



# EESSI - RINA

## Business Messaging Interface (BMI)

*Architecture & Design Specifications*

## Table of Contents

|                                                                                 |    |
|---------------------------------------------------------------------------------|----|
| Table of Contents .....                                                         | 2  |
| 1 EESSI Business Messaging Overview .....                                       | 6  |
| 2 Common API Concepts and Features .....                                        | 10 |
| 2.1 User authentication.....                                                    | 10 |
| 2.2 User authorization .....                                                    | 10 |
| 2.3 Audit trail.....                                                            | 10 |
| 2.3.1 Audit Trail Architecture.....                                             | 10 |
| 2.3.2 Audit Trail Persistence and Retrieval .....                               | 11 |
| 2.3.3 General Approach to audit trail data model.....                           | 11 |
| 2.3.4 Particular approach of audit trailing for Business Messaging Interface .. | 15 |
| 2.4 Exception management.....                                                   | 16 |
| 3 Functions and use cases.....                                                  | 17 |
| 3.1 Configure function .....                                                    | 17 |
| 3.1.1 NA Repositories' Business Message Configure.....                          | 17 |
| 3.1.2 NA Repositories' synchronisation configure .....                          | 18 |
| 3.2 Message exchange function.....                                              | 19 |
| 3.2.1 NA Sends Message to AP.....                                               | 20 |
| 3.2.2 NA Receives Message from AP.....                                          | 21 |
| 3.2.3 NA Receives Status Update from AP.....                                    | 22 |
| 3.3 Repositories synchronisation.....                                           | 22 |
| 3.3.1 Receive IR synchronisation data .....                                     | 23 |
| 3.3.2 Send IR request for synchronisation.....                                  | 24 |
| 3.3.3 Receive CDM synchronisation status request.....                           | 24 |
| 3.3.4 Send CDM status / request for synchronisation .....                       | 25 |
| 3.3.5 Receive CDM Synchronisation data .....                                    | 26 |
| 4 Behaviour.....                                                                | 27 |
| 4.1 Sequence diagrams.....                                                      | 27 |
| 4.1.1 NA Sends Message to AP.....                                               | 29 |
| 4.1.2 NA Receives Message from AP.....                                          | 32 |
| 4.1.3 NA Receives Status Update from AP.....                                    | 34 |
| 4.1.4 Receive IR Synchronisation data .....                                     | 35 |
| 4.1.5 Send IR request for synchronisation.....                                  | 35 |
| 4.1.6 Receive CDM Synchronisation status request.....                           | 36 |
| 4.1.7 Send CDM status/ request for synchronisation.....                         | 37 |
| 4.1.8 Receive CDM Synchronisation data .....                                    | 38 |
| 4.2 State machine diagrams.....                                                 | 39 |
| 4.2.1 Business message state machine .....                                      | 39 |
| 5 RINA Business Messaging Interface .....                                       | 40 |
| 5.1 Service contract.....                                                       | 40 |
| 5.1.1 APClient .....                                                            | 40 |
| 5.1.2 APClientAdmin .....                                                       | 40 |
| 5.1.3 APListener.....                                                           | 41 |
| 5.2 Data contract .....                                                         | 43 |
| 5.2.1 Message model .....                                                       | 43 |
| 5.2.2 Configuration model .....                                                 | 46 |
| 5.2.3 Standard Business Document Header Model.....                              | 52 |
| 5.3 Builders .....                                                              | 57 |
| 5.3.1 SbdhBuilder.....                                                          | 58 |
| 5.3.2 MessageBuilder .....                                                      | 61 |
| 5.4 Fault contract.....                                                         | 64 |
| 5.4.1 MessageProcessingException.....                                           | 64 |
| 5.4.2 MessageTransformerException .....                                         | 65 |
| 5.4.3 MessageTranslationException.....                                          | 65 |
| 5.4.4 ValidationException .....                                                 | 65 |
| 5.4.5 MessageValidationException .....                                          | 65 |
| 5.4.6 StructureValidationException .....                                        | 65 |

|       |                                                                     |     |
|-------|---------------------------------------------------------------------|-----|
| 5.4.7 | MessageSigningException.....                                        | 66  |
| 6     | RINA Business Messaging SOAP Web Service Interface.....             | 67  |
| 6.1   | Service contract.....                                               | 67  |
| 6.1.1 | APClient Web Service.....                                           | 67  |
| 6.1.2 | APListener Web Service.....                                         | 70  |
| 6.2   | Data contract.....                                                  | 77  |
| 6.2.1 | Message model.....                                                  | 77  |
| 6.2.2 | Configuration model.....                                            | 79  |
| 6.2.3 | Standard Business Document Header Model.....                        | 79  |
| 6.3   | Builders.....                                                       | 79  |
| 6.3.1 | SbdhBuilder.....                                                    | 79  |
| 6.3.2 | SoapMessageBuilder.....                                             | 80  |
| 6.4   | Fault contract.....                                                 | 83  |
| 7     | RINA Technical Messaging Interface.....                             | 84  |
| 7.1   | Service contract.....                                               | 84  |
| 7.1.1 | Write.....                                                          | 84  |
| 7.1.2 | Read.....                                                           | 84  |
| 7.2   | Data contract.....                                                  | 84  |
| 7.2.1 | MessageMetaDataType.....                                            | 84  |
| 7.2.2 | SignalMessage.....                                                  | 89  |
| 7.2.3 | PMode.....                                                          | 89  |
| 7.2.4 | Pulling.....                                                        | 89  |
| 7.3   | Builders.....                                                       | 90  |
| 7.3.1 | MmdBuilder.....                                                     | 90  |
| 7.3.2 | PModeBuilder.....                                                   | 91  |
| 7.3.3 | PullConfigurationBuilder.....                                       | 93  |
| 7.4   | Fault contract.....                                                 | 94  |
| 7.4.1 | StructureValidationException.....                                   | 94  |
| 8     | Samples.....                                                        | 95  |
| 8.1   | RINA Business Messaging Interface.....                              | 95  |
| 8.1.1 | How to create a configuration.....                                  | 95  |
| 8.1.2 | How to create the listener implementation.....                      | 96  |
| 8.1.3 | How to setup the ApClient.....                                      | 97  |
| 8.1.4 | How to stop the ApClient.....                                       | 97  |
| 8.1.5 | How to build a business message.....                                | 97  |
| 8.1.6 | How to send a message.....                                          | 98  |
| 8.2   | RINA Business Messaging SOAP Web Service Interface.....             | 99  |
| 8.2.1 | How to create an AP JSON configuration.....                         | 99  |
| 8.2.2 | How to create the listener implementation.....                      | 101 |
| 8.2.3 | How to build a SOAP business message.....                           | 102 |
| 8.2.4 | How to send a message.....                                          | 102 |
| 8.3   | RINA Technical Messaging Interface.....                             | 104 |
| 8.3.1 | How to build the Holodeck PMode for sending messages to AP.....     | 104 |
| 8.3.2 | How to build the Holodeck PMode for retrieving messages from AP.... | 104 |
| 8.3.3 | How to build a Holodeck message for sending.....                    | 104 |
| 8.3.4 | How to write & read MSH messages.....                               | 104 |
| 9     | APPENDIX.....                                                       | 105 |
| 9.1   | ApClient DB solution.....                                           | 105 |
| 9.1.1 | Introduction.....                                                   | 105 |
| 9.1.2 | BMI deployment Scenarios with DB.....                               | 105 |
| 9.1.3 | Database Solution Description.....                                  | 106 |

## Document Control Information

| Document Control                                      | Value                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Project Title</b>                                  | Electronic Exchange of Social Security Information (EESSI)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Document Title</b>                                 | EESSI – RINA - Business Messaging Interface (BMI)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Document Category</b>                              | Architecture & Design Specifications                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Revision</b>                                       | -                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Component Version</b>                              | -                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Last Publication Date</b>                          | 22/12/2020                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Project Milestone</b>                              | EESSI-2020                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Document Status</b>                                | Final                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Sensitivity (TLP)</b><br><b>Distribution terms</b> | <p>Traffic Light Protocol (TLP) = "GREEN"</p> <div data-bbox="734 1055 1165 1256"> <p><b>GREEN</b></p> <p>Only share inside EESSI community<br/>See Author guidance before sharing</p> </div> <p>The distribution of this document is done strictly in line with the Traffic Light Protocol (TLP) established by the European Commission's note AC 790/15 REV for the EESSI project documentation.</p> <p>In line with the note AC 790/15 REV, this document is labelled as TLP = "Green". Therefore, it can be circulated widely within the EESSI community. However, the document or the information herein may not be published or posted on the Internet, nor released outside of the EESSI community.</p> |
| <b>Connected/<br/>Embedded Files</b>                  | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Authors</b>                                        | European Commission, DG EMPL F5, EESSI RINA                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Revised by</b>                                     | European Commission, DG EMPL F5, EESSI QA                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Approved by</b>                                    | European Commission, DG EMPL F5, EESSI PM                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

## Document history

| Project Milestone /<br>Released Component<br>version | Changes/Corrections<br>Description                                               |
|------------------------------------------------------|----------------------------------------------------------------------------------|
| <b>EESSI-2019 / RINA 5.6.2</b>                       | Update the description of the Parameter<br>"ApplicationId"                       |
| <b>EESSI 2020 / RINA 6.2.1</b>                       | Add the filename of the attachment in the BMI-WS                                 |
|                                                      | Align the description to new RINA developments<br>and architectural improvements |

## 1 EESSI Business Messaging Overview

Part of the National Institutions Domain, RINA (Reference Implementation for National Applications) consists in a collection of infrastructure and communication services, foundation, repository and publishing services, business, integration and user interface services which will provide for clerks and their organisations the tools to implement the International Protocol of Data Exchange based on Structured Electronic Documents in Social Security belonging to European Community Member States and Associated States.

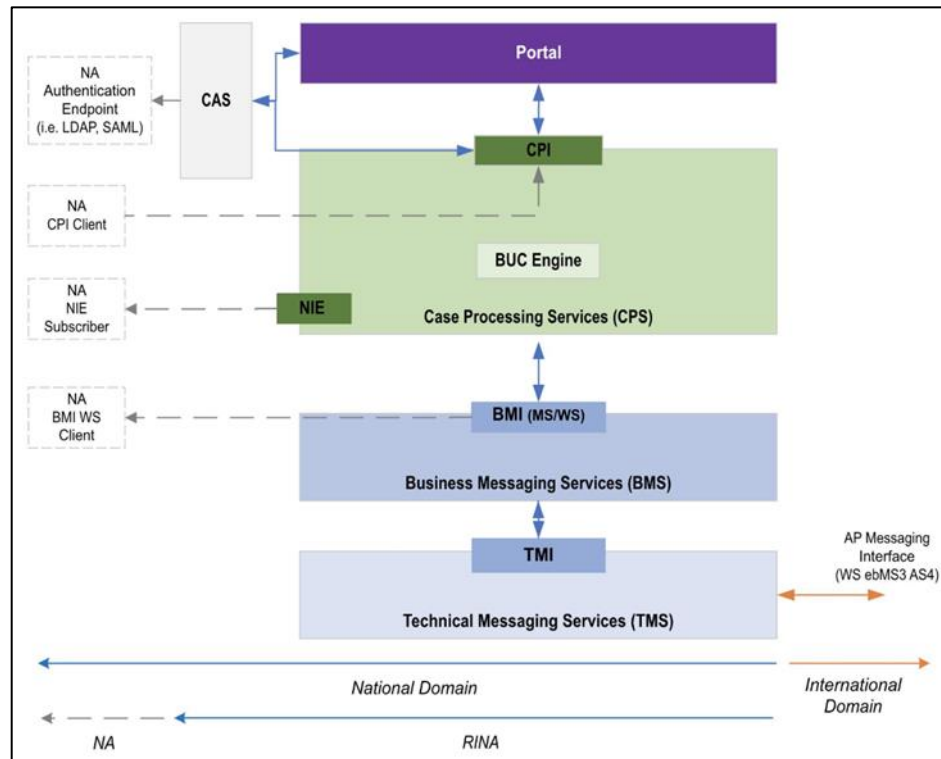


Figure 1 - Overview of RINA interfaces and layers

RINA is based on 4 layers which are built one on top of the other, plus the CAS module:

- *Technical Message Services (TMS)* - provide the mechanism for sending/receiving technical messages to the Access Point. It works based on the protocol ebMS3.0 - AS4 profile. It is an internal layer and cannot be used by third parties, being its unique client the RINA BMS.
- *Business Messaging Services (BMS)* - provide a reusable component for sending business messages to the Access Point via the TMS layer. These services provide the translation (business message to technical message and vice versa), transformation (converting IDs to GUIDs), validation and signing of the business messages for correct exchange with the EESSI environment, and the correct reception of messages from other institutions. It offers the BMI WS interface making available the integration of third party applications to the EESSI ecosystem.
- *Case Processing Services (CPS)* – provide a state full component built on top of the Business Messaging Services that manage cases in a structured manner taking care of all the issues regarding case flow, documents, notifications, user management and provides two interfaces for external access (NIE and CPI)
- *Portal* – provides an UI build on top of the Case Processing Services (CPS) having administration consoles and case Processing modules

- *CAS Authentication Server* – the SSO (Single Sign On) server providing the authentication to users/clients requesting to use a service

The scope of this document is to provide a specification document for the Business Messaging Interface and Technical Messaging Interface, both part of the Business Messaging Services layer. Currently the communication with the BMI layer is only in Java, but in the future, a .NET and Web Service implementation will be available as well.

A short description of the document structure is given below:

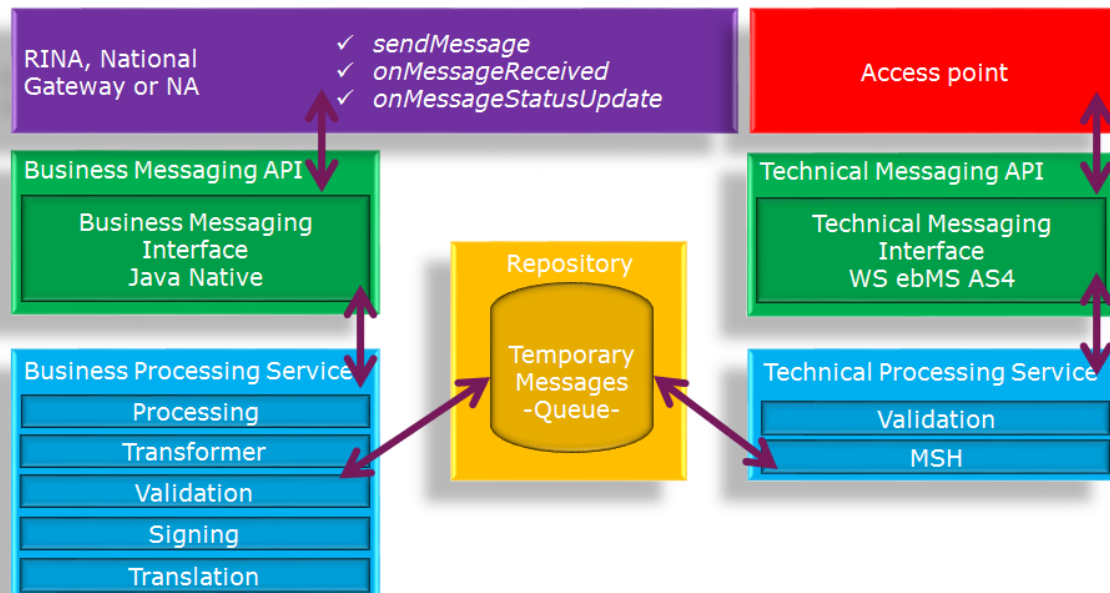
- Service overview – this chapter, an introduction to what this service is about, where its place is and why it is needed;
- Common concepts and features – vertical components, like user authentication & authorization, exception management and audit trail;
- Functions and use cases – a presentation of high level functions and use cases that drive this service component;
- Behaviour diagrams – sequence diagrams that describes the flow between components and state machine diagrams that depicts the states and transitions the message can have;
- Business messaging interface & Technical messaging interface – the two components that the user can use, both having the structure Below:
- Service interfaces – a description of the interfaces provided;
- Service contract – a description for all the methods exposed through the interfaces;
- Data contract – the data model used for this service;
- Builders – the available builders that help in constructing the objects used in the service interfaces;
- Fault contract – the exception model used for this service;
- Samples – some samples that show how to use this library;
- Appendix – presentation of the ApClient database solution.

This software component, the implementation of the BMS Layer, provides a JAVA library that allows national systems or RINA case management service to exchange messages with EESSI access point infrastructure.

From the functional perspective, the implementation of BMS Layer provides two interfaces: one that is able to communicate with the national applications, described in this document and one able to handle the ebMS3/AS4 profile.

This software component uses an ebMS MSH (Message Service Handler) AS4 implementation and therefore it can be thought as an ebMS client that provides core messaging functions like sending or receiving business messages on top of the ebMS transport low level.

The following picture shows the Business Messaging Interface white box view and how the components interact one with each other:



The focus of this document is on the Business Messaging API (the left green box) that provides the following functionality:

- Capabilities to send business messages to a preconfigured EESSI access point using the Message Service Handler;
- Notify a registered listener about a received business message;
- Notify a registered listener about a status update of a business message;
- The status updates of a business message can be either the business message acknowledgement or the message delivery failure;
- Configuration via a file (XML or similar) or by calling the library;
- Configuration capabilities include the setting of AS4 Message Partition Channel (hereunder MPC), application name, inbox endpoint, outbox endpoint, usage of compression, maximum message size, pull or push behaviour and security settings;
- Internal transformer service, in the future configurable to plug in and use a custom transformer library;
- Internal validation service, in the future configurable to delegate the validation capabilities to a custom business message validator;
- Able to convert a technical ebMS3 User Message from a business message and vice versa and to convert an ebMS Error Message or acknowledgement into a message status update;
- The business message statuses can be: "ready", "sent", "delivered", "error";
- The business message consists in a business document header, a structured electronic document and a list of attachments;
- The structured electronic documents and the attachments must be packaged in the business message as input streams in order to handle large amounts of data.
- The business message must accommodate different structure of business document header.

**Note:** Business signature is under discussion and possible subject of change.

Also the JAVA library provides a low level API called the RINA Technical Messaging Interface that implements a file drop concept. The Technical Messages Queue is a file based queue, so this API is able to deliver the following output files to MSH:

- Message Metadata Files – low level send messages in the MSH domain
- MSH P-mode files – configuration file that determines how the MSH behaves

As input files, the following file types are delivered to the user:

- Message Metadata Files – low level received messages in the MSH domain
- Message Status Metadata Files – low level status updates about a particular message

## 2 Common API Concepts and Features

### 2.1 User authentication

N/A

### 2.2 User authorization

N/A

### 2.3 Audit trail

The audit trail is a sequence of chronological records of EESSI system related events to enable the investigation of different type of incidents. These events associate to domain (business) auditing from the perspective of both the case management and application (i.e. configuration elements) context. The information in audit trail assists investigations of suspected incidents and efficient research in terms of user accountability and reconstruction of events.

The audit trail is a fundamental part of EESSI system and this section aims to provide the reader with the audit trail architecture and its components, as well as an overview on the data model.

#### 2.3.1 Audit Trail Architecture

The unique component generating Audit information at the BMS domain is the APClient as depicted in the next picture.

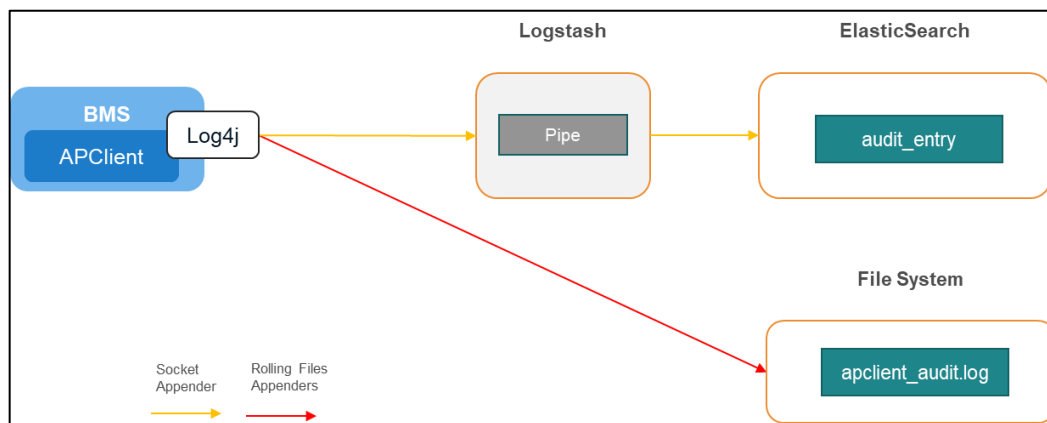


Figure 2: RINA BMS Audit Trail architecture

This is only a part of the full RINA audit mechanism. The full architecture with all the details on components, data flows, repositories and data model can be checked consulted in the *RINA Architecture Overview* document<sup>1</sup>

<sup>1</sup> Document available at Confluence's RINA Architecture space: <https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/EESSI+System+General+Architecture+Documentation>

### 2.3.2 Audit Trail Persistence and Retrieval

*Business Messaging Services (APClient component)* store audit records leveraging the log4j infrastructure.

The default configuration that comes with RINA is to persist the audit events in both a log file using the FileAppender but also to Elasticsearch via the Logstash and SocketAppenders.

Audit Events of BMI can be retrieved by Elasticsearch, or/and log file, depending on the configuration of log4j (*EESSI - RINA - Operations Manual*<sup>2</sup>)

BMS is responsible for generating events related to the business messaging exchange.

### 2.3.3 General Approach to audit trail data model

All different RINA components use the same Audit Trail Data model to enable consistency and easier aggregation.

The following diagram shows the set of UML classes involved in RINA Audit Trail, as well as their relationships. Right after it, a detailed description of each class as well as its attributes is provided to the reader.

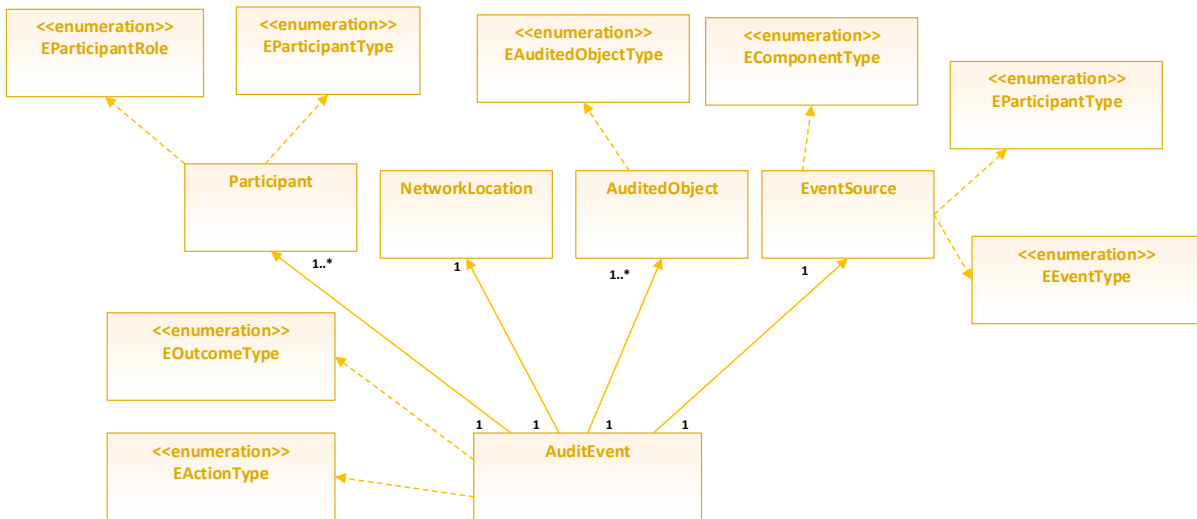


Figure 3: RINA Audit Data Model

Regarding the different domains of the Audit Model, their representation associated for the BMS component layer (regarding the ApClient component) is based on JSON Document for in Elasticsearch database / Audit File.

<sup>2</sup> Document available at Confluence's RINA Operations space: <https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Operations+Documentation>

### 2.3.3.1 Audit Event

An audit event is any triggered event for which a record will be created in the audit-trailing repository.

The audit event is identified by its ID and contains the information detailed in the next table.

| Attribute       | Description                                                                                                   | Data Type           |
|-----------------|---------------------------------------------------------------------------------------------------------------|---------------------|
| id              | The ID of the audit event                                                                                     | String              |
| Action          | The type of action which is executed which can be one of the following: create, read, update, delete, execute | EActionType         |
| userName        | The username of the user who triggered the action                                                             | String              |
| networkLocation | The network location are the identification details (machine & IP) on which the action has been triggered     | NetworkLocation     |
| Date            | Date of the audit event                                                                                       | DateTime            |
| source          | The source of the event which has three attributes (category, component, type)                                | EventSource         |
| outcome         | The outcome of the action triggered which can be success, error or unauthorised                               | EOutcomeType        |
| outcomeError    | When outcome is error or unauthorised, the outcomeError contains the error message.                           | String              |
| auditedObjects  | The audited objects which are identified by id, type and details                                              | List<AuditedObject> |
| participants    | The list of participants to the triggered action.                                                             | List<Participant>   |
| tenantId*       | The tenantId of the tenant that triggered the action.                                                         | String              |

\* *tenantId* is only available in the persisted representation (under File/ES for BMS component layer) – hence not available through the CPI or Admin Portal

### 2.3.3.2 Audited Object

Defines the details about the audit object and important/relevant details of the respective audit object which need to be recorded in the audit trail about respective event.

| Attribute | Description                                                                               | Data Type          |
|-----------|-------------------------------------------------------------------------------------------|--------------------|
| id        | The id of the audited object                                                              | String             |
| Type      | The type of the audited object (e.g. attachment, case credential, policy, document, etc.) | EAuditedObjectType |
| Details   | The relevant details related to the triggered event.                                      | String             |

### 2.3.3.3 Event Source

Details and identify the source of an event.

| Attribute     | Description                                                                                             | Data Type      |
|---------------|---------------------------------------------------------------------------------------------------------|----------------|
| categoryType  | The category of an event source (i.e. messaging, security, business)                                    | ECategoryType  |
| componentType | The component of an event source (e.g. attachment, case, business messaging, technical messaging, etc.) | EComponentType |
| eventType     | The type of an event source which consist in the actual action which triggered the event                | EEventType     |

### 2.3.3.4 Network Location

This defines the network location details for a triggered event. These details will consist in the machine on which the event was triggered and its corresponding IP in the network.

| Attribute | Description                                          | Data Type |
|-----------|------------------------------------------------------|-----------|
| Machine   | The machine from which the action has been triggered | String    |
| Ip        | The IP of the machine                                | String    |

### 2.3.3.5 Participant

This describes the participant(s) which are taking part to a triggered event.

| Attribute | Description                                                     | Data Type        |
|-----------|-----------------------------------------------------------------|------------------|
| id        | The id of the participant                                       | String           |
| type      | The type of the participant which can be person or organisation | EParticipantType |

### 2.3.3.6 EActionType enumeration values and description

**2.3.3.7 The action type describes the actions which correspond to a triggered event. This can be one of the following: create, read, update, delete, execute.**

| Value | Name    | Description                                          |
|-------|---------|------------------------------------------------------|
| cre   | CREATE  | Creation of an audited object type                   |
| re    | READ    | Read action of an audited object type                |
| upd   | UPDATE  | Update an audited object type                        |
| del   | DELETE  | Delete an audited object type                        |
| exec  | EXECUTE | Execution action (when none of the above are happen) |

### 2.3.3.8 EAuditedObjectType enumeration

The audited object type defines the relevant audited objects for a triggered event.

Please note that even though the types for audited objects are similar with the component types these are intended for a different use.

| Value | Name              | Description                                                                                                                                              |
|-------|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| bmsg  | BUSINESS_MESSAGE  | Used when the triggered event involves also an action to a business message                                                                              |
| tmsg  | TECHNICAL_MESSAGE | Used when the triggered event involves also an action to a technical message (received or sent a technical ebMS3 User Message)                           |
| ackm  | BUSINESS_ACK      | Used when the triggered event involves also an action to an acknowledgement technical message (acknowledgement on ebMS3 received or sent Signal Message) |

### 2.3.3.9 ECategoryType enumeration values and description

The category type is used in order to specify the category of an event source.

| Attribute | Name      | Description                                                                               |
|-----------|-----------|-------------------------------------------------------------------------------------------|
| mess      | MESSAGING | Used for event source when the triggered action involves messaging events/messaging layer |

### 2.3.3.10 EComponentType values and description

Defines the component types relevant for an event source.

| Value | Name               | Description                                                                     |
|-------|--------------------|---------------------------------------------------------------------------------|
| bmsg  | BUSINESS_MESSAGING | Will be chosen if the triggered event involves the business messaging component |

### 2.3.3.11 EEventType values and description

Defines all types of the events, which trigger the audit logging process.

| Value                   | Name                                   | Description                                                                                                                                                 |
|-------------------------|----------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| send_bmsg               | SEND_BUSINESS_MESSAGE                  | An event consisting in send a business messages to a preconfigured EESSI access point using the Holodeck AS4. This record exists in BMS ApClient audit log. |
| notif_reg_lsn_bmsg      | NOTIFY_ABOUT_RECEIVED_BUSINESS_MESSAGE | An event consisting in notify a register listener about a received business message. This record exists in BMS ApClient audit log.                          |
| notif_reg_lsn_stat_bmsg | NOTIFY_ABOUT_STATUS_UPDATE             | An event consisting in notify a register listener about a status update of a business message (ack or error). This record exists in BMS ApClient audit log. |

### 2.3.3.12 EOutcomeType values and description

The outcome type is an audit event attributes which specify how a triggered action ended. This can be one of the following: success, error, or unauthorised.

| Value | Name             | Description                                                              |
|-------|------------------|--------------------------------------------------------------------------|
| succ  | SUCCESS          | Used when the action which generates the audit event ends with success.  |
| err   | ERROR            | Used mainly when a HTTP 500 type of exception are occurring              |
| rjct  | UNAUTHORIZE<br>D | Used mainly when a HTTP 401 & HTTP 403 types of exceptions are occurring |

### 2.3.3.13 EParticipantRole enumeration values and description

This section describes the participant role in an audited event.

| Value | Name     | Description                                                                                  |
|-------|----------|----------------------------------------------------------------------------------------------|
| sndr  | SENDER   | Used if the event is in context of messaging and the participant is the sender institution   |
| rcvr  | RECEIVER | Used if the event is in context of messaging and the participant is the receiver institution |
| subj  | SUBJECT  | Used in context of non-messaging events                                                      |

### 2.3.3.14 EParticipantType enumeration values and description

This section describes the participant type in an audited event, which can be person or organisation depending on the event specifics.

| Value | Name             | Description                                          |
|-------|------------------|------------------------------------------------------|
| per   | PERSON           | Used if the audited event participant is a person    |
| org   | ORGANISATIO<br>N | Used if the audited event participant is non-person. |

## 2.3.4 Particular approach of audit trailing for Business Messaging Interface

This mapping aim to illustrate what are the possible associations between all the categories, components, events and their corresponding audited objects (details).

| Category         | Component           | Event                                                                            | Audited Objects                     |
|------------------|---------------------|----------------------------------------------------------------------------------|-------------------------------------|
| <b>Messaging</b> | #Business Messaging | Send a business messages to a preconfigured EESSI access point using the MSH AS4 | BusinessMessage<br>Case<br>Document |
|                  |                     | Notify a register listener about a received business message                     | BusinessMessage<br>Case<br>Document |

| Category | Component | Event                                                                                 | Audited Objects                     |
|----------|-----------|---------------------------------------------------------------------------------------|-------------------------------------|
|          |           | Notify a register listener about a status update of a business message (ack or error) | BusinessMessage<br>Case<br>Document |

## 2.4 Exception management

This section explains how the library manages unexpected behaviour.

When an error occurs during an API call the library will throw a Java exception that is defined in the Fault contract ([5.4](#), [7.4](#)).

There are three generic types of exception that might occur

- Validation exception – this kind of exception is thrown when the message, either business or technical doesn't pass a certain validation check;
- Message translation exception – this kind of exception is thrown when the message could not be converted from the business domain to the technical domain or the other way around;
- Message processing exception – this kind of exception is thrown when there was a problem during the actual sending or receiving of the message or some other error occurred.

In case of the validation exception the API user may update the business message that is being sent and retry to send it.

In case of the translation exception the API user may check the repository queue to see if there is enough space, the attachments are provided or if the business message is valid.

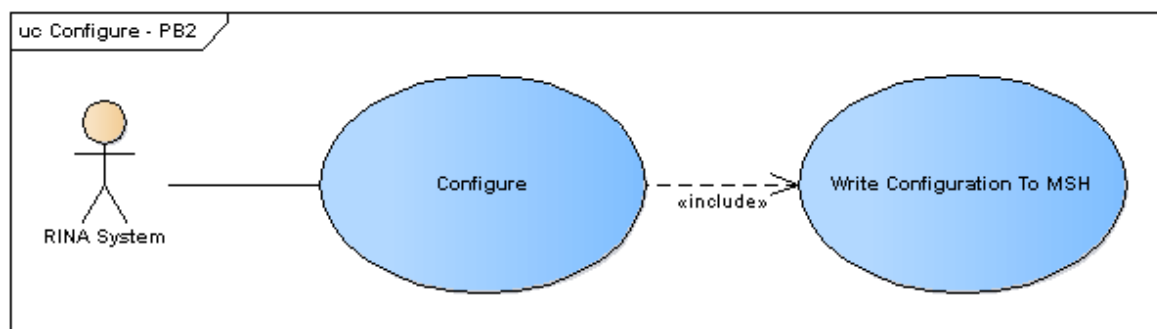
In case of the message processing exception there is not much to do, except for checking the connection configuration and make sure the client is not stopped.

## 3 Functions and use cases

The two top level functions of the EESSI system shown below describe the sum of actions that allow National Applications to configure a Message Handling Service and to send and receive business messages according to the EESSI data exchange contracts, in a secure and reliable way.

### 3.1 Configure function

The configuration function provides a way for the users to configure the library for using with a specific access point.



#### 3.1.1 NA Repositories' Business Message Configure

##### 3.1.1.1 Description

This use case diagram shows that in order to configure the service, the system uses another component to actually write the configuration to a Message Service Handler.

The included uses case may also be fulfilled separately from the main use case.

In any case, the user must be able to configure the library for connecting to the Access Point, and the following properties help in having a flexible connection setup :

- End to end connection is specified by setting up the inbox and outbox URL;
- The institution name and the access point should be part of the configuration to identify the two parties involved in exchanging the messages;
- The messages exchanged must have a known capacity limit so that larger messages are rejected because it exceeds the maximum message size. This could be an issue when very large messages overload the network and it slows down the EESSI system;
- The messages exchanged should support compression to minimize the amount of message size sent across the network;
- The system should be reliable, so that in case of errors, the system tracks down and retries the messages that had errors;
- Also as part of the reliability, the configuration should allow setting up the request interval between retries;
- All the security concerns like the key store and the trust store location and password must be configurable.
- The key store and the trust store installation is out of scope of this use case.

So this use case covers the entire configuration needed to have a secure, reliable end to end connection between the national application and the access point.

The messages exchanged capacity limit could be moved later in the business validation rules that will be executed by the business validation component.

### 3.1.1.2 Support API

The provided methods of the support API, are the following:

- `APClientAdmin` (§ `APClientAdmin`)
- `Write` (§ `Write`)
- `PModeBuilder` (§ `PModeBuilder`)
- `PullConfigurationBuilder` (§ `PullConfigurationBuilder`)

## 3.1.2 NA Repositories' synchronisation configure

### 3.1.2.1 Description

The synchronisation configuration use case describes the settings which can be performed related to the protocol, inbox and outbox services used for repositories' synchronisation.

The NA's administrator can update the above settings within the administrative console of the portal.

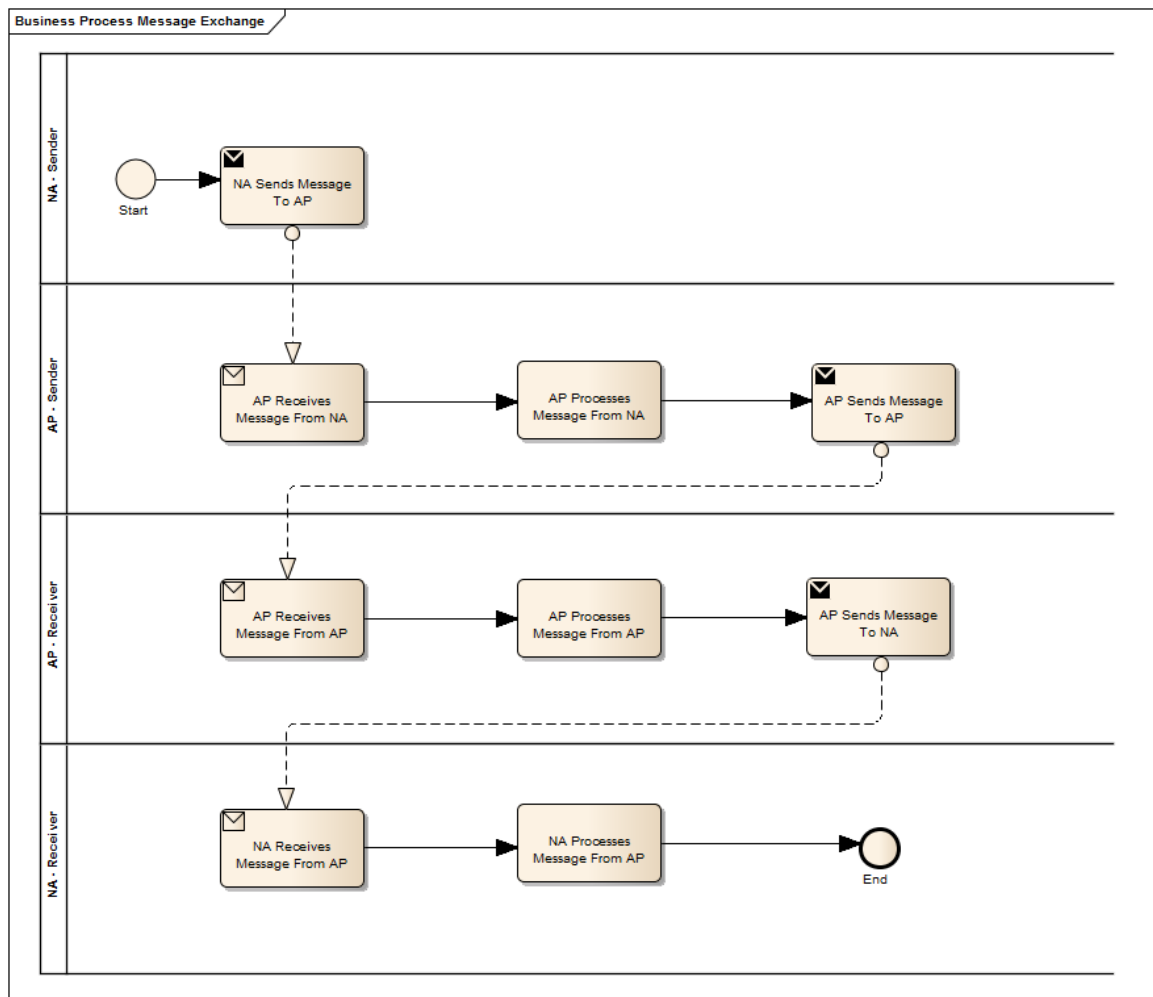
### 3.1.2.2 Support API

The provided methods of the support API, are the following:

- `APClientAdmin` (§ `APClientAdmin`)
- `Write` (§ `Write`)
- `PModeBuilder` (§ `PModeBuilder`)
- `PullConfigurationBuilder` (§ `PullConfigurationBuilder`)

### 3.2 Message exchange function

The message exchange function covers the whole chain of message exchange from the

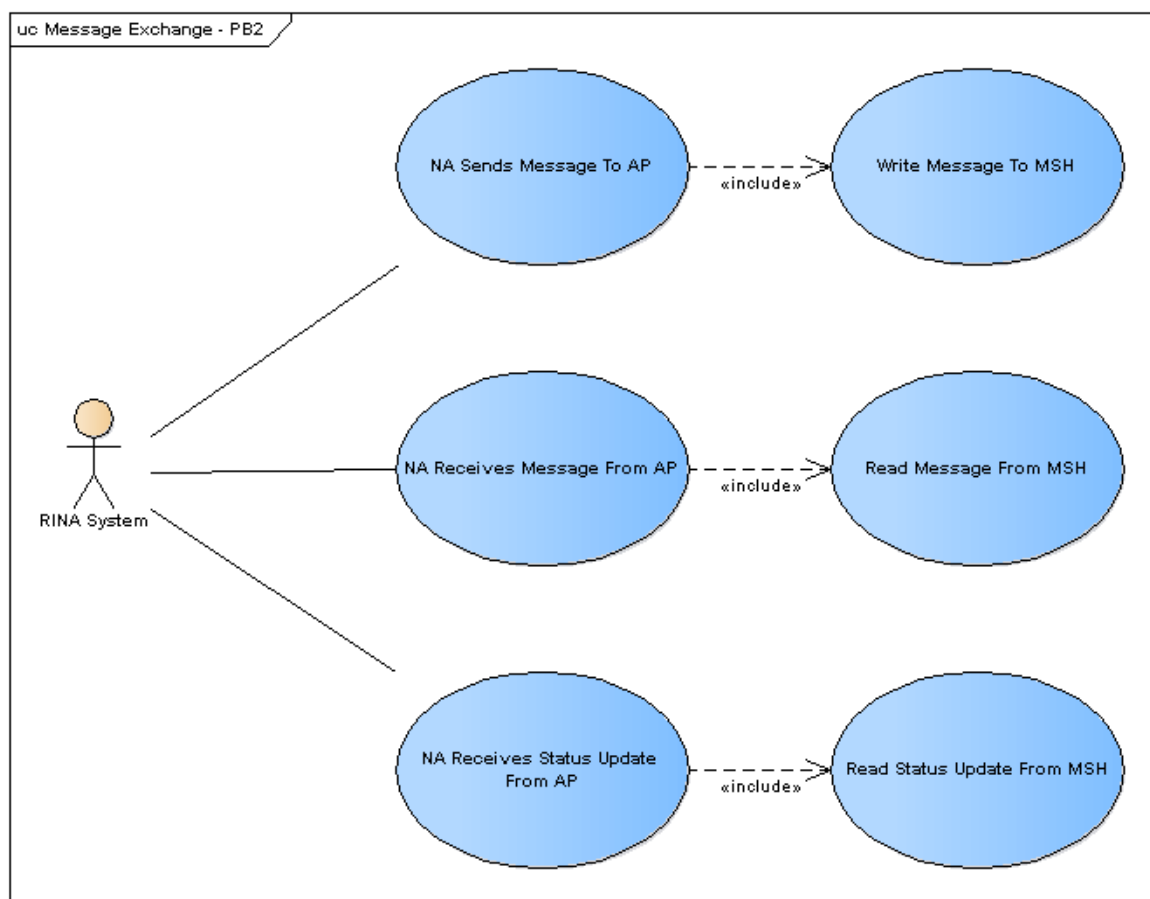


originating National Application, through the Access Points and to the counterparty national application.

The message exchange function provides a way for the users to send and receive business messages transparently over the EESSI system using the ebMS3/AS4 profile.

Please note that in these use cases, the term user refers to a system user which can be RINA, NA or NG.

The following figure shows the use case diagram for this function:



### 3.2.1 NA Sends Message to AP

#### 3.2.1.1 Description

The user must be able to send business messages to the other parties involved in the process relying on a separate component that writes the actual message to the Message Service Handler. The included use case can also be used independent for sending messages.

He is required to build the business message according to the data model provided having all the needed information including the business header, the SED and the attachments if any.

The business messaging interface should however do all the validations needed to be sure that the message is correctly constructed and can be sent. The following checks should be performed before sending the message:

- Check whether the business message is empty;
- Check that the business message contains a valid business header;
- Check that the business message contains a valid SED.

After the business message has been validated, the business message is signed. The technical solution used for business message signing is the XAdES-BES (Basic Electronic Signature) - enveloped format where the signature will be made exclusively for the

business document/ SED. Then the business message is translated into the MSH specific domain and before being put into his queue the message is scanned for malware.

If no malware detected the message is picked up by the MSH that is going to send it using the ebMS standard protocol, more specifically the AS4 profile.

This use case also deals with all the exceptions that may occur during the above steps; so if the validation, translation, transformation or the actual sending fails the user is notified with a message status containing an error description to understand why the operation failed. The retry mechanism is implemented by the MSH.

### 3.2.1.2 Support API

The provided methods of the support API, are the following:

- `APClient` (§ 5.1.1 *APClient*)
- `Write` (§ 7.1.1 *Write*)
- `MmdBuilder` (§ 7.3.1 *MmdBuilder*)

## 3.2.2 NA Receives Message from AP

### 3.2.2.1 Description

The user must be able to receive business messages from the other parties involved in the process relying on a separate component that reads the actual message from the Message Service Handler. The included use case can also be used independent from the main receiving message use case.

The user registers for receiving notifications that can be business messages or status updates.

In case of receiving business messages the MSH receives the standard ebMS/AS4 technical message, translates it into the MSH specific domain and puts it into the input queue for delivering.

The business messaging service will read the message from the input (file) queue and register it to the Dequeue Registry (APClient DB). If it is not already registered, it will pick it up and before translating it to a business message, it will optionally<sup>3</sup> scan the message payload for antimalware.

Following the translation from technical message to business message the business messaging service is also performing the validation of the business message signature.

**Note:** the business signature validation at NA is optional and it will be implemented post first production ready release

Any error in MSH will be translated into an error given to the counter party according to the ebMS AS4 protocol. The duplicate messages are going to be eliminated by the MSH.

---

<sup>3</sup> the antimalware scanning at NA on receive message is an optional antimalware use-case

Any error in the business messaging layer is going to be notified in the receiving user registered listener.

Otherwise the business message is handed over to the user's application that must inform that it was successful or failed in handling the business message.

### 3.2.2.2 Support API

The provided methods of the support API, are the following:

- `APListener` (§ 5.1.3 *APListener*)
- `Read` (§ 7.1.2 *Read*)

## 3.2.3 NA Receives Status Update from AP

### 3.2.3.1 Description

The user must be able to receive status updates about the messages sent to the other parties involved in the process relying on a separate component that reads the actual message status update from the Message Service Handler. The included use case can also be used independent from the main receiving status message update use case.

He registers for receiving notifications that can be business messages or states updates.

In case of receiving status updates the MSH receives the standard ebMS/AS4 signal message, and then it transforms it to a status update in the MSH specific domain.

Any error in MSH will be translated into an error given to the counter party according to the ebMS AS4 protocol. The duplicate status updates are going to be eliminated by the MSH.

Any error in the business messaging layer is going to be notified in the receiving user registered listener.

Otherwise the status update is handed over to the user's application that must inform that it was successful or failed in handling the status update.

### 3.2.3.2 Support API

The provided methods of the support API, are the following:

- `APListener` (§ 5.1.3 *APListener*)
- `Read` (§ 7.1.2 *Read*)

## 3.3 Repositories synchronisation

The Central Service Node (CSN) is the EESSI system that ensures the consistency of information between the different sub-systems of the EESSI ecosystem. This information is stored at the CSN level in two repositories:

- Institution Repository (hereunder IR) – contains all the information related to the institutions exchanging messages within EESSI. The data related to access points is also stored in this repository;

- Common Data Model (hereunder CDM) Repository – contains business use cases metadata, SED metadata and all the versioned information needed by the components of EESSI.

Both the Institution Repository and CDM Repository are automatically replicated at each Access Point. The replication of the information from the Access Point towards National Applications is also possible in an automated or on request demand.

The Institution Repository and the CDM Repository are synchronized from CSN, through the Access Points, towards the National Applications. The synchronisation process can be executed in two manners: automatically and manually. The automatic synchronisation is executed by a scheduler and the manual one by an administrator. In both situations, the Synchronisation Management API is called to trigger the synchronisation itself.

For the purpose of this document the synchronisation of the IR and CDM will be considered.

### ACTORS & ROLES:

An actor is anyone or anything that exchanges data with the technical system. An actor can be a user, external hardware or another system.

The main roles in the repositories synchronisation are:

- **IR Data Source** - The IR Data Source (AP) is the actor that is the owner of the Institution Repository data that needs to be synchronized;
- **IR Data Destination** - The IR Data Destination (NA) is an actor that needs to synchronize with the Institution Repository data from the IR Data Source;
- **CDM Data Source** - The CDM Data Source (AP) is the actor that is the owner of the Common Data Model repository resources that need to be synchronized;
- **CDM Data Destination** - The CDM Data Destination is an actor that needs to synchronize with the Common Data Model repository resources from the CDM Data Source.

### 3.3.1 Receive IR synchronisation data

#### 3.3.1.1 Description

This use-case describes the synchronisation of the overall data regarding the Institution Repository. It covers and describes the following scenario:

- AP sends an IR synchronisation (all data related to Institution Repository) to RINA or National Gateway.

The IR Data Destination (NA) receives the IR Sync Message (SYN001) including the latest IR snapshot version.

#### POST CONDITION:

Following this it updates its IR Data Destination database with the data extracted from SYN001.

#### 3.3.1.2 Support API

The provided methods of the support API, are the following:

- `APListener` (§ 5.1.3 *APListener*)
- `Read` (§ 7.1.2 *Read*)

### 3.3.2 Send IR request for synchronisation

#### 3.3.2.1 Description

This use-case describes the synchronisation of the overall data regarding the Institution Repository. It covers and describes the following scenario:

- National Application requests a synchronisation of the Institution Repository Data from their Access Point.

In this Use Case the trigger come from the IR Data Destination (NA), which is the actor that requests to be synchronized.

The IR Data Destination (NA) requests the IR data by either:

- A manual trigger initiated by an administrator;
- A scheduled trigger (timer based).

The IR Data Destination (NA) creates the Request for IR Sync Message (SYN002) indicating the current snapshot version of the IR data they have and the IR Data Source sends SYN002 to the IR Data Source (AP).

#### POST CONDITION:

Following that the IR Data Destination receives the IR Sync message (SYN001) and updates its IR Data Destination database with the data extracted from SYN001.

#### 3.3.2.2 Support API

The provided methods of the support API, are the following:

- `APClient` (§ 5.1.1 *APClient*)
- `Write` (§ 7.1.1 *Write*)
- `MmdBuilder` (§ 7.3.1 *MmdBuilder*)

### 3.3.3 Receive CDM synchronisation status request

#### 3.3.3.1 Description

This Use Case describes the synchronisation of the overall data regarding the EESSI Common Data Model. It covers and describes the following scenario:

- ✓ AP sends a synchronisation of CDM data to a National Application.

The CDM Data Source (AP) has new/updated CDM resources that needs to be distributed to the CDM Data Destination (NA).

The CDM Data Destination (NA) receives from the CDM Data Source (AP) a CDM Sync Status Request message (SYN004). The SYN004 indicates the updated resource groups available and requesting to the CDM Data Destination to inform him of the resource groups he needs and date and time he last synchronised with the resources of those/that resource group(s).

#### POST CONDITIONS:

The CDM Data Destination (NA) creates a CDM Sync Status message (SYN005) specifying the CDM resource group(s) it needs and the date and time he last synchronised with the resources of those/thats resource group(s) and send this message to the CDM Data Source (AP).

Following the above steps the CDM Data Destination (NA) receives a CDM Sync SED message (SYN003) which indicate whether a delta exists and if yes contains also attached the resources files that are needed for a Resource Type of the Resource Group. These steps are repeated for each Resource Type of the Resource Group required by the CDM data destination (NA).

After receiving the SYN003, the CDM Data Destination (NA) updates its local repository with the new resources and persists/updates the date/time of the successful Resource Group synchronisation.

#### 3.3.3.2 Support API

The provided methods of the support API, are the following:

- [APListener](#) (§ 5.1.3 [APListener](#))
- [Read](#) (§ 7.1.2 [Read](#))

### 3.3.4 Send CDM status / request for synchronisation

#### 3.3.4.1 Description

This Use Case describes the synchronisation of the overall data regarding the EESSI Common Data Model. It covers and describes the following scenario:

- NA requests a synchronisation of CDM data from its AP.

The CDM Data Destination (NA) is an actor that needs to synchronize with the resources from the CDM Data Source (AP) and who is responsible for triggering this Use Case. In this Use Case the Trigger may come from an administrator in the CDM data destination (Human Triggered) or a system scheduler (System Triggered).

The CDM Data Destination (NA) creates the CDM Sync Status (SYN005) specifying the CDM resource groups it needs and the date and time it last synchronised with the resources of those resource groups and send it to the CDM Data Source (AP).

#### POST CONDITIONS:

Following the above use-case, the CDM Data Source (AP) creates the CDM Sync SED message (SYN003) indicating whether a delta exists and attaches to the SED the resources files that are needed for a Resource Type of the Resource Group and send this message to the CDM Data Destination (NA).

These steps are repeated for each Resource Type of the Resource Group required by the CDM data destination (NA).

After receiving the SYN003, the CDM Data Destination (NA) updates their local repository with the new resources and persists/updates the date/time of the successful Resource Group synchronisation.

### 3.3.4.2 Support API

The provided methods of the support API, are the following:

- `APClient` (§ 5.1.1 *APClient*)
- `Write` (§ 7.1.1 *Write*)
- `MmdBuilder` (§ 7.3.1 *MmdBuilder*)

### 3.3.5 Receive CDM Synchronisation data

#### 3.3.5.1 Description

This Use Case describes the synchronisation of the overall data regarding the EESSI Common Data Model. It covers and describes the following scenario:

- NA receive the CDM data from its AP.

The CDM Data Destination (NA) receives a CDM Sync SED message (SYN003) which indicate whether a delta exists and if yes this contains also attached the resources files that are needed for a Resource Type of the Resource Group. These steps are repeated for each Resource Type of the Resource Group required by the CDM data destination (NA).

#### PRE CONDITIONS:

Before the above described scenario, the CDM Data Destination (NA) have to create and send to CDM Data Source (AP) the CDM Sync Status (SYN005) specifying the CDM resource groups it needs and the date and time it last synchronised with the resources of those resource groups. This can be done as part of the above two CDM related scenarios introduced in the previous sections (i.e. *3.3.3 Receive CDM synchronisation status request* and *0 The provided methods of the support API*, are the following:

- `APListener` (§ 5.1.3 *APListener*)
- `Read` (§ 7.1.2 *Read*)

Send CDM status / request for synchronisation)

#### POST CONDITIONS:

After receiving the SYN003, the CDM Data Destination (NA) updates their local repository with the new resources and persists/updates the date/time of the successful Resource Group synchronisation.

#### 3.3.5.2 Support API

The provided methods of the support API, are the following:

- `APListener` (§ 5.1.3 *APListener*)
- `Read` (§ 7.1.2 *Read*)

## 4 Behaviour

### 4.1 Sequence diagrams

The sequence diagrams show:

- the interaction between the business messaging components when sending a business message, receiving a message or receiving a status update.
- the interaction between the technical synchronisation messaging components when receiving IR or CDM synchronisation data or sending IR or CDM request for synchronisation

All the business messages related diagrams have the following internal components:

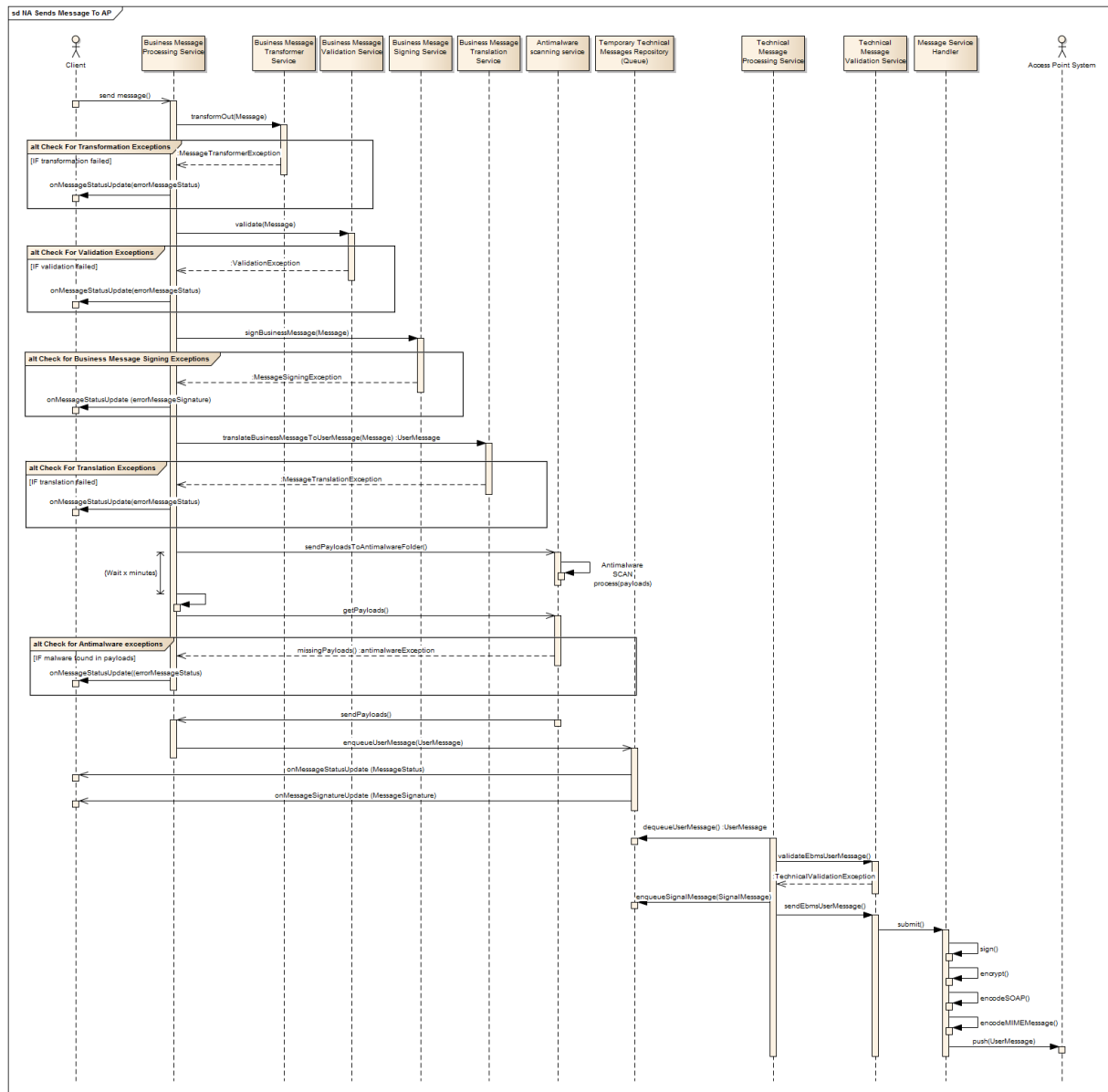
- Business Message Processing Service – this component implements the business messaging interface and it does this by orchestrating all the other business message services;
- Business Message Transformer Service – this component does some fine tuning by transforming the message (for example it may standardize the values that are being sent like the IDs in a GUID format) that is going out or is coming in. The 'in' transformation is the opposite of the 'out' transformation. More transformers can be chained, the output for one transformer is the input for the next transformer and therefore the order they are chained matters; the order of 'in' transformers are the reversed order of 'out' transformers;
- Business Message Validation Service – this component validates the business message structure like the business header, the SED or the attachments. This is where the standard business document is validated against the transaction XSD;
- Business Message Signing Service – this component is responsible for signing business messages using the business certificate of the sending institution;
- Business Message Translation Service – this component makes the conversion between the business domain and the transport domain by translating the business message into MSH specific domain message or the other way around;
- Antimalware Scanning Service – this component allows for integration with Antimalware systems;
- Temporary Technical Message Repository (Queue) – this repository component is used to decouple the business domain from the technical domain:
  - all the exchanged messages that are being sent are queued by the business processing services and de-queued by the technical processing services;
  - all the exchanged messages that are being received are queued by the technical processing services and de-queued by the business processing services;
- Technical Message Processing Service – this component interacts with the queue repository and with the other technical message services;
- Technical Message Validation Service – this component validates the low level ebMS message;

- Message Service Handler – this component is the ebMS/AS4 implementation that provides the low level handling at the ebMS/AS4 to exchange ebMS User and Signal messages with the AP. The main functionalities are:
  - signing;
  - low level error handling and sending receipts;
  - ensure transport level security;
  - conversion to (or decoding from) SOAP message pattern (operation corresponding outgoing/incoming messages);
  - SOAP message structure validation;
  - compression/decompression of the content;
  - technical duplicate detection of incoming messages.

**Note:** *The communication between BMS, TMS and AP is loose coupled. Each subsystem is handling the exceptions and communication between them should be according to the protocol. Error recovery is not in the scope of the document.*

## 4.1.1 NA Sends Message to AP

### 4.1.1.1 Diagram



### 4.1.1.2 Description

This sequence diagram shows the interaction between the business messaging components when the national application sends a business message to the access point.

The process of sending a business message is asynchronous from the user perspective, meaning that the sendMessage API call it will not block to wait for the reply. Rather the API caller will get notified through a separate API onMessageStatusUpdate call about the success or the failure of the initial sending message.

The business message processing service uses the other business processing services to perform its operations:

- it performs a call to the **Business Message Transformer Service** and if there are any transformers, it applies them in the order they were registered (e.g. transforms all IDs into GUIDs format);

- it performs a call to the **Business Message Validation Service** to validate the business message by checking the business header, the SED and the attachments if any. The following steps are performed inside this service:
  - it validates the message structure, checking for required objects and basic relationship between them (for example the SED and attachment IDs from the business message must match the SBDH provided IDs);
  - it serializes the standard business document by translating the business message to a standard business document data transfer object made of SBDH and the SED payload;
  - SBD XML saved in the previous step is validated against the business transaction XSDs that is based on the SBDH XSD and the SED XSD so that the business semantic is also captured;
- it performs a call to the **Business Message Signing Service** to sign the business message. In the signing process XAdES-BES standard will be used and the signature will be made exclusively for the business document / SED and kept in the SED itself;
- it performs a call to the **Business Message Translation Service** to translate the business message into an ebMS MSH user message; this user message has the input Holodeck format (message metadata data format) and contains references to the SBD serialized during validation and to the attachment saved at this step;
- it performs a call to the **Antimalware Scanning Service** by sending the message to the scanning folder. After this it waits for a predefined amount of time in which the antimalware performs the scan. After the waiting time it performs a call to Antimalware Scanning Service to retrieve the payloads and depending on the scan results the payloads or an antimalware exception will be replied by the Antimalware Scanning Service (an additional scanning option is also available – for more details on either option see Architecture document: "*EESSI – Antimalware Interface*");
- if all the operations above succeed, the ebMS user message is put into the queue repository by using the enqueueUserMessage operation of the Temporary Technical Message Repository;
- if any of the operations above fail, a specific exception (TransformationException, ValidationException, MessageTranslationException) is raised, audited, logged and the caller is informed about the issue by receiving a status update with an appropriate error description through the onMessageStatusUpdate callback;
- All the errors are logged.

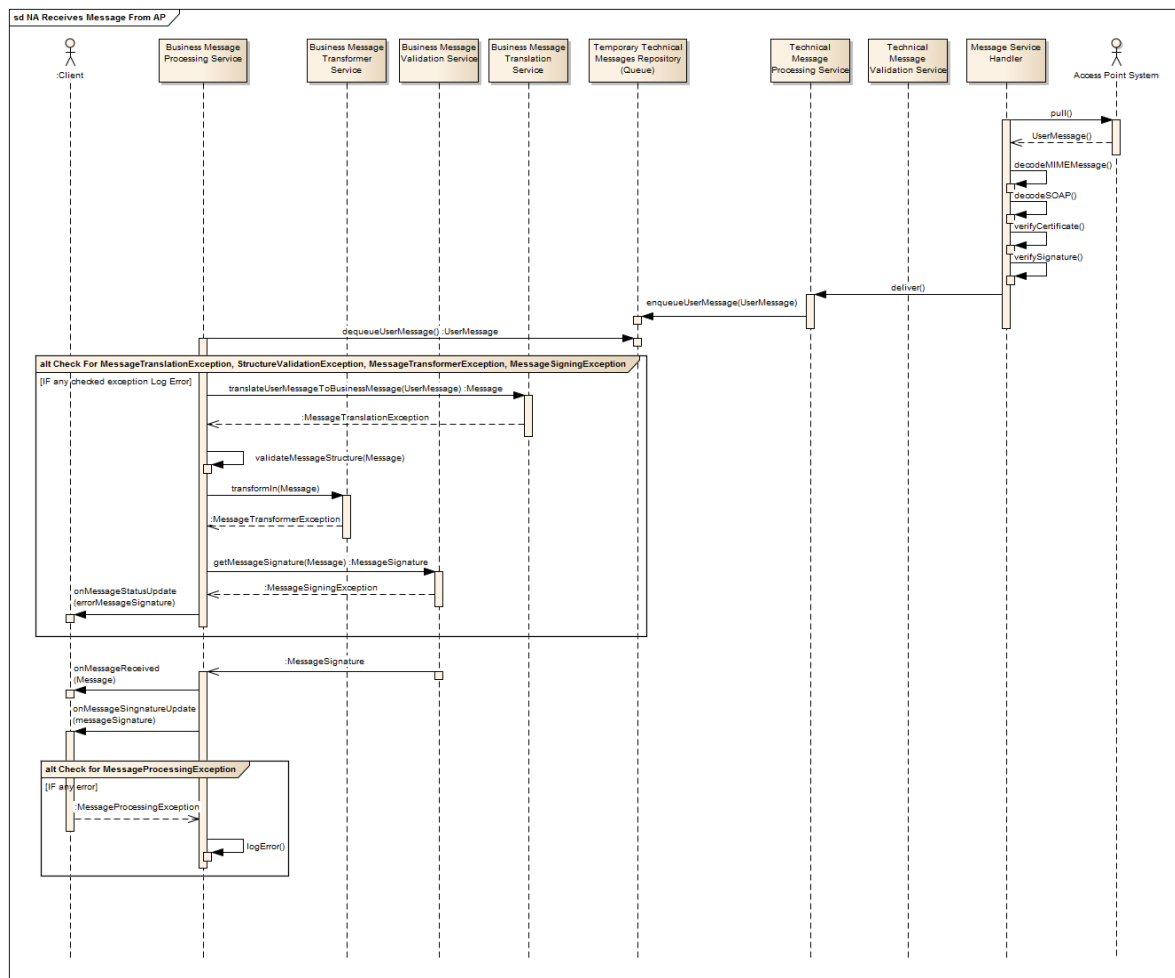
The Technical Message Processing Service takes at some point the ebMS user message from the repository queue and does a technical validation against the Technical Message Validation Service. Should the technical validation fail, an ebMS signal message is put into the queue so that the business message layer can take it and let know the caller about the issue by calling onMessageStatusUpdate.

In the normal successful flow the ebMS message is submitted to the Message Service Handler (MSH) which is the ebMS/AS4 implementation client that is used for low level ebMS/AS4 message exchange.

Once the message is submitted to the Message Service Handler, it is its job to sign the message to ensure security at transport level. The message is then encoded as a SOAP message, compressed and finally is pushed towards the access point.

## 4.1.2 NA Receives Message from AP

### 4.1.2.1 Diagram



**Note 1:** the business signature validation at NA does not appear in the above diagram as it is optional and its implementation depends on the scope of future releases.

**Note 2:** the antimalware scanning at NA on receive message does not appear in the above diagram as is an optional antimalware use-case

**Note 3:** the Dequeue registry (ApClient DB) at NA on receive message does not appear in the above diagram

### 4.1.2.2 Description

This sequence view shows the interaction between the business messaging components when the national application receives a business message from the access point.

The process of receiving a business message starts at the Message Service Handler by pulling messages from the channel. There are two types of ebMS messages that are pulled: User message and Signal message. In this scenario a User message is pulled.

The Message Service Handler validates the ebMS envelope structure, unzips and decodes the ebMS SOAP message and verifies the certificate and the signature. Should this check fail, the Message Service Handler sends an error receipt to the counter party; otherwise the ebMS user message is delivered to the Technical Message Processing Service. The

delivery to the Technical Message Processing Service also depends on whether this message is considered a duplicate (based on the technical metadata content of the ebMS envelope). If so, the message will not be delivered. In any case, a signal message will be sent back to the message sender.

The Technical Message Processing Service will register the incoming message in the Dequeue Registry. The message will be inserted in "DELIVERING" status and thus possible attempts for re-insertion will be rejected, taking also time conditions under consideration. If delivery is successful the status in the Dequeue Registry will become "DELIVERED". More details about the Dequeue Registry (referred also ApClient DB Solution) are described in 9.1

Then, the Technical Message Processing Service performs a call to the Technical Message Validation Service to validate the ebMS user message. If it fails an ebMS signal message is put into the queue, otherwise the user message received is put in the queue. Please note this is under discussion and currently there is no validation and the message is always delivered to the queue.

Validation errors and AP internal errors are propagated back to the sender. If the error is sent by the MSH of the recipient (e.g. improper signature or wrong certificate) then it is also propagated back to the AP and consequently back to the sender.

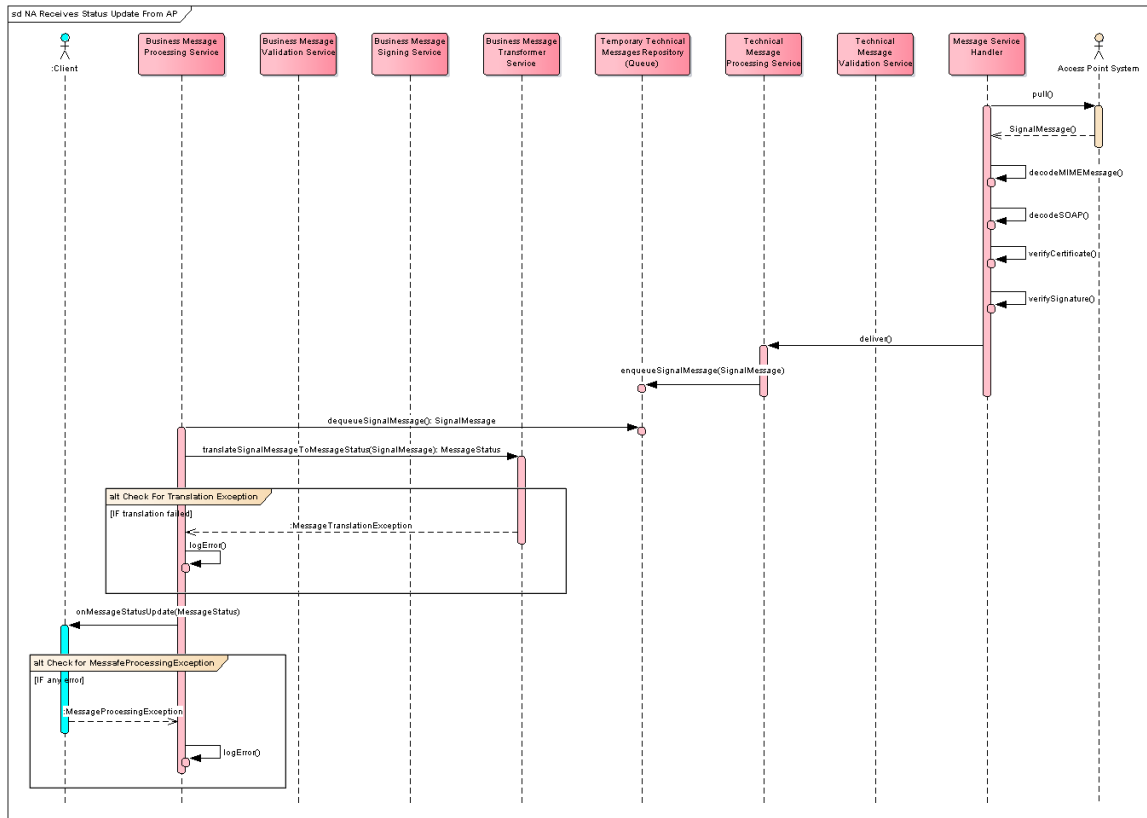
In the case the result could not be provided by AP to holodeck due to connection failure, message could be pulled by holodeck later but not later than configured expiration interval. After expiration, this message is terminated by AP.

At some point the Business Message Processing Service will take the ebMS user message from the queue and then use the other business processing services to perform its operations:

- it performs a call to the Business Message Translation Service to translate the ebMS MSH domain specific user message into a business message; The ebMS message needs to be translated to a Business Message in order to verify the business signatures;
- it validates the message structure, checking for example for nil required objects;
- it performs a call to the Business Message Transformer Service and if there are any transformers it applies them in the reverse order they were registered (e.g. transforms all GUIDs into business IDs);
- it performs an extraction of the SED signature of the message and if all of the operations above succeed, the user is informed about the received Message Signature by the onMessageSignatureUpdate() callback;
- if all the operations above succeed, the user is informed about the received message by the onMessageReceived() callback. Furthermore, the status of the incoming message in the Dequeue Registry is set to "DELIVERED";
- if any of the operations above fail, a specific exception (Message Translation Exception, Structure Validation Exception, Message Transformer Exception) is raised, audited and logged.

### 4.1.3 NA Receives Status Update from AP

#### 4.1.3.1 Diagram



#### 4.1.3.2 Description

This sequence view shows the interaction between the business messaging components when the national application receives a status update from the access point.

The process of receiving a status update business message starts at the Message Service Handler by pulling messages from the channel. There can be two types of ebMS messages that are pulled – user message or signal message. In this scenario a signal message is pulled. The Message Service Handler validates the ebMS envelope structure, unzips and decodes the ebMS SOAP message and verifies the certificate and the signature. Should this check fail, the Message Service Handler will send an error receipt to the counter party; otherwise the ebMS user message is delivered to the Technical Message Processing Service. The delivery to the Technical Message Processing Service also depends on whether this message is considered a duplicate (based on the technical metadata content of the ebMS envelope). If so, the message will not be delivered. In any case, a signal message will be sent back to the message sender.

The Technical Message Processing Service will register the incoming signal message in the Dequeue Registry. The signal message will be inserted in “DELIVERING” status and thus possible attempts for re-insertion will be rejected, taking also time conditions under consideration. If delivery is successful the status in the Dequeue Registry will become “DELIVERED”. More details about the Dequeue Registry (referred also ApClient DB Solution) are described in section 9.1.

The Technical Message Processing Service performs a call to the Technical Message Validation Service to validate the ebMS signal message. If it fails an ebMS signal message is put into the queue.

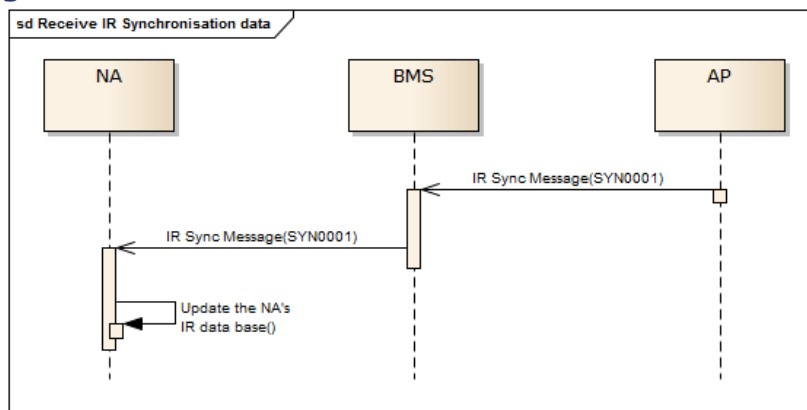
At some point the Business Message Processing Service will take the ebMS signal message from the queue and then use the Business Message Translation Service to translate the ebMS signal message into a message status update.

If this operation succeeds, the user is informed about the status update by the `onMessageStatusUpdate()` callback. Furthermore, the status of the incoming signal message in the Dequeue Registry is set to "DELIVERED".

If any of the operations above fail, a specific exception (Message Translation Exception, Transformation Exception, Message Processing Exception) is raised, audited and logged.

#### 4.1.4 Receive IR Synchronisation data

##### 4.1.4.1 Diagram



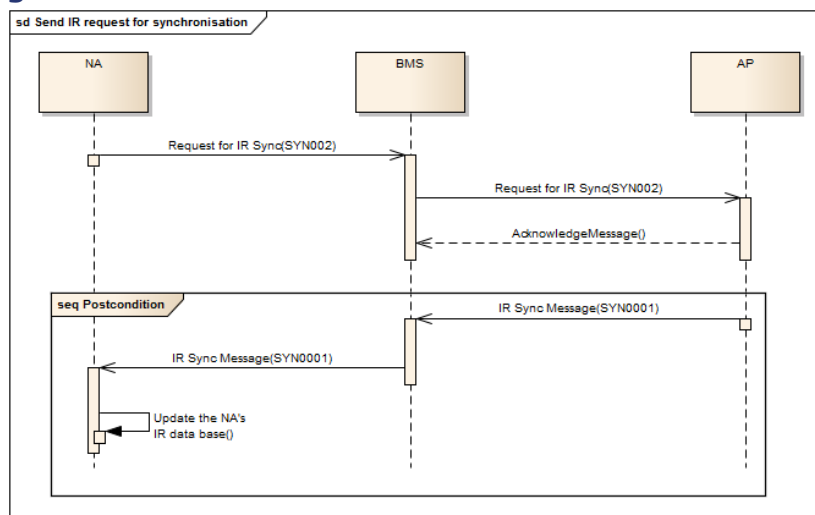
##### 4.1.4.2 Description

The IR Data Destination (NA) receives the IR Sync Message (SYN001) including the latest IR snapshot version.

As the use case post condition, following the receiving the IR Sync Message, the NA updates its IR Data Destination database with the data extracted from SYN001.

#### 4.1.5 Send IR request for synchronisation

##### 4.1.5.1 Diagram



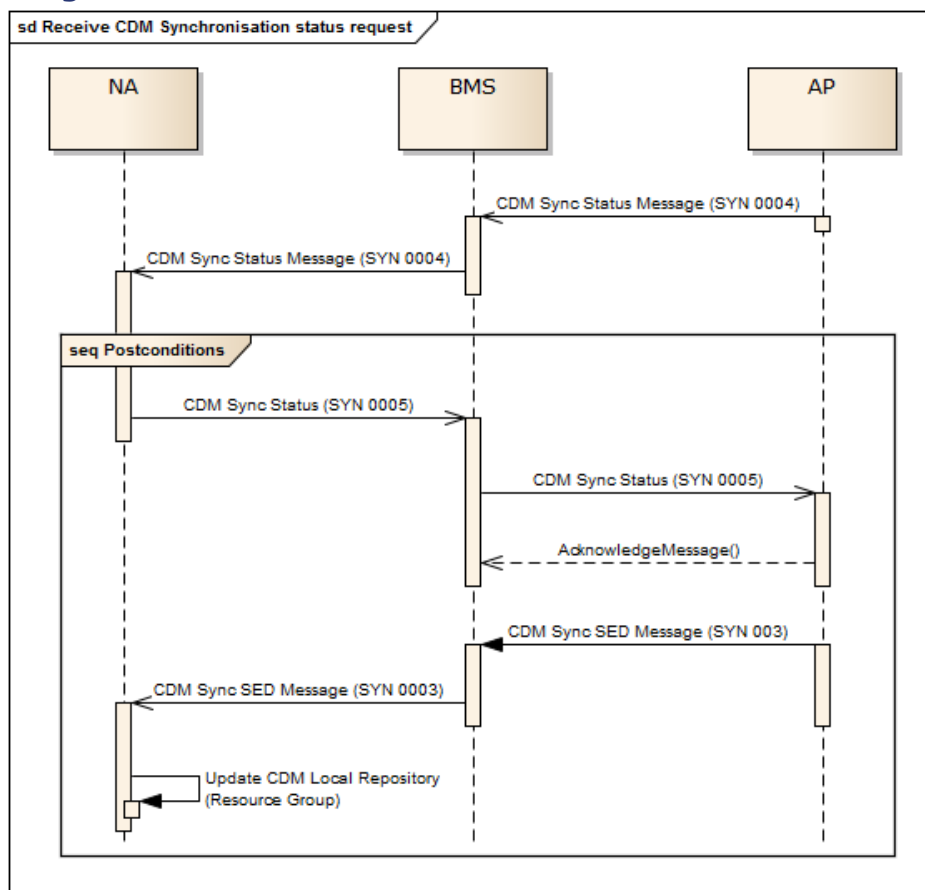
#### 4.1.5.2 Description

The IR Data Destination (NA) creates the Request for IR Sync Message (SYN002) indicating the current snapshot version of the IR data they have and the IR Data Source sends SYN002 to the IR Data Source (AP).

As a post condition of this use case, following the NA sends the SYN002, it receives the IR Sync message (SYN001) and updates its IR database with the data extracted from SYN001.

#### 4.1.6 Receive CDM Synchronisation status request

##### 4.1.6.1 Diagram



##### 4.1.6.2 Description

The CDM Data Destination (NA) receives from the CDM Data Source (AP) a CDM Sync Status Request message (SYN004). The SYN004 indicates the updated resource groups available and requesting to the NA to inform AP of the resource groups he needs and date and time he last synchronised with the resources of those/that resource group(s).

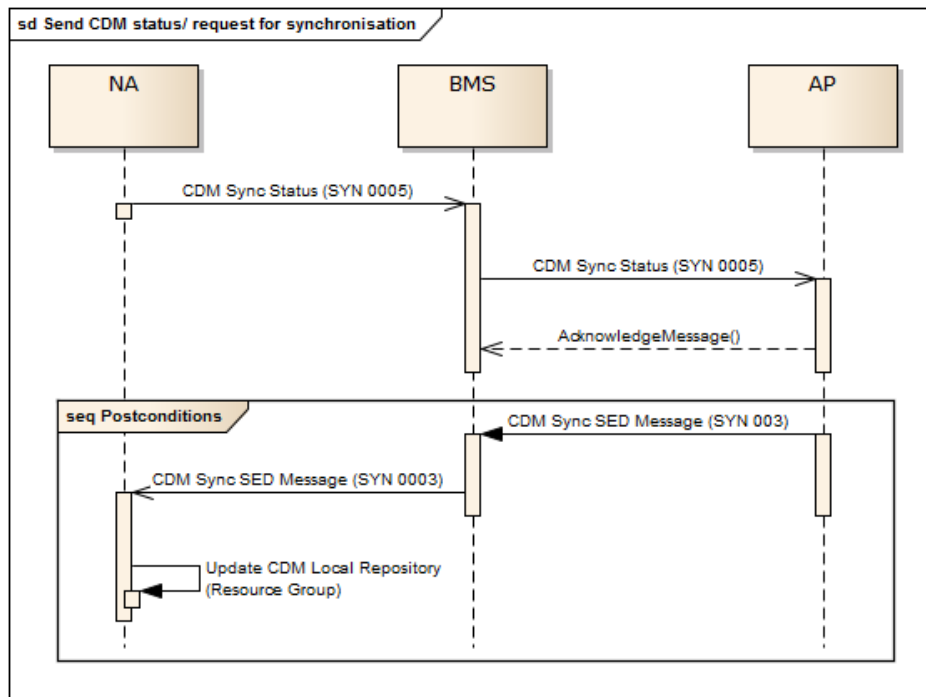
In the above introduced sequence diagram are presented also the post conditions of the current use-case which consists in the following:

- The CDM Data Destination (NA) creates a CDM Sync Status message (SYN005) specifying the CDM resource group(s) it needs and the date and time he last synchronised with the resources of those/that resource group(s) and send this message to the CDM Data Source (AP);

- Following the above the CDM Data Destination (NA) receives a CDM Sync SED message (SYN003) which indicate whether a delta exists and if yes contains also attached the resources files that are needed for a Resource Type of the Resource Group. These steps are repeated for each Resource Type of the Resource Group required by the CDM data destination (NA);
- After receiving the SYN003, the CDM Data Destination (NA) updates its local repository with the new resources and persists/updates the date/time of the successful Resource Group synchronisation.

#### 4.1.7 Send CDM status/ request for synchronisation

##### 4.1.7.1 Diagram



##### 4.1.7.2 Description

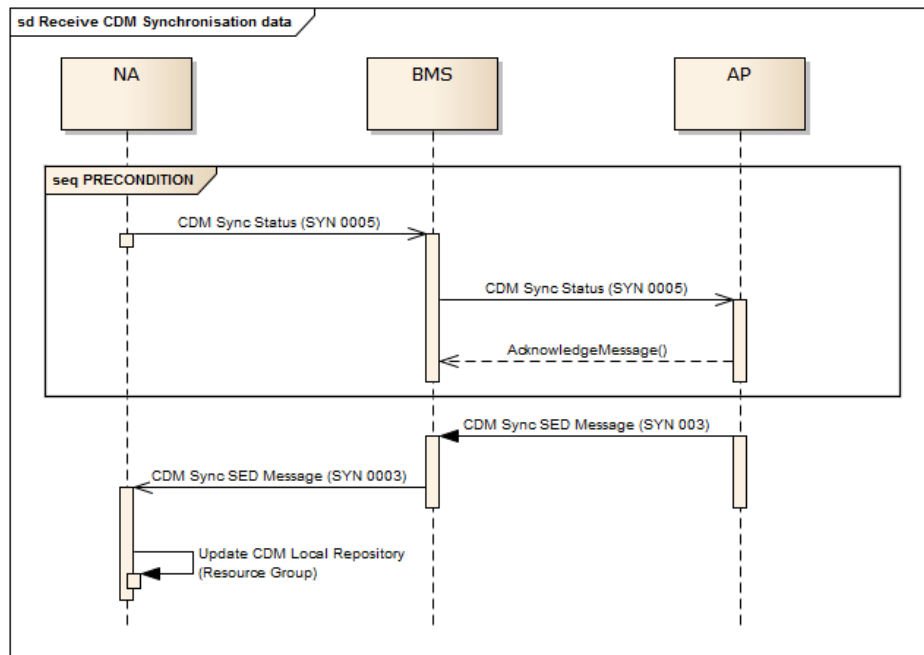
The NA creates the CDM Sync Status (SYN005) specifying the CDM resource groups it needs and the date and time it last synchronised with the resources of those resource groups and send it to the CDM Data Source (AP).

The post conditions of this use-case illustrated in the sequence diagram above consists in the following actions:

- Following the above use-case, the CDM Data Source (AP) creates the CDM Sync SED message (SYN003) indicating whether a delta exists and attaches to the SED the resources files that are needed for a Resource Type of the Resource Group and send this message to the CDM Data Destination (NA);
- These steps are repeated for each Resource Type of the Resource Group required by the CDM data destination (NA);
- After receiving the SYN003, the CDM Data Destination (NA) updates their local repository with the new resources and persists/updates the date/time of the successful Resource Group synchronisation.

## 4.1.8 Receive CDM Synchronisation data

### 4.1.8.1 Diagram



### 4.1.8.2 Description

The NA receives a CDM Sync SED message (SYN003) which indicate whether a delta exists and if yes this contains also attached the resources files that are needed for a Resource Type of the Resource Group. These steps are repeated for each Resource Type of the Resource Group required by the CDM data destination (NA).

This use case will have the following pre and post conditions:

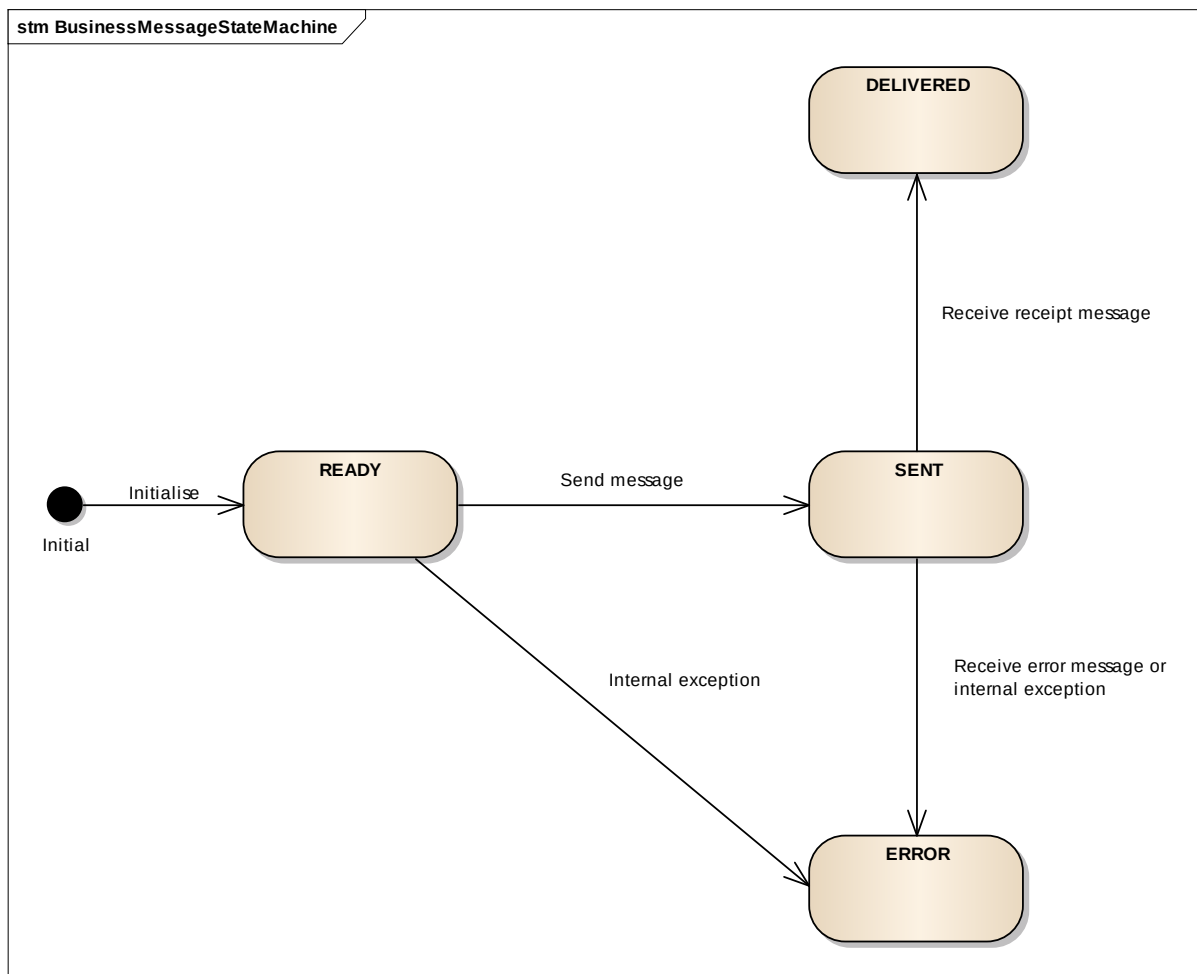
- Precondition:
  - Before the above described scenario, the NA have to create and send to AP the CDM Sync Status (SYN005) specifying the CDM resource groups it needs and the date and time it last synchronised with the resources of those resource groups. This can be done as part of the above two CDM related scenarios introduced in the previous sections;
- Postcondition:
  - After receiving the SYN003, the CDM Data Destination (NA) updates their local repository with the new resources and persists/updates the date/time of the successful Resource Group synchronisation.

## 4.2 State machine diagrams

### 4.2.1 Business message state machine

The business message state machine is used to keep track of the business message status during the sending or receiving of a message

- **READY** - The business message has been initialized and is ready to be handed over to sending MSH
- **SENT** - The business message has been sent to the sending MSH, an update of whether the message has been delivered or had an error will follow
- **ERROR** - The business message could not have been delivered to the counter party, some error occurred during the sending/receiving process or an internal exception occurred
- **DELIVERED** - The business message has been successfully delivered to the counter party



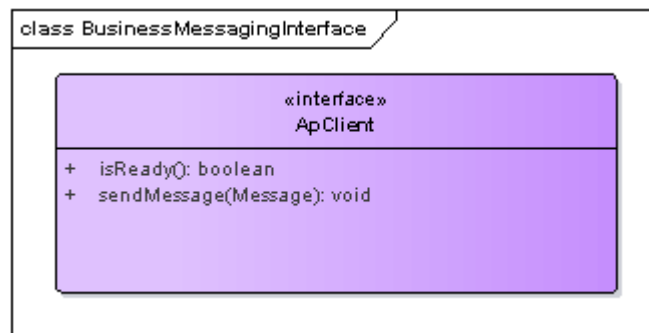
## 5 RINA Business Messaging Interface

### 5.1 Service contract

#### 5.1.1 APClient

Access Point runtime client provides the operations necessary to exchange messages with a single AP. One client is dedicated to support one or many Institutions by using a single channel – Message Partition Channel (MPC). Institution name can be set via ApClientConfiguration.

##### 5.1.1.1 Diagram



##### 5.1.1.2 Description

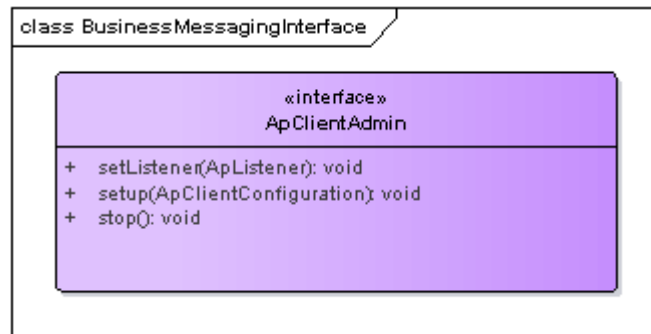
| Name                                                                            | Preconditions                                                              | Input                                                    | Output  | Exceptions                                                                                               |
|---------------------------------------------------------------------------------|----------------------------------------------------------------------------|----------------------------------------------------------|---------|----------------------------------------------------------------------------------------------------------|
| <b>sendMessage</b> -<br>Sends the business message to a configured Access Point | Configured – The client must be configured by calling <code>setup()</code> | <b>message:Message</b> – The business message to be sent | N/A     | <b>MessageProcessingException</b> – Exception that can occur during the processing of a business message |
| <b>isReady</b> –<br>Whether the interface is ready to send business messages    | N/A                                                                        | N/A                                                      | boolean | N/A                                                                                                      |

#### 5.1.2 APClientAdmin

Administration interface that provides capabilities to configure the Message Service Handler: the connection with the Access Point by setting up multiple properties like the access point name, the inbox and outbox URL, the maximum time for retries or the maximum message size. For a complete list for all the properties that can be configured please see the ApClientConfiguration.

Also this interface allows you to register the listener for receiving messages or status updates and to stop the engine.

### 5.1.2.1 Diagram



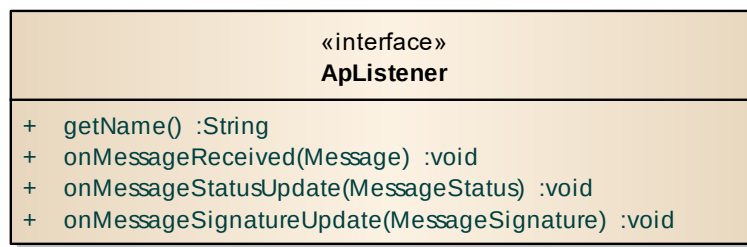
### 5.1.2.2 Description

| Name                                                                                                                                                                                                | Preconditions | Input                                                                                                                                                                        | Output | Exceptions |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|------------|
| <b>setListener</b> - Register a listener in the <code>ApClient</code> runtime in order to receive notifications about the received messages and status updates                                      | N/A           | <b>listener:ApListener</b><br>-<br>The callback to register, the interface the user uses to receive message notifications                                                    | N/A    | N/A        |
| <b>setup</b> - Setup the client configuration, including setting up the MSH with pmodes and pulling configuration. This command starts the file watcher (a thread that looks for incoming messages) | N/A           | <b>configuration:ApClientConfiguration</b> - configuration object, allows to setup properties like access point name, the inbox and outbox URL, the maximum time for retries | N/A    | N/A        |
| <b>stop</b> - Stops <code>ApClient</code> processing functionality                                                                                                                                  | N/A           | N/A                                                                                                                                                                          | N/A    | N/A        |

### 5.1.3 APListener

Listener interface for business application that wants to receive notifications about the received messages and message status updates like delivered or error.

### 5.1.3.1 Diagram

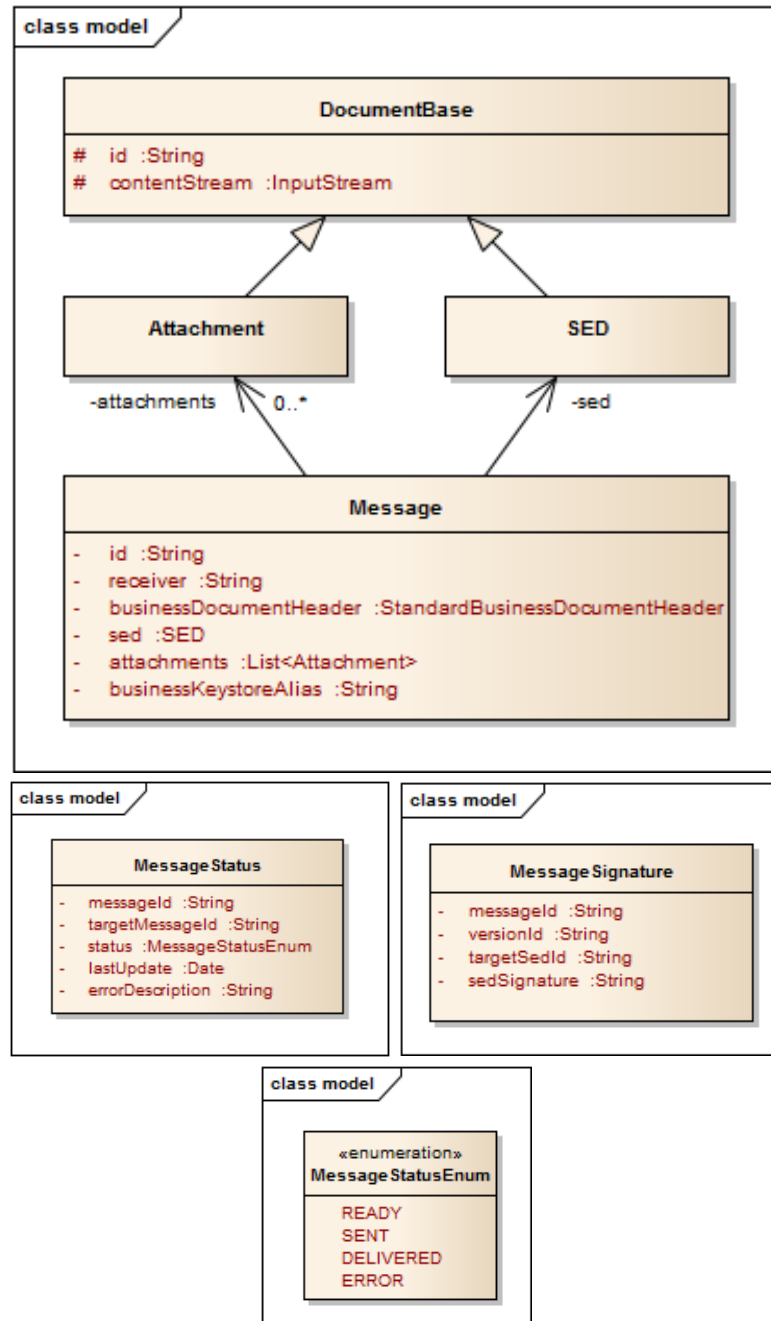


### 5.1.3.2 Description

| Name                                                                                                          | Preconditions                                    | Input                                                                                                            | Output | Exceptions                                                                                                                    |
|---------------------------------------------------------------------------------------------------------------|--------------------------------------------------|------------------------------------------------------------------------------------------------------------------|--------|-------------------------------------------------------------------------------------------------------------------------------|
| <b>getName</b> - The name of the listener                                                                     | N/A                                              | N/A                                                                                                              | String | N/A                                                                                                                           |
| <b>onMessageReceived</b> - Method called when new message is received from the Access Point                   | Configured – the client must be configured first | <b>message:Message</b> – The business message translated from the low level ebMS user message                    | N/A    | <b>MessageProcessingException</b> – The business application can throw this exception on processing the message               |
| <b>onMessageStatusUpdate</b> - Method called when new message status update is received from the Access Point | Configured – the client must be configured first | <b>messageStatus:MessageStatus</b> – The message status that contains information about the state of the message | N/A    | <b>MessageProcessingException</b> – The business application can throw this exception on processing the message status update |
| <b>onMessageSignatureUpdate</b> - Method called when message Signature is created or received                 | N/A                                              | <b>messageSignature:MessageSignature</b> - The signature of the SED of the business message                      | N/A    | <b>MessageSigningException</b> - The business application can throw this exception on message status update                   |

## 5.2 Data contract

### 5.2.1 Message model



#### 5.2.1.1 Attachment

Define attachment that should be sent within BusinessMessage.

Extends DocumentBase.

#### 5.2.1.2 DocumentBase

Base class for Structured Electronic Document or Attachment that is sent within BusinessMessage:

| Attribute     | Description                                                             | Data Type   |
|---------------|-------------------------------------------------------------------------|-------------|
| id            | The attachment or Structured Electronic Document identifier             | String      |
| contentStream | InputStream containing the attachment or Structured Electronic Document | InputStream |

### 5.2.1.3 Message

Business Message represents the Business Document exchanged between Institutions in EESSI network

| Attribute              | Description                                                  | Data Type              |
|------------------------|--------------------------------------------------------------|------------------------|
| id                     | Business Message identifier                                  | String                 |
| receiver               | Where the message is sent to                                 | String                 |
| businessDocumentHeader | Business Document Header that describes the document content | BusinessDocumentHeader |
| sed                    | Structured Electronic Document                               | SED                    |
| attachments            | Attachments list of the Business Message                     | List<Attachment>       |
| businessKeystoreAlias  | The alias/certificate used for business signature            | String                 |

**Note 1:** In case this attribute is equal with "NONE" then this attribute will be ignored as the respective message is a technical message which doesn't need to be signed

**Note 2** If the *businessKeystoreAlias* will not be specified, the default certificate will be used (see 5.2.2.5 *BusinessKeyAliasPasswords*)

### 5.2.1.4 MessageStatus

Status of Business Message identified by targetMessageId

| Attribute       | Description                                              | Data Type         |
|-----------------|----------------------------------------------------------|-------------------|
| messageId       | The id of the acknowledgement or error technical message | String            |
| targetMessageId | the id of the target business message                    | String            |
| status          | The current status of the business message               | MessageStatusEnum |

|                  |                                                    |        |
|------------------|----------------------------------------------------|--------|
| lastUpdate       | the last update time                               | Date   |
| errorDescription | Valid only for status: ERROR.<br>Error description | String |

#### 5.2.1.5 MessageSignature

Class for the signature of Business Message

| Attribute    | Description                                           | Data Type |
|--------------|-------------------------------------------------------|-----------|
| messageId    | The id of the business message                        | String    |
| versionId    | the version Id of the SED                             | String    |
| targetSedId  | the id of the target SED                              | String    |
| sedSignature | The SED's XAdES XML signature of the business message | String    |

#### 5.2.1.6 MessageStatusEnum

All possible states that a business message can take .

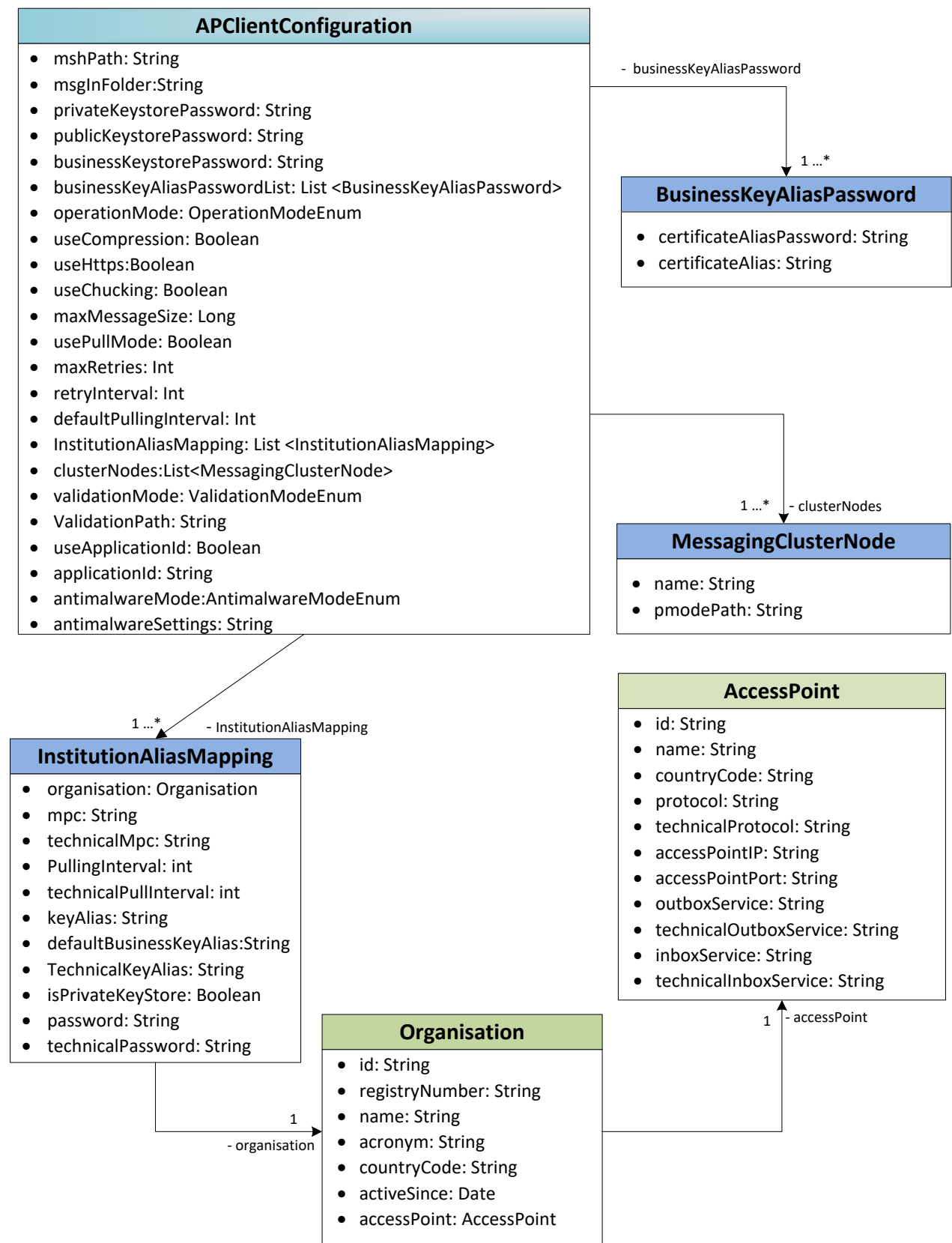
| Attribute | Description                                                            | Data Type         |
|-----------|------------------------------------------------------------------------|-------------------|
| READY     | The message is ready to be read                                        | MessageStatusEnum |
| SENT      | The message has been sent to the MSH                                   | MessageStatusEnum |
| DELIVERED | The message has been delivered to the counter party                    | MessageStatusEnum |
| ERROR     | The message has errors and could not be delivered to the counter party | MessageStatusEnum |

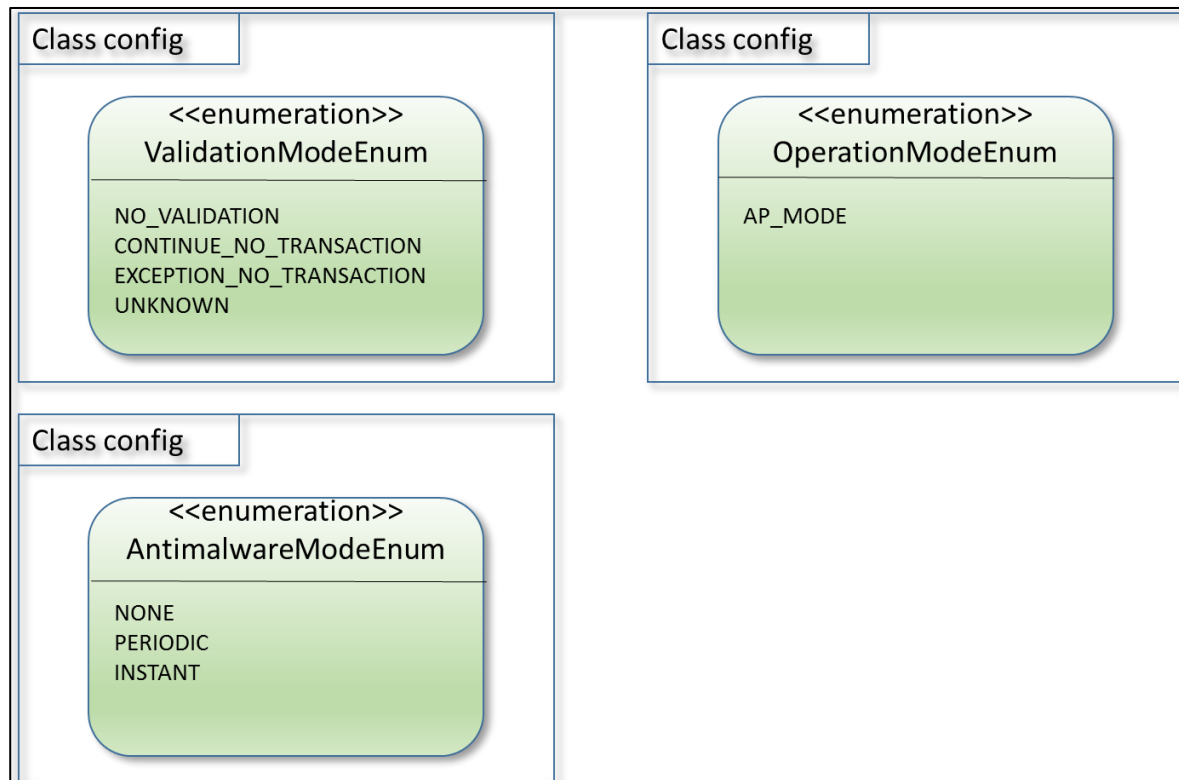
#### 5.2.1.7 SED

Structured Electronic Document is an extension of the DocumentBase used as part of the Business Message

| Attribute     | Description                                                             | Data Type   |
|---------------|-------------------------------------------------------------------------|-------------|
| id            | The attachment or Structured Electronic Document identifier             | String      |
| contentStream | InputStream containing the attachment or Structured Electronic Document | InputStream |

## 5.2.2 Configuration model





### 5.2.2.1 ApClientConfiguration

Configuration class contains the properties to connect to Access Point.

| Attribute                     | Description                                                                                                                                                                                                                                                  | Data Type                      |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------|
| mshPath                       | The root directory path under which the MSH (Messaging Service Handler) file queues are                                                                                                                                                                      | String                         |
| msgInFolder                   | This is ignored in the code. For supporting multitenancy, a specific incoming message folder has been defined programmatically for each tenant, by convention, according to the tenant id.<br><br>This parameter is <b>OBSOLETE</b> and has been deprecated. | String                         |
| privateKeystorePassword       | The password to access the private keystore.<br>Used for ebMS signature                                                                                                                                                                                      | String                         |
| publicKeystorePassword        | The password to access the public truststore<br>Used for ebMS signature                                                                                                                                                                                      | String                         |
| businessKeystorePassword      | The password used in business message signature<br>Used for business signature (SED)                                                                                                                                                                         | String                         |
| businessKeyAliasPassword List | The mapping between aliases for business signatures (certificates & password)                                                                                                                                                                                | List<BusinessKeyAliasPassword> |

|                          |                                                                                                                                                                                                                                                                                                                                                                                                                            |                            |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|
| operationMode            | Determines that the client is connected to an AP. "AP_MODE" is the ONLY valid value for the parameter operationMode when RINA is used in Production/ Quality and/or testing environment                                                                                                                                                                                                                                    | OperationMode Enum         |
| useCompression           | The compression will be used if it is true<br>Values: True/False<br>Default Value: True                                                                                                                                                                                                                                                                                                                                    | boolean                    |
| useHttps                 | This parameter is <b>OBSOLETE</b> and has been deprecated.                                                                                                                                                                                                                                                                                                                                                                 | boolean                    |
| useChunking              | Required to be set to true in order to allow holodeck to use a 'chunking' algorithm for sending messages larger than 1Gb via http.<br>Values: True/False<br>Default value: True                                                                                                                                                                                                                                            | boolean                    |
| maxMessageSize           | Specifies the maximum size of a message<br>Max Value: 2GBytes                                                                                                                                                                                                                                                                                                                                                              | long                       |
| usePullMode              | Defines the transport Mode<br><br>Values: <b>True</b> <sup>*1</sup> for "Pull", <b>False</b> <sup>*2</sup> for "Push"<br><br>The terminology "Push" and "Pull" is referred to the way the messages are retrieved, <u>at the receiving side</u> , by the MSH (Holodeck)<br><br><i>*1PULL: the message is "pulled" by NA (ebMS/AS4 client) from the AP queue</i><br><br><i>*2PUSH: the message is "pushed" from AP to NA</i> | boolean                    |
| maxRetries               | Maximum number of times a message should be resent by Access Point to Holodeck when in Push mode. if no Receipt is received. Setting this value to zero disables the retry functionality.<br>Default value is 0.                                                                                                                                                                                                           | int                        |
| retryInterval            | the internal (in secs) between successive pull attempts from AP to receive messages (for Msg.in)                                                                                                                                                                                                                                                                                                                           | int                        |
| defaultPullingInterval   | Default pulling interval. The time interval between successive pull requests by holodeck to Access Point.                                                                                                                                                                                                                                                                                                                  | int                        |
| institutionAliasMappings | The mapping between aliases of the keys and institutions                                                                                                                                                                                                                                                                                                                                                                   | List<KeystoreAliasMapping> |
| validationMode           | the validation mode that is enforced: NO_VALIDATION (no validation at all), CONTINUE_NO_TRANSACTION (validate and on error continue), EXCEPTION_NO_TRANSACTION (validate and on error generate exception).<br>Enables/Disables validation procedure of transaction SOAP xml against XSD from APClient                                                                                                                      | ValidationMode Enum        |
| validationPath           | the root of the XSD validation path                                                                                                                                                                                                                                                                                                                                                                                        | String                     |

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------|
|                     | The root path of the XSD files used to validate the transaction xml                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                                |
| clusterNodes        | The Nodes of the Cluster in the case of Clustering<br>The name and the path of each cluster                                                                                                                                                                                                                                                                                                                                                                                                                              | List<br><MessagingClusterNode> |
| applicationId       | This is the application ID of the Institution.<br>If useApplicationId = "False" the ApplicationId is not defined<br>Values: True/False (max length: 256 characters.<br>At AP level the length is: 100chars)                                                                                                                                                                                                                                                                                                              | String                         |
| useApplicationId    | Define the usage of "Application ID" parameter value or not. The value of the parameter is used only in the case the flag is "True".<br>Values: True/False<br>Default Value: False                                                                                                                                                                                                                                                                                                                                       | Boolean                        |
| antimalwareMode     | This parameter configures the Antimalware function.<br>The enumeration values are: <ul style="list-style-type: none"> <li>- NONE: no antimalware check is performed,</li> <li>- PERIODIC: antimalware scan is triggered periodically after elapsing set time and</li> <li>- INSTANT: antimalware scan is performed right after message is send/received</li> </ul>                                                                                                                                                       | AntimalwareModeEnum            |
| antimalwareSettings | This parameter contains additional settings for antimalwareMode.<br>If the PERIODIC value has been selected for antimalwareMode, the allowed input represents the time period in secs in which antimalware check will be triggered.<br>If the INSTANT value has been selected for antimalwareMode, antimalwareSettings input contains the command for running the installed antimalware scanner (i.e. for Windows Defender it would be <code>\path to windows defender\ "Windows Defender" \mpcmdrun.exe - scan</code> ) | string                         |

### 5.2.2.2 InstitutionAliasMapping

The mapping between aliases of the keys and institutions

| Attribute    | Description                                                                                                         | Data Type    |
|--------------|---------------------------------------------------------------------------------------------------------------------|--------------|
| organisation | Object for the organisation details                                                                                 | Organisation |
| mpc          | The "business" channel that is dedicated by AP to support the specific institution mpc (message partition channel). | String       |

|                          |                                                                                                                                        |         |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------|---------|
| technicalMpc             | The "technical" channel that is dedicated by AP to support the specific organisation                                                   | String  |
| pullingInterval          | The time interval between successive pull requests by holodeck to Access Point for this specific institution.                          | int     |
| TechnicalPullingInterval | The time interval between successive pull requests by holodeck to Access Point for this specific institution on the technical channel. | int     |
| keyAlias                 | The ebMS alias of key                                                                                                                  | String  |
| DefaultBusinessKeyAlias  | The alias of the corresponding business key with which it is stored inside the business keystore.                                      | String  |
| technicalkeyAlias        | The ebMS alias of key used for the system pmodes                                                                                       | String  |
| isPrivateKeyStore        | True if the alias of key is stored in private <u>keystore</u> . False if the alias of key is stored in public store.                   | Boolean |
| password                 | The password for the private key                                                                                                       | String  |
| technicalPassword        | The technical password for the private key                                                                                             | String  |

### 5.2.2.3 Organisation

Details about an organization.

| Attribute      | Description                                                                                           | Data Type   |
|----------------|-------------------------------------------------------------------------------------------------------|-------------|
| Id             | The id of the organisation. It should be prefixed with the country code and a column (e.g. EL:ELNA01) | String      |
| registryNumber | The registry number of the organization                                                               | String      |
| name           | The name of the organization                                                                          | String      |
| acronym        | The acronym of the organization                                                                       | String      |
| countryCode    | The country code of the country of the organization                                                   | String      |
| activeSince    | The date that the organization is active since                                                        | Date        |
| accessPoint    | The access point that the organization belongs to                                                     | AccessPoint |

#### 5.2.2.4 AccessPoint

Details about an organization.

| Attribute              | Description                                                                | Data Type |
|------------------------|----------------------------------------------------------------------------|-----------|
| id                     | The id of the AccessPoint (AP)                                             | String    |
| name                   | The name of the AccessPoint (AP)                                           | String    |
| countryCode            | The country code of the country that the AccessPoint belongs to            | String    |
| protocol               | The protocol of the AccessPoint, http, https, ...                          | String    |
| technicalProtocol      | The protocol used for the repositories synchronisation                     | String    |
| accessPointIp          | The IP of the AccessPoint                                                  | String    |
| accessPointPort        | The Port of the AccessPoint                                                | String    |
| outboxService          | The outbox service URL of the AccessPoint                                  | String    |
| technicalOutboxService | The technical outbox service URL used for the repositories synchronisation | String    |
| inboxService           | The inbox service url of the AccessPoint                                   | String    |
| technicalInboxService  | The technical inbox service URL used for the repositories synchronisation  | String    |

#### 5.2.2.5 BusinessKeyAliasPasswords

Details about the business messages signatures

| Attribute                | Description                                                       | Data Type |
|--------------------------|-------------------------------------------------------------------|-----------|
| certificateAlias         | The certificate /alias that the SED will be signed                | String    |
| certificateAliasPassword | The password of the certificate/alias that the SED will be signed | String    |

#### 5.2.2.6 MessagingClusterNode

Details about the MessagingClusterNode List

| Attribute | Description                  | Data Type |
|-----------|------------------------------|-----------|
| name      | The name of the cluster Node | String    |
| pmodePath | The path of the cluster Node | String    |

Concerning, a single node installation (no cluster), the name of the Node and the path where the Pmodes are stored in order to be read by MSH (Holodeck) should be put in the "ClusterNode" object.

### 5.2.3 Standard Business Document Header Model

The header this interface uses is the Standard Business Document Header and it is described in separate document, because it is a common component used by other services as well.

This library provides a business model for the SBDH that is encapsulated into the business message.

#### 5.2.3.1 StandardBusinessDocumentHeader

The business model for the SBDH the contains all the metadata information about the business message (please also see the SBDH Implementation Guide for a detailed description of the XSDs pertinent to the SBDH).

| Attribute              | Description                                                                                                                                                                                                                                                                                                                         | Data Type              |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------|
| headerVersion          | The SBDH version                                                                                                                                                                                                                                                                                                                    | String                 |
| sender                 | The sender participant                                                                                                                                                                                                                                                                                                              | PartnerSender          |
| Receivers              | The list of the receiver participants, all receiver participants of the business message must be present in the SBDH.<br><br>When the case type is Multi-lateral the number of Receiving Participants must be one or more.<br><br>When the case type is not Multi-lateral the number of Receiving Participants must be exactly one. | List<PartnerReceiver>  |
| documentIdentification | All the SED information                                                                                                                                                                                                                                                                                                             | DocumentIdentification |
| manifest               | If the SED has one or more attachments, the EESSI SBDH must contain information about each attachment                                                                                                                                                                                                                               | Manifest               |
| businessScope          | This is used to describe business requirements and specific elements of the case and message.                                                                                                                                                                                                                                       | BusinessScope          |

#### 5.2.3.1 PartnerSender

The sender participant.

| Attribute             | Description                                                                                       | Data Type             |
|-----------------------|---------------------------------------------------------------------------------------------------|-----------------------|
| Identifier            | The authority that regulates the identifiers, by default it is 'urn:eu:europa:ec:dgempl:eessi:ir' | PartnerIdentification |
| contactTypeIdentifier | The role of the partner                                                                           | ContactTypeIdentifier |

### 5.2.3.1 PartnerReceiver

The list of the receiver participants, all receiver participants of the business message must be present in the SBDH.

| Attribute             | Description                       | Data Type             |
|-----------------------|-----------------------------------|-----------------------|
| Identifier            | Identification Object for Partner | PartnerIdentification |
| contactTypeIdentifier | The role of the partner           | ContactTypeIdentifier |

### 5.2.3.2 PartnerIdentification

The business model used for the SBDH partner sender or receiver

| Attribute | Description                                                                                                                                          | Data Type |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| value     | The participant must be uniquely identifiable and contains the institution ID that must be taken from the Institution Directory and the country code | String    |
| authority | The authority that regulates the identifiers, by default it is 'urn:eu:europa:ec:dgempl:eessi:ir'                                                    | String    |

### 5.2.3.3 Document Identification

The business model for the SBDH SED

| Attribute   | Description                                                                                                                                  | Data Type |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| typeVersion | The schema version the SED conforms to                                                                                                       | String    |
| type        | The SED/document type (e.g. P2000, S001, etc.) where the case type is Multi-lateral the number of Receiving Participants must be one or more | String    |

|                     |                                                                                                                                                                                                |            |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
|                     | where the case type is not Multi-lateral the number of Receiving Participants must be exactly one                                                                                              |            |
| instanceIdentifier  | This field contains the SED global instance unique identifier within EESSI system according to UUID Version 4 as per IETF RFC4122.                                                             | String     |
| caseAction          | The Sender of a SED can specify the Case Action they are performing when sending the SED, to allow the receiving National Application to fully understand what to do when it receives the SED. | CaseAction |
| creationDateAndTime | This field contains the date and time of the SED creation.<br><br>The date and time encoding format of this field is ISO 8601                                                                  | Date       |

#### 5.2.3.4 CaseActionType

Enumeration of the possible case actions

| Attribute     | Description                                            | Data Type      |
|---------------|--------------------------------------------------------|----------------|
| START         | The case is started                                    | CaseActionType |
| NEW           | The case is already started and a new document is sent | CaseActionType |
| UPDATE        | The case is updated with an updated document           | CaseActionType |
| START_FORWARD | The case is forwarded to another process owner         | CaseActionType |
| NEW_FORWARD   | The case is forwarded to another counter party         | CaseActionType |

#### 5.2.3.5 Manifest

A list of items that are attached to this document

| Attribute     | Description                                                               | Data Type          |
|---------------|---------------------------------------------------------------------------|--------------------|
| numberOfItems | Attachment must be uniquely identifiable                                  | BigInteger         |
| manifestItem  | This block contains information about a specific attachment providing the | List<ManifestItem> |

|  |                                                                                                   |  |
|--|---------------------------------------------------------------------------------------------------|--|
|  | mime type of the item, the URI of the item and the text description and the language of the item. |  |
|--|---------------------------------------------------------------------------------------------------|--|

### 5.2.3.1 ManifestItem

This block contains information about a specific attachment providing the mime type of the item, the URI of the item and the text description and the language of the item.

| Attribute                 | Description                                                                                                                                                                               | Data Type                              |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------|
| contentTypeQualifierCode  | Attachment must be uniquely identifiable                                                                                                                                                  | AllowedAttachmentMimeTypeQualifierCode |
| uniformResourceIdentifier | This field contains the attachment ID                                                                                                                                                     | String                                 |
| description               | This field contains a reference to a section of the SED to which the attachment links, this field must be provided with an XPath expression. (see SBDH Implementation guide)              | String                                 |
| medicalScope              | If the described attachment contains medical data, it should be marked with true and can be opened only if the user has specific medical role. Missing field will be considered as false. | Boolean                                |
| fileName                  | Optional field to describe the original filename attached.                                                                                                                                | String                                 |

### 5.2.3.1 BusinessScope

This is used to describe business requirements and specific elements of the case and message.

| Attribute                 | Description                                                                                                                                                                  | Data Type                              |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------|
| contentTypeQualifierCode  | Attachment must be uniquely identifiable                                                                                                                                     | AllowedAttachmentMimeTypeQualifierCode |
| uniformResourceIdentifier | This field contains the attachment ID                                                                                                                                        | String                                 |
| description               | This field contains a reference to a section of the SED to which the attachment links, this field must be provided with an XPath expression. (see SBDH Implementation guide) | String                                 |

|               |                                                                                                                                                                                           |         |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| medicalScope  | If the described attachment contains medical data, it should be marked with true and can be opened only if the user has specific medical role. Missing field will be considered as false. | Boolean |
| fileName      | Optional filed to describe the original filename attached.                                                                                                                                | String  |
| isMedicalInfo | Whether the SED is considered to hold medical information or not                                                                                                                          | boolean |

### 5.2.3.2 Contact Type Identifier

Enumeration of the possible contact type identifier

| Attribute       | Description                                         | Data Type             |
|-----------------|-----------------------------------------------------|-----------------------|
| CASE_OWNER      | The case owner process                              | ContactTypeIdentifier |
| COUNTERPARTY    | The process counter party                           | ContactTypeIdentifier |
| DATASOURCE      | The data source party used in system messaging      | ContactTypeIdentifier |
| DATADESTINATION | The data destination party used in system messaging | ContactTypeIdentifier |
| INTELLIGENT_RA  | Intelligent routing                                 | ContactTypeIdentifier |

### 5.2.3.3 AllowedAttachmentMimeTypeQualifierCode

Enumeration of the possible Internet media types that can be used for attachments

| Attribute                                          | Description                                                             | Data Type                              |
|----------------------------------------------------|-------------------------------------------------------------------------|----------------------------------------|
| APPLICATION_PDF                                    | application/pdf                                                         | AllowedAttachmentMimeTypeQualifierCode |
| APPLICATION_MSWORD                                 | application/msword                                                      | AllowedAttachmentMimeTypeQualifierCode |
| APPLICATION_VND_MSEXCEL                            | application/vnd.ms-excel                                                | AllowedAttachmentMimeTypeQualifierCode |
| APPLICATION_VND_MSPowerpoint                       | application/vnd.ms-powerpoint                                           | AllowedAttachmentMimeTypeQualifierCode |
| APPLICATION_VND_OPENXMLFORMATS_OFFICEDOCUMENT_WORD | application/vnd.openxmlformats-officedocument.wordprocessingml.document | AllowedAttachmentMimeTypeQualifierCode |

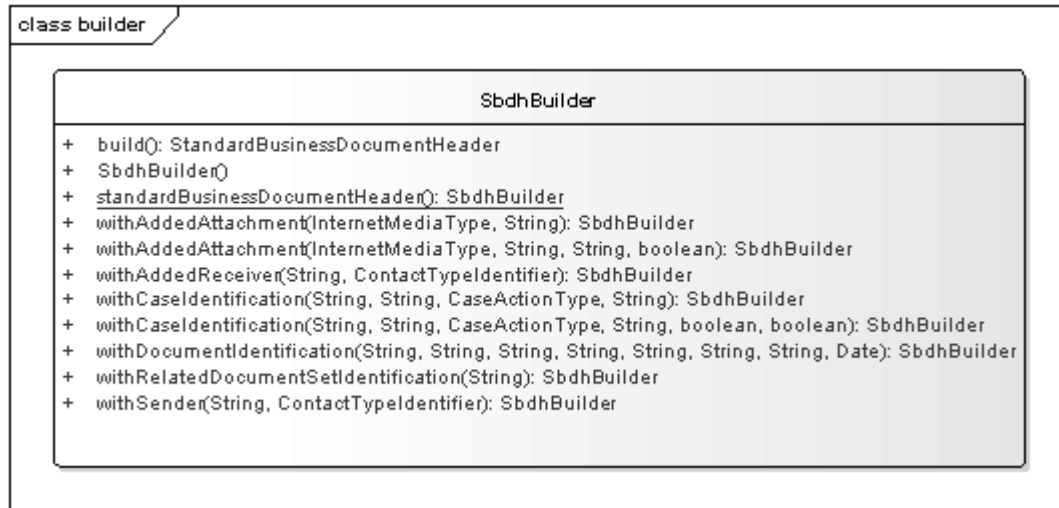
| Attribute                                                                 | Description                                                               | Data Type                              |
|---------------------------------------------------------------------------|---------------------------------------------------------------------------|----------------------------------------|
| PROCESSINGML_DOCUMENT                                                     |                                                                           |                                        |
| APPLICATION_VND_OPENXMLFORMATS_OFFICEDOCUMENT_SPREADSHEETML_SHEET         | application/vnd.openxmlformats-officedocument.spreadsheetml.sheet         | AllowedAttachmentMimeTypeQualifierCode |
| APPLICATION_VND_OPENXMLFORMATS_OFFICEDOCUMENT_PRESENTATIONML_PRESENTATION | application/vnd.openxmlformats-officedocument.presentationml.presentation | AllowedAttachmentMimeTypeQualifierCode |
| APPLICATION_XML                                                           | application/xml                                                           | AllowedAttachmentMimeTypeQualifierCode |
| APPLICATION_ZIP                                                           | application/zip                                                           | AllowedAttachmentMimeTypeQualifierCode |
| APPLICATION_GZIP                                                          | Application/gzip                                                          | AllowedAttachmentMimeTypeQualifierCode |
| IMAGE_JPEG                                                                | image/jpeg                                                                | AllowedAttachmentMimeTypeQualifierCode |
| IMAGE_PNG                                                                 | image/png                                                                 | AllowedAttachmentMimeTypeQualifierCode |
| IMAGE_TIFF                                                                | image/tiff                                                                | AllowedAttachmentMimeTypeQualifierCode |
| TEXT_RTF                                                                  | text/rtf                                                                  | AllowedAttachmentMimeTypeQualifierCode |
| TEXT_XML                                                                  | text/xml                                                                  | AllowedAttachmentMimeTypeQualifierCode |

### 5.3 Builders

The builders in this chapter are useful to easily build the data contract needed by a Java client or a Web Service written in Java.

## 5.3.1 SbdhBuilder

### 5.3.1.1 Diagram



### 5.3.1.2 How to build a StandardBusinessDocumentHeader with SbdhBuilder

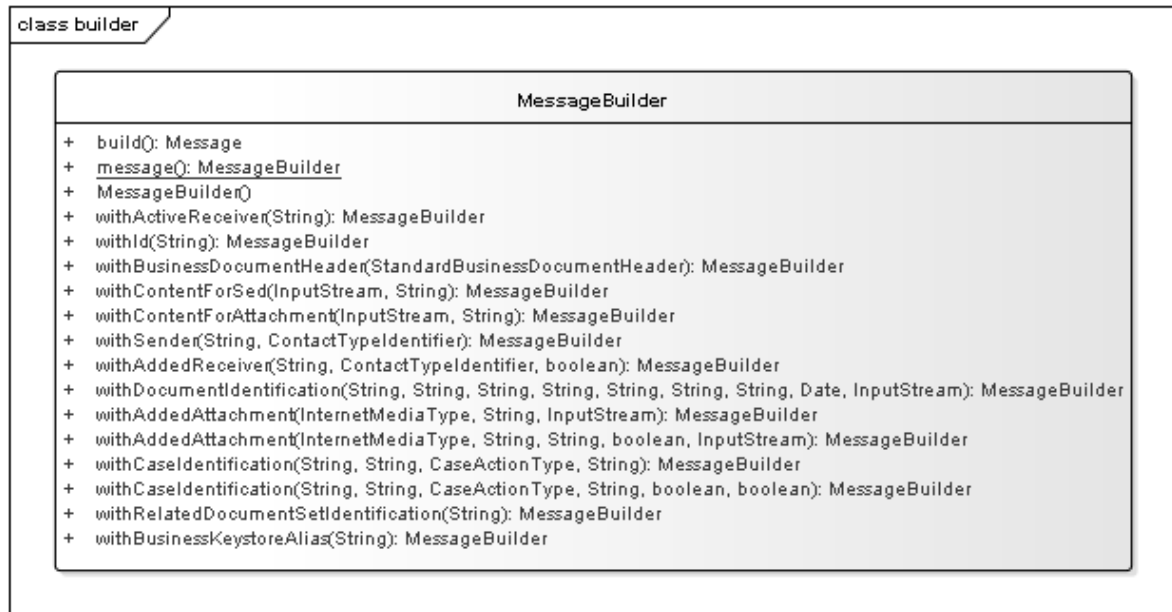
| Name                                                                                                                               | Input                                                                                                                                                                                                                | Output              | Exceptions |
|------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|------------|
| <b>standardBusinessDocumentHeader</b> - Static method for creating a brand new SBDH builder                                        | N/A                                                                                                                                                                                                                  | The new builder     | N/A        |
| <b>withSender</b> - Sending party information                                                                                      | <b>senderIdentifier:String</b> - Sender identifier in the form CountryCodeIso2:institutionId<br><b>senderContactTypeIdentifier: ContactTypeIdentifier</b> - Sender process role (case owner or counter party)        | The current builder | N/A        |
| <b>withAddedReceiver</b> - Receiver party information, if the case is multi-lateral call this method for each receiver participant | <b>receiverIdentifier:String</b> - Receiver identifier in the form CountryCodeIso2:institutionId<br><b>receiverContactTypeIdentifier:ContactTypeIdentifier</b> - receiver process role (case owner or counter party) | The current builder | N/A        |

| Name                                                                                                     | Input                                                                                                                                                                                                                                                                                                                                                                                                                                                    | Output              | Exceptions |
|----------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|------------|
| <b>withDocumentIdentification</b> - SED information                                                      | <b>type:String</b> - SED type, e.g.:P2000, F001, U002<br><b>typeVersion:String</b> - SED schema version, e.g.:3.2<br><b>setIdentifier:String</b> - SED instance identifier<br><b>documentVersion:String</b> - Version of SED instance identifier<br><b>instanceIdentifier:String</b> - SED global instance identifier<br><b>caseActionType:CaseActionType</b> - Enumeration of the possible case actions<br><b>creationDate:Date</b> - SED creation date | The current builder | N/A        |
| <b>withAddedAttachment</b> - Attachment information, call this method for each attachment to the message | <b>mimeType:AllowedAttachmentMimeTypeQualifierCode</b> - Mime type of the attachment<br><b>attachmentIdentifier:String</b> - Unique attachment identifier                                                                                                                                                                                                                                                                                                | The current builder | N/A        |
| <b>withAddedAttachment</b> - Attachment information, call this method for each attachment to the message | <b>mimeType:AllowedAttachmentMimeTypeQualifierCode</b> - Mime type of the attachment<br><b>attachmentIdentifier:String</b> - Unique attachment identifier<br><b>fileName:String</b> - the filename of the attachment as it should be displayed in the UI<br><b>sectionReference:String</b> - The description of the ManifestItem<br><b>isMedicalInfo:Boolean</b> - true if it contains medical data                                                      | The current builder | N/A        |
| <b>withCaseIdentification</b> - Case information                                                         | <b>buc:String</b> - Business use case type, e.g.:P_BUC_01, FB_BUC_01<br><b>version:String</b> - Version of the business use case, e.g.:3.2<br><b>caseIdentifier:String</b> - Case identifier to which the message relates to<br><b>isPersonalProtected:boolean</b> - Whether the case is protected or not, by default this is 'false'                                                                                                                    | The current builder | N/A        |
| <b>withRelatedDocumentSet Identification</b> - Link to a possible related document                       | <b>relatedSetIdentifier:String</b> - Related SED instance identifier                                                                                                                                                                                                                                                                                                                                                                                     | The current builder | N/A        |

| Name                                                                                                                                                                                                                                                                | Input | Output         | Exceptions                          |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|----------------|-------------------------------------|
| <b>build</b> - Constructs the SBDH model<br><b>Preconditions:</b><br>The following with* methods from this table must be called before:<br>withSender<br>withAddedReceiver<br>withDocumentIdentification<br>withCaseIdentification (one of the two forms available) | N/A   | The SBDH model | <b>StructureValidationException</b> |

## 5.3.2 MessageBuilder

### 5.3.2.1 Diagram



### 5.3.2.2 How to build a Message with Message Builder based on an already built SBDH

| Name                                                                                        | Input                                                                                                                                                                                                                     | Output              | Exceptions |
|---------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|------------|
| <b>message</b> - Static method for creating a brand new message builder                     | N/A                                                                                                                                                                                                                       | The new builder     | N/A        |
| <b>withId</b> - Message id                                                                  | <b>id:String</b> – Message id                                                                                                                                                                                             | The current builder | N/A        |
| <b>withActiveReceiver</b> - Selects the active/current receiver from the SBDH receiver list | <b>receiver:String</b> - Receiver identifier in the form<br>CountryCodeIso2:<br>institutionId                                                                                                                             | The current builder | N/A        |
| <b>withBusinessDocumentHeader</b> - SBDH information                                        | <b>businessDocumentHeader:StandardBusinessDocumentHeader</b> - The SBDH model built with the SBDH builder                                                                                                                 | The current builder | N/A        |
| <b>withContentForSed</b> - Adds the actual SED content based on the SED instance identifier | <b>content:InputStream</b> - The InputStream content that needs to be set for the DocumentIdentification from the SBDH<br><b>setIdIdentifier:String</b> - The set identifier for the DocumentIdentification from the SBDH | The current builder | N/A        |

| Name                                                                                                                                                                                                   | Input                                                                                                                                                                                                                                                                       | Output                     | Exceptions                          |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|-------------------------------------|
| <b>withContentForAttachment</b> - Adds the actual attachment content based on the attachment instance identifier                                                                                       | <b>content:InputStream</b> - The InputStream content that needs to be set for the AttachmentIdentification from SBDH that has the attachmentIdentifier<br><b>attachmentIdentifier:String</b> - Identifier for an AttachmentIdentification from the SBDH list of attachments | The current builder        | N/A                                 |
| <b>withBusinessKeystoreAlias</b> - adds the certificate to be used for business message signing                                                                                                        | <b>businessKeystoreAlias:String</b> - identifier for the certificate to be used in the business message signing                                                                                                                                                             | The current builder        | N/A                                 |
| <b>build</b> - Constructs the business message model<br><b>Preconditions:</b> All the with* methods from this table must be called before, except withContentForAttachment if there are no attachments |                                                                                                                                                                                                                                                                             | The business message model | <b>StructureValidationException</b> |

### 5.3.2.3 How to build a Message with Message Builder from scratch

| Name                                                                                                                               | Input                                                                                                                                                                                                                                                                                                                                     | Output              | Exceptions |
|------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|------------|
| <b>message</b> - Static method for creating a brand new message builder                                                            | N/A                                                                                                                                                                                                                                                                                                                                       | The new builder     | N/A        |
| <b>withId</b> - Message id                                                                                                         | <b>id:String</b> - Message id                                                                                                                                                                                                                                                                                                             | The current builder | N/A        |
| <b>withSender</b> - Sending party information                                                                                      | <b>senderIdentifier:String</b> - Sender identifier in the form CountryCodeIso2: institutionId<br><b>senderContactTypeIdentifier:ContactTypeIdentifier</b> - Sender process role (case owner or counter party)                                                                                                                             | The current builder | N/A        |
| <b>withAddedReceiver</b> - Receiver party information, if the case is multi-lateral call this method for each receiver participant | <b>receiverIdentifier:String</b> - Receiver identifier in the form CountryCodeIso2: institutionId<br><b>receiverContactTypeIdentifier:ContactTypeIdentifier</b> - Receiver process role (case owner or counter party)<br><b>isActive:boolean</b> - Whether this newly added receiver is the active/current receiver that gets the message | The current builder | N/A        |

| Name                                                                                                     | Input                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | Output              | Exceptions |
|----------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|------------|
| <b>withDocumentIdentification</b> - SED information                                                      | <b>standard:String</b> - Urn for the SED standard used. This must be 'urn:eu:europa:ec:dgempl:eessi:sed'<br><b>standardVersion:String</b> - Version of the SED standard used. This must be '3.0'<br><b>type:String</b> - SED type, e.g.:P2000, F001, U002<br><b>typeVersion:String</b> - SED schema version, e.g.:3.2<br><b>setIdIdentifier:String</b> - SED instance identifier<br><b>documentVersion:String</b> - Version of SED instance identifier<br><b>instanceIdentifier:String</b> - SED global instance identifier<br><b>creationDate:Date</b> - SED creation date<br><b>contentStream:InputStream</b> - The InputStream content that needs to be set for the SED | The current builder | N/A        |
| <b>withAddedAttachment</b> - Attachment information, call this method for each attachment to the message | <b>mimeType:InternetMediaType</b> - Mime type of the attachment<br><b>attachmentIdentifier:String</b> - Unique attachment identifier<br><b>contentStream:InputStream</b> - The InputStream content that needs to be set for the attachment                                                                                                                                                                                                                                                                                                                                                                                                                                 | The current builder | N/A        |
| <b>withAddedAttachment</b> - Attachment information, call this method for each attachment to the message | <b>mimeType:InternetMediaType</b> - Mime type of the attachment<br><b>attachmentIdentifier:String</b> - Unique attachment identifier<br><b>sectionReference:String</b> - Reference to a section of the SED to which the attachment links in the format of an XPath expression, by default this is 'sed'<br><b>isMedicalInfo:boolean</b> - Whether attachment contains medical information or not, by default this is 'false'<br><b>contentStream:InputStream</b> - The InputStream content that needs to be set for the attachment                                                                                                                                         | The current builder | N/A        |
| <b>withCaseIdentification</b> - Case information                                                         | <b>buc:String</b> - Business use case type, e.g.:P_BUC_01, FB_BUC_01<br><b>version:String</b> - Version of the business use case, e.g:3.2<br><b>action:CaseActionType</b> - Case action, e.g.:START, UPDATE<br><b>caseIdentifier:String</b> - Case identifier to which the message relates to                                                                                                                                                                                                                                                                                                                                                                              | The current builder | N/A        |

| Name                                                                                                                                                                                                                                                                         | Input                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | Output                     | Exceptions                   |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|------------------------------|
| <b>withCaseIdentification</b> - Case information                                                                                                                                                                                                                             | <b>buc:String</b> - Business use case type, e.g.:P_BUC_01, FB_BUC_01<br><b>version:String</b> - Version of the business use case, e.g:3.2<br><b>action: CaseActionType</b> - Case action, e.g.:START, UPDATE<br><b>caseIdentifier:String</b> - Case identifier to which the message relates to<br><b>isPersonalProtected:boolean</b> - Whether the case is protected or not, by default this is 'false'<br><b>isMedicalInfo:Boolean</b> - Whether the case contains medical info, by default this is 'false' | The current builder        | N/A                          |
| <b>withRelatedDocumentSetIdentification</b> - Link to a possible related document                                                                                                                                                                                            | <b>relatedSetIdentifier:String</b> - Related SED instance identifier                                                                                                                                                                                                                                                                                                                                                                                                                                         | The current builder        | N/A                          |
| <b>withBusinessKeystoreAlias</b> - adds the certificate to be used for business message signing                                                                                                                                                                              | businessKeystoreAlias:String - identifier for the certificate to be used in the business message signing                                                                                                                                                                                                                                                                                                                                                                                                     | The current builder        | N/A                          |
| <b>build</b> - Constructs the business message model<br><b>Preconditions:</b> The following with* methods from this table must be called before:<br>withSender<br>withAddedReceiver<br>withDocumentIdentification<br>withCaseIdentification (one of the two forms available) | N/A                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          | The business message model | StructureValidationException |

## 5.4 Fault contract

The only exception that is currently thrown is the `MessageProcessingException`. In case there is a problem during the execution of the callback registered, the user has the option of throwing this exception.

No other exceptions are currently thrown to the user of this interface, however it is worth describing all the internal exceptions that might occur and that are logged. The error description from all the exceptions are filled into the `errorDescription` field from the Message Status that is given as argument to the `onMessageStatusUpdate()` callback.

### 5.4.1 MessageProcessingException

This exception might occur during the processing of a business message, like in the following scenario: the user calls `sendMessage()` on the client interface, there is a problem during the message sending (like for example the client was not properly configured), the library notifies the user about the issue, but the user cannot process the status update so it throws this exception.

### 5.4.2 MessageTransformerException

This exception is thrown during the message transformation in three cases:

- When sending the message
  - Any error during transforming all the IDs (like message id, SED id, attachment id, case id, case set id, related set id) to GUIDs
- When receiving a message
  - Any error during transforming all the GUIDs (like message guid, SED guid, attachment guid, case guid, case set guid, related set guid) to IDs
- When receiving a message status update
  - Any error during transforming all the GUIDs (like message guid, SED guid, attachment guid, case guid, case set guid, related set guid) to IDs

### 5.4.3 MessageTranslationException

This exception might occur during the translation of a business message into an ebMS message or from the ebMS message into the business message, possible causes can be:

- The message cannot be nil
- The SED could not be processed as an input stream
- The SED could not be converted from XML
- The SED could not be converted to XML
- The message could not be persisted into the queue
- The attachment could not be found

### 5.4.4 ValidationException

This is the base validation exception for the more specific `MessageValidationException` and `StructureValidationException`. More specific exception is likely to be implemented later.

It basically indicates that there is a problem during some validation.

### 5.4.5 MessageValidationException

This exception is thrown when validating the business message against the transaction XSDs. The transaction XSDs validates both the SBDH and the SED, so in case there is an issue with the SBDH or the SED this exception will be thrown internally. Possible causes can be:

- The XSD transaction file could not be found;
- There is an exception while parsing the SBD XML;
- There is an exception while parsing the SBD XSD;
- The parser is not configured properly.

### 5.4.6 StructureValidationException

This exception might occur during the structure validation of an object, usually when using a builder.

When using the business message builder, possible causes can be:

- Required objects are missing;
- Input stream for SED is nil;
- Input stream for attachment is nil;

- The SED id does not match the SBDH id;
- The attachment metadata content does not match the SBDH related one;
- The message receiver is not found in the SBDH receiver list;
- Same attachment id is added more than once.

This exception might also occur during the business configuration translation to the Message Service Handler specific configuration (processing modes and pulling configuration), because of the low level configuration builders used.

#### **5.4.7 MessageSigningException**

This exception might occur during the signing of the SED of the business message.

The possible causes can be:

- There is an exception while loading keys from business signature keystore;
- There is an exception while parsing the xml SED to sign;
- There is an exception while signing the xml SED to sign;
- There is an exception while extracting the signature xml value from signed xml SED.

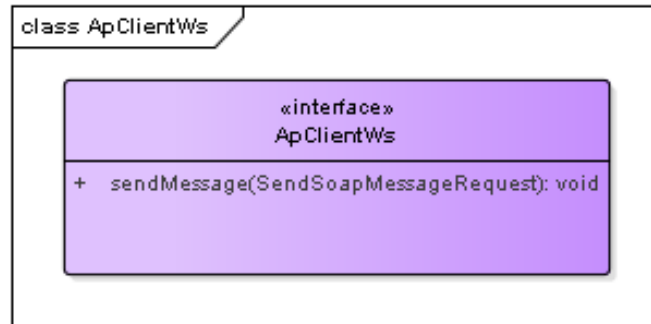
## 6 RINA Business Messaging SOAP Web Service Interface

### 6.1 Service contract

#### 6.1.1 APClient Web Service

APClient Web Service (ApClientWs) is a wrapper of APClient and is published as a SOAP Web Service Interface.

##### 6.1.1.1 Diagram



##### 6.1.1.2 Description

| Name                                                                                                           | Preconditions                                            | Input                                                                                                                   | Output | Exceptions                                                                                               |
|----------------------------------------------------------------------------------------------------------------|----------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------|--------|----------------------------------------------------------------------------------------------------------|
| <b>sendMessage</b> – SOAP Request than contains the SOAP business message to be sent a configured Access Point | Ap Client Web Service should be published and configured | <b>sendSoapMessage Request:</b><br><b>SendSoapMessage Request</b> – The request of the SOAP business message to be sent | N/A    | <b>MessageProcessingException</b> – Exception that can occur during the processing of a business message |

### 6.1.1.1 WSDL (APClient)

The WSDL exposed is the following:

```
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:sch="http://eessi.dgempl.ec.europa.eu/namespaces/soap"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://eessi.dgempl.ec.europa.eu/namespaces/soap"
targetNamespace="http://eessi.dgempl.ec.europa.eu/namespaces/soap">
  <wsdl:types>
    <xs:schema xmlns="http://eessi.dgempl.ec.europa.eu/namespaces/soap"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://eessi.dgempl.ec.europa.eu/namespaces/soap">

      <xs:element name="attachmentIdentification" type="attachmentIdentification" />
      <xs:element name="caseActionType" type="caseActionType" />
      <xs:element name="caseIdentification" type="caseIdentification" />
      <xs:element name="contactTypeIdentifier" type="contactTypeIdentifier" />
      <xs:element name="documentIdentification" type="documentIdentification" />
      <xs:element name="internetMediaType" type="internetMediaType" />
      <xs:element name="partner" type="partner" />
      <xs:element name="sendSoapMessageRequest" type="sendSoapMessageRequest" />
      <xs:element name="soapAttachment" type="soapAttachment" />
      <xs:element name="soapDocumentBase" type="soapDocumentBase" />
      <xs:element name="soapMessage" type="soapMessage" />
      <xs:element name="soapSed" type="soapSed" />
      <xs:element name="standardBusinessDocumentHeader" type="standardBusinessDocumentHeader" />

      <xs:complexType name="sendSoapMessageRequest">
        <xs:sequence>
          <xs:element name="soapMessage" type="soapMessage" />
        </xs:sequence>
      </xs:complexType>

      <xs:complexType name="soapMessage">
        <xs:all>
          <xs:element name="id" type="xs:string" />
          <xs:element minOccurs="0" name="businessKeystoreAlias" type="xs:string" />
          <xs:element name="receiver" type="xs:string" />
          <xs:element name="standardBusinessDocumentHeader" type="standardBusinessDocumentHeader" />
          <xs:element name="soapSed" type="soapSed" />
          <xs:element minOccurs="0" name="soapAttachments">
            <xs:complexType>
              <xs:sequence>
                <xs:element maxOccurs="unbounded" minOccurs="0" name="soapAttachment" type="soapAttachment" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:all>
      </xs:complexType>

      <xs:complexType name="standardBusinessDocumentHeader">
        <xs:all>
          <xs:element minOccurs="0" name="headerVersion" type="xs:string" />
          <xs:element name="sender" type="partner" />
          <xs:element name="receivers">
            <xs:complexType>
              <xs:sequence>
                <xs:element maxOccurs="unbounded" name="partner" type="partner" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element name="documentIdentification" type="documentIdentification" />
          <xs:element minOccurs="0" name="attachments">
            <xs:complexType>
              <xs:sequence>
                <xs:element maxOccurs="unbounded" minOccurs="0" name="attachmentIdentification"
type="attachmentIdentification" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element name="caseIdentification" type="caseIdentification" />
        </xs:all>
      </xs:complexType>

    </xs:schema>
  </wsdl:types>
</wsdl:definitions>
```

```

<xs:complexType name="partner">
  <xs:all>
    <xs:element minOccurs="0" name="authority" type="xs:string" />
    <xs:element name="identifier" type="xs:string" />
    <xs:element name="contactTypeIdentifier" type="contactTypeIdentifier" />
  </xs:all>
</xs:complexType>

<xs:complexType name="documentIdentification">
  <xs:all>
    <xs:element name="type" type="xs:string" />
    <xs:element name="schemaVersion" type="xs:string" />
    <xs:element name="identifier" type="xs:string" />
    <xs:element name="setIdentifier" type="xs:string" />
    <xs:element name="action" type="caseActionType" />
    <xs:element name="version" type="xs:string" />
    <xs:element name="creationDate" type="xs:dateTime" />
    <xs:element minOccurs="0" name="relatedSetIdentifier" type="xs:string" />
  </xs:all>
</xs:complexType>

<xs:complexType name="attachmentIdentification">
  <xs:all>
    <xs:element name="identifier" type="xs:string" />
    <xs:element minOccurs="0" name="fileName" type="xs:string" />
    <xs:element name="internetMediaTypeCode" type="internetMediaType" />
    <xs:element minOccurs="0" name="sectionReference" type="xs:string" />
    <xs:element name="isMedical" type="xs:boolean" />
  </xs:all>
</xs:complexType>

<xs:complexType name="caseIdentification">
  <xs:all>
    <xs:element name="identifier" type="xs:string" />
    <xs:element name="type" type="xs:string" />
    <xs:element name="version" type="xs:string" />
    <xs:element name="isProtectedPerson" type="xs:boolean" />
  </xs:all>
</xs:complexType>

<xs:complexType name="soapSed">
  <xs:complexContent>
    <xs:extension base="soapDocumentBase">
      <xs:sequence />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="soapDocumentBase">
  <xs:all>
    <xs:element name="id" type="xs:string" />
    <xs:element name="contentUrl" type="xs:anyURI" />
  </xs:all>
</xs:complexType>

<xs:complexType name="soapAttachment">
  <xs:complexContent>
    <xs:extension base="soapDocumentBase">
      <xs:sequence />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:simpleType name="contactTypeIdentifier">
  <xs:restriction base="xs:string">
    <xs:enumeration value="DATADESTINATION" />
    <xs:enumeration value="DATASOURCE" />
    <xs:enumeration value="CASE_OWNER" />
    <xs:enumeration value="COUNTERPARTY" />
    <xs:enumeration value="INTELLIGENT_RA" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="internetMediaType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="APPLICATION_PDF" />
  </xs:restriction>
</xs:simpleType>

```

```

        <xs:enumeration value="APPLICATION_MSWORD" />
        <xs:enumeration value="APPLICATION_VND_MS_EXCEL" />
        <xs:enumeration value="APPLICATION_VND_MS_POWERPOINT" />
        <xs:enumeration value="APPLICATION_VND_OPENXMLFORMATS_OFFICEDOCUMENT_WORDPROCESSINGML_DOCUMENT" />
        <xs:enumeration value="APPLICATION_VND_OPENXMLFORMATS_OFFICEDOCUMENT_SPREADSHEETML_SHEET" />
        <xs:enumeration value="APPLICATION_VND_OPENXMLFORMATS_OFFICEDOCUMENT_PRESENTATIONML_PRESENTATION" />
    />

    <xs:enumeration value="APPLICATION_XML" />
    <xs:enumeration value="APPLICATION_ZIP" />
    <xs:enumeration value="APPLICATION_GZIP" />
    <xs:enumeration value="IMAGE_JPEG" />
    <xs:enumeration value="IMAGE_PNG" />
    <xs:enumeration value="IMAGE_TIFF" />
    <xs:enumeration value="TEXT_RTF" />
    <xs:enumeration value="TEXT_XML" />
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="caseActionType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="START" />
        <xs:enumeration value="START_FORWARD" />
        <xs:enumeration value="NEW_FORWARD" />
        <xs:enumeration value="NEW" />
        <xs:enumeration value="UPDATE" />
    </xs:restriction>
</xs:simpleType>

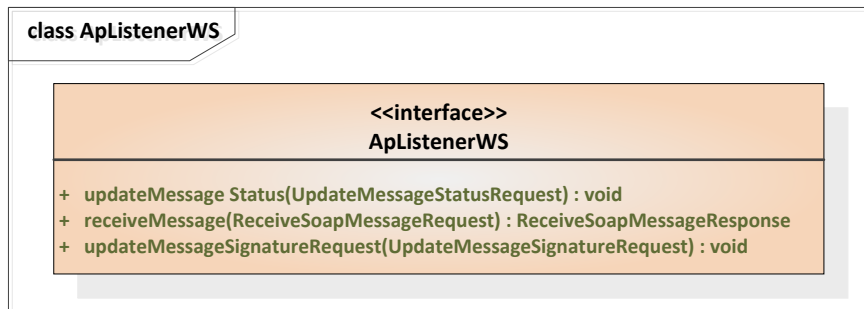
</xs:schema>
</wsdl:types>
<wsdl:message name="sendSoapMessageRequest">
    <wsdl:part element="tns:sendSoapMessageRequest" name="sendSoapMessageRequest">
    </wsdl:part>
</wsdl:message>
<wsdl:portType name="ApClientWsPort">
    <wsdl:operation name="sendSoapMessage">
        <wsdl:input message="tns:sendSoapMessageRequest" name="sendSoapMessageRequest">
        </wsdl:input>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ApClientWsPortSoap11" type="tns:ApClientWsPort">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="sendSoapMessage">
        <soap:operation soapAction="" />
        <wsdl:input name="sendSoapMessageRequest">
            <soap:body use="literal" />
        </wsdl:input>
    </wsdl:operation>
</wsdl:binding>
<wsdl:service name="ApClientWsPortService">
    <wsdl:port binding="tns:ApClientWsPortSoap11" name="ApClientWsPortSoap11">
        <soap:address location="http://localhost:8081/apClient-ws/ws" />
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

### 6.1.2 APListener Web Service

Listener SOAP Web Service interface for business application that wants to receive notifications about the received messages and message status updates like delivered or error.

### 6.1.2.1 Diagram



### 6.1.2.2 Description

| Name                                                                                                        | Preconditions                                              | Input                                                                                                                                                             | Output                                                                                                       | Exceptions                                                                                                                    |
|-------------------------------------------------------------------------------------------------------------|------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| <b>receiveMessage</b> - Method called when new SOAP message is received from the Access Point               | Ap Listener Web Service should be published and configured | <b>ReceiveSoapMessageRequest:</b><br><b>receiveSoapMessageRequest</b> - The request of the SOAP business message transformed from the low level ebMS user message | <b>ReceiveSoapMessageResponse</b> - The SOAP response containing a status                                    | <b>MessageProcessingException</b> - The business application can throw this exception on processing the message               |
| <b>updateMessageStatus</b> - Method called when new message status update is received from the Access Point | Ap Listener Web Service should be published and configured | <b>messageStatus:MessageStatus</b> - The message status that contains information about the state of the message                                                  | <b>updateMessageStatusResponse</b><br>The response containing a status of the message status acknowledgement | <b>MessageProcessingException</b> - The business application can throw this exception on processing the message status update |
| <b>updateMessageSignatureRequest</b> - Method called when the message signature is received                 | Ap Listener Web Service should be published and configured | <b>MessageSignatureRequest:MessageSignatureRequest</b> The request of message singature of the sed of the message                                                 | <b>updateMessageSignatureResponse</b><br>The response containing a status of the signature request           | <b>MessageProcessingException</b> - The business application can throw this exception on processing the message status update |

### 6.1.2.3 WSDL (APListener)

The WSDL exposed is the following:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:sch="http://eessi.dgempl.ec.europa.eu/namespaces/soap"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://eessi.dgempl.ec.europa.eu/namespaces/soap"
targetNamespace="http://eessi.dgempl.ec.europa.eu/namespaces/soap">
  <wsdl:types>
    <xs:schema xmlns="http://eessi.dgempl.ec.europa.eu/namespaces/soap"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://eessi.dgempl.ec.europa.eu/namespaces/soap">

      <xs:element name="attachmentIdentification" type="attachmentIdentification" />
      <xs:element name="caseActionType" type="caseActionType" />
      <xs:element name="caseIdentification" type="caseIdentification" />
      <xs:element name="contactTypeIdentifier" type="contactTypeIdentifier" />
      <xs:element name="documentIdentification" type="documentIdentification" />
      <xs:element name="internetMediaType" type="internetMediaType" />
      <xs:element name="messageSignature" type="messageSignature" />
      <xs:element name="messageStatus" type="messageStatus" />
      <xs:element name="messageStatusEnum" type="messageStatusEnum" />
      <xs:element name="partner" type="partner" />
      <xs:element name="receiveSoapMessageRequest" type="receiveSoapMessageRequest" />
      <xs:element name="receiveSoapMessageResponseStatusEnum" type="receiveSoapMessageResponseStatusEnum" />
      <xs:element name="receiveSoapMessageResponse" type="receiveSoapMessageResponse" />
      <xs:element name="soapAttachment" type="soapAttachment" />
      <xs:element name="soapDocumentBase" type="soapDocumentBase" />
      <xs:element name="soapMessage" type="soapMessage" />
      <xs:element name="soapSed" type="soapSed" />
      <xs:element name="standardBusinessDocumentHeader" type="standardBusinessDocumentHeader" />
      <xs:element name="updateMessageSignatureRequest" type="updateMessageSignatureRequest" />
      <xs:element name="updateMessageSignatureResponse" type="updateMessageSignatureResponse" />
      <xs:element name="updateMessageSignatureResponseEnum" type="updateMessageSignatureResponseEnum" />
      <xs:element name="updateMessageStatusRequest" type="updateMessageStatusRequest" />
      <xs:element name="updateMessageStatusResponseEnum" type="updateMessageStatusResponseEnum" />
      <xs:element name="updateMessageStatusResponse" type="updateMessageStatusResponse" />

      <xs:complexType name="receiveSoapMessageRequest">
        <xs:sequence>
          <xs:element name="soapMessage" type="soapMessage" />
        </xs:sequence>
      </xs:complexType>

      <xs:complexType name="receiveSoapMessageResponse">
        <xs:all>
          <xs:element maxOccurs="1" minOccurs="1" name="status" type="receiveSoapMessageResponseStatusEnum" />
          <xs:element maxOccurs="1" minOccurs="0" name="description" type="xs:string" />
        </xs:all>
      </xs:complexType>

      <xs:complexType name="soapMessage">
        <xs:all>
          <xs:element name="id" type="xs:string" />
          <xs:element minOccurs="0" name="businessKeystoreAlias" type="xs:string" />
          <xs:element name="receiver" type="xs:string" />
          <xs:element name="standardBusinessDocumentHeader" type="standardBusinessDocumentHeader" />
          <xs:element name="soapSed" type="soapSed" />
          <xs:element minOccurs="0" name="soapAttachments">
            <xs:complexType>
              <xs:sequence>
                <xs:element maxOccurs="unbounded" minOccurs="0" name="soapAttachment" type="soapAttachment" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:all>
      </xs:complexType>

      <xs:complexType name="standardBusinessDocumentHeader">
        <xs:all>
          <xs:element minOccurs="0" name="headerVersion" type="xs:string" />
        </xs:all>
      </xs:complexType>
    </xs:schema>
  </wsdl:types>

```

```

<xs:element name="sender" type="partner" />
<xs:element name="receivers">
  <xs:complexType>
    <xs:sequence>
      <xs:element maxOccurs="unbounded" name="partner" type="partner" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="documentIdentification" type="documentIdentification" />
<xs:element minOccurs="0" name="attachments">
  <xs:complexType>
    <xs:sequence>
      <xs:element maxOccurs="unbounded" minOccurs="0" name="attachmentIdentification"
type="attachmentIdentification" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="caseIdentification" type="caseIdentification" />
</xs:all>
</xs:complexType>

<xs:complexType name="partner">
  <xs:all>
    <xs:element minOccurs="0" name="authority" type="xs:string" />
    <xs:element name="identifier" type="xs:string" />
    <xs:element name="contactTypeIdentifier" type="contactTypeIdentifier" />
  </xs:all>
</xs:complexType>

<xs:complexType name="documentIdentification">
  <xs:all>
    <!-- <xs:element name="standard" type="xs:string"/> -->
    <!-- <xs:element name="standardVersion" type="xs:string"/> -->
    <xs:element name="type" type="xs:string" />
    <xs:element name="schemaVersion" type="xs:string" />
    <xs:element name="identifier" type="xs:string" />
    <xs:element name="setIdIdentifier" type="xs:string" />
    <xs:element name="action" type="caseActionType" />
    <xs:element name="version" type="xs:string" />
    <xs:element name="creationDate" type="xs:dateTime" />
    <xs:element minOccurs="0" name="relatedSetIdentifier" type="xs:string" />
  </xs:all>
</xs:complexType>

<xs:complexType name="attachmentIdentification">
  <xs:all>
    <xs:element name="identifier" type="xs:string" />
    <xs:element minOccurs="0" name="fileName" type="xs:string" />
    <xs:element name="internetMediaTypeCode" type="internetMediaType" />
    <xs:element minOccurs="0" name="sectionReference" type="xs:string" />
    <xs:element name="isMedical" type="xs:boolean" />
  </xs:all>
</xs:complexType>

<xs:complexType name="caseIdentification">
  <xs:all>
    <xs:element name="identifier" type="xs:string" />
    <xs:element name="type" type="xs:string" />
    <xs:element name="version" type="xs:string" />
    <!-- <xs:element name="action" type="caseActionType"/> -->
    <xs:element name="isProtectedPerson" type="xs:boolean" />
    <!-- <xs:element name="isMedicalInfo" type="xs:boolean"/> -->
  </xs:all>
</xs:complexType>

<xs:complexType name="soapSed">
  <xs:complexContent>
    <xs:extension base="soapDocumentBase">
      <xs:sequence />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="soapDocumentBase">
  <xs:all>
    <xs:element name="id" type="xs:string" />
    <xs:element name="contentUrl" type="xs:anyURI" />
  </xs:all>
</xs:complexType>

```

```

    </xs:all>
  </xs:complexType>

  <xs:complexType name="soapAttachment">
    <xs:complexContent>
      <xs:extension base="soapDocumentBase">
        <xs:sequence />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="updateMessageSignatureRequest">
    <xs:sequence>
      <xs:element name="messageSignature" type="messageSignature" />
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="updateMessageSignatureResponse">
    <xs:all>
      <xs:element maxOccurs="1" minOccurs="1" name="status" type="updateMessageSignatureResponseEnum" />
      <xs:element maxOccurs="1" minOccurs="0" name="description" type="xs:string" />
    </xs:all>
  </xs:complexType>

  <xs:complexType name="messageSignature">
    <xs:all>
      <xs:element minOccurs="0" name="messageId" type="xs:string" />
      <xs:element minOccurs="0" name="versionId" type="xs:string" />
      <xs:element minOccurs="0" name="targetSedId" type="xs:string" />
      <xs:element minOccurs="0" name="sedSignature" type="xs:string" />
      <xs:element minOccurs="0" name="lastUpdate" type="xs:dateTime" />
    </xs:all>
  </xs:complexType>

  <xs:complexType name="updateMessageStatusRequest">
    <xs:sequence>
      <xs:element name="messageStatus" type="messageStatus" />
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="updateMessageStatusResponse">
    <xs:all>
      <xs:element maxOccurs="1" minOccurs="1" name="status" type="updateMessageStatusResponseEnum" />
      <xs:element maxOccurs="1" minOccurs="0" name="description" type="xs:string" />
    </xs:all>
  </xs:complexType>

  <xs:complexType name="messageStatus">
    <xs:all>
      <xs:element name="messageId" type="xs:string" />
      <xs:element name="targetMessageId" type="xs:string" />
      <xs:element name="status" type="messageStatusEnum" />
      <xs:element minOccurs="0" name="lastUpdate" type="xs:dateTime" />
      <xs:element minOccurs="0" name="errorDescription" type="xs:string" />
    </xs:all>
  </xs:complexType>

  <xs:simpleType name="contactTypeIdentifier">
    <xs:restriction base="xs:string">
      <xs:enumeration value="DATADESTINATION" />
      <xs:enumeration value="DATASOURCE" />
      <xs:enumeration value="CASE_OWNER" />
      <xs:enumeration value="COUNTERPARTY" />
      <xs:enumeration value="INTELLIGENT_RA" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="internetMediaType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="APPLICATION_PDF" />
      <xs:enumeration value="APPLICATION_MSWORD" />
      <xs:enumeration value="APPLICATION_VND_MS_EXCEL" />
      <xs:enumeration value="APPLICATION_VND_MS_POWERPOINT" />
      <xs:enumeration value="APPLICATION_VND_OPENXMLFORMATS_OFFICEDOCUMENT_WORDPROCESSINGML_DOCUMENT" />
      <xs:enumeration value="APPLICATION_VND_OPENXMLFORMATS_OFFICEDOCUMENT_SPREADSHEETML_SHEET" />
      <xs:enumeration value="APPLICATION_VND_OPENXMLFORMATS_OFFICEDOCUMENT_PRESENTATIONML_PRESENTATION" />
    </xs:restriction>
  </xs:simpleType>
  </xs:all>
</pre>

```

```
<xs:enumeration value="APPLICATION_XML" />
<xs:enumeration value="APPLICATION_ZIP" />
<xs:enumeration value="APPLICATION_GZIP" />
<xs:enumeration value="IMAGE_JPEG" />
<xs:enumeration value="IMAGE_PNG" />
<xs:enumeration value="IMAGE_TIFF" />
<xs:enumeration value="TEXT_RTF" />
<xs:enumeration value="TEXT_XML" />
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="caseActionType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="START" />
    <xs:enumeration value="START_FORWARD" />
    <xs:enumeration value="NEW_FORWARD" />
    <xs:enumeration value="NEW" />
    <xs:enumeration value="UPDATE" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="messageStatusEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SENT" />
    <xs:enumeration value="DELIVERED" />
    <xs:enumeration value="READY" />
    <xs:enumeration value="ERROR" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="receiveSoapMessageResponseStatusEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="DELIVERED" />
    <xs:enumeration value="ERROR" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="updateMessageStatusResponseEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SUCCESS" />
    <xs:enumeration value="ERROR" />
  </xs:restriction>
</xs:simpleType>

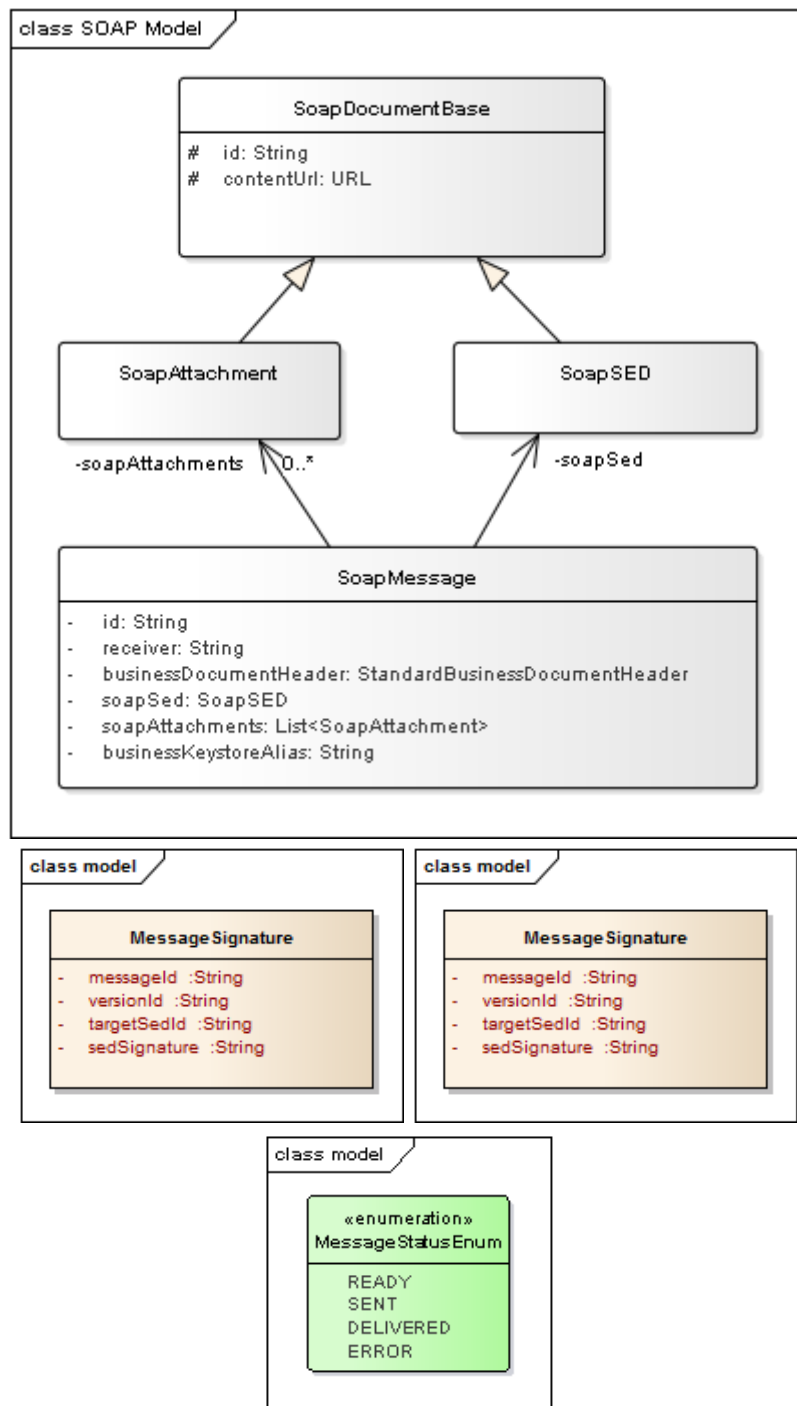
<xs:simpleType name="updateMessageSignatureResponseEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SUCCESS" />
    <xs:enumeration value="ERROR" />
  </xs:restriction>
</xs:simpleType>

</xs:schema>
</wsdl:types>
<wsdl:message name="receiveSoapMessageRequest">
  <wsdl:part element="tns:receiveSoapMessageRequest" name="receiveSoapMessageRequest">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="updateMessageSignatureRequest">
  <wsdl:part element="tns:updateMessageSignatureRequest" name="updateMessageSignatureRequest">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="updateMessageStatusResponse">
  <wsdl:part element="tns:updateMessageStatusResponse" name="updateMessageStatusResponse">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="receiveSoapMessageResponse">
  <wsdl:part element="tns:receiveSoapMessageResponse" name="receiveSoapMessageResponse">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="updateMessageSignatureResponse">
  <wsdl:part element="tns:updateMessageSignatureResponse" name="updateMessageSignatureResponse">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="updateMessageStatusRequest">
  <wsdl:part element="tns:updateMessageStatusRequest" name="updateMessageStatusRequest">
  </wsdl:part>
</wsdl:message>
```

```
<wsdl:portType name="ApListenerWsPort">
  <wsdl:operation name="receiveSoapMessage">
    <wsdl:input message="tns:receiveSoapMessageRequest" name="receiveSoapMessageRequest">
    </wsdl:input>
    <wsdl:output message="tns:receiveSoapMessageResponse" name="receiveSoapMessageResponse">
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="updateMessageSignature">
    <wsdl:input message="tns:updateMessageSignatureRequest" name="updateMessageSignatureRequest">
    </wsdl:input>
    <wsdl:output message="tns:updateMessageSignatureResponse" name="updateMessageSignatureResponse">
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="updateMessageStatus">
    <wsdl:input message="tns:updateMessageStatusRequest" name="updateMessageStatusRequest">
    </wsdl:input>
    <wsdl:output message="tns:updateMessageStatusResponse" name="updateMessageStatusResponse">
    </wsdl:output>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ApListenerWsPortSoap11" type="tns:ApListenerWsPort">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
  <wsdl:operation name="receiveSoapMessage">
    <soap:operation soapAction="" />
    <wsdl:input name="receiveSoapMessageRequest">
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="receiveSoapMessageResponse">
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="updateMessageSignature">
    <soap:operation soapAction="" />
    <wsdl:input name="updateMessageSignatureRequest">
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="updateMessageSignatureResponse">
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="updateMessageStatus">
    <soap:operation soapAction="" />
    <wsdl:input name="updateMessageStatusRequest">
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="updateMessageStatusResponse">
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:service name="ApListenerWsPortService">
  <wsdl:port binding="tns:ApListenerWsPortSoap11" name="ApListenerWsPortSoap11">
    <soap:address location="http://localhost:8082/apClient-ws-listener/ws" />
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

## 6.2 Data contract

### 6.2.1 Message model



#### 6.2.1.1 SoapAttachment

Define attachment that should be sent within SOAP BusinessMessage.

Extends SoapDocumentBase.

### 6.2.1.2 SoapDocumentBase

Base class for Structured Electronic Document or Attachment that is sent within SOAP BusinessMessage:

| Attribute  | Description                                                                      | Data Type |
|------------|----------------------------------------------------------------------------------|-----------|
| id         | The attachment or Structured Electronic Document identifier                      | String    |
| contentUrl | URL referring to the content of the attachment or Structured Electronic Document | URL       |

### 6.2.1.3 SoapMessage

SOAP Business Message represents the Business Document exchanged between Institutions in EESSI network

| Attribute              | Description                                                  | Data Type              |
|------------------------|--------------------------------------------------------------|------------------------|
| id                     | Business Message identifier                                  | String                 |
| receiver               | Where the message is sent to                                 | String                 |
| businessDocumentHeader | Business Document Header that describes the document content | BusinessDocumentHeader |
| soapSed                | SOAP Structured Electronic Document                          | SoapSED                |
| soapAttachments        | SOAP Attachments list of the Business Message                | List<SoapAttachment>   |
| businessKeystoreAlias  | The alias/certificate used for business signature            | String                 |

### 6.2.1.4 MessageStatus

Please refer to previous chapter about ApClient for details.

### 6.2.1.5 MessageSignature

Please refer to previous chapter about ApClient for details

### 6.2.1.6 MessageStatusEnum

Please refer to previous chapter about ApClient for details.

### 6.2.1.7 SoapSED

Base class for SOAP Structured Electronic Document or Attachment that is sent within SOAP BusinessMessage

| Attribute  | Description                                                                  | Data Type |
|------------|------------------------------------------------------------------------------|-----------|
| id         | The attachment or Structured Electronic Document identifier                  | String    |
| contentUrl | URL referring to the content of attachment or Structured Electronic Document | URL       |

### 6.2.2 Configuration model

Remains the same with ApClient. Please refer to the previous chapter for details.

### 6.2.3 Standard Business Document Header Model

Remains the same with ApClient. Please refer to the previous chapter for details.

## 6.3 Builders

The builders in this chapter are useful to easily build the data contract needed by a Java client or a Web Service written in Java.

### 6.3.1 SbdhBuilder

Remains the same with ApClient. Please refer to the previous chapter for details.

## 6.3.2 SoapMessageBuilder

### 6.3.2.1 Diagram



### 6.3.2.2 How to build a SOAP Message with SOAP Message Builder based on an already built SBDH

| Name                                                                                        | Input                                                                                                                                                                                                            | Output              | Exceptions |
|---------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|------------|
| <b>message</b> - Static method for creating a brand new message builder                     | N/A                                                                                                                                                                                                              | The new builder     | N/A        |
| <b>withId</b> - Message id                                                                  | <b>id:String</b> – Message id                                                                                                                                                                                    | The current builder | N/A        |
| <b>withActiveReceiver</b> - Selects the active/current receiver from the SBDH receiver list | <b>receiver:String</b> - Receiver identifier in the form @CountryCodeIso2: institutionId                                                                                                                         | The current builder | N/A        |
| <b>withBusinessDocument Header</b> - SBDH information                                       | <b>businessDocumentHeader: StandardBusinessDocumentHeader</b> - The SBDH model built with the SBDH builder                                                                                                       | The current builder | N/A        |
| <b>withContentForSed</b> - Adds the actual SED content based on the SED instance identifier | <b>content:URL</b> - The URL of the content that needs to be set for the DocumentIdentification from the SBDH<br><b>setIdIdentifier:String</b> - The set identifier for the DocumentIdentification from the SBDH | The current builder | N/A        |

| Name                                                                                                                                                                                                   | Input                                                                                                                                                                                                                                                                          | Output                     | Exceptions                          |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|-------------------------------------|
| <b>withContentForAttachment</b> - Adds the actual attachment content based on the attachment instance identifier                                                                                       | <b>content:URL</b> - The URL that refers to the content that needs to be set for the AttachmentIdentification from SBDH that has the attachmentIdentifier<br><b>attachmentIdentifier:String</b> - Identifier for an AttachmentIdentification from the SBDH list of attachments | The current builder        | N/A                                 |
| <b>withBusinessKeystoreAlias</b> - adds the certificate to be used for business message signing                                                                                                        | businessKeystoreAlias:String - identifier for the certificate to be used in the business message signing                                                                                                                                                                       | The current builder        | N/A                                 |
| <b>build</b> - Constructs the business message model<br><b>Preconditions:</b> All the with* methods from this table must be called before, except withContentForAttachment if there are no attachments |                                                                                                                                                                                                                                                                                | The business message model | <b>StructureValidationException</b> |

### 6.3.2.3 How to build a SOAP Message with SOAP Message Builder from scratch

| Name                                                                                                                               | Input                                                                                                                                                                                                                                                                                                                                     | Output              | Exceptions |
|------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|------------|
| <b>message</b> - Static method for creating a brand new message builder                                                            | N/A                                                                                                                                                                                                                                                                                                                                       | The new builder     | N/A        |
| <b>withId</b> - Message id                                                                                                         | <b>id:String</b> - Message id                                                                                                                                                                                                                                                                                                             | The current builder | N/A        |
| <b>withSender</b> - Sending party information                                                                                      | <b>senderIdentifier:String</b> - Sender identifier in the form CountryCodeIso2: institutionId<br><b>senderContactTypeIdentifier:ContactTypeIdentifier</b> - Sender process role (case owner or counter party)                                                                                                                             | The current builder | N/A        |
| <b>withAddedReceiver</b> - Receiver party information, if the case is multi-lateral call this method for each receiver participant | <b>receiverIdentifier:String</b> - Receiver identifier in the form CountryCodeIso2: institutionId<br><b>receiverContactTypeIdentifier:ContactTypeIdentifier</b> - Receiver process role (case owner or counter party)<br><b>isActive:boolean</b> - Whether this newly added receiver is the active/current receiver that gets the message | The current builder | N/A        |

| Name                                                                                                          | Input                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Output              | Exceptions |
|---------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|------------|
| <b>withDocumentIdentification</b> - SED information                                                           | <b>standard:String</b> - Urn for the SED standard used. This must be 'urn:eu:europa:ec:dgempl:eessi:sed'<br><b>standardVersion:String</b> - Version of the SED standard used. This must be '3.0'<br><b>type:String</b> - SED type, e.g.:P2000, F001, U002<br><b>typeVersion:String</b> - SED schema version, e.g.:3.2<br><b>setIdIdentifier:String</b> - SED instance identifier<br><b>documentVersion:String</b> - Version of SED instance identifier<br><b>instanceIdentifier:String</b> - SED global instance identifier<br><b>creationDate:Date</b> - SED creation date<br><b>contentUrl:URL</b> - The Url that refers to the content that needs to be set for the SOAP SED | The current builder | N/A        |
| <b>withAddedAttachment</b> - SOAP Attachment information, call this method to the message                     | <b>mimeType:InternetMediaType</b> - Mime type of the attachment<br><b>attachmentIdentifier:String</b> - Unique attachment identifier<br><b>contentUrl:URL</b> - The URL that refers to the content that needs to be set for the attachment                                                                                                                                                                                                                                                                                                                                                                                                                                      | The current builder | N/A        |
| <b>withAddedAttachment</b> - SOAP Attachment information, call this method for each attachment to the message | <b>mimeType:InternetMediaType</b> - Mime type of the attachment<br><b>attachmentIdentifier:String</b> - Unique attachment identifier<br><b>sectionReference:String</b> - Reference to a section of the SED to which the attachment links in the format of an XPath expression, by default this is 'sed'<br><b>isMedicalInfo:boolean</b> - Whether attachment contains medical information or not, by default this is 'false'<br><b>contentUrl:URL</b> - The Url that refers to the content that needs to be set for the SOAP attachment<br><b>fileName:String</b> - the filename of the attachment as it should be displayed in the UI                                          | The current builder | N/A        |
| <b>withCaseIdentification</b> - Case information                                                              | <b>buc:String</b> - Business use case type, e.g.:P_BUC_01, FB_BUC_01<br><b>version:String</b> - Version of the business use case, e.g.:3.2<br><b>action:CaseActionType</b> - Case action, e.g.:START, UPDATE<br><b>caseIdentifier:String</b> - Case identifier to which the message relates to                                                                                                                                                                                                                                                                                                                                                                                  | The current builder | N/A        |

| Name                                                                                                                                                                                                                                                                         | Input                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | Output                     | Exceptions                          |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|-------------------------------------|
| <b>withCaseIdentification</b> - Case information                                                                                                                                                                                                                             | <b>buc:String</b> - Business use case type, e.g.:P_BUC_01, FB_BUC_01<br><b>version:String</b> - Version of the business use case, e.g.:3.2<br><b>action: CaseActionType</b> - Case action, e.g.:START, UPDATE<br><b>caseIdentifier:String</b> - Case identifier to which the message relates to<br><b>isPersonalProtected:boolean</b> - Whether the case is protected or not, by default this is 'false'<br><b>isMedicalInfo:Boolean</b> - Whether the case contains medical info, by default this is 'false' | The current builder        | N/A                                 |
| <b>withRelatedDocumentSet Identification</b> - Link to a possible related document                                                                                                                                                                                           | <b>relatedSetIdentifier:String</b> - Related SED instance identifier                                                                                                                                                                                                                                                                                                                                                                                                                                          | The current builder        | N/A                                 |
| <b>withBusinessKeystoreAlias</b> – <b>adds the certificate to be used for business message signing</b>                                                                                                                                                                       | businessKeystoreAlias:String – identifier for the certificate to be used in the business message signing                                                                                                                                                                                                                                                                                                                                                                                                      | The current builder        | N/A                                 |
| <b>build</b> - Constructs the business message model<br><b>Preconditions:</b> The following with* methods from this table must be called before:<br>withSender<br>withAddedReceiver<br>withDocumentIdentification<br>withCaseIdentification (one of the two forms available) | N/A                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | The business message model | <b>StructureValidationException</b> |

## 6.4 Fault contract

Remains the same with ApClient. Please refer to the previous chapter for details.

## 7 RINA Technical Messaging Interface

### 7.1 Service contract

The service provided for the technical messaging interface follows the file drop concept where files that need to be sent are written in a specific output folder and files that are received are read from a specific input folder.

#### 7.1.1 Write

There are two types of file that are serialized:

- Configuration
  - Pmode files as defined in 7.2.3
  - Pulling configuration as defined in 7.2.4
- Output messages as defined in 7.2.1

The Holodeck path for the pulling configuration file is `<HOLODECK_PATH>/conf` and for the pmodes is `<HOLODECK_PATH>/conf/pmodes`

The Holodeck path for the output messages is `<HOLODECK_PATH>/data/msg_out`

#### 7.1.2 Read

There are two types of files that can be de-serialized:

- Input messages as defined in 7.2.1
- Input status updates as defined in 7.2.2

The Holodeck path for the input messages is `<HOLODECK_PATH>/data/msg_in/<TENANT ID>`

### 7.2 Data contract

#### 7.2.1 MessageMetadataType

This type is generated from the Holodeck messagemetadata.xsd attached below and defined in the eu.ec.dgempl.apclient.holodeck.dto.mmd package:

```
<?xml version="1.0"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
  targetNamespace="http://holodeck-b2b.org/schemas/2014/06/mmd" attributeFormDefault="unqualified"
  xmlns:tns="http://holodeck-b2b.org/schemas/2014/06/mmd"
  xmlns:xm1="http://www.w3.org/XML/1998/namespace">

  <xs:annotation>
    <xs:documentation xml:lang="en">This schema defines the structure of the message meta-data (or short
MMD) document used by Holodeck B2B.
XML documents of this type are used when exchanging messages between Holodeck B2B and the connected business
application. When sending the Producer should supply the MMD and when delivering a message to the Consumer
Holodeck B2B will supply the MMD. Note that delivery is done by a delivery class and that it may choose not
to supply the MMD because the meta-data is provided otherwise.</xs:documentation>
  </xs:annotation>
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
    schemaLocation="http://www.w3.org/2001/03/xml.xsd"/>
  <xs:element name="MessageMetaData" type="tns:MessageMetaDataType">
  </xs:element>
  <xs:complexType name="MessageMetaDataType">
    <xs:sequence>
      <xs:annotation>
        <xs:documentation/>
      </xs:annotation>
      <xs:element minOccurs="0" name="MessageInfo">
        <xs:annotation>
          <xs:documentation>MessageInfo contains some technical details that are used to identify
the UserMessage and its relation to another UserMessage. As none of the contained information is required
when submitting the user message this element is optional.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="tns:MessageInfo">
              <xs:attribute name="mpc" type="tns:non-empty-string">
                <xs:annotation>
                  <xs:documentation>The message partition channel the user message is
assigned to. As described in the ebMS Core Specification it is not required to specify the MPC. If no MPC is
given for a message it assumed to be assigned to the default MPC.
                </xs:documentation>
              </xs:annotation>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element minOccurs="0" name="PartyInfo" type="tns:PartyInfo">
        <xs:annotation>
          <xs:documentation>PartyInfo contains information on the sender and [intended] receiver
of the user message. NOTE that the sender and receiver are defined by the roles an MSH acts in when
exchanging user messages. These roles are defined in section 2.1.3 of the ebMS Core Specification.
This element is optional as information about the sender and receiver is normally included in the P-Mode
configuration and not needed during message submission. When however the P-Mode id is not directly specified
in the message meta-data the information in this element can be used for determining the P-Mode to
use.</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element minOccurs="1" name="CollaborationInfo" type="tns:CollaborationInfo">
        <xs:annotation>
          <xs:documentation>Meta data about the context of the message exchange, i.e. the business
collaboration. Most of the information included in this element can be contained in the P-Mode. The
ConversationId child element is however REQUIRED as it indicates the business level conversation the message
exchange is part of and which is known only when the message is submitted.
Also REQUIRED when submitting a message is the P-Mode id in the AgreementRef child element. This enables
Holodeck B2B to determine the P-Mode that configures the message processing.</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element minOccurs="0" name="MessageProperties">
        <xs:annotation>
          <xs:documentation>Additional user defined, i.e. by the business application, message
properties. These properties will not be used by Holodeck B2B to determine the P-Mode.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element maxOccurs="unbounded" ref="tns:Property">
              <xs:annotation>
                <xs:documentation>The Property element represents user defined, i.e. non
standard, properties. </xs:documentation>
                After saving the draft, optional comments can be added, and the
SED can be sent. Before transmission, the receiving institutions have to be selected (see Error! Reference s

```

source not found.). An alarm can be set when sending the reminder in the same way as for any other document sent (cf. section Error! Reference source not found.).

```

        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="PayloadInfo" type="tns:PayloadInfo" minOccurs="0">
  <xs:annotation>
    <xs:documentation>Information about the payloads of the user message. As the payloads
contain the actual business information to exchange normally a user message will contain at least one
payload.</xs:documentation>
  </xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
<xs:complexType name="MessageInfo">
  <xs:sequence>
    <xs:element minOccurs="0" name="Timestamp" type="xs:dateTime">
      <xs:annotation>
        <xs:documentation>The timestamp reflect the time when the message header is created (see
5.2.2.1 of the ebMS Core Specification). As the header will be created by Holodeck B2B when sending the
message the Timestamp element SHOULD NOT be used when submitting the user message. </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element minOccurs="0" name="MessageId" type="tns:non-empty-string">
      <xs:annotation>
        <xs:documentation>The submitter MAY set the MessageId if it requires a specific id. It
MUST ensure that the supplied value is a globally unique identifier conforming to the MessageId format as
defined in [RFC2822]. When no message-id is specified Holodeck B2B will assign one and return this to the
submitter of the message. </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element minOccurs="0" name="RefToMessageId" type="tns:non-empty-string">
      <xs:annotation>
        <xs:documentation>Indicates that this user message is a response to another user
message. If the submitted message is a response message the RefToMessageId MUST be set by the submitter,
Holodeck B2B will not try to detect message relations.</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="PartyInfo">
  <xs:sequence>
    <xs:element minOccurs="0" name="From" type="tns:TradingPartner">
      <xs:annotation>
        <xs:documentation>Contains information about the identification of the party that is the
[intended] receiver of the user message, i.e. this is the MSH acting in the Receiving
role.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element minOccurs="0" name="To" type="tns:TradingPartner">
      <xs:annotation>
        <xs:documentation>Contains information about the identification of the party that is the
sender of the user message, i.e. this is the MSH acting in the Sending role.</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="TradingPartner">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="PartyId" type="tns:PartyId">
      <xs:annotation>
        <xs:documentation>The PartyId element contains an identifier of the party involved in
the message exchange. When identifying business partners a naming scheme can be used. This can be indicated
using the type attribute. For more information see section 5.2.2.4 of the ebMS Core Specification.
If this element occurs multiple all instances MUST identify the same business partner. This element can not
be used for multi-cast.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element minOccurs="1" name="Role" type="tns:non-empty-string">
      <xs:annotation>
        <xs:documentation>A identification of the business role the party acts in. This is
(probably) different from the ebMS role (Sending/Receiving) the party acts in.</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>

```

```

</xs:complexType>
<xs:complexType name="PartyId">
  <xs:simpleContent>
    <xs:extension base="tns:non-empty-string">
      <xs:attribute name="type" type="tns:non-empty-string" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="CollaborationInfo">
  <xs:sequence>
    <xs:element minOccurs="0" name="AgreementRef" type="tns:AgreementRef">
      <xs:annotation>
        <xs:documentation>This element contains information about the agreement that governs the
message exchange. See also section 5.2.2.7 of the ebMS Core Specification. Holodeck B2B uses the pmode
attribute from this element to determine which P-Mode configuration governs the message exchange.
</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element minOccurs="0" name="Service" type="tns:Service">
      <xs:annotation>
        <xs:documentation>The service that is [supposed] to handle the user
message.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element minOccurs="0" name="Action" type="xs:token">
      <xs:annotation>
        <xs:documentation>The operation/activity requested to be executed.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element minOccurs="1" name="ConversationId" type="xs:token">
      <xs:annotation>
        <xs:documentation>The ConversationId element identifies the business level conversation
the message exchange is part of. Because this is only known to the business application it MUST be supplied
when the message is submitted. </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="Service">
  <xs:simpleContent>
    <xs:extension base="tns:non-empty-string">
      <xs:attribute name="type" type="tns:non-empty-string" use="optional" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="AgreementRef">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="type" type="tns:non-empty-string" use="optional" />
      <xs:attribute name="pmode" type="tns:non-empty-string" use="optional" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="PayloadInfo">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="PartInfo" type="tns:PartInfo">
      <xs:annotation>
        <xs:documentation>Information about a specific payload. There is one REQUIRED
information element, the location attribute, which MUST contain the location where the data of the payload
can be found. </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="PartInfo">
  <xs:sequence>
    <xs:element minOccurs="0" name="Schema" type="tns:Schema" />
    <xs:element minOccurs="0" name="Description" type="tns:Description" />
    <xs:element minOccurs="0" name="PartProperties" type="tns:PartProperties" />
  </xs:sequence>
  <xs:attribute default="attachment" name="containment">
    <xs:annotation>
      <xs:documentation>Defines how the payload should be included in the message. There are three
options:
1) Include as a SOAP attachment [value="attachment"] - This is default way to package a payload as it can
contain both text as well as binary data;

```

2) Include in the SOAP body [value="body"] - When the payload is an XML document it can be included in the SOAP body. For WS-I conformance there SHOULD only be one payload in the SOAP body;

3) External to the message [value="external"] - The payload is stored at another location and only referred to from within the ebMS message. This option is defined in the ebMS Core Specification, but how such payload must be processed is left to implementation. It is therefore NOT RECOMMENDED to use this containment option.</xs:documentation>

```
</xs:annotation>
<xs:simpleType>
  <xs:restriction base="xs:normalizedString">
    <xs:enumeration value="body"/>
    <xs:enumeration value="attachment"/>
    <xs:enumeration value="external"/>
  </xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="uri" type="xs:anyURI">
  <xs:annotation>
```

**<xs:documentation>**Each payload included in the user message, with exception of the one included in the SOAP body, is referenced from the ebMS header using the href attribute in the PartInfo element. For external payloads this attribute **MUST** be supplied and identify the location where the payload is stored.

Because payloads may be related to each other the business application may wish to set these references. It is however RECOMMENDED to use the PartProperties for this. If a reference is given for a payload included as an attachment the uri MUST be a valid MIME Content-id as defined in [RFC2392]. If the reference is for a payload contained in the SOAP body the uri MUST be a valid xml id (as specified in <http://www.w3.org/TR/xml-id/>). The value of this attribute SHOULD only contain the id itself and no prefix ("cid:" or "#"), Holodeck B2B will add the correct prefix.</xs:documentation>

```
</xs:annotation>
</xs:attribute>
<xs:attribute name="mimeType" type="tns:non-empty-string" use="optional">
  <xs:annotation>
    <xs:documentation>The MIME type of payload content. This is an OPTIONAL attribute, when no
MIME type is given Holodeck B2B will try to detect the correct MIME type based on the payload
content.</xs:documentation>
```

```

        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="location" type="xs:anyURI" use="optional">
        <xs:annotation>
            <xs:documentation>The location where Holodeck B2B can retrieve the payload content. This
attribute MUST be specified for payloads contained in the message, i.e. containment is body or attachment.
Not used for external payloads</xs:documentation>

```

```
</xs:annotation>
</xs:attribute>
</xs:complexType>
<xs:complexType name="Schema">
  <xs:attribute name="location" type="xs:anyURI" use="required"/>
  <xs:attribute name="version" type="tns:non-empty-string" use="optional"/>
  <xs:attribute name="namespace" type="tns:non-empty-string" use="optional"/>
</xs:complexType>
<xs:complexType name="PartProperties">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" minOccurs="1" ref="tns:Property"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="Property" type="tns:Property"/>
<xs:complexType name="Property">
  <xs:annotation>
    <xs:documentation>NOTE: This type definition differs from the definition of the Property type in
the XML Schema Definition from the ebMS Core Specification as this definition includes the type attribute.
The omission of the type attribute is a known issue to the OASIS TC (see https://tools.oasis-
open.org/issues/browse/EBXMLMSG-2) and will be resolved in a future release of the XML Schema. For now it
means it is NOT RECOMMENDED to use the type attribute as interoperability between MSHs can not be
guaranteed!</xs:documentation>
  </xs:annotation>
</xs:complexType>
</xs:schema>
```

```

</xs:annotation>
<xs:simpleContent>
  <xs:extension base="tns:non-empty-string">
    <xs:attribute name="name" type="tns:non-empty-string" use="required"/>
    <xs:attribute name="type" type="tns:non-empty-string" />
  </xs:extension>
</xs:simpleContent>
</xs:complexType>
<xs:complexType name="Description">
  <xs:simpleContent>
    <xs:extension base="tns:non-empty-string">
      <xs:attribute ref="xml:lang" use="required"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

```

```
</xs:simpleContent>
</xs:complexType>
<xs:simpleType name="non-empty-string">
  <xs:restriction base="xs:string">
    <xs:minLength value="1"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>
```

The XSD contains all the documentation necessary to understand the DTO model.

When using the RINA Business Messaging Interface for sending/receiving a message, the business Message is translated to/from this MessageMetaData DTO.

### 7.2.2 SignalMessage

This type is generated from the Holodeck signalmetadata.xsd attached below and defined in the eu.ec.dgempl.apclient.holodeck.dto.mmd package:

The XSD contains all the documentation necessary to understand the DTO model.

When using the RINA Business Messaging Interface this SignalMessage DTO is translated into the business MessageStatus that indicates whether the message was delivered or there was an error.

### 7.2.3 PMode



This type is generated from the Holodeck messametadata.xsd attached below and defined in the eu.ec.dgempl.apclient.holodeck.dto.mmd package:

The XSD contains all the documentation necessary to understand the DTO model.

When using the RINA Business Messaging Interface for configuring, the business ApClientConfiguration is translated to a list of this PMode DTO.

### 7.2.4 Pulling



This type is generated from the Holodeck pullingconfig.xsd attached below and defined in the eu.ec.dgempl.apclient.holodeck.dto.mmd package:

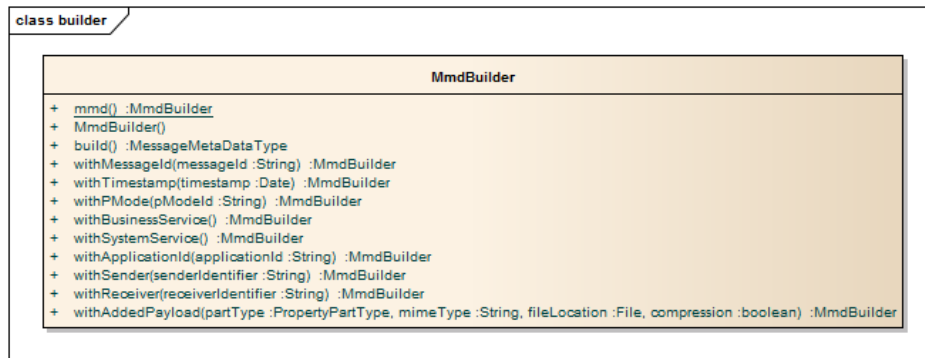
The XSD contains all the documentation necessary to understand the DTO model.

When using the RINA Business Messaging Interface for configuring, the business ApClientConfiguration generates the technical pulling configuration as well.

## 7.3 Builders

### 7.3.1 MmdBuilder

#### 7.3.1.1 Diagram



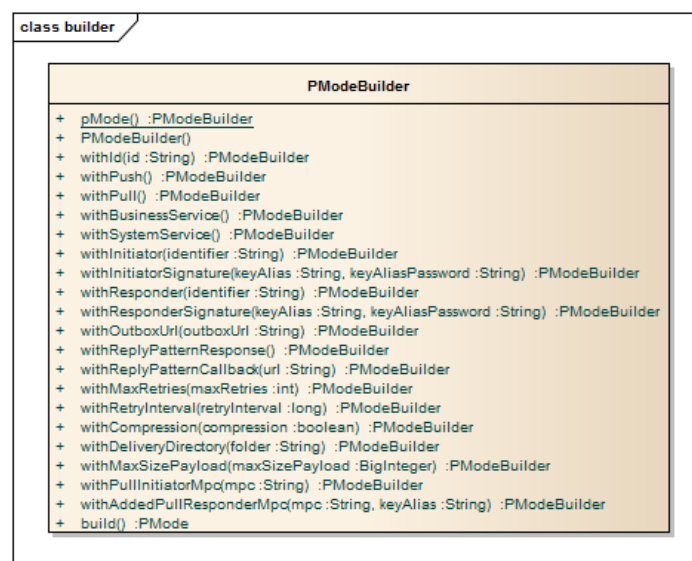
#### 7.3.1.2 Description

| Name                                                                                     | Input                                                                                                                                                                                                                                                                                                                                                                                        | Output              | Exceptions |
|------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|------------|
| <b>mmd</b> - Static method for creating a brand new message meta-data builder            | N/A                                                                                                                                                                                                                                                                                                                                                                                          | The new builder     | N/A        |
| <b>withMessageId</b> - Message id                                                        | <b>messageId:String</b> - The unique message Id used for identify the message. Duplicates message Ids will be rejected by Holodeck                                                                                                                                                                                                                                                           | The current builder | N/A        |
| <b>withTimestamp</b> - Use this if you don't want Holodeck to create the timestamp       | <b>timestamp:Date</b> - The date the message header is created                                                                                                                                                                                                                                                                                                                               | The current builder | N/A        |
| <b>withPMode</b> - MSH process mode id                                                   | <b>pModeId:String</b> - The P-Mode Id that is used to send the message                                                                                                                                                                                                                                                                                                                       | The current builder | N/A        |
| <b>withSender</b> - Sending party information                                            | <b>senderIdentifier:String</b> - The sender party identifier involved in the message exchange. For EESSI this is the institution id as retrieved from the Institution Repository that sends the message                                                                                                                                                                                      | The current builder | N/A        |
| <b>withReceiver</b> - Receiver party information                                         | <b>receiverIdentifier:String</b> - The receiver party identifier involved in the message exchange. For EESSI this is the institution id as retrieved from the Institution Repository that receives the message                                                                                                                                                                               | The current builder | N/A        |
| <b>withAddedPayload</b> - Use this to add the SED and also to add all attachments if any | <b>partType:PropertyPartType</b> - The payload type included in the message. For EESSI the payload containment is always attachment and partType specifies whether the payload is SED or Attachment<br><b>mimeType:String</b> - The MIME type of payload content<br><b>fileLocation:File</b> - The file where Holodeck B2B can retrieve the payload content. Please notice that the filename | The current builder | N/A        |

| Name                                                                                                                                                                 | Input                                                                                                                                                                                   | Output               | Exceptions |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------|------------|
|                                                                                                                                                                      | from the location is going to be used as an href in ebMS, so this must be unique.<br><b>compression:boolean</b> - Whether the payload should be compressed or not, by default it is not |                      |            |
| <b>withBusinessService</b> - Marks the message as a business message: setups the business partyId roles, collaboration service name and other business specific info | setups the business partyId roles, collaboration service name and other business specific info                                                                                          | The current builder  | N/A        |
| <b>withSystemService</b> - Marks the message as a system message: setups the system partyId roles, collaboration service name and other system specific info         | setups the system partyId roles, collaboration service name and other system specific info                                                                                              | The current builder  | N/A        |
| <b>withApplicationId</b> - Sending application party information                                                                                                     | <b>applicationId</b> The sender party application identifier involved in the message exchange                                                                                           | The current builder  | N/A        |
| <b>build</b> - Constructs the message meta-data DTO<br><b>Preconditions:</b> All the with* methods from this table must be called before, except withTimestamp       | N/A                                                                                                                                                                                     | The user message DTO | N/A        |

## 7.3.2 PModeBuilder

### 7.3.2.1 Diagram



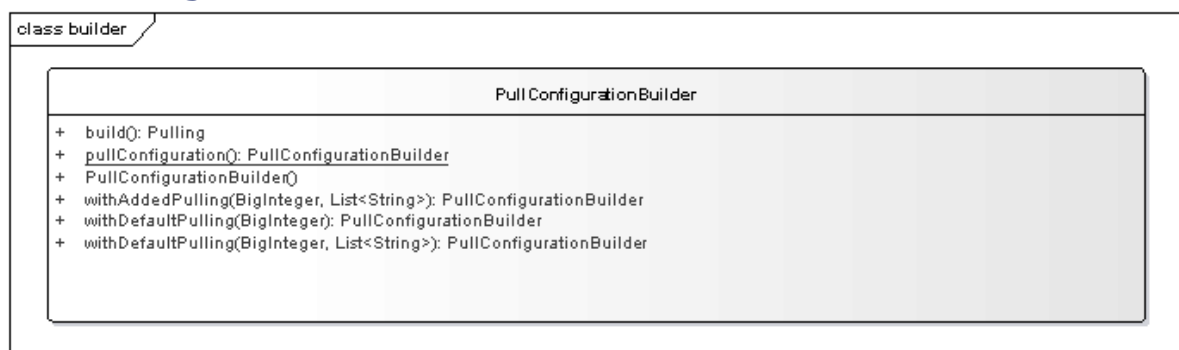
### 7.3.2.2 Description

| Name                                                                                                                   | Input                                                                                                                                                      | Output              | Required |
|------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|----------|
| <b>pMode</b> - Static method for creating a brand new p-mode builder                                                   | N/A                                                                                                                                                        | The new builder     | N/A      |
| <b>withId</b> - P-mode id                                                                                              | <b>id:String</b> - The identifier of this P-Mode.                                                                                                          | The current builder | N/A      |
| <b>withPush</b> - This setups the PUSH message exchange pattern                                                        | N/A                                                                                                                                                        | The current builder | N/A      |
| <b>withPull</b> - This setups the PULL message exchange pattern                                                        | N/A                                                                                                                                                        | The current builder | N/A      |
| <b>withInitiator</b> - P-mode initiator party id                                                                       | <b>identifier:String</b> - Initiator party id in format CountryCodeIso2:organization Id                                                                    | The current builder | N/A      |
| <b>withInitiatorSignature</b> - Signature information for the initiator party                                          | <b>keyAlias:String</b> - Initiator key store alias<br><b>keyAliasPassword:String</b> - Initiator public key password                                       | The current builder | N/A      |
| <b>withResponder</b> - P-mode responder party id                                                                       | <b>identifier:String</b> - Responder party id in format CountryCodeIso2:organizationId                                                                     | The current builder | N/A      |
| <b>withResponderSignature</b> - Signature information for the responder party                                          | <b>keyAlias:String</b> - Responder key store alias<br><b>keyAliasPassword:String</b> - Responder public key password                                       | The current builder | N/A      |
| <b>withOutboxUrl</b> - Receiving MSH connection details                                                                | <b>outboxUrl:String</b> - The URL location where the receiving MSH is listening for messages                                                               | The current builder | N/A      |
| <b>withReplyPatternResponse</b> - The Receipt signal will be sent synchronously in the transport back-channel response | N/A                                                                                                                                                        | The current builder | N/A      |
| <b>withReplyPatternCallback</b> - The Receipt signal will be sent as a callback to the URL                             | <b>url:String</b> - The URL where the Receipt will be sent                                                                                                 | The current builder | N/A      |
| <b>withMaxRetries</b> - Sets the maximum number of retries, by default there are no retries                            | <b>maxRetries:int</b> - Maximum of times a message should be resent if no Receipt is received. Setting this value to zero disables the retry functionality | The current builder | N/A      |
| <b>withRetryInterval</b> - Sets the interval for the next send retry, by default this is set to 30 seconds             | <b>retryInterval:long</b> - The time interval in seconds before a message is resent, the time to wait for a Receipt                                        | The current builder | N/A      |
| <b>withMaxSizePayload</b> - Sets the maximum size of the payload                                                       | <b>maxSizePayload:BigInteger</b> - The maximum SBD or attachment size that can be processed by the MSH specified in KB                                     | The current builder | N/A      |

| Name                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | Input                                                                                          | Output              | Required                            |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------|---------------------|-------------------------------------|
| <b>withCompression</b> - Whether the compression is enabled or not, by default compression is disabled                                                                                                                                                                                                                                                                                                                                                                               | <b>compression:boolean</b> - Enables compression feature. By default, compression is disabled. | The current builder | N/A                                 |
| <b>withPullInitiatorMpc</b> - Sets the pull initiator mpc                                                                                                                                                                                                                                                                                                                                                                                                                            | <b>mpc:String</b> - The sub-channel the pull initiator uses to retrieve its messages           | The current builder | N/A                                 |
| <b>withBusinessService</b> - Marks the p-mode as a business p-mode                                                                                                                                                                                                                                                                                                                                                                                                                   | setups the business partyId roles, collaboration service name and other business specific info | The current builder | N/A                                 |
| <b>withSystemService</b> - Marks the p-mode as a system p-mode                                                                                                                                                                                                                                                                                                                                                                                                                       | setups the system partyId roles, collaboration service name and other system specific info     | The current builder | N/A                                 |
| <b>build</b> - Constructs the p-mode DTO<br><br><b>Preconditions:</b> The following with* methods from this table must be called before in case of PUSH mode:<br>withId<br>withPush<br>withInitiator<br>withInitiatorSignature<br>withOutboxUrl<br><br>The following with* methods from this table must be called before in case of PULL mode:<br>withId<br>withPull<br>withInitiator<br>withInitiatorSignature<br>withOutboxUrl<br>withReplyPatternCallback<br>withPullInitiatorMpc |                                                                                                | The p-mode DTO      | <b>StructureValidationException</b> |

### 7.3.3 PullConfigurationBuilder

#### 7.3.3.1 Diagram



### 7.3.3.2 Description

| Name                                                                                                                                                                                                                                                                  | Input                                                                                                                                                                | Output                        | Exceptions                                 |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------|--------------------------------------------|
| <b><i>pullConfiguration</i></b> - Static method for creating a brand new pull configuration builder                                                                                                                                                                   | N/A                                                                                                                                                                  | The new builder               | N/A                                        |
| <b><i>withDefaultPulling</i></b> - Sets the default pulling interval in seconds                                                                                                                                                                                       | <b><i>defaultInterval:BigInteger</i></b> - The interval in seconds between two pull requests                                                                         | The current builder           | N/A                                        |
| <b><i>withDefaultPulling</i></b> - Sets the default pulling interval for a list of p-modes                                                                                                                                                                            | <b><i>defaultInterval:BigInteger</i></b> - The interval in seconds between two pull requests<br><br><b><i>pmodes:List&lt;String&gt;</i></b> - The list of pmode ids  | The current builder           | N/A                                        |
| <b><i>withAddedPulling</i></b> - Sets the specific pulling interval for a list of p-modes                                                                                                                                                                             | <b><i>specificInterval:BigInteger</i></b> - The interval in seconds between two pull requests<br><br><b><i>pmodes:List&lt;String&gt;</i></b> - The list of pmode ids | The current builder           | N/A                                        |
| <b><i>build</i></b> - Constructs the pulling configuration DTO<br><br><b>Preconditions:</b> The following with* methods from this table must be called before:<br><br>withDefaultPulling(defaultInterval)<br><br>The other with* are not intended for production use. | N/A                                                                                                                                                                  | The pulling configuration DTO | <b><i>StructureValidationException</i></b> |

## 7.4 Fault contract

### 7.4.1 StructureValidationException

This exception might occur when using any of the builders on the build () method.

If the object was not built correctly (using all the required with\* methods or passing bad parameters) this exception will be thrown along with a message saying why the object could not be constructed.

## 8 Samples

### 8.1 RINA Business Messaging Interface

#### 8.1.1 How to create a configuration

First step in configuring the ApClient is building the configuration object ApClientConfiguration.

Below you can find how this can be done for a communication via AP when sending messaging via push and retrieving messages via pull:

```
ApClientConfiguration apClientConfiguration = new ApClientConfiguration();
apClientConfiguration.setMshPath("c:\\all\\holodeck-b2b-2.1\\");
apClientConfiguration.setOperationMode(OperationModeEnum.AP_MODE);
apClientConfiguration.setUsePullMode(true);
apClientConfiguration.setValidationMode(ValidationModeEnum.NO_VALIDATION);
apClientConfiguration.setValidationPath("c:\\all\\holodeck-b2b-2.1\\repository\\bmp\\");
apClientConfiguration.setPrivateKeyStorePassword("<psw_of_PrivateKeystore>");
apClientConfiguration.setPublicKeyStorePassword("<psw_of_PublicKeystore>");
apClientConfiguration.setUseCompression(true);
apClientConfiguration.setMaxMessageSize(2097152);
apClientConfiguration.setRetryInterval(3600);
apClientConfiguration.setMaxRetries(24);
apClientConfiguration.setDefaultPullingInterval(10);
```

The configuration contains a list of InstitutionAliasMapping that encapsulates information about the local party and all parties that connects to the local party directly:

```
List<InstitutionAliasMappingInstitutionAliasMapping> InstitutionAliasMappingInstitutionAliasMappingList =
new ArrayList<>();
apClientConfiguration.setInstitutionAliasMappings(InstitutionAliasMappingInstitutionAliasMappingList);
```

In our case, the operation mode selected is OperationModeEnum.AP\_MODE, so the local party communicates only with the Access Point in order to send messages to all the other institutions involved in the case.

Below you can see how the InstitutionAliasMappingInstitutionAliasMapping looks like for the Access Point:

```
InstitutionAliasMappingInstitutionAliasMapping InstitutionAliasMappingInstitutionAliasMappingAp = new
oInstitutionAliasMappingInstitutionAliasMapping();
InstitutionAliasMappingInstitutionAliasMappingList.add(InstitutionAliasMappingInstitutionAliasMappingAp);
InstitutionAliasMappingInstitutionAliasMappingAp.setIsPrivateKeystore(false);
InstitutionAliasMappingInstitutionAliasMappingAp.setKeyAlias("beap1btk.cloudapp.net (eessi-ca)");
InstitutionAliasMappingInstitutionAliasMappingAp.setMpc("http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/defaultMPC/BE:BEAP1");
InstitutionAliasMappingInstitutionAliasMappingAp.setPassword("password");
InstitutionAliasMappingAp.setOrganisation(organisationAp);
InstitutionAliasMappingInstitutionAliasMappingAp.setTechnicalKeyAlias("beap1btk.cloudapp.net (eessi-ca)");
InstitutionAliasMappingInstitutionAliasMappingAp.setTechnicalPassword("password");
InstitutionAliasMappingInstitutionAliasMappingAp.setTechnicalMpc("http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/defaultMPC/BE:BEAP1:System");
```

Next we need to define the AccessPoint that the local NA communicates with:

```
AccessPoint accessPoint = new AccessPoint();
accessPoint.setId("BEAP1");
accessPoint.setName("Access Point for Belgium");
accessPoint.setCountryCode("BE");
accessPoint.setProtocol("https");
accessPoint.setAccessPointIp("beap1btk.cloudapp.net");
accessPoint.setAccessPointPort("443");
accessPoint.setOutboxService("/EESSI/BusinessMessaging/v1.0/Outbox/Service.svc");
```

```
accessPoint.setInboxService("/EESSI/BusinessMessaging/v1.0/Inbox/Service.svc");  
accessPoint.settechnicalOutboxService("/EESSI/SystemMessaging/v1.0/Outbox/Service.svc");  
accessPoint.settechnicalInboxService("/EESSI/SystemMessaging/v1.0/Inbox/Service.svc");
```

The AccessPoint is part of an Organisation, just like any institution:

```
Organisation organisationAp = new Organisation();  
organisationAp.setAccessPoint(accessPoint);  
organisationAp.setId("BE:BEAP1");  
organisationAp.setRegistryNumber("registry");  
organisationAp.setName("Access Point Organisation for Belgium");  
organisationAp.setAcronym("beap1");  
organisationAp.setCountryCode("BE");  
organisationAp.setActiveSince(new Date());
```

Below you can see how the InstitutionAliasMapping looks like based on the local Access Point and Organisation:

```
InstitutionAliasMapping InstitutionAliasMappingRina = new InstitutionAliasMapping();  
InstitutionAliasMappingList.add(InstitutionAliasMappingRina);  
InstitutionAliasMappingRina.setIsPrivateKeyStore(true);  
InstitutionAliasMappingRina.setKeyStoreAlias("bena08");  
InstitutionAliasMappingRina.setMpc("http://docs.oasis-open.org/ebxml-  
msg/ebms/v3.0/ns/core/200704/defaultMPC/BE:BENA08");  
InstitutionAliasMappingRina.setPassword("123456aA");  
InstitutionAliasMappingRina.setOrganisation(organisationRina);
```

```
AccessPoint accessPointLocalHolodeck = new AccessPoint();  
accessPointLocalHolodeck = new AccessPoint();  
accessPointLocalHolodeck.setId("HKNA08");  
accessPointLocalHolodeck.setName("Holodeck for BENA08 institution");  
accessPointLocalHolodeck.setCountryCode("BE");  
accessPointLocalHolodeck.setProtocol("https");  
accessPointLocalHolodeck.setAccessPointIp("be08.cloudapp.net");  
accessPointLocalHolodeck.setAccessPointPort("7070");  
accessPointLocalHolodeck.setOutboxService("/msh");  
accessPointLocalHolodeck.setInboxService("/msh");
```

```
Organisation organisationAp = new Organisation();  
organisationRina.setId("BE:BENA08");  
organisationRina.setRegistryNumber("registry");  
organisationRina.setName("BENA08 institution");  
organisationRina.setAcronym("BENA08");  
organisationRina.setCountryCode("BE");  
organisationRina.setActiveSince(new Date());  
organisationRina.setAccessPoint(accessPointLocalHolodeck);
```

### 8.1.2 How to create the listener implementation

In order to receive business messages and status updates about sent business messages the user must implement ApListener interface as in this simple logging implementation:

```
public class IntegrationApListener implements ApListener {  
  
    private final String organisationId;  
  
    public IntegrationApListener (final String organisationId) {  
        super();  
        this.organisationId = organisationId;  
    }  
  
    @Override  
    public String getName () {  
        logger.info("getName () - " + getName());  
        return "SampleListener";  
    }  
}
```

```
}

@Override
public void onMessageStatusUpdate(MessageStatus messageStatus) throws MessageProcessingException {
    logger.info("onMessageStatusUpdate() - " + messageStatus);
}

@Override
public void onMessageReceived(Message message) throws MessageProcessingException {
    logger.info("onMessageReceived() - " + message);
}

@Override
public void onMessageSignatureUpdate(MessageSignature messageSignature) throws
MessageProcessingException {
    logger.info("onMessageSignatureUpdate() - " + message);
}
}
```

### 8.1.3 How to setup the ApClient

Once there is an ApClientConfiguration built and an ApLISTER implementation available, the ApClient setup is straightforward:

First, take the ApClientImpl singleton instance, then set the callback and call setup with the configuration object:

```
ApClientManager apClientManager = ApClientManagerImpl.instance();
apClientManager.setupClients(config);

Collection<ApClient> clients = apClientManager.getClients();
for (ApClient client: clients) {
    String orgId = client.getOrganisationId();
    ApListener listener = new IntegrationApListener(orgId);
    ((ApClientAdmin) client).setListener(listener);
}
```

The library is ready to be used, the user can now send and receive messages.

### 8.1.4 How to stop the ApClient

When the ApClient is no longer user, it has to be stopped because there is a file watcher thread that looks for input files. If it is not stopped the thread may hang out.

Please use the stop() command so that the library can properly do its cleanup:

```
client.stop();
```

### 8.1.5 How to build a business message

There are two builders available that can be used for constructing a business Message: one that is based on the SBDH and one that builds a message from scratch:

#### 8.1.5.1 How to build a business message with SBDH

First build the standard business header using the SbdhBuilder:

```
StandardBusinessDocumentHeader sbdh = SbdhBuilder
    .standardBusinessDocumentHeader()
    .withSender("NO: 889640782", ContactTypeIdentifier.CASE_OWNER)
    .withAddedReceiver("AT: 9999", ContactTypeIdentifier.COUNTERPARTY)
    .withAddedReceiver("AT: 9998", ContactTypeIdentifier.COUNTERPARTY)
    .withDocumentIdentification("urn:eu:europa:ec:dgempl:ees si:sed", "1.0", "P2000", "3.2",
"75b21c33-22b3-4554-9707-5fa829ee8ab1", "3.2", "75b21c33-22b3-4554-9707-5fa829ee8ab1", new Date())
    .withAddedAttachment(InternetMediaType.APPLICATION_PDF, "75b21c33-22b3-4554-9707-5fa829ee8ab1")
    .withAddedAttachment(InternetMediaType.APPLICATION_PDF, "75b21c33-22b3-4554-9707-5fa829ee8ab2", "sed:-ClaimKind/individualClaim[1]", true)
    .withCaseIdentification("P_BUC_01", "3.2", CaseActionType.START, "75b21c33-22b3-4554-9707-5fa829ee8ab1", true, true)
```

```
.withRelatedDocumentSetIdentification("85b21c33-22b3-4554-9707-5fa829ee8ab1")  
.withBusinessKeystoreAlias("aliasInst3")  
.build();
```

Once the standard business header is built, the user can use the sbdh version of the message builder as below:

```
Message message = MessageBuilder  
.message()  
.withId("12345678-22b3-4554-9707-5fa829ee8ab1")  
.withActiveReceiver("AT:9998")  
.withBusinessDocumentHeader(sbdh)  
.withContentForSed(inputStreamSed, "75b21c33-22b3-4554-9707-5fa829ee8ab1")  
.withContentForAttachment(inputStreamAttachement1, "75b21c33-22b3-4554-9707-5fa829ee8ab1")  
.withContentForAttachment(inputStreamAttachement1, "75b21c33-22b3-4554-9707-5fa829ee8ab2")  
.build();
```

### 8.1.5.2 How to build a business message from scratch

Use this version of message builder to build the message in one shot:

```
Message message = MessageBuilder  
.message()  
.withId("12345678-22b3-4554-9707-5fa829ee8ab1")  
.withSender("NO:889640782", ContactTypeIdentifier.CASE_OWNER)  
.withAddedReceiver("AT:9999", ContactTypeIdentifier.COUNTERPARTY, true)  
.withAddedReceiver("AT:9998", ContactTypeIdentifier.COUNTERPARTY, false)  
.withDocumentIdentification("urn:eu:europa:ec:dgempl:eesi:sed", "1.0", "P2000", "3.2",  
"75b21c33-22b3-4554-9707-5fa829ee8ab1", "3.2", "75b21c33-22b3-4554-9707-5fa829ee8ab1", new Date(),  
inputStreamSed)  
.withAddedAttachment(InternetMediaType.APPLICATION_PDF, "75b21c33-22b3-4554-9707-  
5fa829ee8ab1", inputStreamAttachement1)  
.withAddedAttachment(InternetMediaType.APPLICATION_PDF, "75b21c33-22b3-4554-9707-  
5fa829ee8ab2", inputStreamAttachement2)  
.withCaseIdentification(("P_BUC_01", "3.2", CaseActionType.START, "75b21c33-22b3-4554-  
9707-5fa829ee8ab1")  
.withRelatedDocumentSetIdentification("85b21c33-22b3-4554-9707-5fa829ee8ab1")  
.build();
```

### 8.1.6 How to send a message

Once the client is configured and the business message is built, sending the message is as simple as:

```
final ApClientManager apClientManager = ApClientManagerImpl.instance();  
apClientManager.send(message);
```

## 8.2 RINA Business Messaging SOAP Web Service Interface

In previous chapter we described how we build the configuration Object `ApClientConfiguration`.

Also, all the fields from this object are described in the `ApClientConfiguration` object from the 'Data contract' chapter in the 'Configuration model' subchapter.

For `Ap Client Web Interface` we need the same configuration, but in a JSON file format.

In the next two section we provide sample of `apClientConfiguration.json` in AP mode.

For more details please refer to previous mentioned chapters.

### 8.2.1 How to create an AP JSON configuration

In this example we want to configure the institution with id: 'pb2ecna01' to work with an AP that has the id: 'pb2ecap01'.

In this case the selected operation mode is 'AP\_MODE' and we want to receive the messages from AP by pulling, this is why the 'usePullMode' is set to 'true'.

Holodeck is installed in 'mshPath' and there is no validation on the messages ('validationMode' is set to 'NO\_VALIDATION'). If we want to have validation enabled, we can switch to 'EXCEPTION\_NO\_TRANSACTION' and provide the location of the XSDs in 'validationPath' field.

The most important structure is the 'institutionAliasMappings' that contains information about the institutions and the access points it belongs to.

In this example the institution 'pb2ecna01' has the following details:

- The name is "NA 01 of EESSI"
- It is located in Belgium as the country code is "BE"
- The Message Partition Channel used for retrieving business messages is "<http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/defaultMPC/BE:pb2ecna01>" while for retrieving system messages is <http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/defaultMPC/BE:pb2ecna01:System>
- "isPrivateKeyStore" is very important as it shows whether this is the local institution or not. (Only the local institution has this field set to 'true', all the other it communicates with have this field set to 'false')

The institution Access Point has the following details:

- The id is "pb2ecap01" and its name "int-sync-ap"
- It is located in Belgium as the country code is "BE"
- The business URL for sending messages is composed as following: `<protocol>://<accessPointIp>:<accessPointPort>/<outboxService>`
- The business URL for receiving messages is composed as following: `<protocol>://<accessPointIp>:<accessPointPort>/<inboxService>`
- The system URL for sending messages is composed as following: `<technicalProtocol>://<accessPointIp>:<accessPointPort>/<technicalOutboxService>`
- The system URL for receiving messages is composed as following: `<technicalProtocol>://<accessPointIp>:<accessPointPort>/<technicalInboxService>`

Below is a sample configuration file : \EESSI\BMIWS\Tomcat\conf\apClientConf.json

```
{
  "mshPath": "C:\\EESSI\\Share\\",
  "msgInFolder": null,
  "privateKeystorePassword": "supervisor",
  "publicKeystorePassword": "supervisor",
  "businessKeystorePassword": "supervisor123",
  "businessKeyAliasPasswordList": [
    {
      "certificateAlias": "rona05business",
      "certificateAliasPassword": "businessPassword"
    }
  ],
  "operationMode": "AP_MODE",
  "useCompression": true,
  "useHttps": true,
  "useChunking": true,
  "maxMessageSize": 51200,
  "usePullMode": true,
  "maxRetries": 24,
  "retryInterval": 3600,
  "defaultPullingInterval": 30,
  "institutionAliasMappings": [
    {
      "organisation": {
        "id": "RO:rona05",
        "registryNumber": null,
        "name": "rona05",
        "acronym": "rona05",
        "countryCode": "RO",
        "activeSince": null,
        "accessPoint": {
          "id": "TESTRO",
          "name": "TESTRO",
          "countryCode": "RO",
          "protocol": "https",
          "technicalProtocol": "https",
          "accessPointIp": "EESSTESTRO.westeurope.cloudapp.azure.com",
          "accessPointPort": "443",
          "outboxService": "/eessi/BusinessMessaging/v1.0/Outbox/Service.svc",
          "technicalOutboxService": "/eessi/SystemMessaging/v1.0/Outbox/Service.svc",
          "inboxService": "/eessi/BusinessMessaging/v1.0/Inbox/Service.svc",
          "technicalInboxService": "/eessi/SystemMessaging/v1.0/Inbox/Service.svc"
        }
      },
      "mpc": "http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/defaultMPC/RO:rona05",
      "technicalMpc": "http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/defaultMPC/RO:rona05:SYSTEM",
      "pullingInterval": 30,
      "technicalPullingInterval": 30,
      "keyAlias": "rona05ebms",
      "defaultBusinessKeyAlias": "rona05business",
      "technicalKeyAlias": "rona05ebms",
      "isPrivateKeyStore": true,
      "password": "ebmsPassword",
      "technicalPassword": "ebmsPassword"
    }
  ],
  "clusterNodes": [
    {
      "name": "node1",
      "pmodePath": "C:\\EESSI\\Share\\conf\\pmodes\\"
    }
  ],
  "validationMode": "EXCEPTION_NO_TRANSACTION",
  "validationPath": "C:\\EESSI\\Share\\repository\\bpm\\",
  "useApplicationId": false,
  "applicationId": "",
  "antimalwareMode": "PERIODIC",
  "antimalwareSettings": "30",
}
```

## 8.2.2 How to create the listener implementation

In order to receive SOAP business messages and status updates about sent business messages the user must implement ApListener Web Service interface as in this simple logging implementation:

```
public class IntegrationApListenerWs implements ApListenerWs {

    @Override
    @PayloadRoot(namespace = Constants.SOAP_NAMESPACE, localPart = "updateMessageSignatureRequest")
    public void updateMessageSignature(@RequestPayload final UpdateMessageSignatureRequest
updateMessageSignatureRequest) {
        logger.info("MessageSignature received: " +
updateMessageSignatureRequest.getMessageSignature().getSedSignature());
    }

    @Override
    @PayloadRoot(namespace = Constants.SOAP_NAMESPACE, localPart = "updateMessageStatusRequest")
    public void updateMessageStatus(@RequestPayload final UpdateMessageStatusRequest
updateMessageStatusRequest) {
        logger.info("MessageStatus received: " + updateMessageStatusRequest.getMessageStatus().toString());
    }

    @Override
    @PayloadRoot(namespace = Constants.SOAP_NAMESPACE, localPart = "receiveSoapMessageRequest")
    @ResponsePayload
    public ReceiveSoapMessageResponse receiveMessage(@RequestPayload final ReceiveSoapMessageRequest
receiveSoapMessageRequest) {

        ReceiveSoapMessageResponse response = new ReceiveSoapMessageResponse();
        try {
            final SoapMessage soapMessage = receiveSoapMessageRequest.getSOAPMessage();
            logger.info("Message received: " + soapMessage.toString());
            final String id = soapMessage.getId();

            // Example of sending a semantic error
            if (StringUtils.isBlank(id)) {
                response = new ReceiveSoapMessageResponse(ReceiveSoapMessageResponseStatusEnum.ERROR, "Id
should NOT be blank!");
            }

            // Here the message should be handled...

        } catch (Exception e) {
            response = new ReceiveSoapMessageResponse(ReceiveSoapMessageResponseStatusEnum.ERROR,
e.getCause().getMessage());
        }

        return response;
    }
}
```

### 8.2.3 How to build a SOAP business message

There are two builders available that can be used for constructing a business Message: one that is based on the SBDH and one that builds a message from scratch:

#### 8.2.3.1 How to build a SOAP business message with SBDH

First build the standard business header using the SbdhBuilder:

```
StandardBusinessDocumentHeader sbdh = SbdhBuilder
    .standardBusinessDocumentHeader()
    .withSender("NO:889640782", ContactTypeIdentifier.CASE_OWNER)
    .withAddedReceiver("AT:9999", ContactTypeIdentifier.COUNTERPARTY)
    .withAddedReceiver("AT:9998", ContactTypeIdentifier.COUNTERPARTY)
    .withDocumentIdentification("urn:eu:europa:ec:dgempl:eesi:sed", "1.0", "P2000", "3.2",
"75b21c33-22b3-4554-9707-5fa829ee8ab1", "3.2", "75b21c33-22b3-4554-9707-5fa829ee8ab1", new Date())
    .withAddedAttachment(InternetMediaType.APPLICATION_PDF, "75b21c33-22b3-4554-9707-
5fa829ee8ab1")
    .withAddedAttachment(InternetMediaType.APPLICATION_PDF, "75b21c33-22b3-4554-9707-
5fa829ee8ab2", "sed:-ClaimKind/individualClaim[1]", true)
    .withCaseIdentification("P_BUC_01", "3.2", CaseActionType.START, "75b21c33-22b3-4554-
9707-5fa829ee8ab1", true, true)
    .withRelatedDocumentSetIdentification("85b21c33-22b3-4554-9707-5fa829ee8ab1")
    .withBusinessKeystoreAlias("aliasinst3")
    .build();
```

Once the standard business header is built, the user can use the sbdh version of the message builder as below:

```
SoapMessage soapMessage = SoapMessageBuilder
    .message()
    .withId("12345678-22b3-4554-9707-5fa829ee8ab1")
    .withActiveReceiver("AT:9998")
    .withBusinessDocumentHeader(sbdh)
    .withContentForSed(sedContentUrl, "75b21c33-22b3-4554-9707-5fa829ee8ab1")
    .withContentForAttachment(attachmentContentUrl1, "75b21c33-22b3-4554-9707-5fa829ee8ab1")
    .withContentForAttachment(attachmentContentUrl2, "75b21c33-22b3-4554-9707-5fa829ee8ab2")
    .build();
```

#### 8.2.3.2 How to build a business message from scratch

Use this version of message builder to build the message in one shot:

```
SoapMessage soapMessage = SoapMessageBuilder
    .message()
    .withId("12345678-22b3-4554-9707-5fa829ee8ab1")
    .withSender("NO:889640782", ContactTypeIdentifier.CASE_OWNER)
    .withAddedReceiver("AT:9999", ContactTypeIdentifier.COUNTERPARTY, true)
    .withAddedReceiver("AT:9998", ContactTypeIdentifier.COUNTERPARTY, false)
    .withDocumentIdentification("urn:eu:europa:ec:dgempl:eesi:sed", "1.0", "P2000", "3.2",
"75b21c33-22b3-4554-9707-5fa829ee8ab1", "3.2", "75b21c33-22b3-4554-9707-5fa829ee8ab1", new
Date(), sedContentUrl)
    .withAddedAttachment(InternetMediaType.APPLICATION_PDF, "75b21c33-22b3-4554-9707-
5fa829ee8ab1", attachmentContentUrl1)
    .withAddedAttachment(InternetMediaType.APPLICATION_PDF, "75b21c33-22b3-4554-9707-
5fa829ee8ab2", attachmentContentUrl2)
    .withCaseIdentification(("P_BUC_01", "3.2", CaseActionType.START, "75b21c33-22b3-4554-
9707-5fa829ee8ab1")
    .withRelatedDocumentSetIdentification("85b21c33-22b3-4554-9707-5fa829ee8ab1")
    .build();
```

### 8.2.4 How to send a message

Once the client is configured, the web service is published and the SOAP business message is built, in order to send the message you need

- To create a SendSoapMessageRequest

```
SendSoapMessageRequest sendSoapMessageRequest = new SendSoapMessageRequest();  
sendSoapMessageRequest.setSOAPMessage(createMessage(UUID.randomUUID().toString()));
```

- Create a web services proxy/client instance to the ApClient WSDL

```
client = new ApClientWsClient("http://localhost:8080/apclientws/ws");
```

- And the send:

```
apClientWs.sendMessage(sendSoapMessageRequest);
```

## 8.3 RINA Technical Messaging Interface

### 8.3.1 How to build the Holodeck PMode for sending messages to AP

```
PMode pushInitPmode = PModeBuilder
    .pMode()
        .withId("pmode-BENA08-BEAP1-push-init")
        .withPush()
        .withInitiator("BE:BENA08")
        .withInitiatorSignature("keyStoreAlias", "keyStorePassword")

    .withOutboxUrl("https://beap1btk.cloudapp.net:443/EESSI/Messaging/v0.4.1/Outbox/Service.svc")
        .withMaxRetries(3)
        .withRetryInterval(30)
        .withCompression(true)
    .build();
```

### 8.3.2 How to build the Holodeck PMode for retrieving messages from AP

```
PMode pullInitPmode = PModeBuilder
    .pMode()
        .withId("pmode-BENA08-BEAP1-pull-init")
        .withPull()
        .withInitiator("BE:BENA08")
        .withInitiatorSignature("keyStoreAlias", "keyStorePassword")
        .withOutboxUrl("https://beap1btk.cloudapp.net:443/EESSI/Messaging/v0.4.1/Inbox/Service.svc")

    .withReplyPatternCallback("https://beap1btk.cloudapp.net:443/EESSI/Messaging/v0.4.1/Outbox/Service.svc")
        .withPullInitiatorMpc("http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/defaultMPC/BE:BENA08")
    .build();
```

### 8.3.3 How to build a Holodeck message for sending

```
Pulling pulling = PullConfigurationBuilder
    .pullConfiguration()
        .withDefaultPulling(Big Integer.valueOf(30))
    .build();
```

### 8.3.4 How to write & read MSH messages

Once the technical objects are built, they can be serialized/desterialized using any XML binding framework into the appropriate Holodeck folders.

Here is a table that summarized all the important information regarding the technical messages:

| Technical message type | Holodeck default location | Extension | Read/Write |
|------------------------|---------------------------|-----------|------------|
| Pmode                  | Conf/pmodes               | .xml      | Write      |
| Pulling configuration  | Conf/                     | .xml      | Write      |
| Send message           | Data/msg_out              | .mmd      | Write      |
| Receive message        | Data/msg_in/<TENANT ID>/  | .mmd      | Read       |
| Receive status update  | Data/msg_in/<TENANT ID>/  | .smd      | Read       |

## 9 APPENDIX

### 9.1 ApClient DB solution

#### 9.1.1 Introduction

The ApClient has been modified in order to incorporate a persistence layer since it consists of a core part of BMS module in either of the two supported deployment scenarios:

- Full stack RINA and
- BMI-WS deployment

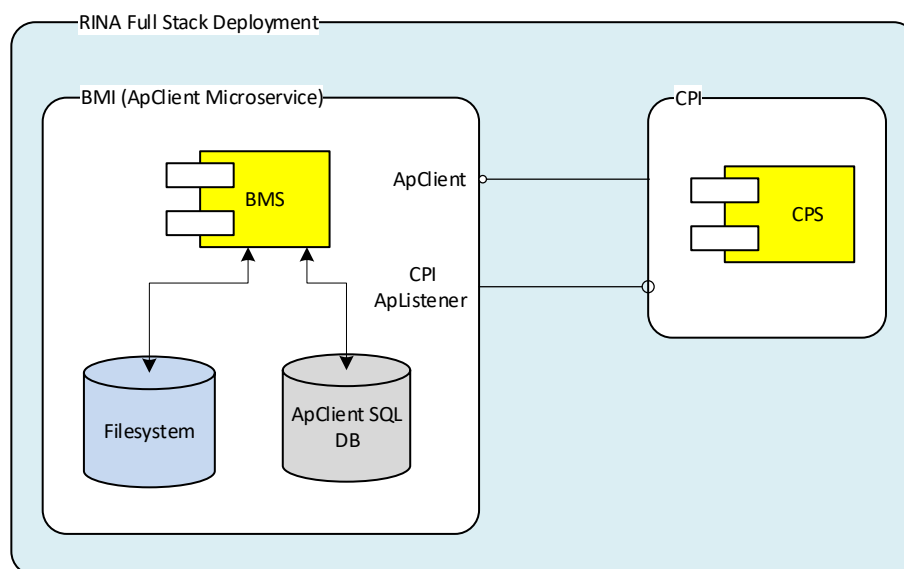
This enhancement stems from a pertinent observation corresponding to detected failures of processing incoming messages under particular conditions.

Specifically, any error during the processing of an incoming message in BMI 'clients' (like Case Processing Services (CPS) or National Applications with access to BMI-WS) could never be recovered and the message is lost. Furthermore, other errors arose when the Holodeck messages were delivered twice to the BMI Listeners.

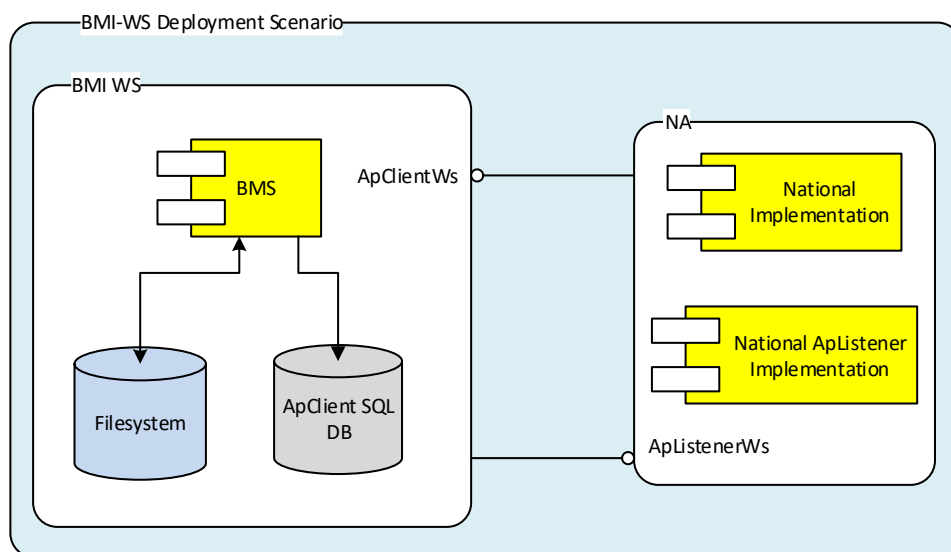
Possible cause of these situations was the fact that there was no way for the CPI Services to re-process messages which had already been processed. This was mainly due to lack of persistence layer for storing the incoming message status (Referenced also as Dequeue Registry in the document above)

#### 9.1.2 BMI deployment Scenarios with DB

The two diagrams below depict, with a considerable level of abstraction, the fact that the DB is now an inseparable part of the BMI module in either of the two proposed BMI deployment scenarios.



In the above diagram one can observe (albeit coarsely) the fact that the ApClient (micro) service now incorporates the DB layer. So, whenever a new incoming message is received, the filesystem event that is triggered results in inserting this information in the DB. Following the execution of a synchronous call to the ApListener (which calls BUC Engine), the outcome of that communication defines the status of the message.



In the second diagram (BMI-WS Deployment Scenario) the logic is similar. The main difference is that now the interfaces between the National (back-end) Implementations and the BMI module are defined through a WSDL contract. Furthermore, the implementation of the ApListener in this case is part of the National Implementation (hence not included inside a provided business processing engine, i.e. Bonita for Full-stack deployment option).

### 9.1.3 Database Solution Description

The incorporated persistence layer is based on a simple DAO mechanism which is responsible for storing the message status (along with other message properties). The DAO is using the EBMS message Id as a reference for each message. The DAO stores the data into a new DB table (**message**).

Each time a new incoming message arrives, a new record (row) is inserted in that table containing fields associated to the EBMS *message ID*, *status* and *timestamp* (related) information.

At this stage, two states (field *STATUS*) have been defined that can take the following values:

- **DELIVERING** (1): The message is being processed by the ApClient but has not been successfully delivered to the ApListener as of yet.
- **DELIVERED** (2): The message has been already successfully delivered by ApClient to the ApListener.

In conjunction with the above, the following fallback mechanism has been implemented in ApClient component:

- Messages in **DELIVERING** status remain in the physical message folder.
- In the current implementation, any ApClient Service restart will result to the mechanism trying to handle all the messages in **DELIVERING** status, for a longer than 5 minute (as default value) period of time. In such cases, the ApClient tries to process such messages by delivering them again to the listener.
- If delivered successfully, ApClient will update their state in the DB table to **DELIVERED**.
- If NOT, the status remains in **DELIVERING** state. This implies that the ApClient will attempt to handle it again on next service restart.

A second table (**parameter**) has also been introduced holding system parameters. In particular, one of the parameters stored in this table is the one with key '**ap\_client\_configuration**'. The *value* of that parameter holds the JSON configuration needed for the initialization of the ApClient.

In the case of BMI (as internal component of RINA fullstack deployment) the *value* is re-populated each time the Pmode configuration is modified using the GUI of RINA (via the Admin Portal), whereas in the case of BMI-WS component it simply contains a copy of the *ApClient.conf* file. The latter is because every time the service is restarted it is simply repopulated from the ApClient.conf JSON file, thus the value of the parameter does not affect the operation of the service.

However, this is not the case for fullstack RINA. It must be accentuated that this modification was imperative since the service has been redesigned to work in a 'distributed' mode. The ApClient is currently deployed separately and for this reason it needs to store its configuration in a private storing location in order to be able to initialise its components in case of failure.

Storing the configuration also allows for the automatic reconfiguration of multiple nodes of ApClient by taking advantage of the presence of the *version* field. This is achieved by increasing the version by 1, each time the configuration is modified. In addition, each node of the cluster holds a local variable for the version stored in local memory. Periodically each node compares the version stored in the DB with the one in memory and, if the DB value corresponds to a bigger number, it reloads the configuration from the DB. This mechanism is used for synchronizing the configuration of all nodes in the cluster. Note that, in the case of clustering, only one node (picked randomly by the Apache proxy) will be informed about the modification in the messaging configuration.

A third table (**dead\_queue**) has also been added. This table acts as a FIFO stack of messages for which an error has been received by the ApListener. This table includes both outgoing and incoming messages. In case of such failure the message is placed at the bottom of the stack. A process running at regular intervals is responsible for handling all the failing messages. This process pops the top message from the stack and resends it to the ApListener. If the failure takes place again then the message is pushed back to the bottom of the stack.

### 9.1.3.1 Database schema description

The new ApClient DB schema contains the following two tables:

| message    |             |                          |          |               |
|------------|-------------|--------------------------|----------|---------------|
| NAME       | PROPERTY    | TYPE                     | NULLABLE | DEFAULT VALUE |
| ID         | Primary Key | VARCHAR(255)             | Not Null |               |
| STATUS     |             | INT8                     | Not Null |               |
| FILENAME   |             | VARCHAR(255)             | Not Null |               |
| CREATED_AT |             | TIMESTAMP WITH TIME ZONE | Not Null | now()         |
| UPDATED_AT |             | TIMESTAMP WITH TIME ZONE | Not Null | now()         |

| Parameter |             |              |          |               |
|-----------|-------------|--------------|----------|---------------|
| NAME      | PROPERTY    | TYPE         | NULLABLE | DEFAULT VALUE |
| KEY       | Primary Key | VARCHAR(255) | Not Null |               |
| VALUE     |             | TEXT         | Not Null |               |
| VERSION   |             | INT4         | Null     | 0             |

| dead_queue |             |              |          |               |
|------------|-------------|--------------|----------|---------------|
| NAME       | PROPERTY    | TYPE         | NULLABLE | DEFAULT VALUE |
| ID         | Primary Key | VARCHAR(255) | Not Null |               |
| DIRECTION  | Primary Key | VARCHAR(3)   | Not Null |               |
| MSG_TYPE   |             | VARCHAR(30)  | Not Null |               |
| MSG        |             | TEXT         | Not Null |               |

|               |  |                             |          |       |
|---------------|--|-----------------------------|----------|-------|
| <b>TSTAMP</b> |  | TIMESTAMP WITH TIME<br>ZONE | Not Null | now() |
|---------------|--|-----------------------------|----------|-------|

The usage of the **message** table has already been explained in the previous section. It suffices to notice that it is important for storing the message status **and** the physical location where the message is actually stored.

The parameter table is used for storing system-wide parameters as explained also in the previous section.

The usage of the **dead\_queue** table has also been explained. What is worth noticing is that the field tstamp is used for ordering the messages in the stack.