EUROPEAN COMMISSION
Directorate-General for Internal Market, Industry, Entrepreneurship and SMEs

Consumer, Environmental and Health Technologies
**Health Technology and Cosmetics**

29/05/2019

# Eudamed Data Exchange Services and Entity Models Introduction

*Eudamed Data Exchange (DTX) service and entity models introductory document*

# *Table of contents*

# 1 Approach and Purpose of the Document

This document assumes that readers are familiar with the EUDAMED MDR general propose and CEF eDelivery building block (EUSEND) of the Connecting Europe Facility (CEF).

This document addresses data exchange related architecture and data modelling applicable introductory guidelines, in the context of performing machine – to – machine *(M2M)* connectivity of external organisations with EUDAMED MDR application for data exchange of actors, actor registrations and devices data. It also aims to stand as a living document helping the economical operators' representatives but not limited to, be introduced into technical aspects concerning M2M data exchange and services definition between EUDAMED and their internal data repositories.

In order to perform M2M EUDAMED Data Exchange (EUDAMED DTX) between various private or business entities or public administrations, it is required to use an additional component (building block), provided by EUSEND program. EUSEND allows the possibility to transmit data between third parties by electronic means and provides evidence relating to the handling of the transmitted data, including the proof of sending and receiving the data, and that protects transmitted data against the risk of loss, theft, damage or any unauthorised alterations.

*The eDelivery building block of the Connecting Europe Facility (CEF) enables Businesses and Public Administrations (both are hereinafter referred to as 'organisations'), to exchange electronic data and documents in digital format in an interoperable, secure, reliable and trusted way , with other organisations (and indirectly with citizens).*

This document states the first EUDAMED Data Exchange Service / Entity Model and data exchange protocols based on the functional specifications and use cases and targeted reader stakeholders are the Eudamed DTX participants and their IT / software backend.

The concepts and terminology that are of key importance in understanding the concepts around data exchange are introduced below:

- *entity model* - represents the canonical model of business entities (e.g. actor, actor registration, device, certificate, etc.) ;
- *service model* - models how information _data) is shaped / transferred in order to be transported encapsulated and to access the right service;
- *data exchange communication patterns* - defines how the data is exchanged (transported) between participants;
- *service definition* - describes how to represent and to access the business capability.

# 2 Performing EUDAMED Data Exchange - high level architecture

## 2.1 Eudamed Data exchange scenarios

Currently, two scenarios are available in order to exchange data between a public or private organisation and Eudamed MDR database (accessible through exposed EUDAMEDN services) as described in Figure 1.

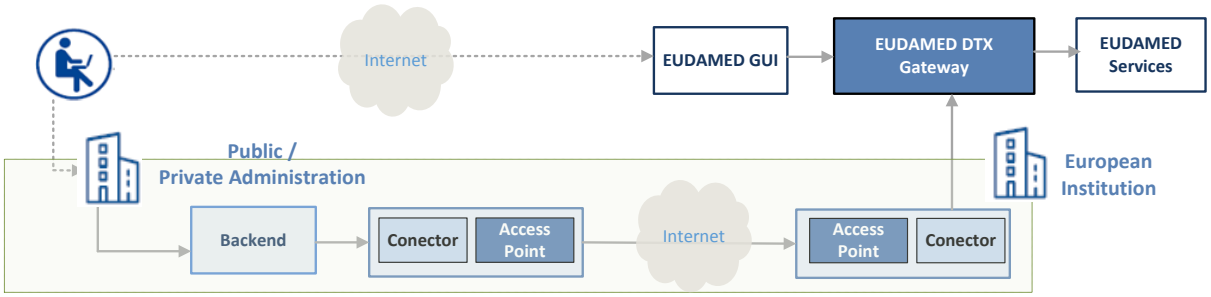



Figure 1 Available performing data exchange scenarios with Eudamed

*First scenario, named manual bulk data exchange***,** describes a data exchange user (as a EUDAMED user) that access EUDAMED GUI (web application) and manually upload an XML formatted file containing exchanged domain entities. The files will be process by a EUDAMED DTX Gateway engine that will serve the user's request in an asynchronous way by accessing EUDAMED service and notify the result into the same GUI interface.

*The second scenario, named M2M data exchange*, describe a data exchange user (as a public administration / private organisation backend) that automatically submits the XML formatted file containing exchanged domain entities to EUMDAMED services using a dedicated CEF eDelivery access point (AP) and a secure communication protocol assured by the AS4 protocol.

As noticed, the similarities between the two scenarios consists in using a common XML data exchange service and entity model to represent the data to be exchanged and a common processing gateway to serve and manage the requests.

The current document aims to provide introductory knowledge around the M2M scenario, but most of the information provided (such as data model and service model) are reusable also for the first scenario.

## 2.2 M2M Data exchange architectural view

The EUDAMED M2M Data Exchange defines a set of building blocks that should be specified and implemented to enable the information exchange between partners and EUDAMED (Figure 2):

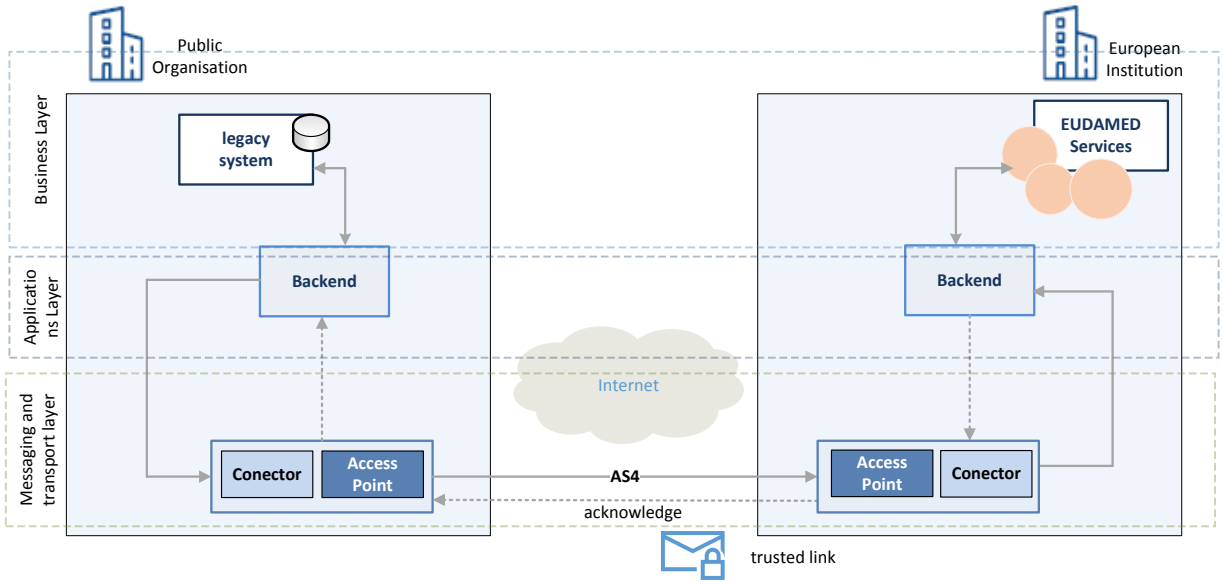



Figure 2 EUDAMED data exchange integration - high level architecture

**Public / Private Organisation** – this block generically represent the organisation's legacy business domain from which the data is exchanged to and from EUDAMED. In this category, a particular organisation must be included: **3rd Party Provider –** this entity can substitute the functionality of an existing manufacturer that exchanged data on his behalf.

**Organisation Backend -** this block represents a dedicated information exchange gateway. It implements the specific communication protocol, service and entity data exchanged between the organisations and EUDAMED. It also can have an adaptation role for mapping the data model in the context of the communication with the actor's legacy systems or human interface. The EUDAMED partner is responsible to provide an implementation for this building block. It will have to be compliant with the EUDAMED service and data model described in the following sections.

**eDelivery Access Point –** instances deployed on both Actor premises and Commission EUDAMED**.** These blocks will ship the messages from the EUDAMED partner infrastructure to the corresponding EUDAMED deployed eDelivery access point.

**EUDAMED DTX Gateway (Backend) -** this module takes care of the data exchange messages requests, including the security, access control and reliability aspects. It also can have an adaptation role for mapping the requests to the destination EUDAMED M2M Data exchange services and in charge of constructing the response messages and acknowledgements to be sent back to the requester.

## 2.3 Service and entity models cohesion

The data exchanged in Eudamed DTX is based on two independent data models. One used to describe the communication protocols and target services called ***service model*** and the ***entity model,*** used to represent the business domain entities involved in data exchanged. There is an encapsulation process of those models as shown in the below Figure 3.

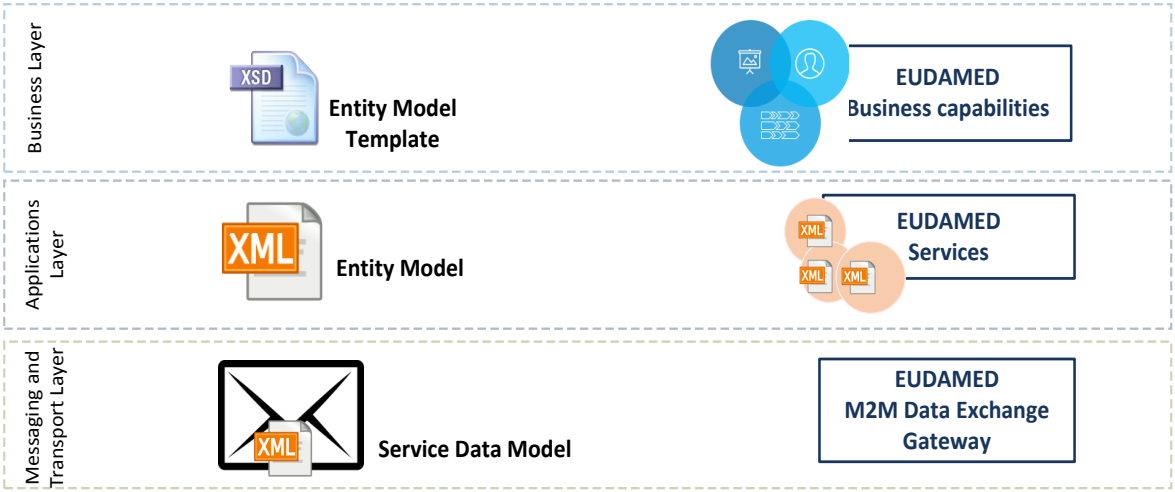



Figure 3 Service and entity models cohesion

# 3 EUDAMED DTX Services and Messaging Patterns

The following subsections describe how to implement each communication pattern using service descriptions and message objects. All the communication patterns are asynchronous, i.e., the request of information and the response from the provider are separate processes, only loosely linked by correlation numbers.

## 3.1 Operations and messages

Each service is a collection of operations. An operation is the callable part of a service. A *Device* service may, for instance, have operations like *Download Devices* (or simpler *download*) and *Upload Devices* (or simpler *upload).* The inputs and outputs of operations are called messages. A message can be defined as an autonomous unit of information. It contains one data ***(entity)*** part, carrying whatever information the sender wishes to convey, and one metadata ***(service)*** part with information needed to route the message to its correct destination together with any information the receiver will need to successfully process the message. The best practices recommend that messages should be as self sufficient as possible; holding a logic entity of information in its data part.

## 3.2 Implementing the information exchange patterns

The following subsections describe the main data exchange message structures to implement supported communication pattern.

### 3.2.1 Pull

The Pull pattern is based on the *need-to-know* principle. In this pattern, the Actors (Manufacturer, Competent Autorities (CA), Authorise Representative (AR), etc.), as customers of data request, request a piece of information (e.g. Device, Certificate, Actor registration, etc.) to the EUDAMED provider through the Pull operation using the PullRequest message. The EUDAMED provider replies using the PullResponse message back to consumer. The PullResponse message contains, as payload, the information requested in the PullRequest. The pull request and response messages exchange protocol is described in Figure 4.

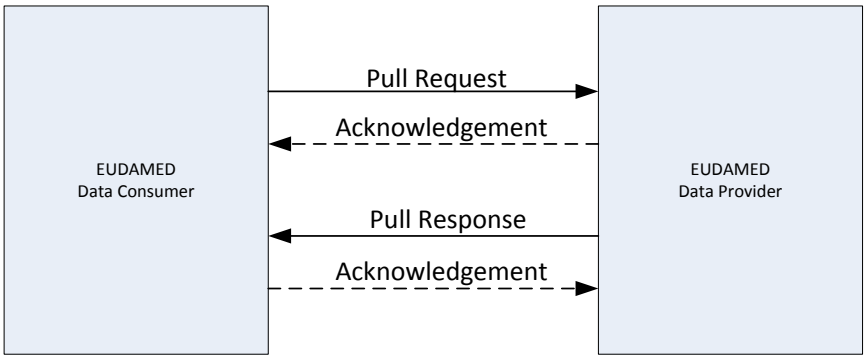



Figure 4 Pull request and pull response communication message exchange pattern

### 3.2.2 Push

The Push pattern is based on the *responsibility-to-share* principle. In this pattern, the EUDAMED data provider sends a piece of information to the EUDAMED data consumer using the Push operation. The push pattern in described into the Figure 5.

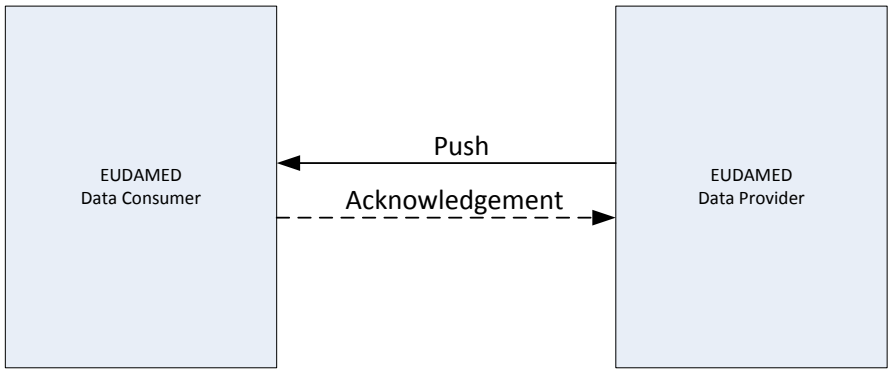



Figure 5 Push communication message exchange pattern

# 4 Service and Entity Model Overview

## 4.1 Service model

The Service Model describes the EUDAMED data structures used to perform information exchange. Data Exchange Service model is used for shaping the transport protocols (including security, reliability, etc.). In EUDAMED, services hide the complexity of the EUDAMED infrastructure and functionalities and the heterogeneity of functional modules behind standards-based interfaces.

The Service model is designed to be flexible and adaptable to several use cases and is composed by the following elements:

- information about the type of message (from the data exchange messaging patterns pull/push);
- service destination information (each operation will represent a specific functionality);
- means to query / filter (criteria) the request;
- security rules that may apply depending on the actor that performs it;
- information available as response, eventually number of provided entities, pagination and versioning capabilities.

**Messaging *(MessageType)***: message types and related properties, such as:

- **Message type *(Push, PullRequest, PullResponse, Acknowledge)***: identification, description of the root of the messages, etc.
- **Payload *(Criteria Payload)* -** canonical entities data models and related attached resources (e.g. attachments, etc.), or querying criteria
- **Reports *(ElementReportsType):*** describing the reports to be submitted with the processing / request status in case of an initial Push or PullRequest messages.

**Service definition *(ServiceType)***: description of the services provided by the EUDAMED.

- **serviceID** - unique identifier of a service in EUDAMED;
- **serviceOperation** - Supported by the service (e.g. download, upload, update, etc.);
- **serviceAccessToken -** Bearer security authentication token to gain access permission to the requested data by the requester as EUDAMED actor or 3rd party (acting on behalf of the actor mentioned in nodeCode).
- **serviceVersion –** in case of multi versioned compatible service, allows to specify what version of the current service is invoked.

**Addressing *(NodeType)***: logical networking information related to the actors and AP used in data exchange process:

- **nodeCode** - Contains the EUDAMED code (e.g. SRN, CA identifier, NB code etc.) of the party that performs the call of service. In case of multi profile endpoint (e.g. 3rd party companies), it contains the actor code on behalf of the request is performed.
- **nodeID** - Identify the requester (actor/3rd party integrators) of the message eDelivery endpoint.

### 4.1.1 Message identifiers

Message objects can contain three types of identifiers:

- *MessageID*: Identifier of the message. It is unique for the Eudamed participants who created the message;
- *CorrelationID*: This identifier correlates the request and response messages of/to a service;
- *ConversationID*: This identifier correlates the messages that share an operational need. For instance, in order to download all manufacturer' details, several paginated queries must be sent by the requester in order to obtain the full list of the manufacturers.

### 4.1.2 Service query

The specification of the Pull operation for the Pull patterns introduce the query mechanism that consists of specific query classes containing criteria supported by the related services. Using this mechanism, EUDAMED actors, information consumers, can request to a EUDAMED provided service all the data entities that are matching the requested criteria.

Considering an example in which a Eudamed DTX *Actor* service consumer (e.g. Belgium CA) wants to download all the registered manufacturers from Belgium. To achieve this, it will send to EUDMAED a pull request message where he adds BE as country of the respective organization as a criteria inside the PullRequest message. The EUDAMED service provider should reply with all the MF actor entities whose organization country is BE.

### 4.1.3 The business acknowledgement mechanism

The business acknowledgement mechanism between EUDAMED participants is designed to be asynchronous and achieved through out sending an *acknowledgement* message each time a message a Push message is received by EUDAMED DTX gateway. The scope of this acknowledgment is to notify the sender if the message was successfully delivered and provide a status of the information processed.

A detailed sequence diagram of the two acknowledgement mechanism is described in Figure 6.

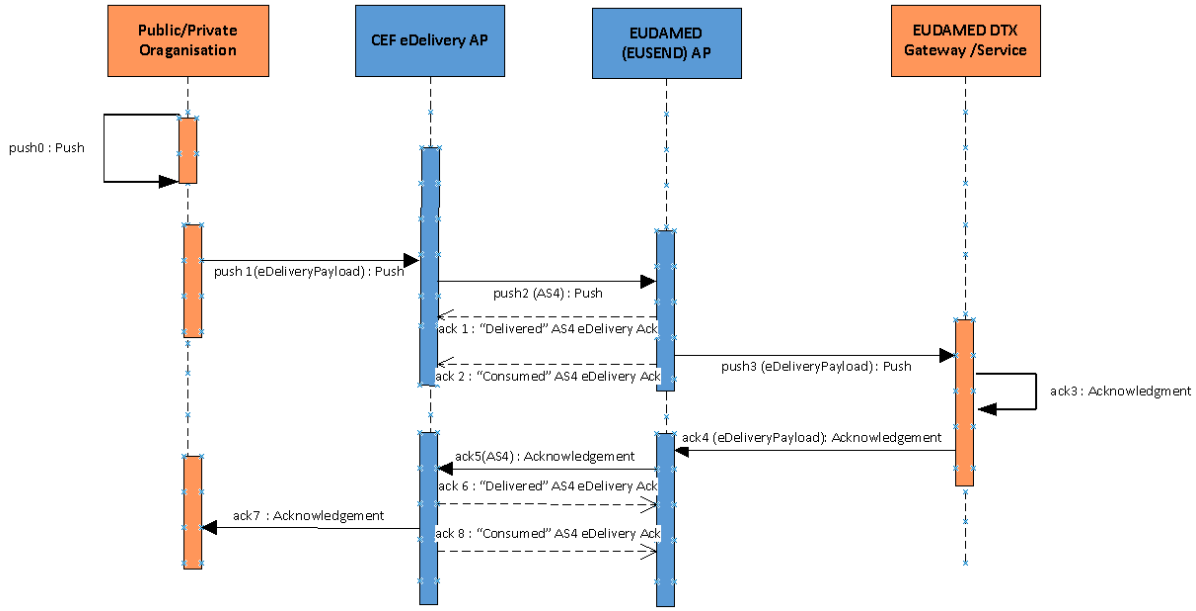



Figure 6 Types of acknowledgments provided by EUDAMED DTX.

## 4.2 Canonical domain entity structures and generic modelling aspects

The EUDAMED entity model represents the canonical representation of domain entities used to exchange information across various EUDAMED partners (backend platforms) and EUDAMED application. The Entity model is under development and takes into account the existing data standards used in the systems for medical devices in Europe in order to facilitate the integration of these systems to EUDAMED.

## 4.3 Entity model

The majority of main business entities contain the abstract superclass *Entity* to express the versioning capability.

Generally, most of the entities exchanged (trough the exposed services) are interrelated (have dependencies between them). To model this, a concept of link has been introduced. In this way, it is possible to isolate the information related to a specific subdomain and only to reference the other related entities.

# 5 XSD Data Model Representation

## 5.1 XSD Packages view

The current paragraph describes the structure and usage of the provided data (service and entity models) in XSD format. The provided XSD packages have the following composition as show in below Figure 7.

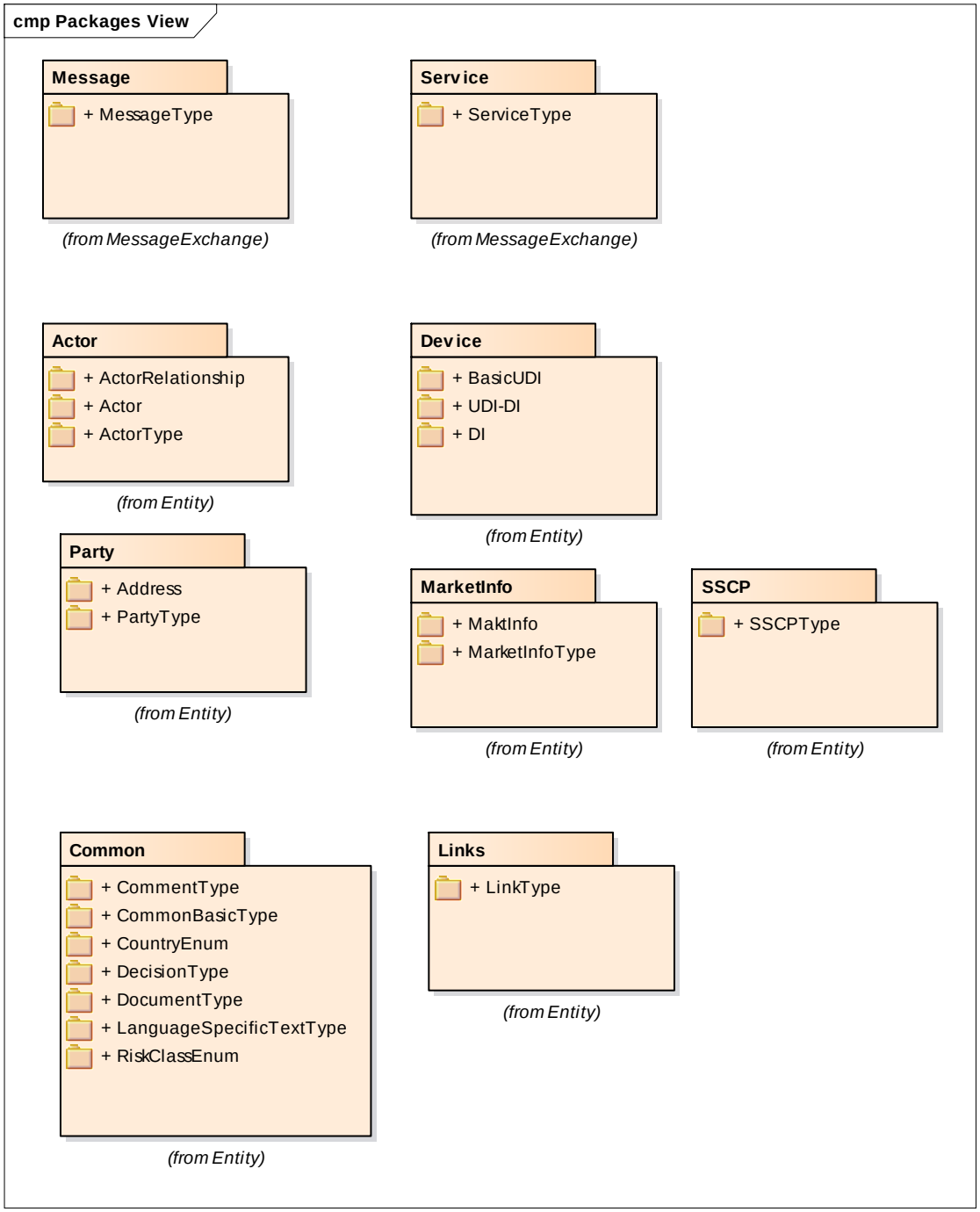
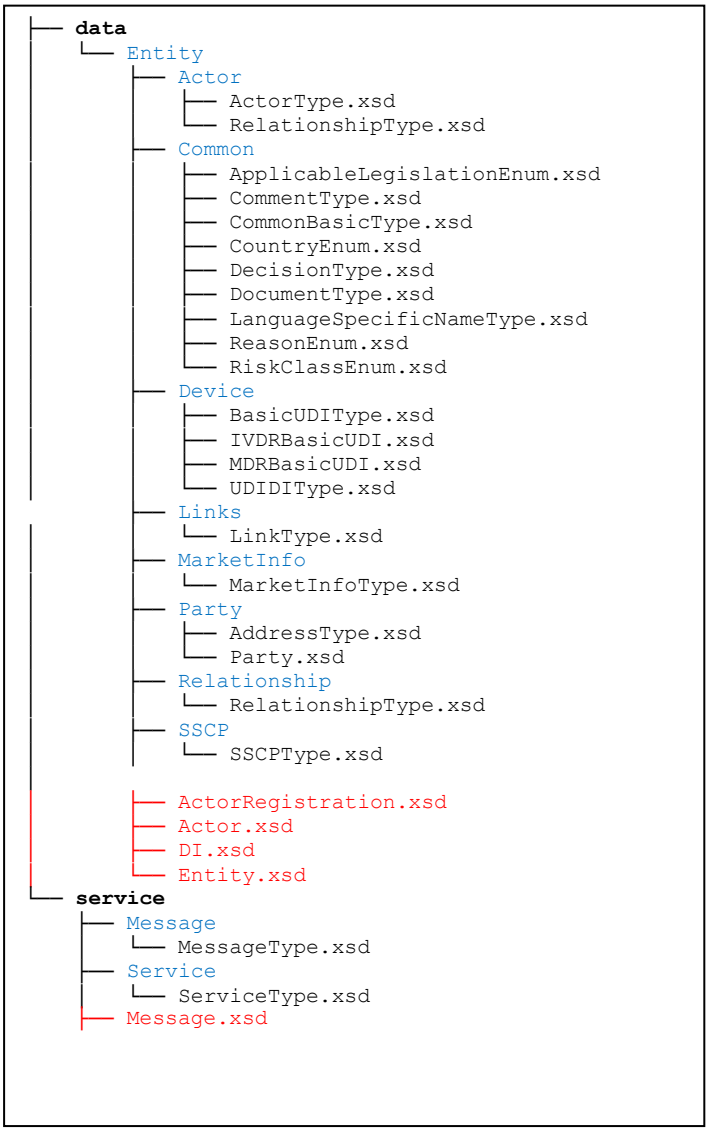



Figure 7 Package structure of the EUDAMED DTX XSD models

The package organisation of the DTX model offer flexibility and reusability of components. The structure is composed into three main groups of packages:

- service package;
- entity models package;
- commons (reusable components) package.

## 5.2 XSD File structure organisation




The ***data*** directory stores the Eudamed DTX *Entity* model packages and dependencies as:

- ***entity model packages*** (e.g. Actor, ActorRegistration, UDI-DI, etc.), stored into a dedicated subfolder (package)
- ***additional transversal packages*** (e.g. Commons (language, countries, etc.), Links, etc.), holds some common data models
- ***top level elements (for the main entities)*** - interfaces to related packages*:*
    - ActorRegistration.xsd
    - Actor.xsd,
    - DI.xsd

The ***service*** directory stores the EUDAMED DTX *Service* model (*Messaging*) packages and dependencies. This package contains the structure of data to be used for creating the service envelope of the business entities that are transmitted between the EUDAMED DTX parties. It is structured into the following packages:

- ***messaging model package*** - describe message types and metadata *(*Pull / Push / Acknowledgment), information about the senders and destinations of the message, reliability and status reports, payload to accommodate the entities transmitted, etc.;
- ***service representation model package*** - models the requester call back and destination service the current message is targeted.
- ***top level elements*** *file (for the message types)* - interfaces to related packages:
    - `Message.xsd`.

`Message.xsd` is the starting point (top level elements) that point to the three main message types needed to start creating a message: Pull request / Pull response, Push, Acknowledgment.

**Important!** The bridge between the two models representation (service and entity model) is the payload section. The payload encapsulates elements representing entities (from entity model package). A basic message structure would consist of the following main elements:

```
<PullRequest/PullResponse/Acknowlegment>
      .. ..
    <messageID>
      .. ..
    <sender>
        <node>
        <service>
    <recipient>
        <node/>
        <service>

    <payload>
        <Entities xsi:type="MFActorType"</Entities>
         <Entities xsi:type="MFActorType"</Entities>

</PullRequest/PullResponse/Acknowlegment>
```

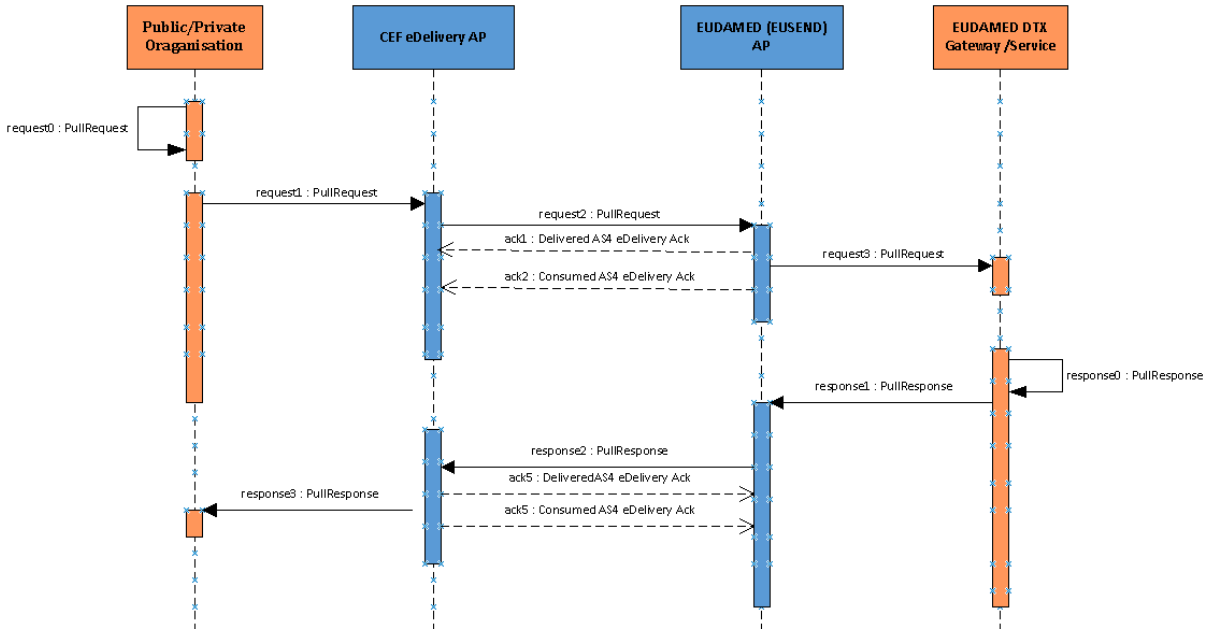
# 6 EUDAMED Message Exchange Patterns

## 6.1 Pull Pattern

### 6.1.1 Generic pull request and pull response

The following UML sequence schema describes the message exchange between a public / private organisation (an EUDAMED actor) and EUDAMED services using the *Pull* communication pattern.

This process is invoked every time there is a need of requesting / downloading of data store by EUDAMED.




1. ***request0 -*** *build a* ***pullrequest*** message (Eudamed service model) compliant xml message.

   ***Attributes selection:***

   - **messageID**: a unique identifier, issued by submitter
   - **correlationID**: an identifier that will correlate the request to the response or to the acknowledgements, issued by the requester
   - **sender/service/ServiceID**: identifier of the callback service (for responses and acknowledgements)
   - **sender/node/nodeCode**: identifier of the EUDAMED unique number of the requester (e.g. SRN, CA number, etc.)
   - **sender/node/nodeID**: identifier of the eDelivery ***partyID***
   - **recipient/service/serviceID**: identifier of the Eudamed service
   - **recipient/service/serviceOperation**: identifier of the Eudamed service operation to uniquely define the service scope
   - **recipient/service/serviceAccessToken**: the EUDAMED bearer security token attached to the requester service
   - **pageNumber:** in case of paginated response, the requester can orchestrate multiple page response and ask for a specific page on page to be provided in the response
   - **pageSize**: required maximum number of entities on a specific response page
   - **payload/Entities**: it may contain the main service entity in case of a query by example criteria

2. ***request1 (PullRequest)*** **– message** send by the requester backend to his CEF eDelivery access point AP. The request message (Eudamed service model) is embedded into the payload of a SOAP eDelivery compliant message.

   ***Attributes selection:***

- **envelope/header/messaging/usermessage/partyinfo/from/partyID**: a unique identifier of the requester's eDelivery AP
- **envelope/header/messaging/usermessage/partyinfo/to/partyID**: a unique identifier of the EUDAMED MDR eDelivery AP
- **envelope/body/submitRequest/payload:** holds the Base64 format of the *request0* message

3. ***request2 (PullRequest)* – message** send between the two CEF eDelivery AP. The format of the message is AS4 compliant. The message will be marked with *SENDING status*. The message format is not subject of treatment into the current document.

4. ***ack1 (AS4)* – acknowledgement message** indicates that current message has been delivered to the service provider CEF eDelivery AP. The message will be marked as *DELIVERED*. The message format is not subject of treatment into the current document.

5. ***request3 (PullRequest) – (same as request1) message** delivered to the EUDAMED services backend processor.*

6. ***ack2 (AS4)* - acknowledgement message** indicates that current message has been delivered to the EUDAMED service by the corresponded CEF eDelivery AP. The message will be marked as *DELETED* in the management interface of the requester eDelivery AP management interface.

7. ***response0* -** *build* ***pullresponse*** message (Eudamed service model) compliant xml message to be sent to the requester

   ***Attributes selection:***
   - **messageID**: a unique identifier, issued by submitter
   - **correlationID**: same as correlationID from the request message
   - **responseCode:** status code of the service call (success or matching error code)
   - **sender/service/ServiceID**: identifier of the EUDMAED initiator service
   - **sender/node/nodeCode**: identifier of the EUDAMED eDelivery party identifier
   - **sender/node/nodeID**: identifier of the eDelivery partyID
   - **recipient/service/serviceID**: requester generic callback service
   - **recipient/service/serviceOperation**: requester callback service operation (as specified in the request message)
   - **payload/entities**: contain the list of entities of the same type that were collected thought the service invocation

8. ***response1 - PullResponse – asynchronous*** message send by EUDAMED DTX service backend to his CEF eDelivery instance. The protocol and format are the same as request1

   ***Attributes selection:***
   - **envelope/header/messaging/usermessage/partyinfo/to/partyID**: a unique identifier of the requester's eDelivery AP
   - **envelope/header/messaging/usermessage/partyinfo/from/partyID**: a unique identifier of the EUDAMED MDR eDelivery AP
   - **envelope/body/submitRequest/payload:** holds the Base64 format of the *response0* message.

9. ***response2 - PullResponse* –** message send by EUDAMED eDelivery AP to the initial requester AP. The protocol and format are the same as *request2*
10. ***ack3 (AS4)* – acknowledgement message -** same as *ack1*.

11. ***response3 - PullRequest message* -** same as *response2.*

12. ***ack4 (AS4)* - acknowledgement message** – same as *ack2*.

### 6.1.2 Use case – CA downloads paginated devices (Device DOWNLOAD service)

**Process:**

| Step | Description | Messages Exchanged |
|---|---|---|
| 1 | **Competent Authority CA (ex. CA BE identified by number `CA-BE-000000555`)** creates and sends a *PullRequest* message requesting from EUDAMED Device service the list of his registered devices for the Manufacturer SRN **`BE-MF-000001201`** . | request0 – PullRequest<br>request1 – eDelivery push<br>request2 – eDelivery As4 push<br>request 3 – same as request1 |
| 2 | **CA** receives an *eDelivery AS4 ack* message indicating that the message was delivered and consumed by **EUDAMED** backend services. | ack1, ack2: *AS4 ack* |
| 3 | **EUDAMED DEVICE service** creates and sends a *PullResponse* message containing the first page of list of devices that belongs to Belgium manufacturer. | response0 – PullResponse<br>response1 - eDelivery push<br>response2 - eDelivery As4 push<br>response3 - same as response1 |
| 4 | **EUDAMED eDelivery AP** receives an *eDelivery AS4 ack* message indicating that the message was delivered and consumed by **CA BE**. | ack3, ack4: *AS4 ack* |

**Messages:**

- ***request0 PullRequest Message***

Belgium CA constructs and sends a ***PullRequest*** message containing a payload with DIDownloadCriteria criteria along with the service *(DEVICE)*, operation *(GET)* and submitter identifier request details

```
<message:PullRequest
.............
xmlns:device="https://ec.europa.eu/tools/eudamed/dtx/datamodel/Entity/Device/v1"
xmlns:party="https://ec.europa.eu/tools/eudamed/dtx/datamodel/Party/v1">

    <message:correlationID>a754cdd7-3602-45b4-a993-c25adb18a60e</message:correlationID>
    <message:creationDateTime>2019-05-22T06:58:45.223+02:00</message:creationDateTime>
    <message:messageID>a64a5a1f-da86-4810-a0de-27d0338811a9</message:messageID>

    <message:recipient>
        <message:node>
            <service:nodeActorCode>EUDAMED</service:nodeActorCode>
            <service:nodeID>eDelivery:EUDAMED</service:nodeID>
        </message:node>
        <message:service>
            <service:serviceAccessToken>34345242342252342342 34</service:serviceAccessToken>
            <service:serviceID>DEVICE</service:serviceID>
            <service:serviceOperation>GET</service:serviceOperation>
        </message:service>
    </message:recipient>
    <message:payload/>
    <message:sender>
        <message:node>
            <service:nodeActorCode>CA-BE-000000555</service:nodeActorCode>
            <service:nodeID>eDelivery:CA-BE-000000555</service:nodeID>
        </message:node>
        <message:service>
            <service:serviceID>REPLY_SERVICE</service:serviceID>
            <service:serviceOperation>GET</service:serviceOperation>
        </message:service>
    </message:sender>
    <message:pageNumber>1</message:pageNumber>
    <message:pageSize>10</message:pageSize>
    <message:criteriaPayload>
        <message:diDownloadCriteria>
            <service:MFActorCode>BE-MF-000001201</service:MFActorCode>
        </message:diDownloadCriteria>
    </message:criteriaPayload>
</message:PullRequest>
```

- ***request1 eDelivery pull request message***

The GW Consumer discovers the GW Provider address/id through the internal services and delivers the request2 PullRequest message to the final destination.

- ***response0 PullResponse Message***

The *EUDAMED* creates a *PullResponse message* with paginated requested information (Devices) to complete the request.

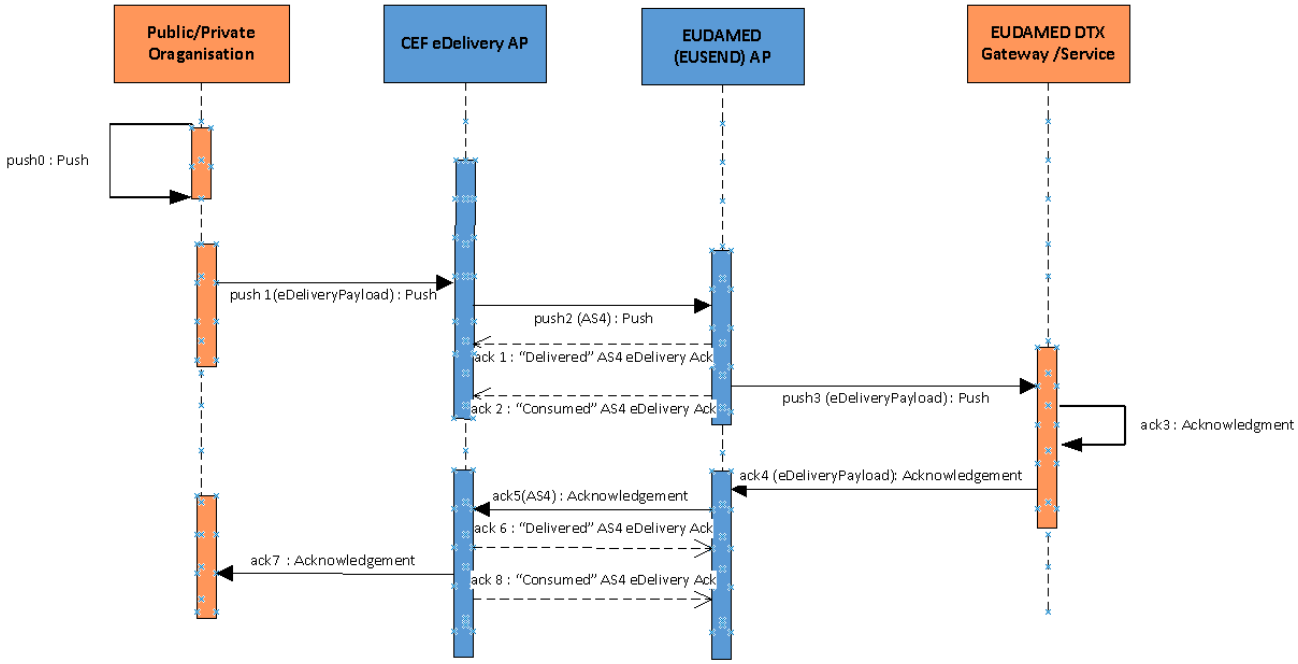
```
<message:PullResponse
.........
xmlns:device="https://ec.europa.eu/tools/eudamed/dtx/datamodel/Entity/Device/v1">

    <message:correlationID>a754cdd7-3602-45b4-a993-c25adb18a60e</message:correlationID>
    <message:creationDateTime>2019-05-22T06:43:12.459+02:00</message:creationDateTime>
    <message:messageID>a64a5a1f-da86-4810-a0de-27d0338811a9</message:messageID>

    <message:recipient>
        <message:node>
            <service:nodeActorCode>AC-BE-000000555</service:nodeActorCode>
            <service:nodeID>eDelivery:AC-BE-000000555</service:nodeID>
        </message:node>
        <message:service>
            <service:serviceID>REPLY_SERVICE</service:serviceID>
            <service:serviceOperation>GET</service:serviceOperation>
        </message:service>
    </message:recipient>
    <message:payload>
        <device:Device xsi:type="device:IVDRDeviceType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
            <device:IVDRBasicUDI>
                .......
                <budi:modelName>
                    <budi:model>1001/202</budi:model>
                    <budi:name>Clear-View Sub-Q</budi:name>
                </budi:modelName>
                 <budi:MFActorCode>BE-MF-000001201</budi:MFActorCode>
            </device:IVDRBasicUDI>
            <device:IVDRUDIDIData>
                <udidi:identifier>
                    <udidi:DICode>UDICODE1</udidi:DICode>
                    <udidi:issuingEntityCode>GS1</udidi:issuingEntityCode>
                </udidi:identifier>
                .........
            </device:IVDRUDIDIData>
        </device:Device>
         <device:Device xsi:type="device:MDRDeviceType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
             <device:MDRBasicUDI>
                    .................
                    <budi:riskClass>IIb</budi:riskClass>
                    <budi:model>SCE8-03-05</budi:model>
                    <budi:identifier>
                           <udidi:DICode>M991CVS1277777</udidi:DICode>
                           <udidi:issuingEntityCode>GS1</udidi:issuingEntityCode>
                    </budi:identifier>
                    <budi:MFActorCode>BE-MF-000001201</budi:MFActorCode>
                    <budi:applicableLegislation>MDR</budi:applicableLegislation>
                    ..............
             </device:MDRBasicUDI>
             <device:MDRUDIDIData>
                    ..............
             </device:MDRUDIDIData>
        </device:Device>
      </message:payload>
      ......
      <message:maxPageNumber>1</message:maxPageNumber>
      <message:pageNumber>1</message:pageNumber>
      <message:pageSize>10</message:pageSize>
      <message:responseCode>SUCCESS</message:responseCode>
</message:PullResponse>
```

## 6.2 Push Pattern

### 6.2.1 Generic push and acknowledgment

The following UML sequence schema describes the message exchange between a public / private organisation (a EUDAMED actor) and EUDAMED services using the *Push* communication pattern.

This process is invoked every time there is a need of uploading / updating of data to EUDAMED MDR.




1. ***push0 -*** *build a* ***Push*** message (Eudamed service model) compliant xml message.
   ***Attributes selection:***
   - **messageID**: a unique identifier, issued by submitter
   - **correlationID**: an identifier that will correlate the request to the response or to the acknowledgements, issued by the requester
   - **sender/service/ServiceID**: identifier of the callback service (for responses and acknowledgements)
   - **sender/node/nodeCode**: identifier of the EUDAMED unique number of the requester (e.g.SRN, CA number, etc.)
   - **sender/node/nodeID**: identifier of the eDelivery partyID
   - **sender/node/nodeProfileToken**: the EUDAMED bearer security token attached to the requester
   - **recipient/service/serviceID**: identifier of the Eudamed service
   - **recipient/service/serviceOperation**: identifier of the Eudamed service operation to uniquely define the service scope
   - **payload/Entities**: contain the main service accepted entity

2. ***push1 (Push)*** **– message** send by the data supplier backend to his CEF eDelivery AP. The push message (Eudamed data model) is embedded into the payload of a soap eDelivery compliant message.
   ***Attributes selection:***
   - **envelope/header/messaging/usermessage/partyinfo/from/partyID**: a unique identifier of the requester's eDelivery AP
   - **envelope/header/messaging/usermessage/partyinfo/to/partyID**: a unique identifier of the EUDAMED MDR eDelivery AP

- **envelope/body/submitRequest/payload:** holds the Base64 format of the *request0* message

3. ***push2 (Push)* – message** send between the two CEF eDelivery AP. The format of the message is AS4 compliant. The message will be marked as *SENDING* in the management interface. The message format is not subject of treatment into the current document.
4. ***ack1 (AS4)* – acknowledgement message** indicates that current message has been delivered to the service provider CEF eDelivery AP. The message will be marked as *DELIVERED*. The message format is not subject of treatment into the current document.
5. ***push3 (Push) – (same as push1) message** delivered to the EUDAMED services backend processor.*
6. ***ack2 (AS4)* - acknowledgement message** indicates that current message has been delivered to the EUDAMED service by the corresponded CEF eDelivery AP. The message will be marked as *DELETED* in the management interface of the requester eDelivery AP management interface.
7. ***ack3 (Acknowlegment) -** build **Acknowledgment*** message (Eudamed service model) compliant xml message to be sent to the requester after invoking the EUDAMED service

   ***Attributes selection:***
   - **messageID**: a unique identifier, issued by submitter
   - **correlationID**: same as correlationID from the request message
   - **sender/service/ServiceID**: identifier of the EUDMAED initiator service
   - **sender/node/nodeCode**: identifier of the EUDAMED eDelivery party identifier
   - **sender/node/nodeID**: identifier of the eDelivery partyID
   - **recipient/service/serviceID**: requester generic callback service
   - **recipient/service/serviceOperation**: requester callback service operation (as specified in the request message)
   - **ackCode**: status code of the service call (success or matching error code)
   - **payload/report**: contain a report of processing statuses and details for each entity that have been sent through the push message.
8. ***ack4 - (Push eDelivery SOAP) – asynchronous*** message send by EUDAMED service provider backend to his CEF eDelivery instance. The protocol and format are the same as *push1*

   ***Attributes selection:***
   - **envelope/header/messaging/usermessage/partyinfo/to/partyID**: a unique identifier of the requester's eDelivery AP
   - **envelope/header/messaging/usermessage/partyinfo/from/partyID**: a unique identifier of the EUDAMED MDR eDelivery AP
   - **envelope/body/submitRequest/payload:** holds the Base64 format of the *ack0* message.
9. ***ack5 – (AS4) –*** message send by EUDAMED eDelivery AP to the initial requester AP. The protocol and format are the same as *push2*
10. ***ack6 – (AS4)* – acknowledgement message -** same as *ack1* delivered to the data provider party backend
11. ***ack7 – (Push eDelivery SOAP) message -**.follow the protocol and format as push1 message*
12. ***ack8 (AS4)* - acknowledgement message** – same as *ack2*.

### 6.2.2 Use case – NON EU MF uploads own device information

**Process:**

| Step | Description | Messages Exchanged |
|---|---|---|
| 1 | **NONEU MF with SRN `JP-MF-000033020`** creates and sends a *Push* message to upload to EUDAMED *DEVICE Upload* service the list of his devices. A callback service is specifies in order the eudamed service to know where to acknowledge the processing of the request. | push0 – Push<br>push1 – eDelivery SOAP push<br>push2 – eDelivery As4 push<br>push3 – same as push1 |
| 2 | **MF** receives an *eDelivery AS4 ack* message indicating that the message was delivered and consumed by **EUDAMED** backend services. | ack1, ack2: *AS4 ack* |
| 3 | **EUDAMED DEVICE Upload service** process the message containing the devices entities and build a report on the result of each entity process outcome. | Ack3 – Acknowledgement<br>Ack4 - eDelivery push<br>Ack5 - eDelivery As4 push |
| | **MF** receives the acknowledgment message from his eDelivery AP on his callback service | Ack6 - same as ack4 |
| 4 | **EUDAMED eDelivery AP** receives an *eDelivery AS4 ack* message indicating that the message was delivered and consumed by **MF**. | ack7, ack8: *AS4 ack* |

**Messages:**

***push0 - Push Message***

**NONEU MF** with SRN JP-MF-000033020 constructs and sends a ***Push*** message containing a payload with devices – IVDR, MDR to be transmitted to Eudamed DTX Device upload service

```
<message:Push>
    <message:correlationID>a754cdd7-3602-45b4-a993-c25adb18a60e</message:correlationID>
    <message:creationDateTime>2019-05-22T07:00:28.066+02:00</message:creationDateTime>
    <message:messageID>a64a5a1f-da86-4810-a0de-27d0338811a9</message:messageID>
    <message:recipient>
        <message:node>
            <service:nodeActorCode>EUDAMED</service:nodeActorCode>
            <service:nodeID>eDelivery:EUDAMED</service:nodeID>
        </message:node>
        <message:service>
            <service:serviceAccessToken>9434524234225234234239</service:serviceAccessToken>
            <service:serviceID>DEVICE</service:serviceID>
            <service:serviceOperation>POST</service:serviceOperation>
        </message:service>
    </message:recipient>
    <message:payload>
          <device:Device xsi:type="device:IVDRDeviceType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
            <device:IVDRBasicUDI>
                    ..............................
                <budi:ARActorCode>BE-AR-000033010</budi:ARActorCode>
                <budi:MFActorCode>JP-MF-000033020</budi:MFActorCode>
                <budi:specialDevice>ORTHOPAEDIC</budi:specialDevice>
                <budi:applicableLegislation>IVDD</budi:applicableLegislation>
                    ................................
            </device:IVDRBasicUDI>
            <device:IVDRUDIDIData>
                <udidi:identifier>
                    <udidi:DICode>UDICODE1</udidi:DICode>
                    <udidi:issuingEntityCode>GS1</udidi:issuingEntityCode>
                </udidi:identifier>
                    ......................................
            </device:IVDRUDIDIData>
        </device:Device>
        <device:Device xsi:type="device:MDRDeviceType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
            <device:MDRBasicUDI>
            <e:state>REGISTERED</e:state>
            <e:version>1.0</e:version>
            <e:versionDate>2018-09-21T00:00:00+02:00</e:versionDate>
            <budi:riskClass>IIb</budi:riskClass>
    <budi:model>SCE8-03-05</budi:model>
    <budi:identifier>
        <udidi:DICode>M991CVS1277777</udidi:DICode>
        <udidi:issuingEntityCode>GS1</udidi:issuingEntityCode>
    </budi:identifier>
    <budi:ARActorCode>BE-AR-000000077</budi:ARActorCode>
    <budi:MFActorCode>JP-MF-000033020</budi:MFActorCode>
    <budi:applicableLegislation>MDR</budi:applicableLegislation>
.......................................
</device:MDRBasicUDI>
 <device:MDRUDIDIData>
            ..............................
        </device:MDRUDIDIData>
        </device:Device>
    </message:payload>
    <message:sender>
        <message:node>
            <service:nodeActorCode>JP-MF-000033020</service:nodeActorCode>
            <service:nodeID>eDelivery:JP-MF-000033020</service:nodeID>
        </message:node>
        <message:service>
            <service:serviceID>REPLY_SERVICE</service:serviceID>
            <service:serviceOperation>GET</service:serviceOperation>
        </message:service>
```

**Ack3** ***- Acknowledgment Message***

EUDAMED service process the message containing the device entities and build a response with and acknowledgment message with the processing status and report.

```
<message:Acknowledgement>
    <message:conversationID>0c4e479e-32a8-4439-9eea-4cd6c25c5745</message:conversationID>
    <message:correlationID>a754cdd7-3602-45b4-a993-c25adb18a60e</message:correlationID>
    <message:creationDateTime>2019-05-29T00:53:47.165+02:00</message:creationDateTime>
    <message:messageID>a64a5a1f-da86-4810-a0de-27d0338811a9</message:messageID>
    <message:recipient>
        <message:node>
            <service:nodeActorCode>JP-MF-000033020</service:nodeActorCode>
            <service:nodeID>eDelivery:JP-MF-000033020</service:nodeID>
        </message:node>
        <message:service>
            <service:serviceID>REPLY_SERVICE</service:serviceID>
            <service:serviceOperation>GET</service:serviceOperation>
        </message:service>
    </message:recipient>
    <message:payload />
    <message:sender>
        <message:node>
            <service:nodeActorCode>EUDAMED</service:nodeActorCode>
            <service:nodeID>eDelivery:EUDAMED</service:nodeID>
        </message:node>
        <message:service>
            <service:serviceID>DEVICE</service:serviceID>
            <service:serviceOperation>GET</service:serviceOperation>
        </message:service>
    </message:sender>
    <message:responseCode>SUCCESS</message:responseCode>
</message:Acknowledgement>
```

# 7 Data Security and Integrity

## 7.1 High Level Architecture for EUDAMED Message Exchange

The EUDMAED DTX solution, in the context of Machine to Machine data exchange (M2M) requires composition of three independent building blocks as described in the integration architecture diagram (Figure 8).

From security perspective, the security controls groups should be seen as being distributed over all components in order to perform:

- message encryption and integrity;
- message authentication;
- authorisation;
- service capabilities;
- information exchange patterns.

The EUSEND / eDelivery building block defines the use of Access Points implementing the AS4 Messaging Protocol according to the guidelines defined in the e-SENS profile/ implementation guidelines.

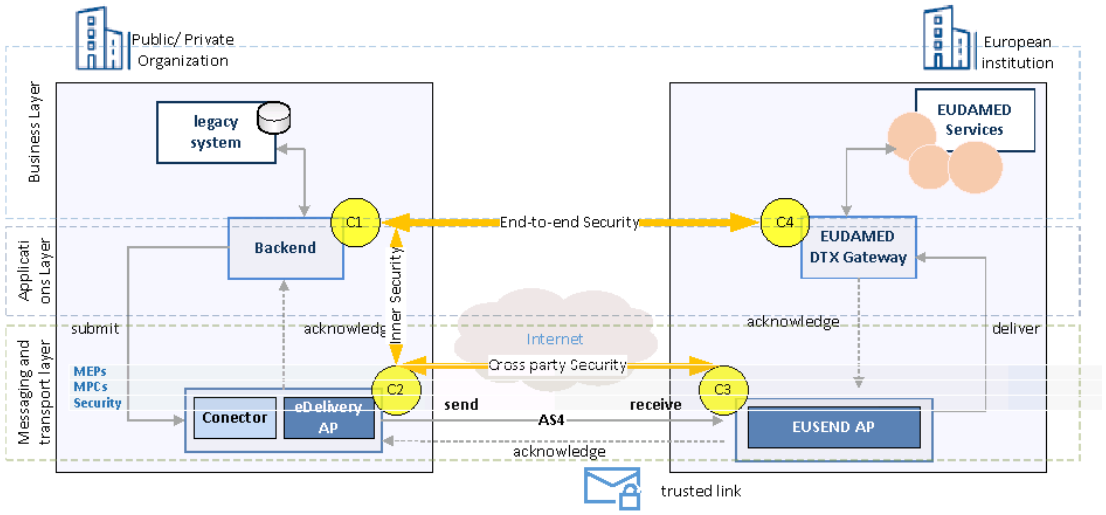



Figure 8 EUDAMED DTX solution integration architecture for M2M

It is given to EUDAMED DTX Service to rely on the transversal protocols and functionalities provided by CEF (EUSEND) components / functionalities layer as presented:

- **Message exchange patterns** (MEPs) based on the ebMS 3.0 / e-SENS AS4 ebHandler profile;
- **Reliability -** synchronous acknowledgments of receipt;
- **Message Partition Channels (MPCs) –** facilitate implementation of different communication protocol polices and message prioritization[1]**;**
- **Security / Authentication / Authorization –** supporting the EUDAMED Data Integration security policy aspects in terms of participants authentication and massage exchange authorization, digital signature and encryption on the data, non-repudiation mechanism as well as the duplicate messages detection;

This architectural solution delegates some functionalities/responsibilities related to the communication protocol from the EUDAMED DTX Gateway to the transport layer, EUSEND.

[1] Not in the scope of the current implementation of EUDAMED DTX Solution

CEF EUSEND / e-Delivery building blocks are relaying on B2B communication protocols with a specific conformance profile (CP) AS4 ebHandler for the ebMS 3.0 (e-SENS profile). Compared to the AS4 ebHandler Conformance Profile, e-SENS profile, from a security point of view, updates or adds some functionality:

- Transport Layer Security (TLS), if handled in the AS4 handler, is profiled and is mandatory;
- The WS-Security version is the 1.1.1 version;
- Algorithms specified for securing messages at the Message Layer are updated to current guidelines and use of signing and encryption is mandatory;
- All payloads are exchanged in separate MIME parts;
- Receipts and errors are reported synchronously only;
- WS-Security support is limited to the X.509 Token Profile. The use of *UserName* Tokens is not supported. When using digital signatures or encryption, an AS4 MSH implementation is required to use the Web Services Security X.509 Certificate Token Profile. The certificate authority (CA) / PKI support instance is currently provided by CEF (EUSEND service).

## 7.2 Implemented security capabilities at EUDAMED DTX service model

***Security Token Authentication of Services / Messages***

An additional authorisation mechanism is set in place at the application level of the EUDAMED DTX solution architecture. This aims to add C1-C4 authentication in situations where XML digital signatures are not possible (3rd party providers). The solution bases on security token profile or bearer token principle (randomise 128 long sequence characters). Access tokens are managed inside EUDAMED application and provided to the data owners (EO, CA, NB, etc.) and are identifying an entity and a service access.

A responding MSH mayrespond with an error if a received ebMS message does not meet the security policy of the responding MSH. For example, a security policy might indicate that messages with unsigned parts of the SOAP Body or eb:Messaging Container element are unauthorized for further processing.

***Electronic Time Stamp of Message***

Data in electronic form which binds other data in electronic form to a particular time establishing evidence that the latter data existed at that time. It establishes existence of the data at a given time (ensures date and time of the data).