MEF Standard

MEF 125.0.1

Amendment to MEF 125: LSO Cantata and LSO Sonata – Subscriber Ethernet Product Schemas and Developer Guide

January 2024

Disclaimer

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

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

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

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

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

Table of Contents

List of Figures

1 List of Contributing Members

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

- Amartus

2 Abstract

This document is an Amendment to MEF 125 LSO Cantata and LSO Sonata - Subscriber Ethernet Product Schemas and Developer Guide [7]. The purpose of this amendment is as follows:

- presentation of different Subscriber Ethernet configurations
- show the basic differences between Subscriber Ethernet technologies
- provide examples for actions (add, modify, delete)
- deliver basic API steps walkthrough to order a Subscriber Ethernet product

3 Introduction

This document delivers only informative New Appendix A and provides Postman collection examples.

In this amendment, changes are shown as follows:

- Instructions for how to apply the amendment are shown in *blue italics*
- In content modified by the amendment, the text to be removed is shown with ~~red strikethrough~~
- In content modified by the amendment, the text to be added is shown in red
- The content added by the amendment (Appendix A) is shown in standard black color.

4 Changes to Section 2 Abstract

Add the following paragraph to the end of section 2:

Also included in the MEF-LSO GitHub repository is a Postman file that contains informative examples illustrating the use of the Subscriber Ethernet payloads. This file is not part of this standard but is referred to in Appendix A.

- documentation/productSchema/carrierEthernet/subscriberEthernet/MEF 125.0.1 Appendix A.postman_collection.json

5 New Appendix A

Insert the content below into the document as Appendix A.

Appendix A Usage examples (Informative)

This appendix provides an extensive set of examples to cover:

- different configuration variants
- basic all API steps walkthrough to order a Subscriber Ethernet product
- common modifications
- deletion of a product

The examples are delivered in two forms:

- as part of this document (in parts) – to allow comments and rich explanation
- as a Postman collection – for ease of use in testing.

The following terms are used in Appendix A:

- EPL - ethernetPrivateLineEvc
- EVPL – ethernetVirtualPrivateLineEvc
- EP-LAN – ethernetPrivateLineEvc
- EP-LAN EVC End Point - ethernetPrivateLanEvcEp
- EVP-LAN – ethernetVirtualPrivateLanEvc
- EVP-LAN EVC End Point - ethernetVirtualPrivateLanEvcEp
- EP-TREE – ethernetPrivateTreeEvc
- EP-TREE EVC End Point - ethernetPrivateTreeEvcEp
- EVP-TREE – ethernetVirtualPrivateTreeEvc
- EVP-TREE EVC End Point - ethernetVirtualPrivateTreeEvcEp
- UNI – carrierEthernetSubscriberUni

A.1 High-Level flow

The Cantata and Sonata Interface Reference Points are formed from a set of APIs that serve different functions in the end-to-end flow. Figure A1-1 shows all of the functions and their sequence.

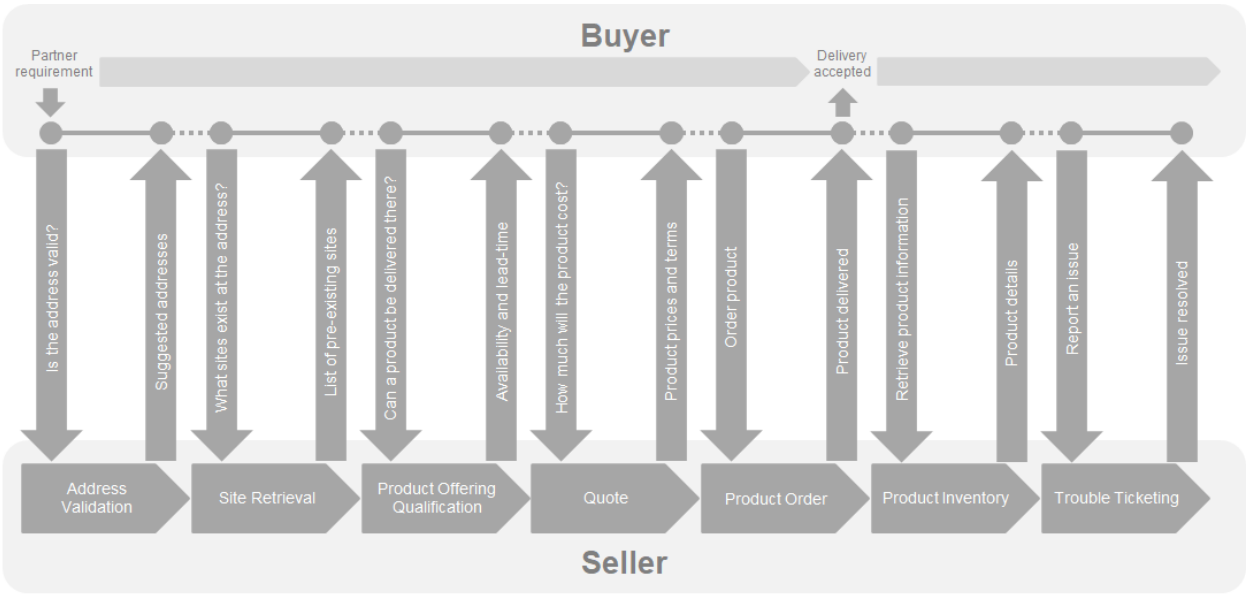

Figure A1-1 – Cantata and Sonata End-to-End Function Flow

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

All of the above-mentioned APIs are provided in the SDK together with accompanying Developer Guides. Please refer to these documents for more details and examples of particular functional APIs.

A.1 Integration of product specifications into the APIs.

The above-mentioned APIs are product-agnostic in that they serve as a business interaction level between the Buyer and the Seller, and they do not contain any product-specific information in their specifications. To pass product-specific information, an extension pattern must be used. This applies to four APIs that carry product-specific information: POQ, Quote, Product Order, and Product Inventory.

The extension hosting type in the API data model is "MEFProductConfiguration". The "@type" attribute of that type must be set to a value that uniquely identifies the product specification (Figure A1-2). A unique identifier for MEF standard product specifications is in URN format and is assigned by MEF. This identifier is provided as root schema "$id" and in product specification documentation. In this case, this will be one of:

- urn:mef:lso:spec:cantata-sonata:epl-evc:v1.0.0:all
- urn:mef:lso:spec:cantata-sonata:evpl-evc:v1.0.0:all
- urn:mef:lso:spec:cantata-sonata:eplan-evc:v1.0.0:all
- urn:mef:lso:spec:cantata-sonata:evplan-evc:v1.0.0:all
- urn:mef:lso:spec:cantata-sonata:eptree-evc:v1.0.0:all
- urn:mef:lso:spec:cantata-sonata:evptree-evc:v1.0.0:all
- urn:mef:lso:spec:cantata-sonata:eplan-evc-endpoint:v1.0.0:all
- urn:mef:lso:spec:cantata-sonata:evplan-evc-endpoint:v1.0.0:all
- urn:mef:lso:spec:cantata-sonata:eptree-evc-endpoint:v1.0.0:all
- urn:mef:lso:spec:cantata-sonata:evptree-evc-endpoint:v1.0.0:all
- urn:mef:lso:spec:cantata-sonata:carrier-ethernet-subscriber-uni:v1.0.0:all

Use of non-MEF standard product definitions is allowed. In such a case the schema identifier must be agreed upon between the Buyer and the Seller.

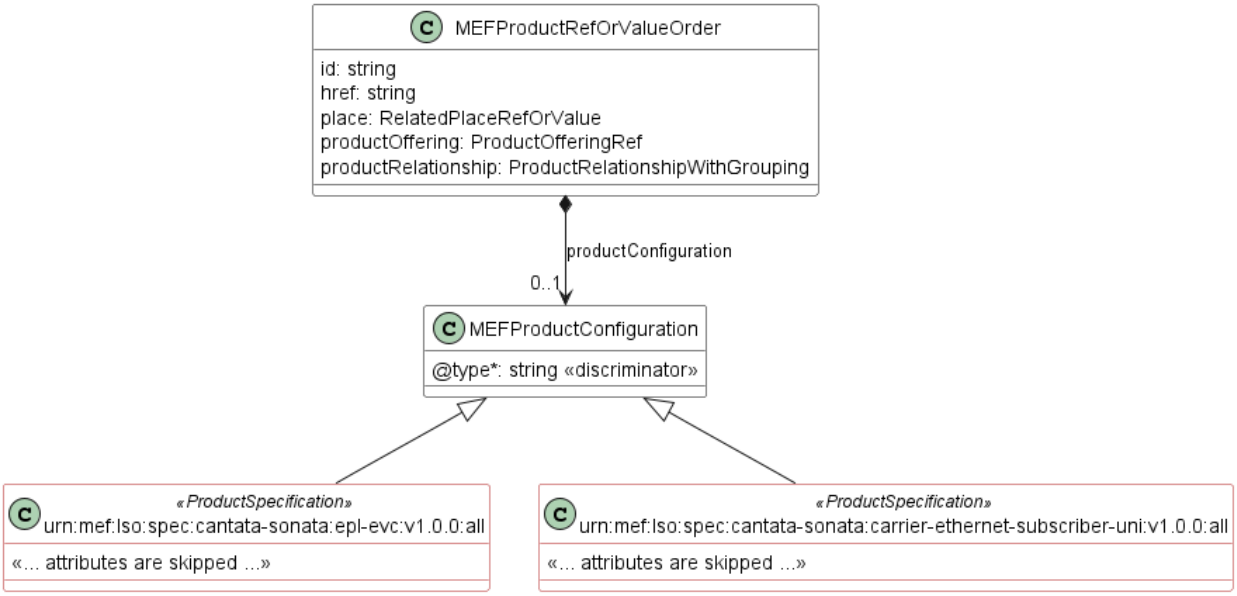

Figure A1-2 – The Extension Pattern

Product specifications are provided as JSON schemas without the "MEFProductConfiguration" context. Product-specific attributes are introduced via the "MEFProductRefOrValue" (defined by the Buyer). This entity has the "productConfiguration" attribute of type "MEFProductConfiguration" which is used as an extension point for product-specific attributes. The example result of such binding in a request payload may look like this (for POQ):

```
{
  "instantSyncQualification": true,
  "externalId": "BuyerPoq-00002a",
  "provideAlternative": false,
  "projectId": "BuyerProject2a",
  "productOfferingQualificationItem": [
    {
      "id": "item-001",
      "action": "add",
      "product": {
        "productOffering": {
          "id": "000073"
        },
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:cantata-sonata:epl-evc:v1.0.0:all"
          "listOfCosNames": ["low"],
          "availableMegLevel": "6",
          "carrierEthernetSls": [],
          "maximumFrameSize": 1522,
          "evcEndPointA": {},
          "evcEndPointZ": {}
<<the rest of the attributes omitted>>
          …
        }
```

POQ API part

Subscriber Ethernet Product part

```
      }
    }
  ]
}
```

A.2 action: add

This section guides through all the steps of Sonata and Cantata APIs that need to be performed to successfully order a Subscriber Ethernet product.

Note: Sellers are free to mandate some of these steps.

Note: As the examples of particular steps in many cases will replicate the product-specific information, in some of the snippets some parts of it will be omitted for better readability.

There are common rules for all request items for creation requests (POQ, Quote, Order):

- "item.action" must be set to "add"
- "item.product.id" must not be provided
- "product.productConfiguration" must contain all desired configurations.

A.2.1 Use Case 1: Address Validation

For detailed guidance on how to use the Address Validation API, please refer to MEF 121 [5].

The first step of the process is the Address Validation. This step aligns the address representation between the Buyer and the Seller. This is to overcome the very common problem of different address representations in various countries and systems. The Buyer sends a representation of the address that is intended to be used in further steps (most likely an installation place). The question is "Dear Seller – do you recognize and understand this address?". Additionally, the Buyer may also ask the Seller to provide alternatives if there is no clear match. If the provided address is found, the Seller responds with a matching address in "bestMatchGeorgraphicAddress". This can include an id that can be used in further steps to avoid the need for Address resolution.

Note: The Seller doesn't need to provide the Id of the returned Address, yet it is recommended.

Note: The Seller's response might come with some enhancements to the Address. It is up to the Seller's discretion what makes the best match and an alternative.

The Buyer in the request places one of 4 possible representations of the Address (FieldedAddress, FormattedAddress, MEFGeographicPoint, or GeographicAddressLabel). The following Figure and snippet present an example request:

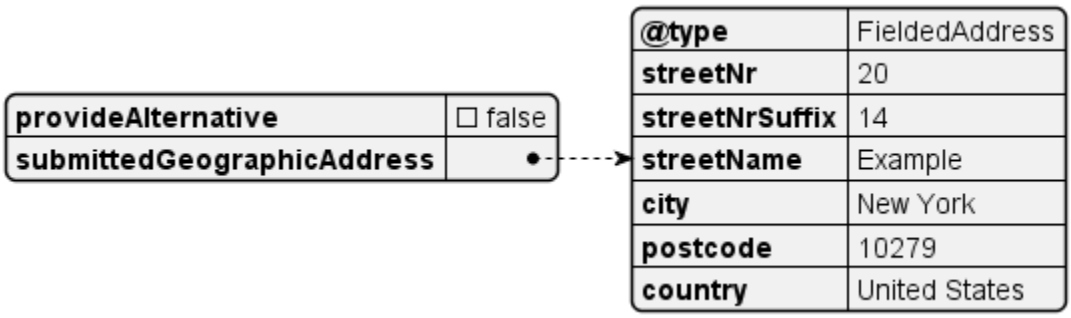

Figure A1-3 – UC1: Address Validation request

Example Address Validation Request:

```
{
  “provideAlternative”: false,
  “submittedGeographicAddress”: {
    “@type”: “FieldedAddress”,
    “streetNr”: “20”,
    “streetNrSuffix”: “14”,
    “streetName”: “Example”,
    “city”: “New York”,
    “postcode”: “10279”,
    “country”: “United States”
  }
}
```

In the response, the Seller repeats the submitted address for reference and populates the “bestMatchGeographicAddress” and/or the “alternateGeographicAddress”. In the example, the Seller matches the best-match address, which has a little more details than the one in the request. The Seller also provides the address id (“NewYorkAddress-id-1”) that the Buyer will refer to in later steps.

Note: The identifiers must be unique. In all examples, the identifiers are human-readable to make it easier to read and match across the use cases.

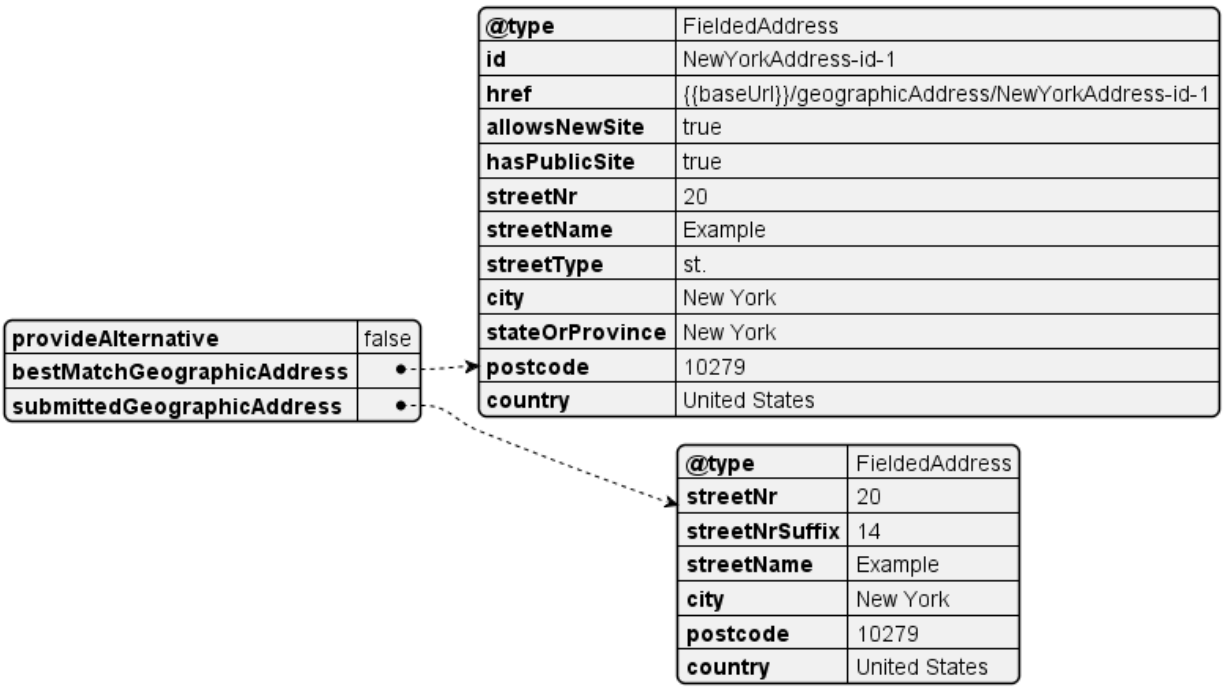

Figure A1-4 – UC1: Address Validation response

Seller's response:

```
{
  "provideAlternative": "false",
  "bestMatchGeographicAddress": {
    "@type": "FieldedAddress",
    "id": "NewYorkAddress-id-1",
    "href": "{{baseUrl}}/geographicAddress/NewYorkAddress-id-1",
    "allowsNewSite": "true",
    "hasPublicSite": "true",
    "streetNr": "20",
    "streetName": "Example",
    "streetType": "st.",
    "city": "New York",
    "stateOrProvince": "New York",
    "postcode": "10279",
    "country": "United States"
  },
  "submittedGeographicAddress": {
    "@type": "FieldedAddress",
    "streetNr": "20",
    "streetNrSuffix": "14",
    "streetName": "Example",
    "city": "New York",
    "postcode": "10279",
    "country": "United States"
  }
}
```


A.2.2 Use Case 2a: POQ - new EPL, new UNIs

For detailed guidance on how to use the Product Offering Qualification API, please refer to MEF 87 [3].

The Product Offering Qualification step is designed for the Buyer to ask the question "Dear Seller, are you able to provide a certain product (based on "productOffering" and "productConfiguration") at a given location"? The Seller responds with qualification confidence:

- green - The Seller has high confidence that this Product can be delivered,
- yellow - The Seller believes they can deliver the Product but is not highly confident,
- red - The Seller cannot deliver the Product as specified.

In case of yellow or red, additionally, the Seller may return (if requested) an alternative Product Offering, that might fulfill the Buyer's needs.

The topology of the Use Case 2 is presented in Figure A1-5.

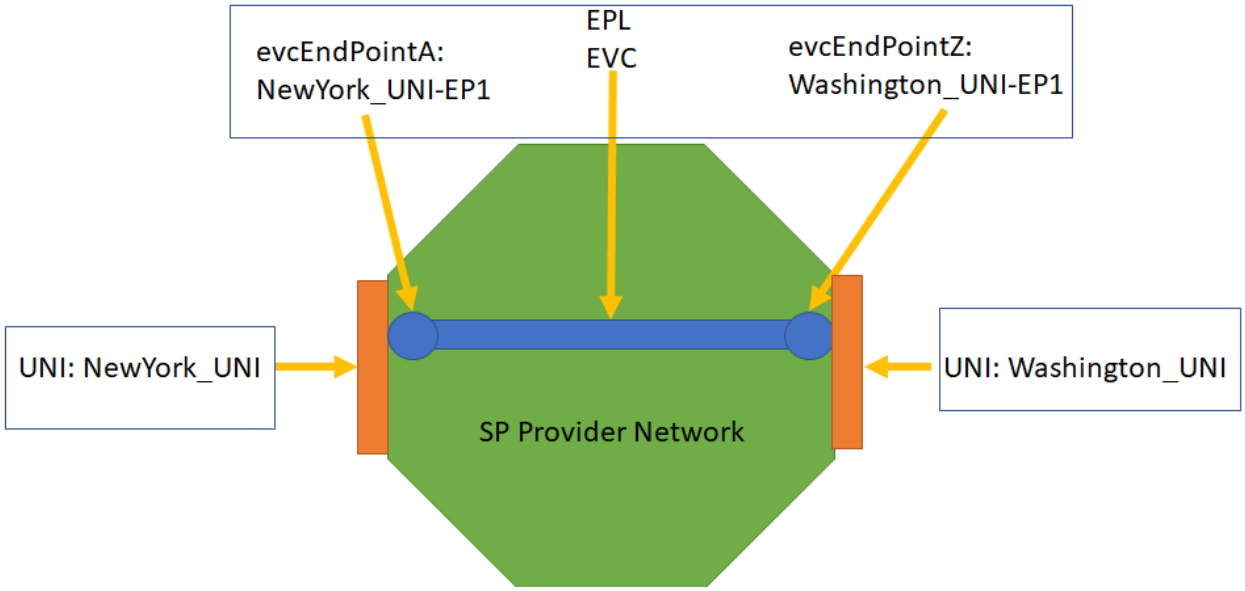

Figure A1-5 – UC2a: EPL Setup Diagram

This setup involves:

- Creation of the UNIs
    - id=”NewYork_UNI”
        - place: New York (Address id acquired in Use Case 1)
    - id=”Washington_UNI”
        - place: Washington (Address id acquired in Use Case 1)
- Creation of the EPL including:
    - configuration of a new EVC End Point with id=”NewYork_UNI-EP1”, at the UNI with id=”NewYork_UNI”, which is also created within the same request.
    - configuration of a new EVC End Point with id=”Washington_UNI-EP1”, at the UNI with id=”Washington_UNI”, which is also created within the same request.

The diagram aggregates the scope of a particular product configuration into rectangles. This is to stress that the EVC End Points are parts of the EPL configuration. They are not individually orderable products (this is the case in point-to-point connections).

It is very important to understand the pattern of integrating the product configuration (so-called “payload”) with the functional product-agnostic API (“envelope”). As explained in MEF 125 [7], the EPL product model is composed of 2 elements (products):

- the EPL EVC itself. It contains the “evcEndPointA” and “evcEndPointZ” attributes, which carry EVC End Point configuration information,
- the UNIs

The information about one single product is carried within the Product Offering Qualification (POQ) API by a single "productOfferingQualificationItem" being subject to qualification. One POQ Request can carry more than one POQ Items, that may or may not be related to each other.

There are 2 ways to reference products:

- existing Products – present in the Inventory at the moment of issuing the request, to which the Buyer has the "product.id". These must be referenced by "productOfferingQualificationItem.product." with appropriate "product.id" and "relationshipType". The Product Specification defines what roles must be used during referencing other products as specified in Chapter 13.

- newly created or modified products – ones being created or modified by other POQ Items in the same POQ request, so there is a relation between the Items within a POQ. These must be referenced using the "productOfferingQualificationItem.qualificationItemRelationship" by the target Item "id" and the "relationshipType" (CONNECTS_TO_UNI_A) and (CONNECTS_TO_UNI_Z).

In this use case, both the EPL and the UNI products are created or, to be more precise, a request to qualify if the creation of both of them is possible. Since three products are being subject to qualification, the POQ request contains 3 items with "action=add". The EPL POQ Item has 2 relations:

- to the first UNI (NewYork_UNI), which is being qualified in the same request – by "productOfferingQualificationItem.qualificationItemRelationship"

- to the second UNI (Washington_UNI), which is being qualified in the same request – by "productOfferingQualificationItem.qualificationItemRelationship"

An instance diagram in Figure A1-6 shows an extracted part from the request, to present the most important integration-related attributes. The product configurations attached to a POQ request are highlighted with green color, and the product relations are highlighted with a bold font.

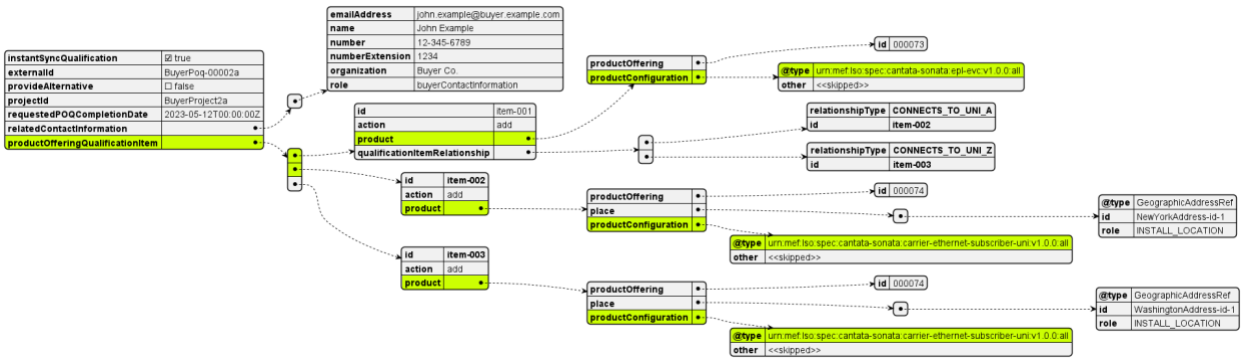

Figure A1-6 – UC2a: POQ Request, envelope part

The sequence diagram below (Figure A1-7) shows a set of logical steps for building the POQ request:

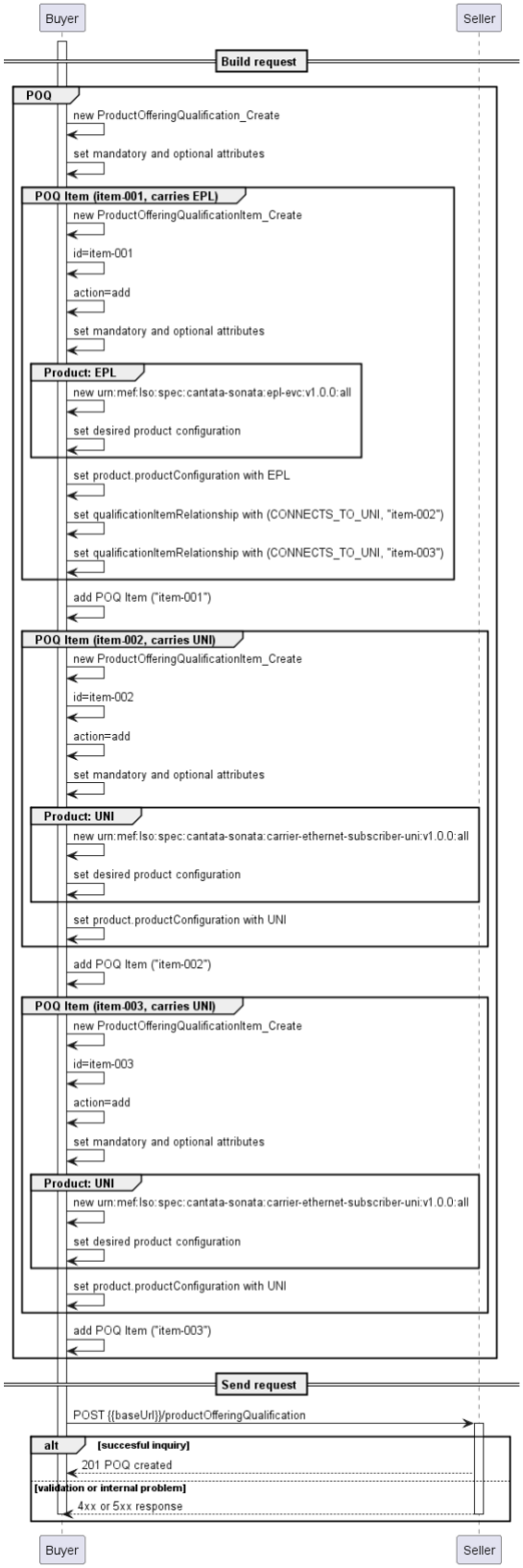

Figure A1-7 – UC2a: POQ request building steps

The instance diagram for the whole EPL configuration is too big to be presented so it is split and presented in parts. Figure A1-8 shows the main EPL attributes. This diagram is attached to Figure A1-6 as the node with "@type=urn:mef:lso:spec:cantata-sonata:epl-evc:v1.0.0:all". The attributes that are skipped on this level are marked with a "<<skipped>>" label and will be presented on the next diagrams.

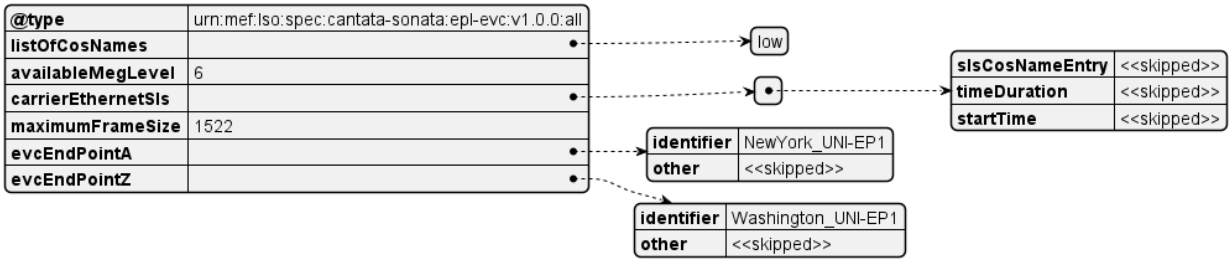

Figure A1-8 – UC2a: EPL basic attributes

The structures defining the "carrierEthernetSls", "evcEndPointA", and the "evcEndPointZ" are complex and presented in Figure A1-9 and Figure A1-10:

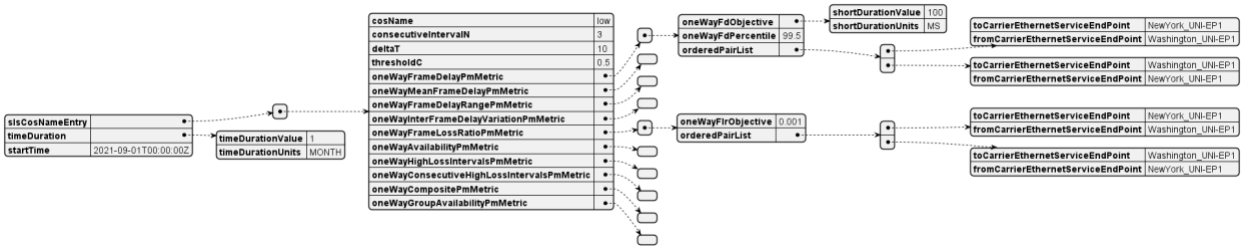

Figure A1-9 – UC2a: EPL Carrier Ethernet SLS

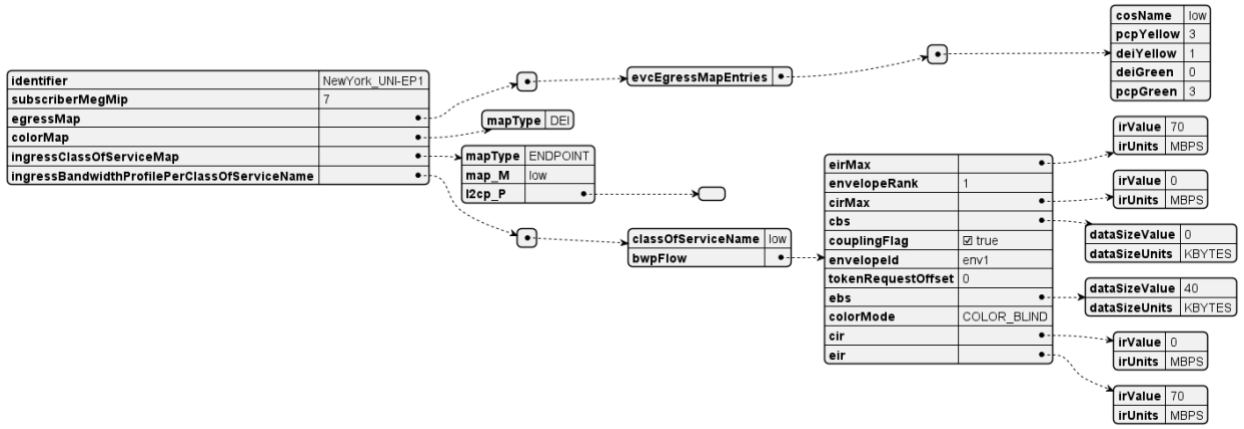

Figure A1-10 – UC2a: EPL EVC End Point

Figure A1-11 in this use case presents the UNI product configuration:

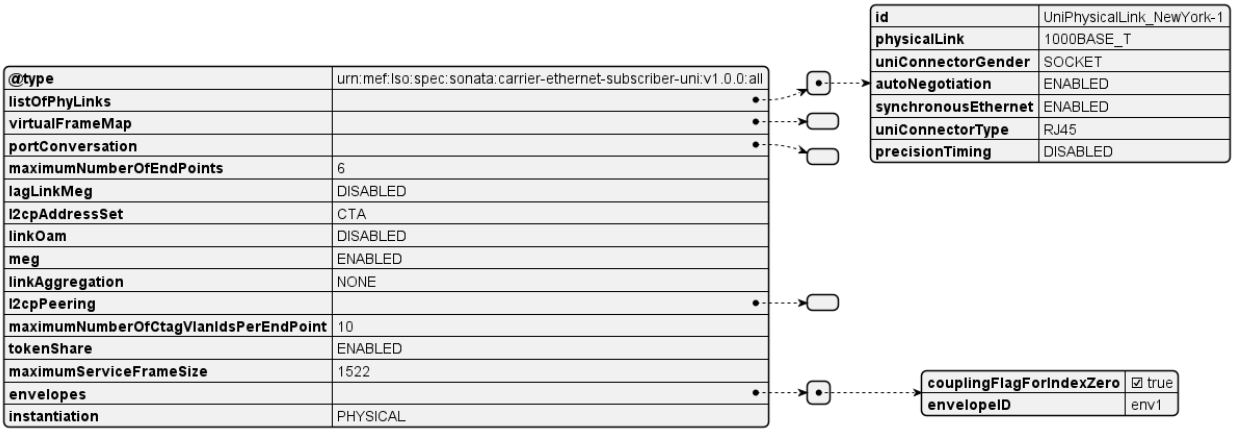

Figure A1-11 – UC2a: UNI

A full POQ Request example:


```
{
  "instantSyncQualification": false,
  "externalId": "BuyerPoq-00002a",
  "provideAlternative": false,
  "requestedPOQCompletionDate": "2023-10-12T00:00:00Z",
  "projectId": "BuyerProject2a",
  "relatedContactInformation": [
    {
      "emailAddress": "john.example@buyer.com",
      "name": "John Example",
      "number": "12-345-6789",
      "numberExtension": "1234",
      "organization": "Buyer Co.",
      "role": "buyerContactInformation"
    }
  ],
  "productOfferingQualificationItem": [
    {
      "id": "item-001",
      "action": "add",
      "qualificationItemRelationship": [
        {
          "relationshipType": "CONNECTS_TO_UNI_A",
          "id": "item-002"
        },
        {
          "relationshipType": "CONNECTS_TO_UNI_Z",
          "id": "item-003"
        }
      ],
```

```
      "product": {
        "productOffering": {
          "id": "000073"
        },
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:cantata-sonata:epl-evc:v1.0.0:all",
          "listOfCosNames": ["low"],
          "availableMegLevel": "6",
          "carrierEthernetSls": [
            {
              "slsCosNameEntry": [
                {
                  "cosName": "low",
                  "consecutiveIntervalN": 3,
                  "deltaT": 10,
                  "thresholdC": 0.5,
                  "oneWayFrameDelayPmMetric": [
                    {
                      "oneWayFdObjective": {
                        "shortDurationValue": 100,
                        "shortDurationUnits": "MS"
                      },
                      "oneWayFdPercentile": 99.5,
                      "orderedPairList": [
                        {
                          "toCarrierEthernetServiceEndPoint": "NewYork_UNI-EP1",
                          "fromCarrierEthernetServiceEndPoint": "Washington_UNI-
EP1"
                        },
                        {
                          "toCarrierEthernetServiceEndPoint": "Washington_UNI-
EP1",
                          "fromCarrierEthernetServiceEndPoint": "NewYork_UNI-EP1"
                        }
                      ]
                    }
                  ],
                  "oneWayMeanFrameDelayPmMetric": [],
                  "oneWayFrameDelayRangePmMetric": [],
                  "oneWayInterFrameDelayVariationPmMetric": [],
                  "oneWayFrameLossRatioPmMetric": [
                    {
                      "oneWayFlrObjective": 0.001,
                      "orderedPairList": [
                        {
                          "toCarrierEthernetServiceEndPoint": "NewYork_UNI-EP1",
```

```
                        "fromCarrierEthernetServiceEndPoint": "Washington_UNI-
EP1"
                      },
                      {
                        "toCarrierEthernetServiceEndPoint": "Washington_UNI-
EP1",
                        "fromCarrierEthernetServiceEndPoint": "NewYork_UNI-EP1"
                      }
                    ]
                  }
                ],
                "oneWayAvailabilityPmMetric": [],
                "oneWayHighLossIntervalsPmMetric": [],
                "oneWayConsecutiveHighLossIntervalsPmMetric": [],
                "oneWayCompositePmMetric": [],
                "oneWayGroupAvailabilityPmMetric": []
              }
            ],
            "timeDuration": {
              "timeDurationValue": 1,
              "timeDurationUnits": "MONTH"
            },
            "startTime": "2022-10-12T00:00:00Z"
          }
        ],
        "maximumFrameSize": 1522,
        "evcEndPointA": {
          "identifier": "NewYork_UNI-EP1",
          "subscriberMegMip": "7",
          "egressMap": [
            {
              "evcEgressMapEntries": [
                {
                  "cosName": "low",
                  "pcpYellow": "3",
                  "deiYellow": "1",
                  "deiGreen": "0",
                  "pcpGreen": "3"
                }
              ]
            }
          ],
          "colorMap": {
            "mapType": "DEI"
          },
          "ingressClassOfServiceMap": {
```

```
      "mapType": "ENDPOINT",
      "map_M": "low",
      "l2cp_P": []
    },
    "ingressBandwidthProfilePerClassOfServiceName": [
      {
        "classOfServiceName": "low",
        "bwpFlow": {
          "eirMax": {
            "irValue": 70,
            "irUnits": "MBPS"
          },
          "envelopeRank": 1,
          "cirMax": {
            "irValue": 0,
            "irUnits": "MBPS"
          },
          "cbs": {
            "dataSizeValue": 0,
            "dataSizeUnits": "KBYTES"
          },
          "couplingFlag": true,
          "envelopeId": "env1",
          "tokenRequestOffset": 0,
          "ebs": {
            "dataSizeValue": 40,
            "dataSizeUnits": "KBYTES"
          },
          "colorMode": "COLOR_BLIND",
          "cir": {
            "irValue": 0,
            "irUnits": "MBPS"
          },
          "eir": {
            "irValue": 70,
            "irUnits": "MBPS"
          }
        }
      }
    ]
  },
  "evcEndPointZ": {
    "identifier": "Washington_UNI-EP1",
    "subscriberMegMip": "7",
    "egressMap": [
      {
```

```
      "evcEgressMapEntries": [
        {
          "cosName": "low",
          "pcpYellow": "3",
          "deiYellow": "1",
          "deiGreen": "0",
          "pcpGreen": "3"
        }
      ]
    }
  ],
  "colorMap": {
    "mapType": "DEI"
  },
  "ingressClassOfServiceMap": {
    "mapType": "ENDPOINT",
    "map_M": "low",
    "l2cp_P": []
  },
  "ingressBandwidthProfilePerClassOfServiceName": [
    {
      "classOfServiceName": "low",
      "bwpFlow": {
        "eirMax": {
          "irValue": 70,
          "irUnits": "MBPS"
        },
        "envelopeRank": 1,
        "cirMax": {
          "irValue": 0,
          "irUnits": "MBPS"
        },
        "cbs": {
          "dataSizeValue": 0,
          "dataSizeUnits": "KBYTES"
        },
        "couplingFlag": true,
        "envelopeId": "env1",
        "tokenRequestOffset": 0,
        "ebs": {
          "dataSizeValue": 40,
          "dataSizeUnits": "KBYTES"
        },
        "colorMode": "COLOR_BLIND",
        "cir": {
          "irValue": 0,
```

```
                    "irUnits": "MBPS"
                  },
                  "eir": {
                    "irValue": 70,
                    "irUnits": "MBPS"
                  }
                }
              }
            ]
          }
        }
      }
    },
    {
      "id": "item-002",
      "action": "add",
      "relatedContactInformation": [
        {
          "number": "+12-345-678-90",
          "emailAddress": "LocationContact@example.com",
          "role": "locationContact",
          "name": "Location Contact"
        }
      ],
      "product": {
        "productOffering": {
          "id": "000074"
        },
        "place": [
          {
            "@type": "GeographicAddressRef",
            "id": "NewYorkAddress-id-1",
            "role": "INSTALL_LOCATION"
          }
        ],
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:sonata:carrier-ethernet-subscriber-
uni:v1.0.0:all",
          "listOfPhyLinks": [
            {
              "id": "UniPhysicalLink_NewYork-1",
              "physicalLink": "1000BASE_T",
              "uniConnectorGender": "SOCKET",
              "autoNegotiation": "ENABLED",
              "synchronousEthernet": "ENABLED",
              "uniConnectorType": "RJ45",
```

```
            "precisionTiming": "DISABLED"
          }
        ],
        "virtualFrameMap": [],
        "portConversation": [],
        "maximumNumberOfEndPoints": 6,
        "lagLinkMeg": "DISABLED",
        "l2cpAddressSet": "CTA",
        "linkOam": "DISABLED",
        "meg": "ENABLED",
        "linkAggregation": "NONE",
        "l2cpPeering": {},
        "maximumNumberOfCtagVlanIdsPerEndPoint": 10,
        "tokenShare": "ENABLED",
        "maximumServiceFrameSize": 1522,
        "envelopes": [
          {
            "couplingFlagForIndexZero": true,
            "envelopeID": "env1"
          }
        ],
        "instantiation": "PHYSICAL"
      }
    }
  },
  {
    "id": "item-003",
    "action": "add",
    "relatedContactInformation": [
      {
        "number": "+12-345-678-90",
        "emailAddress": "LocationContact@example.com",
        "role": "locationContact",
        "name": "Location Contact"
      }
    ],
    "product": {
      "productOffering": {
        "id": "000074"
      },
      "place": [
        {
          "@type": "GeographicAddressRef",
          "id": "WashingtonAddress-id-1",
          "role": "INSTALL_LOCATION"
        }
```

```
        ],
        "productConfiguration": {
          "@type": "urn:mef:lso:spec:sonata:carrier-ethernet-subscriber-
uni:v1.0.0:all",
          "listOfPhyLinks": [
            {
              "id": "UniPhysicalLink_Washington-1",
              "physicalLink": "1000BASE_T",
              "uniConnectorGender": "SOCKET",
              "autoNegotiation": "ENABLED",
              "synchronousEthernet": "ENABLED",
              "uniConnectorType": "RJ45",
              "precisionTiming": "DISABLED"
            }
          ],
          "virtualFrameMap": [],
          "portConversation": [],
          "maximumNumberOfEndPoints": 6,
          "lagLinkMeg": "DISABLED",
          "l2cpAddressSet": "CTA",
          "linkOam": "DISABLED",
          "meg": "ENABLED",
          "linkAggregation": "NONE",
          "l2cpPeering": {},
          "maximumNumberOfCtagVlanIdsPerEndPoint": 4094,
          "tokenShare": "ENABLED",
          "maximumServiceFrameSize": 1522,
          "envelopes": [
            {
              "couplingFlagForIndexZero": true,
              "envelopeID": "env1"
            }
          ],
          "instantiation": "PHYSICAL"
        }
      }
    }
  ]
}
```


A.2.3 Use Case 2b: POQ - new EVPL, new UNIs

A detailed description of the Product Offering Qualification envelope part is covered in Use Case 2a. This section will focus only on the unique features of the EVPL Product (in contrast to the EPL Product described above).

EPL and EVPL are very similar, however, there are a few differences. EPL is port-based so the whole port is dedicated to the EPL EVC. The EVPL is VLAN-based and several Virtual Private Services such as EVPL EVCs can share the same UNI.

The topology of the use case is identical. Note that the identifiers overlap with the previous use case for the sake of ease of comparison. In real-life EPL and EVPL can not share the same UNI.

The setup of Use Case 2b is presented in Figure A1-12.

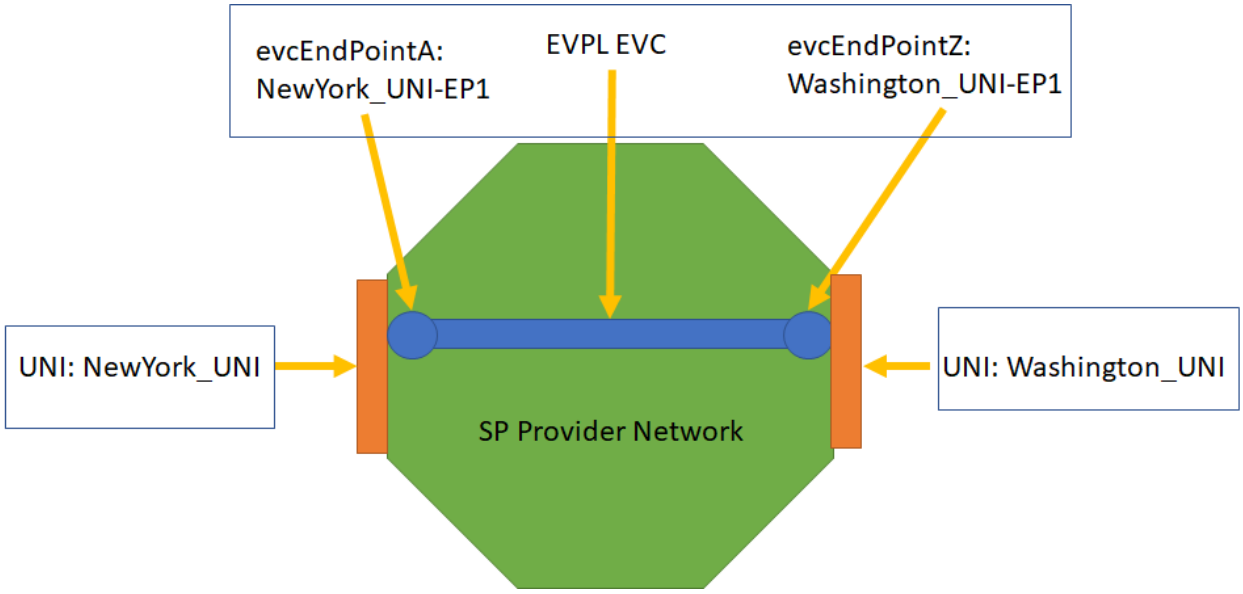

Figure A1-12 – UC2b: EVPL Topology Diagram

This topology involves:

- Creation of two UNIs
  - id=”NewYork_UNI”
  - id=”Washington_UNI”
- Creation of the EVPL, including:
  - configuration of a new EVC End Point with id=”NewYork_UNI-EP1”, at the UNI with id=”NewYork_UNI”, which is also created within the same request.
  - configuration of a new EVC End Point with id=”Washington_UNI-EP1”, at the UNI with id=”Washington _UNI”, which is also created within the same request.

Figure A1-13 shows the main EVPL attributes with highlighted differences compared to EPL.

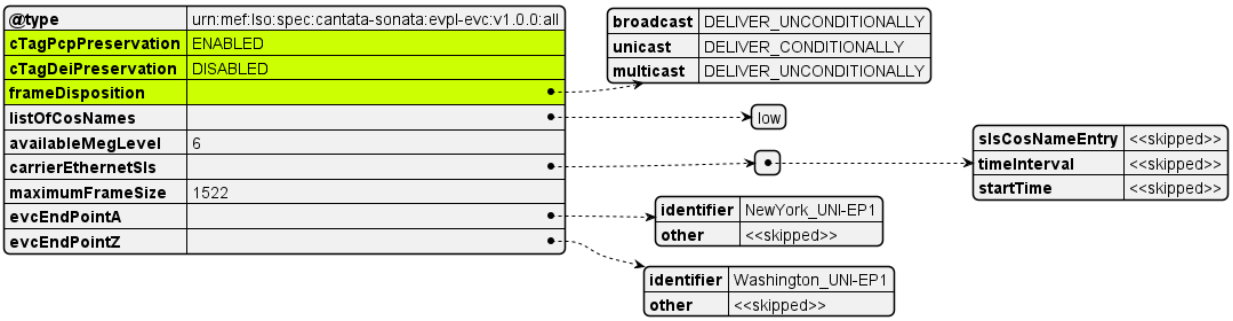

Figure A1-13 – UC2b: EVPL basic attributes

The structures defining the EVC End Point are complex and presented in the following figures:

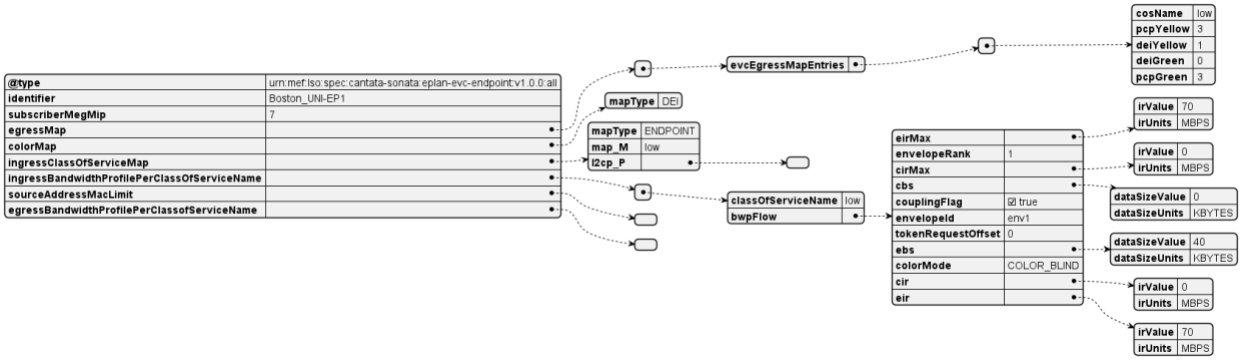

Figure A1-14 – UC2b: EVPL EVC End Point

A.2.4 Use Case 2c: POQ - new EP-LAN, new UNIs, new EVC End Points

This section describes the unique features of the EP-LAN technology.

The EP-LAN product model is composed of 3 elements (products):

- EP-LAN EVC
- UNIs
- EVC End Points

The difference between the EPL (Ethernet Private Line) and the EP-LAN (Ethernet Private LAN) is that an EPL is a point-to-point connection that always has exactly two endpoints. Thus parameters of the endpoints are covered within the EPL product parameter (as "evcEndPointA" and "evcEndPointZ"). EP-LAN is a multipoint connection so the number of EVC End Points is not restricted and may change during the product lifecycle. Thus the EVC End Points are separately orderable products with two product relationships pointing to an EVC and UNI.

The topology of Use Case 2c is presented in Figure A1-15.

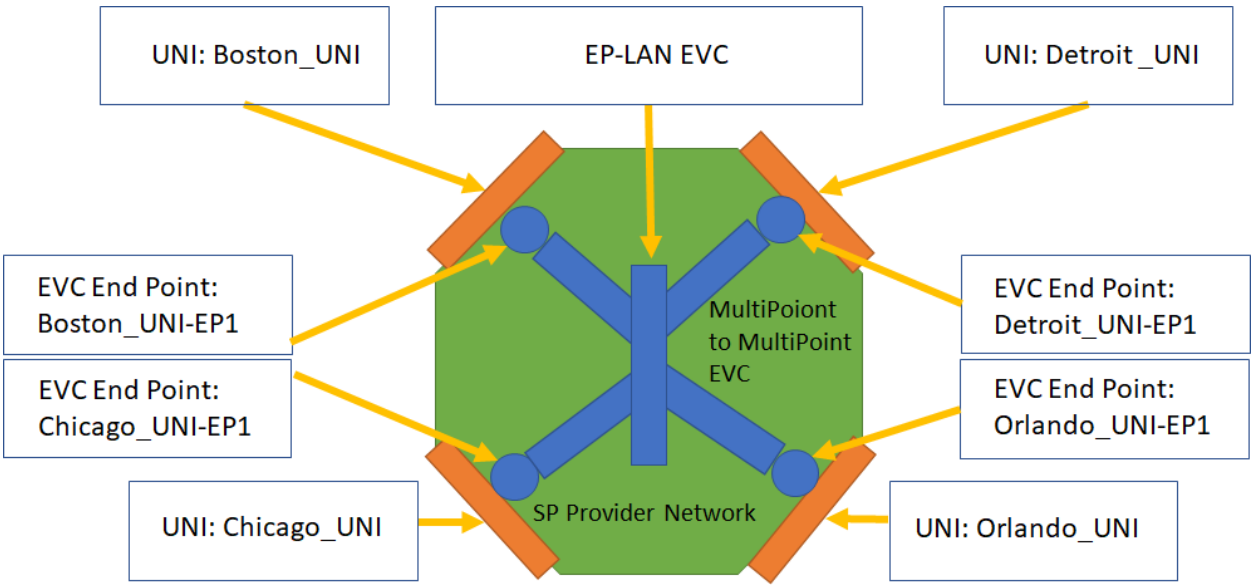

Figure A1-15 – UC2c: EP-LAN Setup Diagram

This topology involves:

- Creation of the EP-LAN,
- Creation of the UNIs
  - id=”Boston_UNI”
  - id=”Chicago_UNI”
  - id=”Detroit _UNI”
  - id=”Orlando_UNI”
- Creation of the EVC End Points:
  - id=”Boston_UNI-EP1”
  - id=”Chicago_UNI-EP1”
  - id=”Detroit_UNI-EP1”
  - id=”Orlando_UNI-EP1”

The diagram aggregates the scope of a particular product configuration into rectangles. However, unlike the EPL configuration, the EVC End Points are separate products.

An instance diagram in Figure A1-16 shows part of the request, to present the request structure-related attributes. The product URNs are highlighted with green color, and the product relations are highlighted with a bold font.

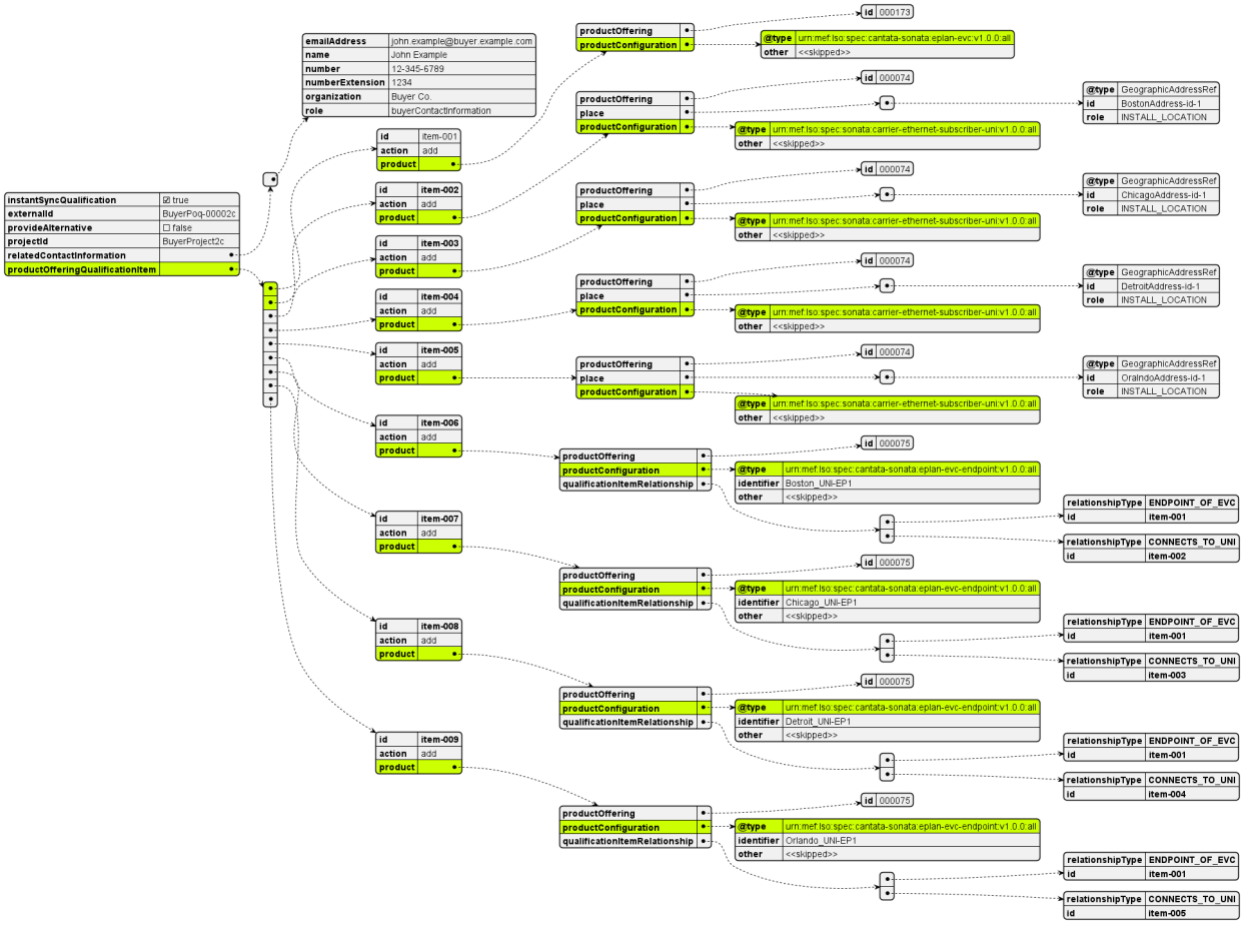

Figure A1-16 – UC2c: POQ Request, envelope part

The instance diagram for the whole EP-LAN configuration is too big to be presented so it is split and presented in parts. Figure A1-17 shows the basic EP-LAN attributes. This diagram is attached to Figure A1-16 as the node with "@type=urn:mef:lso:spec:cantata-sonata:eplan-evc:v1.0.0:all". The attributes that are skipped on this level are marked with a "<<skipped>>" label and will be presented on the next diagrams.

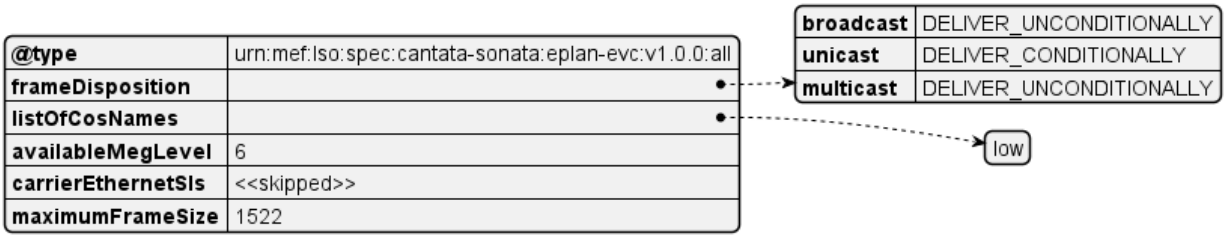

Figure A1-17 – UC2c: EP-LAN basic attributes

The "carrierEthernetSls" shows an example of an SLS specification between two EVC End Points: "Boston_UNI-EP1" and "Detroit_UNI-EP1":

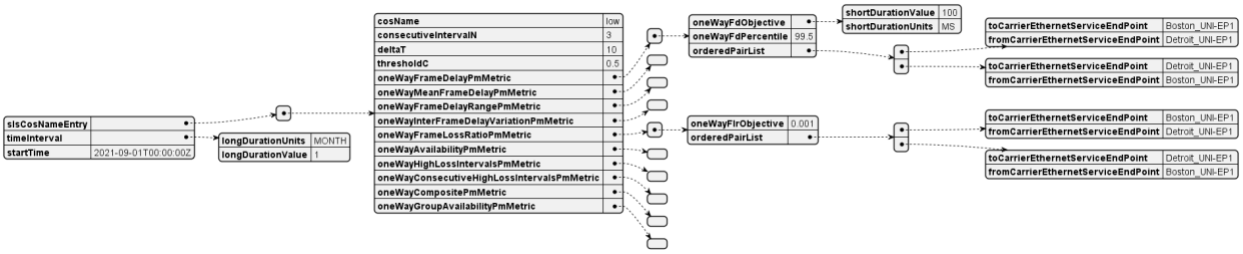

Figure A1-18 – UC2c: EP-LAN: Carrier Ethernet SLS

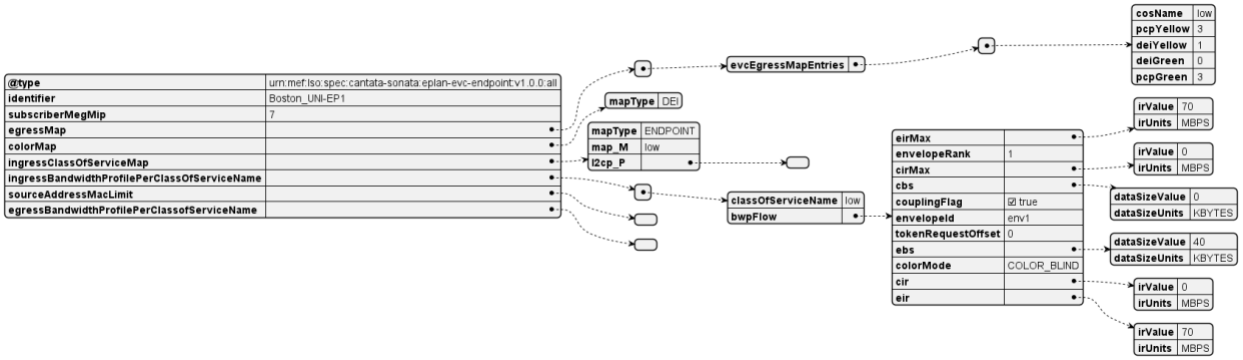

Figure A1-19 – UC2c: EP-LAN: EVC End Point

The last figure in this use case presents the UNI product configuration.

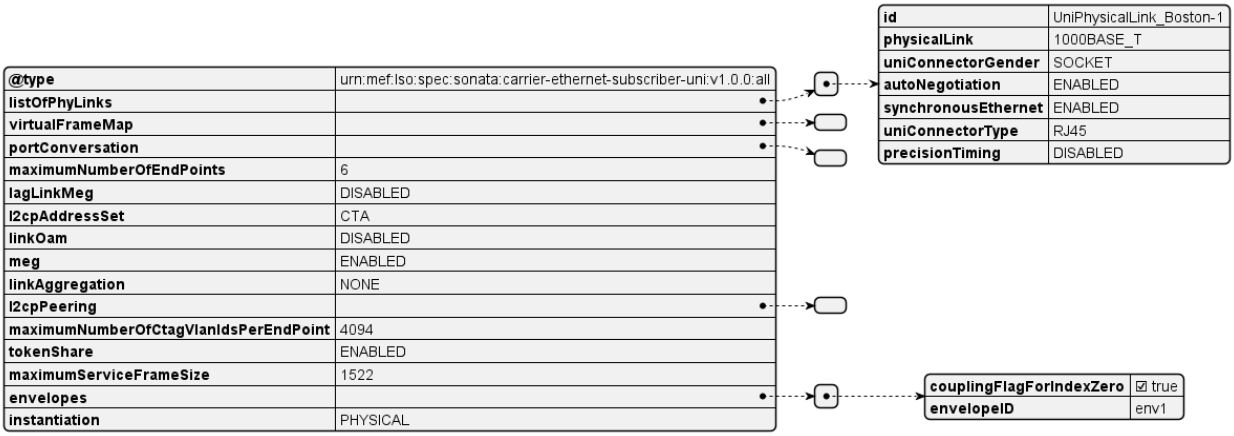

Figure A1-20 – UC2c: Subscriber Ethernet UNI

A.2.5 Use Case 2d: POQ - new EVP-LAN, new UNIs, new EVC End Points

The difference between EP-LAN and EVP-LAN is the same as one between the EPL and EVPL. This is because the Virtual one is VLAN based and the EVC End Points can share a UNI with other VLAN-based EVC End Points.

The topology of Use Case 2d is presented in Figure A1-21. It is similar to the one for EP-LAN.

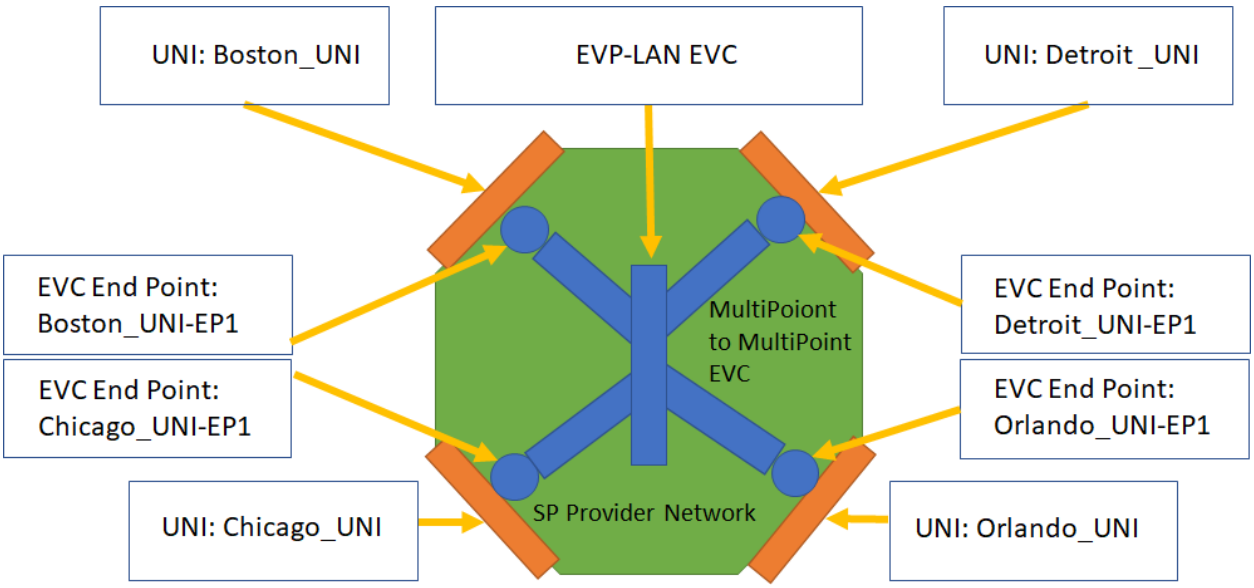

Figure A1-21 – UC2d: EVP-LAN Topology Diagram

This topology involves:

- Creation of the EVP-LAN,
- Creation of the UNIs
  - id=”Boston_UNI”
  - id=”Chicago_UNI”
  - id=”Detroit _UNI”
  - id=”Orlando_UNI”
- Creation of the EVC End Points:
  - id=”Boston_UNI-EP1”
  - id=”Chicago_UNI-EP1”
  - id=”Detroit_UNI-EP1”
  - id=”Orlando_UNI-EP1”

The instance diagram for the whole EVP-LAN configuration is too big to be presented as a whole so it is split and presented in parts. Figure A1-22 shows the basic EVP-LAN attributes. Differences to EP-LAN are highlighted.

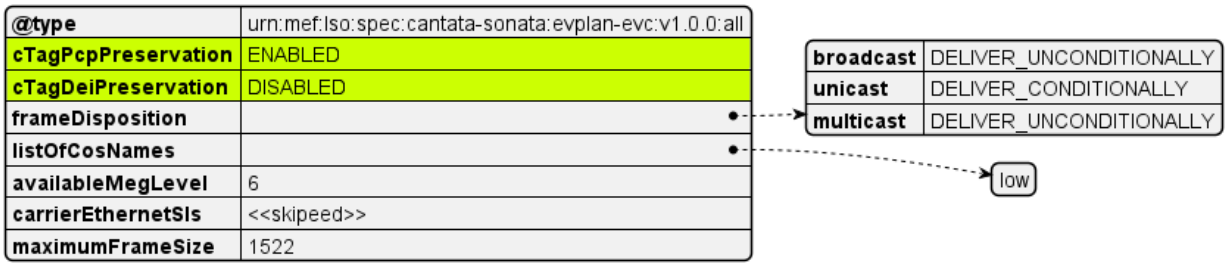

Figure A1-22 – UC2d: EVP-LAN basic attributes

The structures defining the "evcEndPoint" are complex and presented in the following figures:

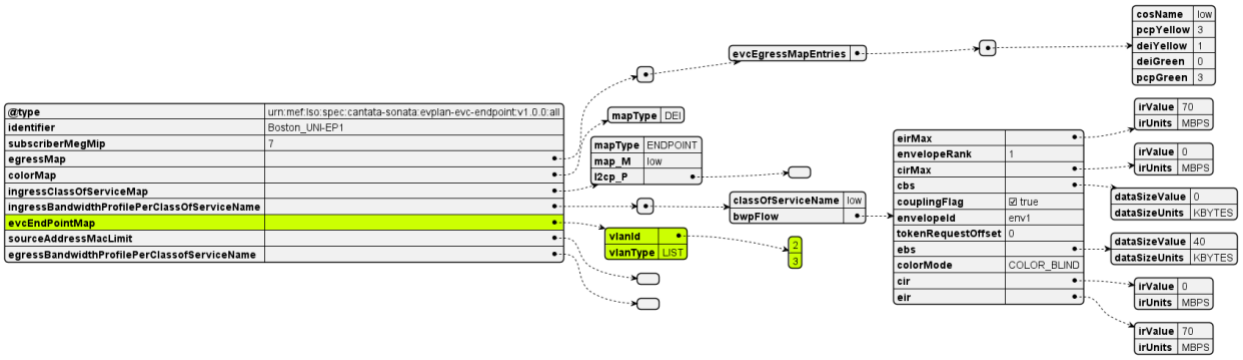

Figure A1-23 – UC2d: EVP-LAN EVC End Point

A.2.6 Use Case 2e: POQ - new EP-TREE, new UNIs, new EVC End Points

The topology of this Use Case is presented in Figure A1-24.

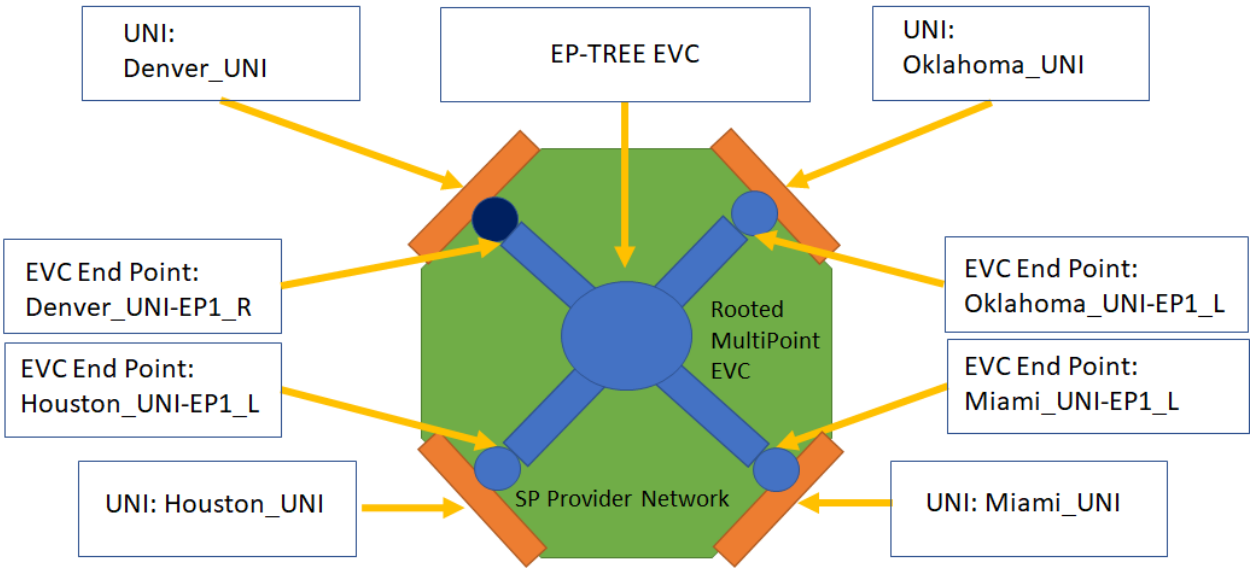

Figure A1-24 – UC2e: EP-TREE Setup Diagram

This setup involves:

- Creation of the EP-TREE

- Creation of the UNIs
    - id=”Denver_UNI”
    - id=”Houston_UNI”
    - id=”Oklahoma_UNI”
    - id=”Miami _UNI”
- Creation of the EVC End Points:
    - id=”Denver_UNI-EP1_R”
    - id=”Houston _UNI-EP1_L”
    - id=”Oklahoma_UNI-EP1_L”
    - id=”Miami_UNI-EP1_L”

EP-TREE is very similar to EP-LAN. The difference is that EP-TREE is a rooted-multipoint topology. EVC End Points used by EP-TREE are marked being either a ROOT or a LEAF (whereas all EP-LAN EVC Endpoint are ROOTs). This is marked by defining product or item relationship type towards the EP-TREE with “ROOT_ENDPOINT_OF_EVC” or “LEAF_ENDPOINT_OF_EVC”. Figure A1-25 shows a multi-item POQ request example of such configuration (truncated to present only item relationships).

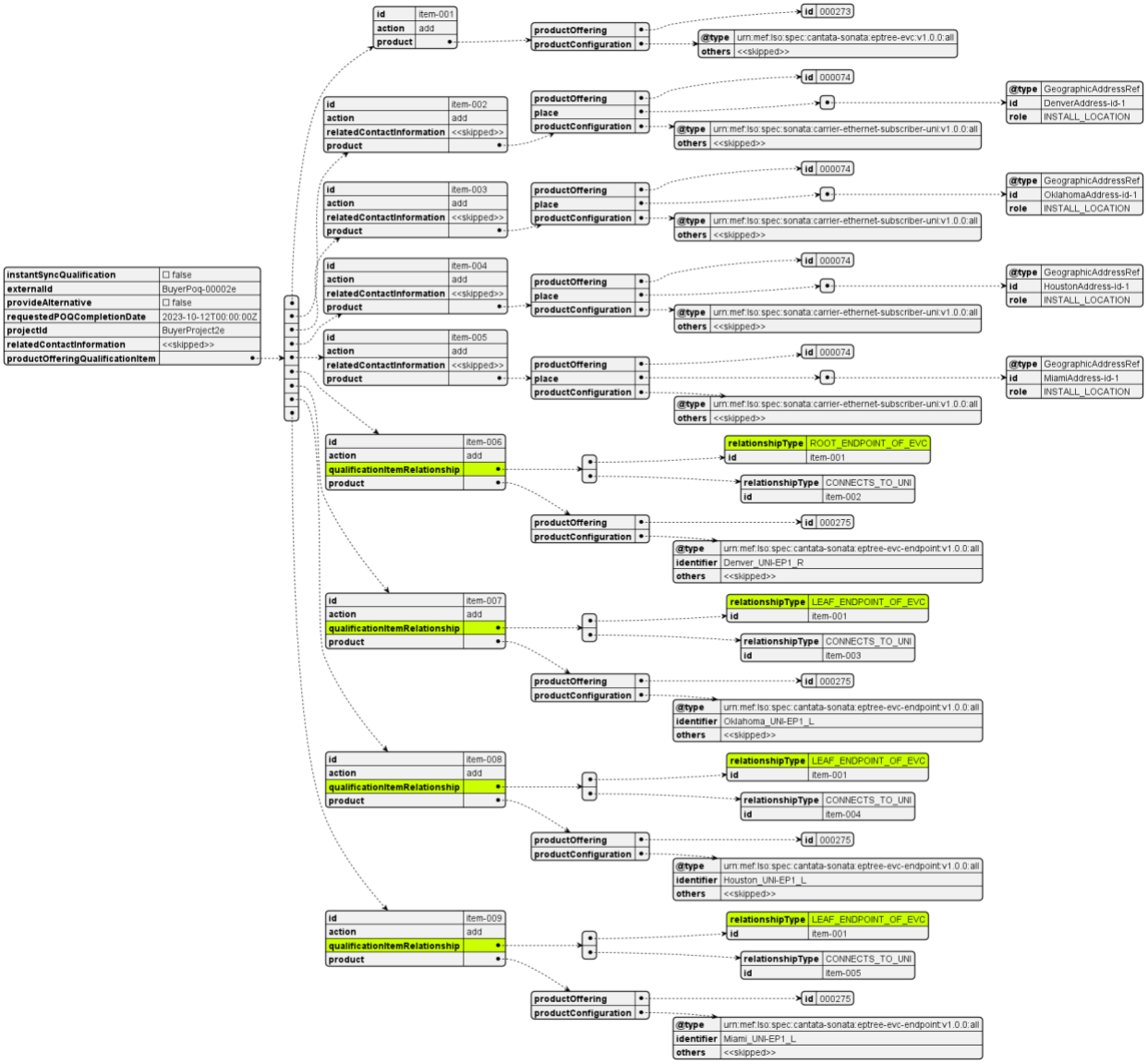

Figure A1-25 – UC2e: POQ Request, envelope part

A.2.7 Use Case 2f: POQ - new EVP-TREE, new UNIs, new EVC End Points

The difference between EP-TREE and EVP-TREE is the same as one between EP-LAN and EVP-LAN. This is because the Virtual one is VLAN based and the EVC End Points can share a UNI with other VLAN-based EVC End Points.

The topology of this Use Case is presented in Figure A1-26.

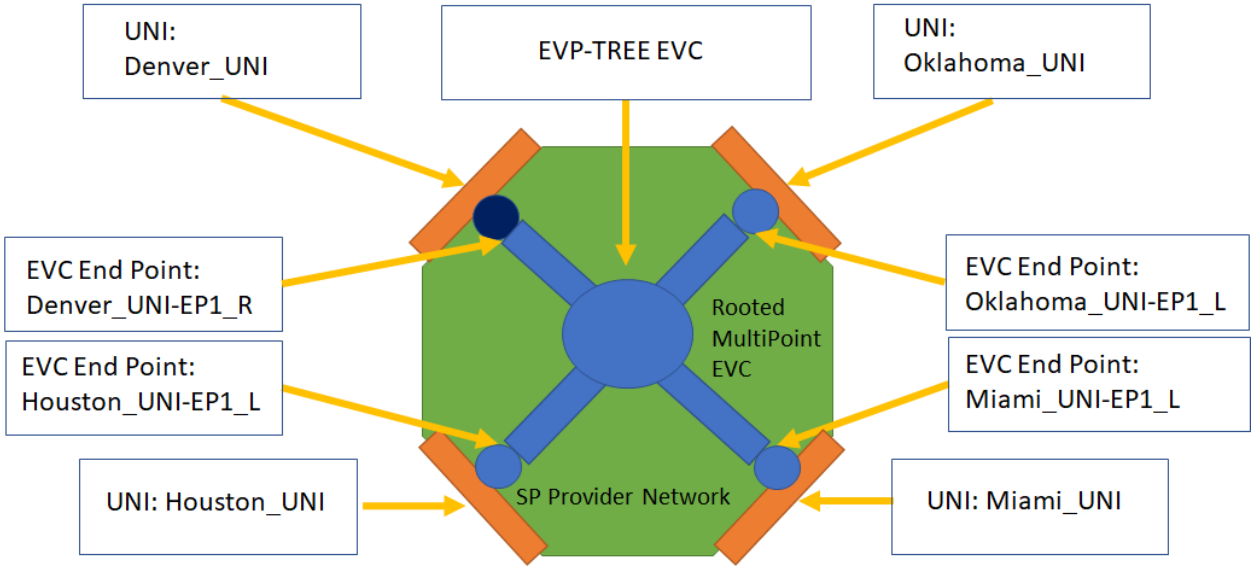

Figure A1-26 – UC2f: EVP-TREE Topology Diagram

This setup involves:

- Creation of the EVP-TREE
- Creation of the UNIs
  - id=”Denver_UNI”
  - id=”Houston_UNI”
  - id=”Oklahoma _UNI”
  - id=”Miami _UNI”
- Creation of the EVC End Points:
  - id=”Denver_UNI -EP1_R”
  - id=”Houston _UNI-EP1_L”
  - id=”Oklahoma _UNI-EP1_L”
  - id=”Miami _UNI-EP1_L”

There are no new specific configurations of EVP-TREE to cover so there are no more examples in this section. A full request example is available in the request collection.

A.2.8 Use Case 3a: POQ - new EPL, new UNIs

Use Case 3 presents the difference between EPL and EVPL. Sub use case 3a adds new EPL aside topology built-in Use Case 2a. It assumes the creation of a new EPL between the existing Address – “NewYorkAddress-id-1” and the new "SanFranciscoAddress-id-1". Since EPL is port-based – a new UNI in New York must be created.

The topology of the Use Case 3a is presented in Figure A1-27. Newly added products are highlighted with a frame.

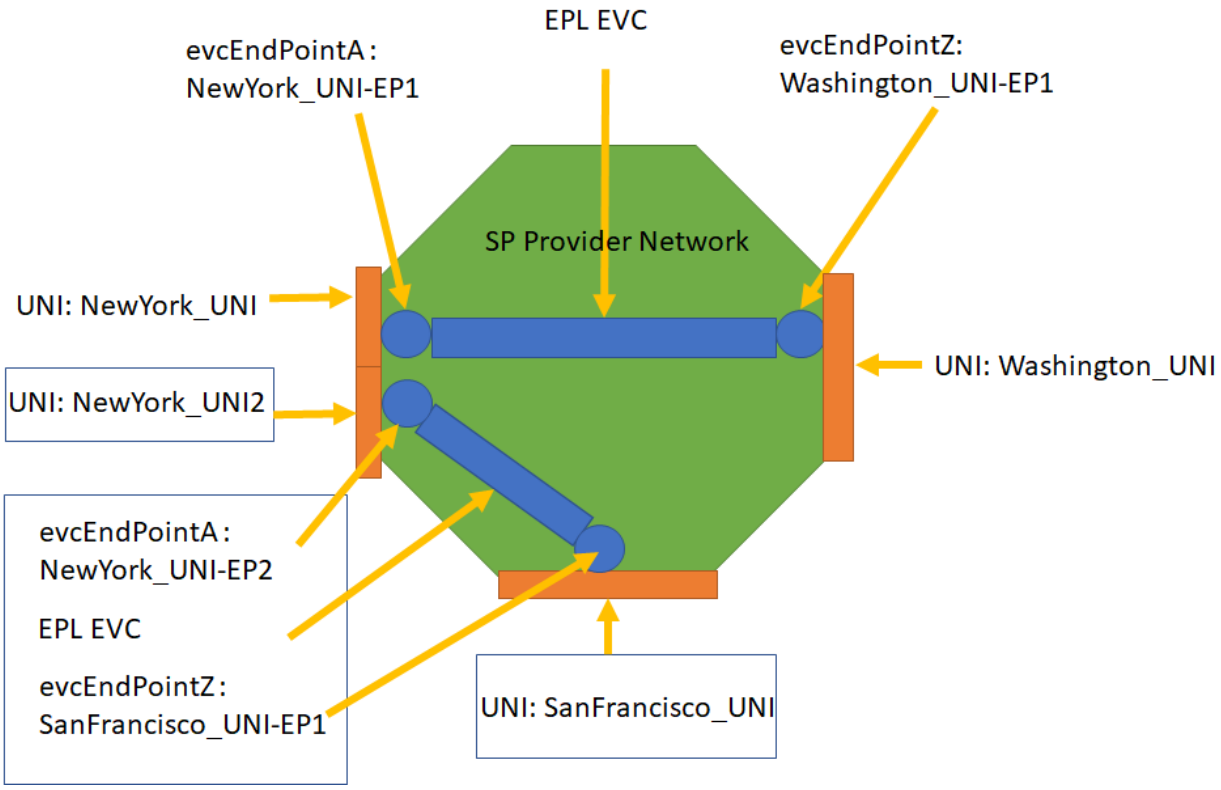

Figure A1-27 – UC3a: EPL – modified setup diagram

This topology involves:

- Creation of 2 new UNIs
    - id=”NewYork_UNI2”,
    - id=”SanFrancisco_UNI”,
- Creation of the new EPL, including:
    - configuration of a new EVC End Point with id=”NewYork_UNI-EP2”, at the new UNI with id=”NewYork_UNI2”
    - configuration of a new EVC End Point with id=”SanFrancisco_UNI-EP1”, at the new UNI with id=”SanFrancisco_UNI”

A structure of a request for a new EPL with UNIs is presented in Figure A1-28:

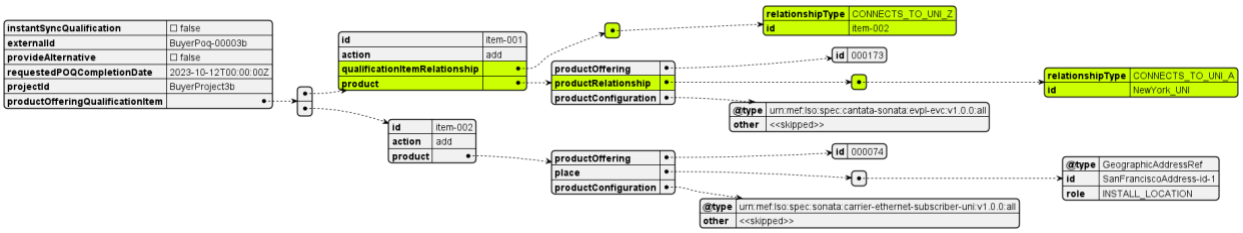

Figure A1-28 – UC3a: EPL relationships

A.2.9 Use Case 3b: POQ - new EVPL, existing UNI, new UNI

EVPL is a point-to-point Ethernet Service in which frames are mapped to the Service End Point based on VLAN IDs. This allows multiple "Virtual Private" Services (each based on a different set of VLAN IDs) to terminate at the same UNI. This use case extends the topology of Use Case 2b with the addition of a new EVPL Service. It assumes the creation of a new EVPL between the existing Address – "NewYorkAddress-id-1" and the new "SanFranciscoAddress-id-1". Since EVPL is VLAN-based – an existing NewYork_UNI can be reused. Note that it is referenced with "productRelationship" instead of "qualificationItemRelationship".

The topology of the Use Case 3b is presented in Figure A1-29:

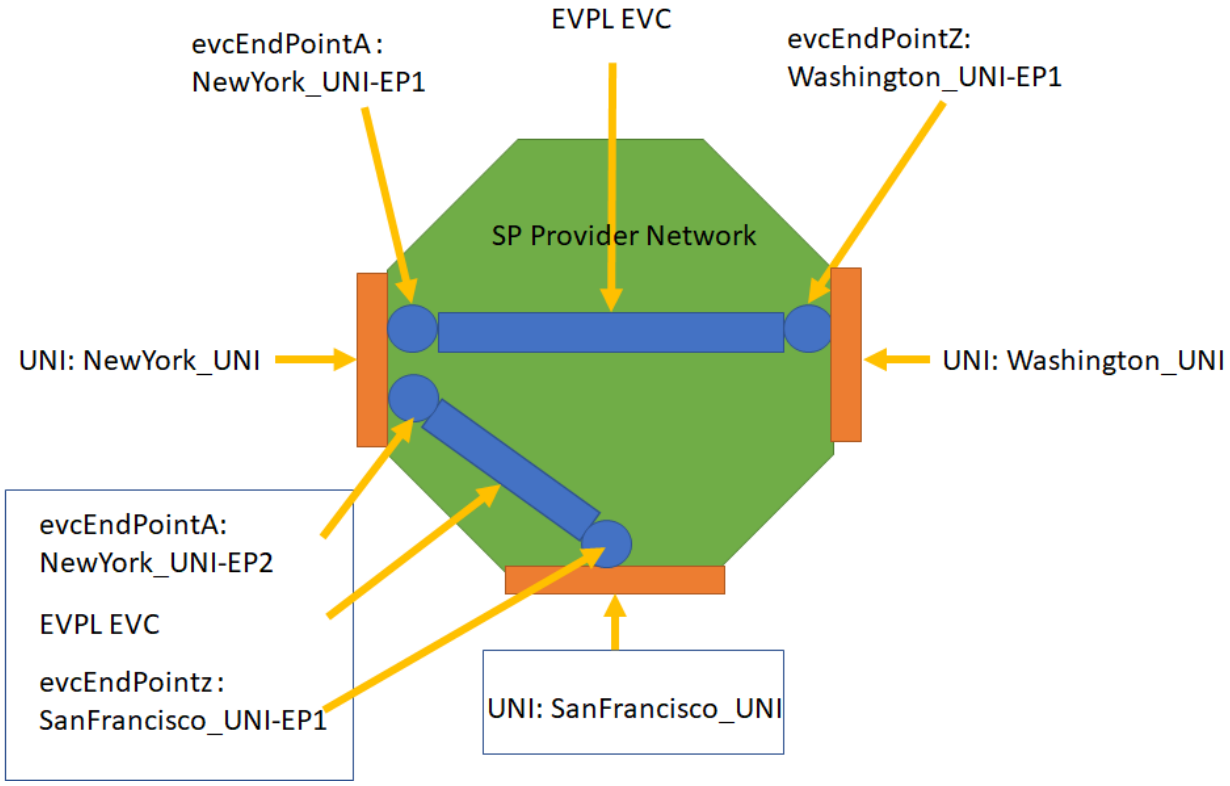

Figure A1-29 – UC3b: EVP – modified Setup Diagram

This topology involves:

- Creation of 1 new UNI

- id=”SanFrancisco_UNI”,

- Creation of the new EVPL, including:

  - configuration of a new EVC End Point with id=”NewYork_UNI-EP2”, at the already existing UNI with id=”NewYork_UNI”, the one that was created in Use Case 2b (assuming it was successfully ordered)

  - configuration of a new EVC End Point with id=”SanFrancisco_UNI-EP1”, at the new UNI with id=”SanFrancisco_UNI”

A structure of a request for a new EVPL and a UNI is presented in Figure A1-30:

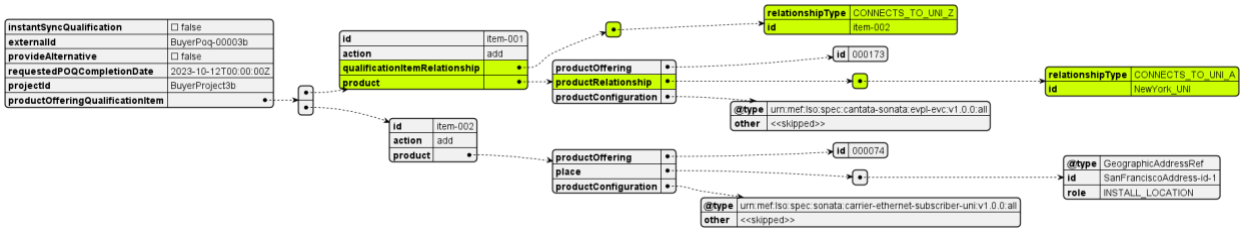

Figure A1-30 – UC3b: EVPL relationships

A.2.10 Use Case 4: Quote – new EPL

For detailed guidance on how to use the Quote Management API, please refer to MEF 115 [4].

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

This use case is the next step after Use Case 2. It asks for a quotation for the installation of the EPL and UNI products, with configuration as described in Use Case 2a.

The Quote API carries product information the same way as the POQ. The same steps in request building and rules of referencing existing products or ones in the same request, as described in section A.2.2, apply. Due to this, only an example built on Use Case 2a is provided to show the details of the Quote envelope. The Quote requests for the remainder of the examples from Use Case 2 can be built using the item structure and product configurations taken from respective POQ requests.

Figure A1-31 presents a diagram of the structure of a Quote request, with product information skipped.

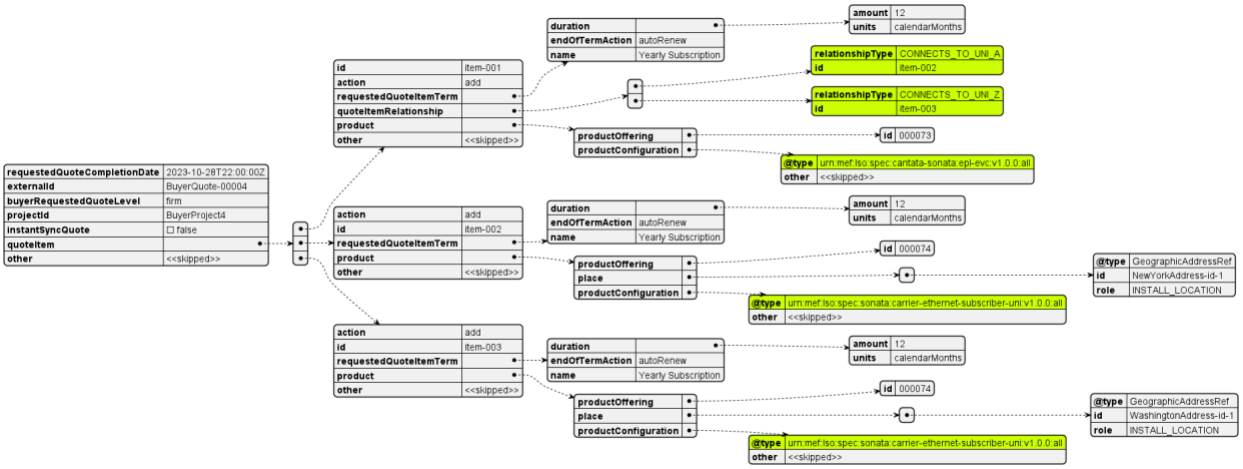

Figure A1-31 – UC4: EPL Quote Request

The most important attributes to set in the quote request are:

- "instantSyncQuote" – to state the preference of receiving an instant (synchronous) response or a deferred (asynchronous) one. In the latter case, the Seller only sends back an acknowledged response and proceeds with the quotation. The Buyer may choose to register for notification or perform a periodical poll.

- "requestedQuoteCompletionDate" – if an instant response is not required this specifies the requested response time.

- "buyerRequestedQuoteLevel" - 3 different types of quotes are managed:

    - **Budgetary**: A Quote that is provided quickly and with very little analysis such that the Buyer can get an idea of how much the requested Product Offering could cost. Any charges specified are subject to change.

    - **Firm - Subject to Feasibility Check**: A Quote that is provided to the Buyer based on some, but not a complete, pre-order analysis. At this stage, the Seller may not be willing to perform any further work on the Quote and requests that the Buyer use the Firm – Subject to Feasibility Check Quote to proceed to the Order process. Ordering is possible based on the Firm – Subject to Feasibility Check Quote with some stipulations as to how the cost identified during delivery is addressed. The Monthly Recurring Charges specified in the Quote Response are final. Non-Recurring Charges specified in the Quote Response are subject to change and new Non-Recurring Charges may be identified during fulfillment.

    - **Firm**: A Quote provided to the Buyer based on a complete pre-order analysis. All Monthly Recurring Charges and Non-Recurring Charges specified on a Firm Quote are committed. A Firm Quote may expire at some date specified by the Seller.

- "requestedQuoteItemTerm" – to specify the term (also known as commitment)

In the response, the Seller confirms (most likely) the “quoteLevel”, “quoteItemTerm” and provides a price per each quote item. An example of price specification is shown below:


```
  "quoteItemPrice": [
 {
   "name": "Monthly Plan 25",
   "priceType": "recurring",
   "recurringChargePeriod": "month",
   "price": {
     "taxRate": 16,
     "dutyFreeAmount": {
       "unit": "EUR",
       "value": 25,
     },
     "taxIncludedAmount": {
       "unit": "EUR",
       "value": 29,
     },
   },
 }
],
```


Note: The Seller may require the Buyer to perform POQ before sending a Quote request.

A.2.11 Use Case 5: Product Order – new EPL

For detailed guidance on how to use the Product Order Management API, please refer to MEF 123 [6].

Product Order allows the Buyer to request the Seller to initiate and complete the fulfillment process of an installation of a Product Offering, an update to an existing Product, or a disconnect of an existing Product at the address defined by the Buyer.

This use case is the next step after use case 4. It places an order for the installation of the EPL and UNI products, which were qualified and quoted in use cases 2 and 4.

The Order API carries product information the same way as the POQ and Quote. The same steps in request building and rules of referencing existing products or ones in the same request, as described in section A.2.2, apply. Due to this, only an example built on Use Case 2a /4 is provided to show the details of the Product Order envelope. The Product Order requests for the remainder of the examples from Use Case 2 can be built using the item structure and product configurations taken from respective POQ or Quote requests.

An example Product Order request can be found in the Postman collection. Figure A1-32 presents it with product information skipped for readability. Note that there are many required related contact information to be provided – please check inside the example in the Postman collection.

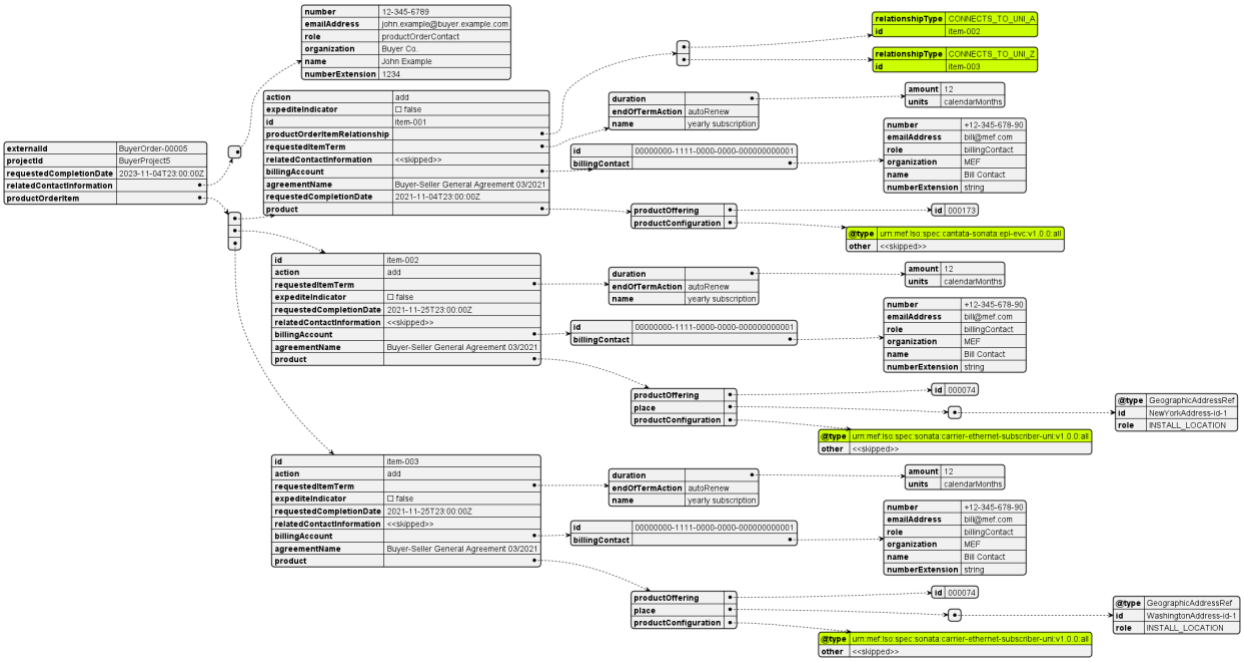

Figure A1-32 – UC5a: EPL Product Order request

Again, there are a few attributes to be set by the Seller in the request like "requestedCompletionDate", "expediteIndicator" or "billingAccount" together with the required contact information.

The Seller responds with an acknowledgment confirmation and then starts processing the order. The order fulfillment process is longer than a simple request-response one of the previous steps (POQ, Quote) and the state machine is more complex. The process may also be more interactive due to charge negotiation, possible request updates, etc.

Product order API offers many more use cases like updating, expediting, or canceling an order request and additional charge negotiation.

A.3 action: modify

The mechanism of building a modification request for both envelope and payload for all steps is practically the same as for the create request.

The differences are in the following common rules (POQ, Quote, Order):

- "item.action" must be set to "modify"
- "item.product.id" of the product to be updated must be provided
- "product.productConfiguration" must contain the full desired configuration (not only the updated values)
- "product.productOffering" must not be changed

- The Subscriber Ethernet Products do not allow “product.productRelationship”, and “product.place” to be changed.

A.3.1 Use Case 6: POQ – EPL - bandwidth change

Use case 6 presents a POQ request for an EPL bandwidth change. The change is made only for the attributes of the EPL product, so the request contains only one item for EPL (the UNI product is not modified). The change is made by updating the following attributes’ value from 70 to 100:

- “evcEndPointA.ingressBandwidthProfilePerClassOfServiceName.bwpFlow.eir.irValue”,
- “evcEndPointA.ingressBandwidthProfilePerClassOfServiceName.bwpFlow.eirMax.irValue”,
- “evcEndPointZ.ingressBandwidthProfilePerClassOfServiceName.bwpFlow.eir.irValue”,
- “evcEndPointZ.ingressBandwidthProfilePerClassOfServiceName.bwpFlow.eirMax.irValue”

Note that since there are no accompanying items for UNIs, relationships information "CONNECTS_TO_UNI_A" and "CONNECTS_TO_UNI_Z" must be provided with the use of “product.productRelationship” attributes to point to the existing UNI instances with “id”=”NewYork_UNI” and “id”=”Washington_UNI” respectively.

When requesting an update of the product, the “action” must be set to “modify” and an “id” of the product to be modified must be provided. In this example, the “id” is “EPL”.

The diagram below shows a POQ request and detailed EPL configuration for modification, highlighting the changes compared to the creation request.

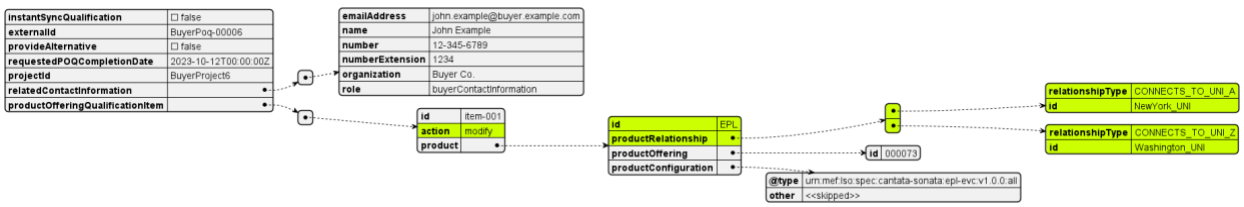

Figure A1-33 – UC6: EPL POQ request for modification

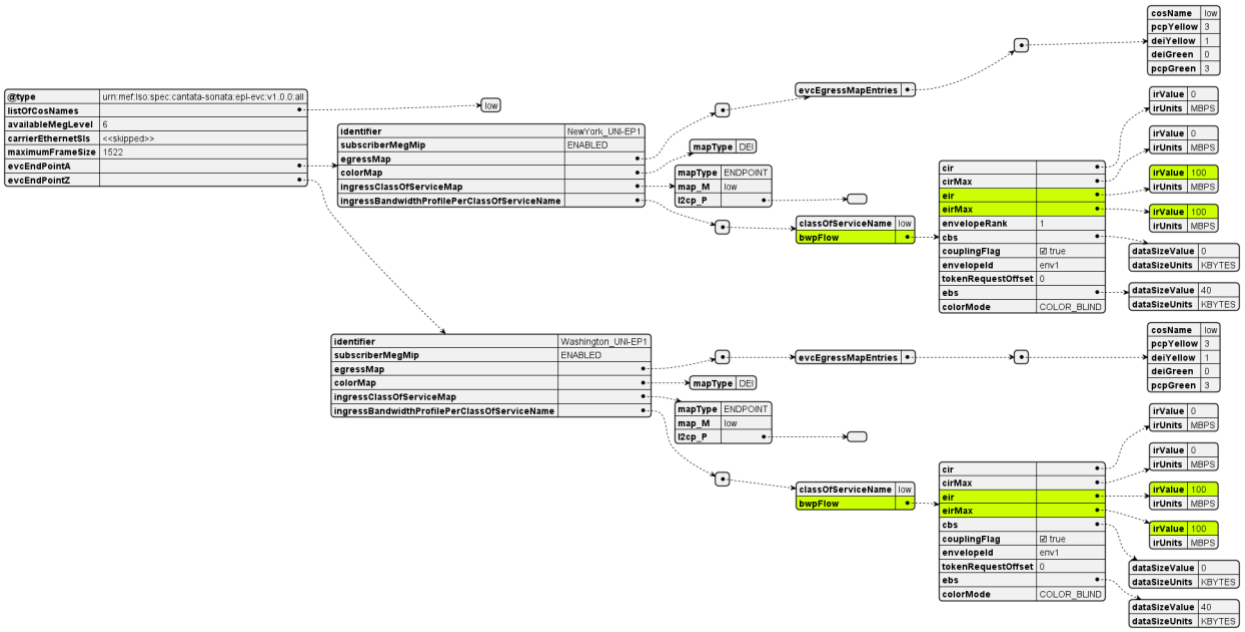

Figure A1-34 – UC6: EPL modified attributes

A full example request can be found in the attached Postman collection.

A.3.2 Use Case 7: POQ - EVP-LAN - add UNI and EVC End Point

Use case 7 presents POQ for adding a new connection point to a multi-point EVP-LAN topology of Use Case 2d. This requires the addition of one new EVC End Point at a new or existing UNI. It also requires an EVC End Point Map that specifies which VLANs are mapped to the new EVC End Point.This example shows the creation of a new UNI.

Figure A1-35 depicts the topology of this Use Case. The light blue items at the bottom are the newly created products. The EVC End Point references EVP-LAN EVC existing product (assuming its “id” = “EVP-LAN”), there is no update of any configuration of the EVC itself, so the POQ (and also further Quote and Product Order) only contains items for the EVC and the EVC End Point.

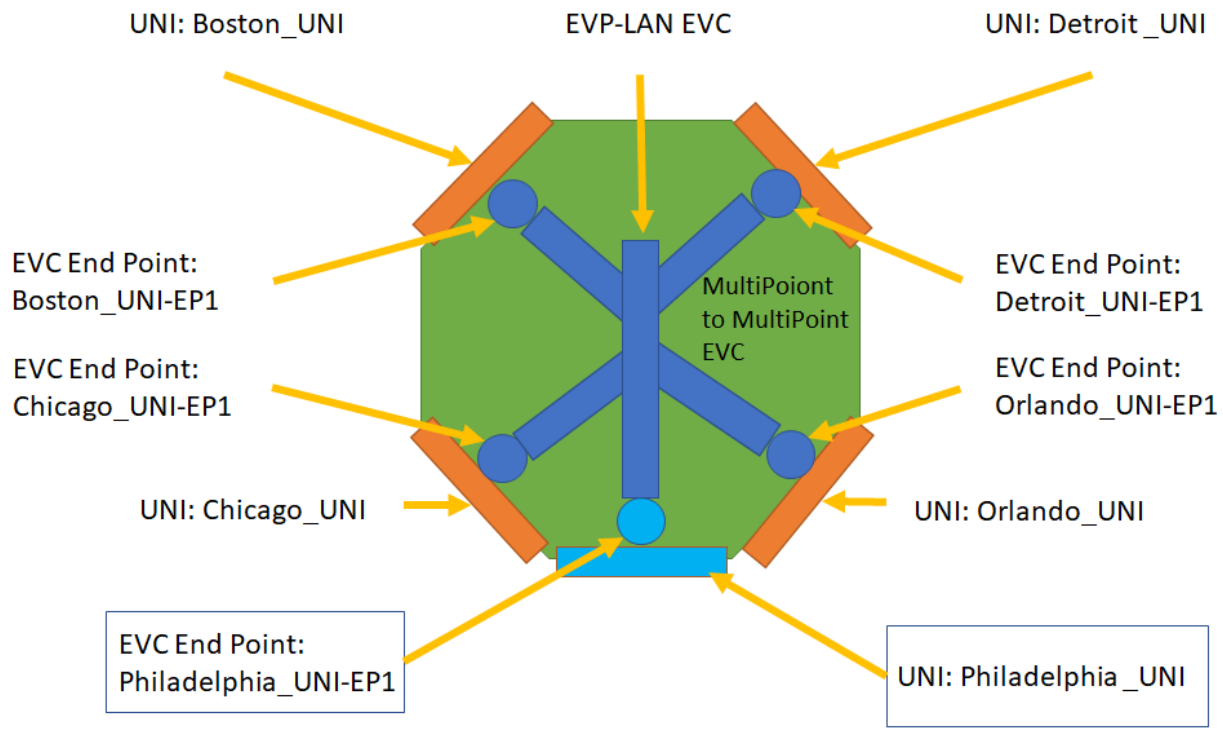

Figure A1-35 – UC7: EVP-LAN - add UNI and EVC End Point

Note that since there is no item with the EVC, the EVC End Point relationship information "ENDPOINT_OF_EVC" must be provided with the use of the "productOfferingQualificationItem.qualificationItemRelationship" attribute pointing to the existing EVP-LAN EVC product instance with "id"="EVP-LAN".

The diagram below shows a POQ request for the addition of an EVC End Point and the UNI, highlighting the relationships and discrepancies with the creation request.

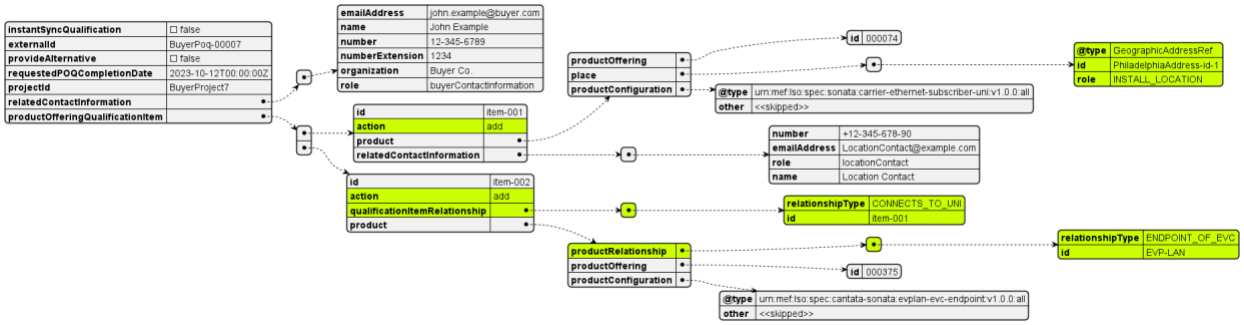

Figure A1-36 – UC7: EVP-LAN POQ modify request

A full example request can be found in the attached Postman collection.

A.3.3 Use Case 8: POQ – EP-TREE - remove UNI and EVC End Point

Use case 8 provides an example of reducing the number of connection points of the EP-TREE created in Use Case 2e. This requires decommissioning (removal) of the EVC End Point and the

UNI (refer to section A.4 for general rules of the “delete” action). The topology of this use case with removed items shown in red color is presented in Figure A1-37. The deletion request must only contain the “id” of the product to be removed. The products to be deleted must not provide any product configuration or relations to other products and/or items. The change in the number of EVC End Points does not need any update of the EVC, so the request contains only 2 simple items.

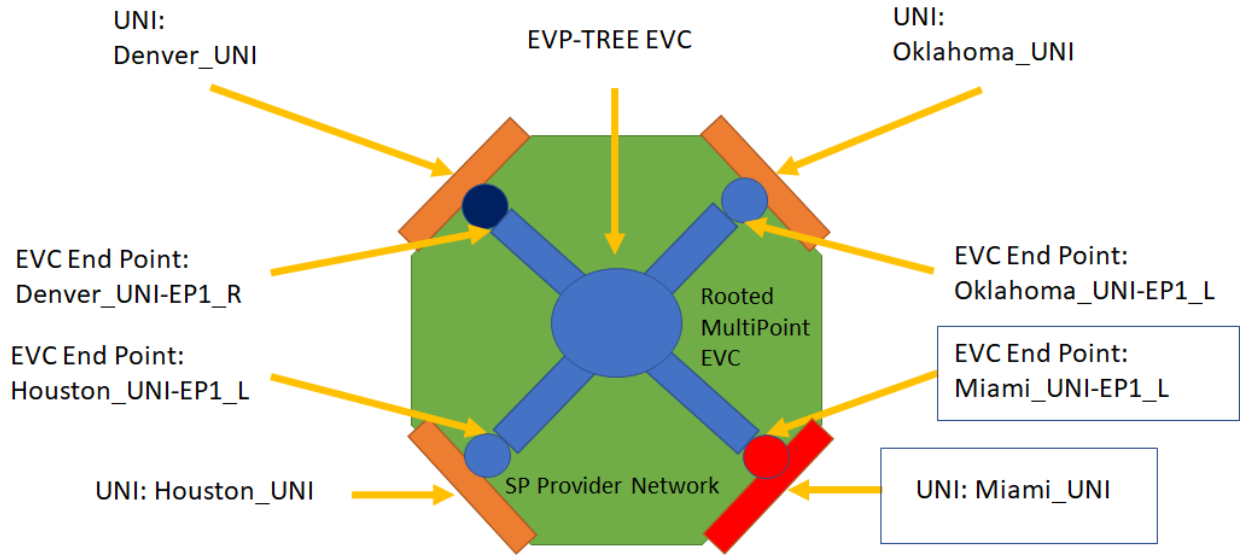

Figure A1-37 – UC8: EP-TREE – modification topology

The diagram below shows a POQ request for modification, highlighting the changes compared to the creation request. (Figure A1-38)

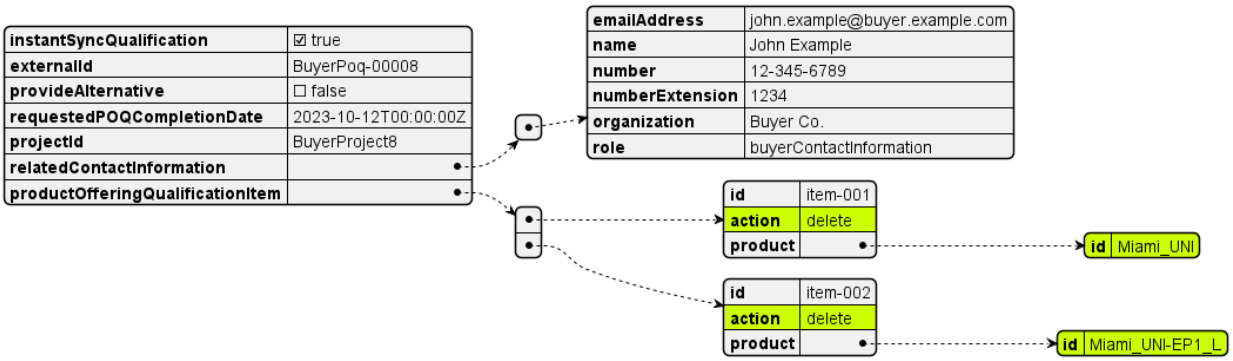

Figure A1-38 – UC8: EP-TREE – modification POQ request

A full example request can be found in the attached Postman collection.

A.3.4 Use Case 9: Product Order – EVPL - VLAN change at the UNI

In this case, an order to enhance the list of VLAN IDs that are mapped to the EVC End Point is enhanced from [1, 2] (based on Use Case 2b) to [1, 2, 3]. This is done with the order request that provides the configuration only of the modified object – EVC (no need to provide all of the items

for components building the full product). Figure A1-39 shows the diagram of the product configuration in the request highlighting the modified attribute.

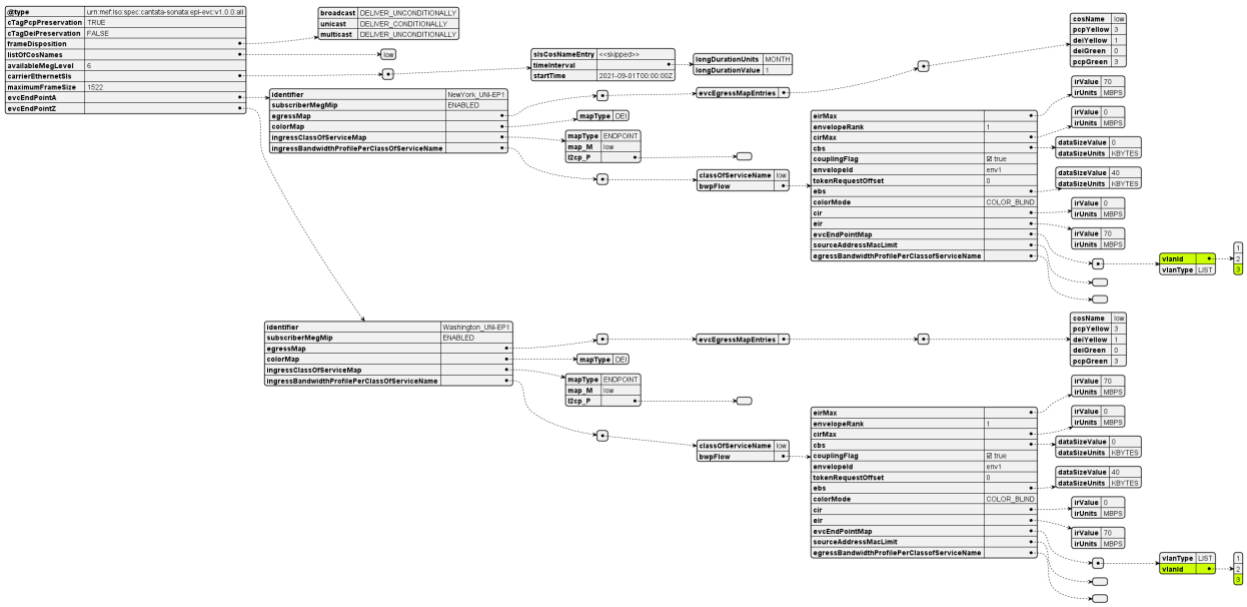

Figure A1-39 – UC9: EVPL Order modification request

A full example request can be found in the attached Postman collection.

A.3.5 Use Case 10: EPL - move to a different Location

In the case of moving the Buyer's office to another building, existing connections must be updated. However, this cannot be realized by a single update of the "place" attribute of the UNI product. As stated in [R29]:

*For a CHANGE to a Subscriber Ethernet UNI product, the Related Place **MUST NOT** be changed from the value present in the Product Inventory.*

This also cannot be realized by updating EPL's product reference to another UNI created at the new location, as per [R24]:

*For a CHANGE to an EPL EVC or EVPL EVC product, the relationship to the Subscriber UNI **MUST NOT** be changed from the value present in the Product Inventory.*

So, the argument is both business and technical. To realize this use case, the following requests must be performed:

1. Creation of a new UNI at the new location
2. Deletion of an old EPL
3. Creation of a new EPL

4. Deletion of an old UNI

Step 1 as potentially requiring physical installation should be performed earlier to prepare for a switchover. Steps 2 and 3 should be coordinated to ensure minimal downtime.

This use case as being built upon already described steps is not part of the attached postman collection.

A.3.6 Use Case 11: EVP-LAN – move to a different Location

Since EVP-LAN is multipoint technology, it is possible to "move" it simply by applying the steps described in Use Cases 7 and 8 – adding new Endpoint(s) and UNI(s) to it and removing the old ones.

This use case as being built upon already described steps is not part of the attached postman collection.

A.4 action: delete

Delete requests are very straightforward, as they only carry the product "id".

The following common rules apply for deletion operation:

- "item.action" must be set to "delete"
- "item.product.id" of the product to be deleted must be provided
- "product.productConfiguration" must not be provided
- no other item attribute may be provided (except for the optional "billingAccount" in Order)

A.4.1 Use Case 12: Product Order – EPL - decommission

Decommissioning of an EPL product requires the deletion of all of its components, which are the EPL EVC and the UNIs. Deletion of EPL created in Use Case 2a can be ordered with a request that is presented in Figure A1-40:

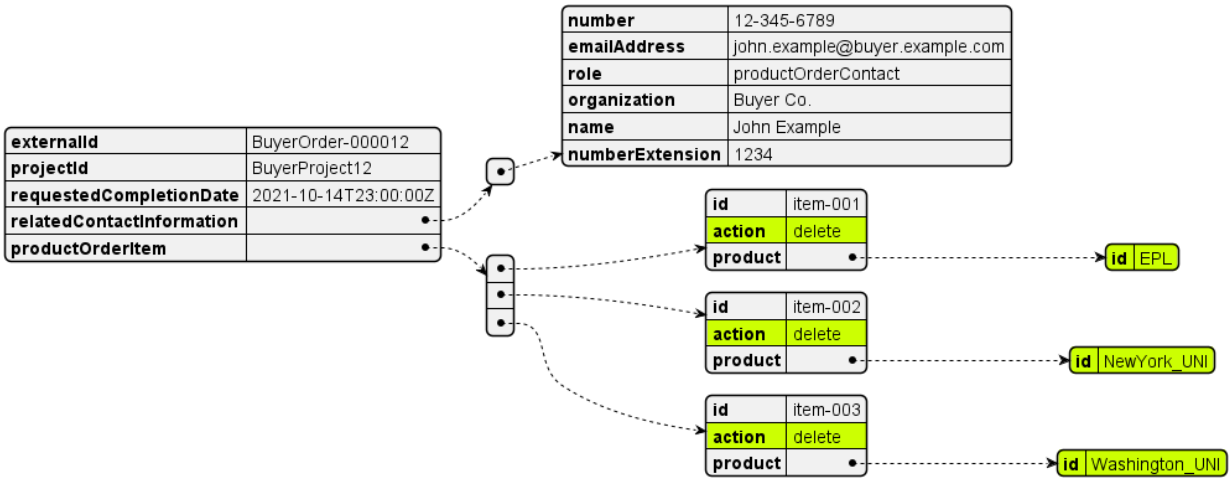

Figure A1-40 – UC10a: EPL Order deletion request

Note: A disconnect request may result in additional charges (if not quoted earlier).

6 References

[1] IETF RFC 2119, *Key words for use in RFCs to Indicate Requirement Levels*, by Scott Bradner, March 1997

[2] IETF 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.

[3] MEF 87, *LSO Cantata and LSO Sonata Product Offering Qualification API – Developer Guide*, May 2022

[4] MEF 115, *LSO Cantata and LSO Sonata Quote Management API – Developer Guide*, May 2022

[5] MEF 121, *LSO Cantata and LSO Sonata Address Management API – Developer Guide*, May 2022

[6] MEF 123, *LSO Cantata and LSO Sonata Product Order Management API – Developer Guide*, May 2022

[7] MEF 125, *LSO Cantata and LSO Sonata - Subscriber Ethernet Product Schemas and Developer Guide*, July 2022

Appendix A Acknowledgements (Informative)

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

- Manfred **ARNDT**
- Michał **ŁĄCZYŃSKI**
- Marcin **NATURALNY**
- Larry **SAMBERG**
-