFaultManagementJobStateChangeEvent	Not present

7.3.7. Type FaultManagementJobProcessEventPayload

Description: The identifier of the Fault Management Job Process, including:

- Modify Fault Management Job
- Cancel Fault Management Job being the subject of this event.

Name	Type	Description	MEF 133.1
href	string	Hyperlink to access the Fault Management Job Process.	Not present
id*	string	ID of the Fault Management Job Process.	Not present

7.3.8. Type FaultManagementJobReportPreparationErrorEvent

Description:

Inherits from:

- Event

Name	Type	Description
eventType*	FaultManagementJobReportPreparationErrorEventType	
event*	FaultManagementJobReportPreparationErrorEventPayload	

7.3.9. enum FaultManagementJobReportPreparationErrorEventType

Description: Indicates the type of Fault Management Job event.

Value	MEF 133.1
faultManagementJobReportPreparationErrorEvent	FM Job Report Failed

7.3.10. Type FaultManagementJobReportPreparationErrorEventPayload

Description: The identifier of the Fault Management Job being the subject of this event and reason for report preparation failure.

Name	Type	Description
href	string	Hyperlink to access the Fault Management Job.
id*	string	ID of the Fault Management Job.
reportPreparationFailedReason*	string	Reason for Report preparation failure.

7.3.11. Type FaultManagementJobReportReadyEvent

Description:

Inherits from:

- Event

Name	Type	Description	MEF 133.1
eventType*	FaultManagementJobReportReadyEventType		Not present
event*	FaultManagementJobReportReadyEventPayload		Not present

7.3.12. enum FaultManagementJobReportReadyEventType

Description: Indicates the type of Fault Management Job event.

Value	MEF 133.1
faultManagementJobReportReadyEvent	FM Collection ready

7.3.13. Type FaultManagementJobReportReadyEventPayload

Description: The identifier of the Fault Management Job and Report ID being the subjects of this event.

Name	Type	Description	MEF 133.1
href	string	Hyperlink to access the Fault Management Job.	Href
id*	string	ID of the Fault Management Job.	FM Job Identifier

Name	Type	Description	MEF 133.1
reportHref	string	Hyperlink to access the Fault Management Report.	Not present
reportId*	string	ID of generated Fault Management Report.	Report Identifier

7.3.14. Type FaultManagementReportEvent

Description:

Inherits from:

- Event

Name	Type	Description	MEF 133.1
eventType*	FaultManagementReportEventType		Not present
event*	FaultManagementReportEventPayload		Not present

7.3.15. enum FaultManagementReportEventType

Description: Indicates the type of Fault Management Report event.

Value	MEF 133.1
faultManagementReportCreateEvent	Not present
faultManagementReportStateChangeEvent	Not present

7.3.16. Type FaultManagementReportEventPayload

Description: The identifier of the Fault Management Report being the subject of this event.

Name	Type	Description	MEF 133.1
href	string	Hyperlink to access the Fault Management Report.	Not present
id*	string	ID of the Fault Management Report.	Report Identifier

8. References

- JSON Schema draft 7, JSON Schema: A Media Type for Describing JSON Documents and associated documents, by Austin Wright and Henry Andrews, March 2018. Copyright © 2018 IETF Trust and the persons identified as the document authors. All rights reserved.
- [MEF55.1] MEF 55.1, Lifecycle Service Orchestration (LSO): Reference Architecture and Framework, February 2021
- [MEF128] MEF 128, LSO API Security Profile, July 2022
- MEFW133.1 Allegro, Interlude and Legato Fault Management and Alarm API BR&UC, January 2025
- [MEF152] [MEF 152] Carrier Ethernet Payload Schema/Guide for SOAM
- [MEF153] [MEF 153] IP/IPVPN Schema/Guide for SOAM
- [MEF154] [MEF 154] SD-WAN Schema/Guide for SOAM
- [OAS-v3] Open API 3.0, February 2020
- [REST] Chapter 5: Representational State Transfer (REST) Fielding, Roy Thomas, Architectural Styles and the Design of Network-based Software Architectures (Ph.D.).
- [RFC2119] RFC 2119, Key words for use in RFCs to Indicate Requirement Levels, by S. Bradner, March 1997
- [RFC3986] RFC 3986 Uniform Resource Identifier (URI): Generic Syntax, January 2005
- [RFC8174] RFC 8174, Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words, by B. Leiba, May 2017, Copyright © 2017 IETF Trust and the persons identified as the document authors. All rights reserved.
- [TMF630] TMF 630 TMF630 API Design Guidelines 4.2.0