



EESSI – RINA

Architecture Overview

Architecture & Design Specifications

Table of Contents

1. Introduction.....	7
1.1 Scope.....	7
1.2 Structure of the document.....	7
1.3 Reference and Applicable documents	8
1.4 Audience	10
1.5 Definitions, Acronyms and Abbreviations	10
2. EESSI and RINA: Core Functionalities	11
2.1 EESSI Application Domains and Functions	11
2.2 RINA Core Functionalities	13
3. RINA Layers and Core Interfaces	16
3.1 RINA Portal.....	17
3.2 RINA Case Processing Services (CPS)	17
3.3 RINA Business Messaging Services (BMS)	18
3.4 RINA Technical Messaging Services (TMS).....	18
3.5 RINA CAS.....	19
4. RINA Logical Components View	20
4.1 RINA PORTAL.....	21
4.1.1 RINA Clerk Consoles.....	21
4.1.2 RINA Administration Consoles	22
4.1.3 RINA Shared Consoles.....	28
4.1.4 Proxies to external Services.....	29
4.2 RINA Case Processing Services (CPS)	30
4.2.1 RINA Case Processing Interface (CPI).....	30
4.2.2 RINA National Systems APIs	31
4.2.3 RINA Business Services	32
4.2.4 RINA Data Access	34
4.2.5 CPS Repository.....	35
4.3 RINA Business Messaging Services (BMS)	37
4.3.1 RINA Business Messaging Interface (BMI)	37
4.3.2 RINA National Systems Services	38
4.3.3 RINA Business Message Processing Services	39
4.3.4 BMS Repository.....	39
4.3.5 RINA Technical Messaging Services (TMS)	40
4.3.6 RINA Technical Messaging Interface (TMI).....	41
4.4 RINA Foundation Services.....	42
4.4.1 Localisation	42
4.4.2 RINA Authentication Services (CAS Server)	42
4.4.3 RINA Logging Services	43
4.4.4 RINA Log Repository.....	43
5. RINA Repositories.....	44
5.1 Elastic Search (ES)	44
5.2 File System.....	44
5.3 RINA SQL Database	45
5.4 ApClient(BMS) DB.....	45
5.5 Holodeck DB	46
5.6 RINA Logical Repos mapping	46
6. RINA Case Processing Services Behaviour	47
6.1 CPS: Create Case	47
6.2 CPS: Create and Send Document.....	50
6.3 CPS: Receive Message	53
7. RINA Audit Trail	56
7.1 Audit Trail principles.....	56
7.2 Audit Trail architecture.....	56

7.2.1	CPS and Audit Trail.....	57
7.2.2	BMS and Audit Trail.....	57
7.3	Audit Trail Data Model	58
7.3.1	Audit Event.....	59
7.3.2	Audited Object.....	59
7.3.3	Event Source	60
7.3.4	Network Location.....	60
7.3.5	Participant	60
7.3.6	EActionType enumeration values and description.....	61
7.3.7	EAuditedObjectType enumeration.....	61
7.3.8	ECategoryType enumeration values and description.....	63
7.3.9	EComponentType enumeration values and description.....	63
7.3.10	EEventType values and description	64
7.3.11	EOutcome Type enumeration values and description	72
7.3.12	EParticipantRole enumeration values and description.....	72
7.3.13	EParticipantType enumeration values and description	72
7.3.14	Audit events types and audited Objects	73
7.4	Audit model CS DB Domain representation.....	75
7.5	NA and Audit Trail.....	76
8.	RINA Technical Logging.....	77
8.1	Logging architecture (default).....	77
8.2	Log Records context and content.....	78
8.3	Logging levels.....	79
8.4	RINA log files	80
ANNEX A.	Technical Logging and Exception Management Principles in RINA	83
ANNEX B.	RINA default logging format	85

Table of Figures

Figure 1: EESSI Core Functionalites & Application Domains	11
Figure 2: RINA Layers and Core Interfaces.....	16
Figure 3: RINA Logical Components View	20
Figure 4: RINA Portal Logical Components	21
Figure 5: RINA CPS Logical Components.....	30
Figure 6: RINA BMS Logical Components	37
Figure 7: RINA Foundation Services Logical Components	42
Figure 8: Create Case Sequence Diagram	49
Figure 9: Create and Send Document Sequence Diagram.....	52
Figure 10: Receive Message Sequence Diagram.....	55
Figure 11: RINA Audit Trail architecture	57
Figure 12: RINA Audit Data Model.....	58
Figure 13: RINA Audit Model CS DB Domain Representation	76
Figure 14: RINA Logging Architecture	78

Table of Tables

Table 1: Reference documents list	10
Table 2: Definitions and Acronyms list	10
Table 3: RINA core functionalities	15
Table 4: RINA Logical and Physical Repositories	46
Table 5: RINA Logging Levels	80
Table 6: RINA Log Files (with component and location mapping).....	82

Document Control Information

Document Control	Value
Project Title	Electronic Exchange of Social Security Information (EESSI)
Document Name	EESSI – RINA - Architecture Overview
Document Category	Architecture & Design Specifications
Revision	-
Component Version	-
Last Publication Date Project Milestone	22/12/2020 EESSI-2020
Document Status	Final
Sensitivity (TLP) Distribution terms	Traffic Light Protocol (TLP) = "GREEN" 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. 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.
Connected/Embedded Files	None
Authors	European Commission, DG EMPL F5, EESSI ARCH / RINA
Revised by	European Commission, DG EMPL F5, EESSI QA/QC
Approved by	European Commission, DG EMPL F5, EESSI PMO team

Document history

Project Milestone	Changes/Corrections Description
EESSI-2020	Initial Document

1. Introduction

The purpose of this document is to provide a structural overview and a high level description of the EESSI Reference Implementation for National Application (RINA) architecture.

This document should be used by the different stakeholders of EESSI project that need to comprehend the high-level picture of RINA architecture. Additional details are available for each architecture layer and interface in the related specification documents, as referred in this document.

1.1 Scope

This document focuses on providing mostly a functional/logical description of RINA architecture.

It follows a top-down approach, introducing the reader in the main functionalities to be provided by the EESSI Ecosystem and framing RINA into it. Then, it describes RINA from a blackbox viewpoint, depicting its layers and core interfaces.

The core of the whole document is a detailed presentation on the RINA logical components from a whitebox perspective. These components are then framed into a behavioural view, describing the most relevant scenarios related with Case, Document and Message management.

In addition, it provides with a high-level view on RINAs repositories, audit and technical logging architecture.

This document does not describe the RINA technical design. Information on this domain it's provided by other artefacts such as the RINA BUC Engine Architecture, CPI, BMI and NIE interfaces and the SQL Database Schema (all included in the list of referenced documents).

1.2 Structure of the document

The present document is divided in the following sections:

- Section 1: Introduction, providing the reader with the purpose and scope of the document, as well as the list of referenced documents and acronyms list.
- Section 2: EESSI and RINA Core Functionalities. The aim of this section is to illustrate the reader with a high-level view on the core functionalities to be delivered with the EESSI ecosystem and how RINA fits into them. This section content will act as reference on the main RINA functionalities, framing the next 2 sections in the RINA Layers and Logical components.
- Section 3: RINA Layers and Core Interfaces. It describes the RINA four main layers as well as its core interfaces (both public – CPI, NIE, BMI WS- and private –BMI MS and TMI-)
- Section 4: RINA Logical Components View. Core of this document, this section provides the reader with the full architecture, dependencies as well as description of all the RINA logical components.

- Section 5: RINA Repositories, contains a high-level view of all the physical repositories that RINA has, providing with a table mapping logical and physical repositories.
- Section 6: RINA Case Processing Services Behaviour, provides the reader with the sequence diagrams depicting the behaviour of the most relevant RINA scenarios: Create Case, Create and Send Document and Receive a Message.
- Section 7: RINA Audit Trail. This section contains the RINA audit architecture, together with the full description of its logical data model and dependencies.
- Section 8: RINA Technical Logging, provides the reader with the RINA logging architecture as well as the logging levels and a list of all files created by RINA.

1.3 Reference and Applicable documents

#	Artefact/Document	Location
R1	EESSI Glossary	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/EESSI+Glossary
R2	GTM requirements 2013 11 06	https://citnet.tech.ec.europa.eu/CITnet/confluence/download/attachments/865090523/GTM%20requirements%202013%2011%2006.xlsx?version=1&modificationDate=1549464522007&api=v2
R3	EESSI - RINA - Functional Specs - Administration Portal	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Architecture+Documentation
R4	EESSI - RINA - Functional Specs - Case Management	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Architecture+Documentation
R5	EESSI - RINA - Identity and Access Management (IAM)	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Architecture+Documentation
R6	EESSI - RINA - Archiving Specifications	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Architecture+Documentation
R7	EESSI - RINA - Security Specs	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Architecture+Documentation

R8	EESSI - RINA - Case Processing Interface (CPI)	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA+Architecture+Documentation
R9	EESSI - RINA - Localisation Specifications	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA+Architecture+Documentation
R10	EESSI - RINA - National Information Exchange Interface (NIE)	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA+Architecture+Documentation
R11	EESSI - RINA - Business Messaging Interface (BMI)	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA+Architecture+Documentation
R12	EESSI - RINA - Case Processing Interface (CPI) - Reference.pdf	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA+Architecture+Documentation
R13	EESSI - RINA - Business Use Case Engine Architecture	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA+Architecture+Documentation
R14	EESSI - RINA - BUC Technical Specifications	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA+Architecture+Documentation
R15	EESSI - RINA - User Manual	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Operations+Documentation
R16	EESSI - RINA - Administration Manual	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Operations+Documentation
R17	EESSI - RINA - Operations Manual	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Operations+Documentation
R18	EESSI - RINA - Operations - Security Guide	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Operations+Documentation
R19	EESSI - Architecture Overview	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/EESSI+System+General+Architecture+Documentation

R20	EESSI – Component Architecture	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/EESSI+System+General+Architecture+Documentation
R21	EESSI – AS4 Messaging Profile	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/EESSI+System+General+Architecture+Documentation
R22	EESSI 2020 - RINA 6.2.1 - SQL DB Schema - Physical Data Model	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Architecture+Documentation
R23	EESSI 2020 - RINA 6.2.1 - SQL DB Schema - Figure.gif	https://citnet.tech.ec.europa.eu/CITnet/confluence/display/EESSI/RINA++Architecture+Documentation
R24	Holodeck B2B documentation	http://holodeck-b2b.org/documentation/
R25	eUI Framework documentation	https://webgate.ec.europa.eu/fpfis/wikis/spaces/viewspace.action?key=eUI

Table 1: Reference documents list

1.4 Audience

This document targets the RINA solution and technical architect as well as developers.

1.5 Definitions, Acronyms and Abbreviations

Please see the "EESSI Business and Technical Glossary [R1]".

In addition, please find below some RINA related acronyms used in this document.

Abbreviation	Full Form
APClient	BMS core technical component
BMI	Business Messaging Interface
BMI MS	Business Messaging Interface – Micro Service
BMI WS	Business Messaging Interface – Web Service
BMS	Business Messaging Services
CAS	Central Authentication Server
CPI	Case Processing Interface
CPIAPListener	Registered Listener of BMI MS to publish messages to CPS
CPS	Case Processing Services
Holodeck B2B	TMS core technical component
NIE	National Information Exchange
REST (CPS)	CPI Interface Technical nature

Table 2: Definitions and Acronyms list

2. EESSI and RINA: Core Functionalities

This section shows a high level view on the core functionalities that the EESSI ecosystem provide, as well as its applications, defining the role that RINA plays within it. Then it lists the summary of all RINA functionalities, framing - and providing a rationale to- the architectural layers and logical components that conforms it.

2.1 EESSI Application Domains and Functions

The next diagram illustrates the how the core functionalities of EESSI span across its application domains.

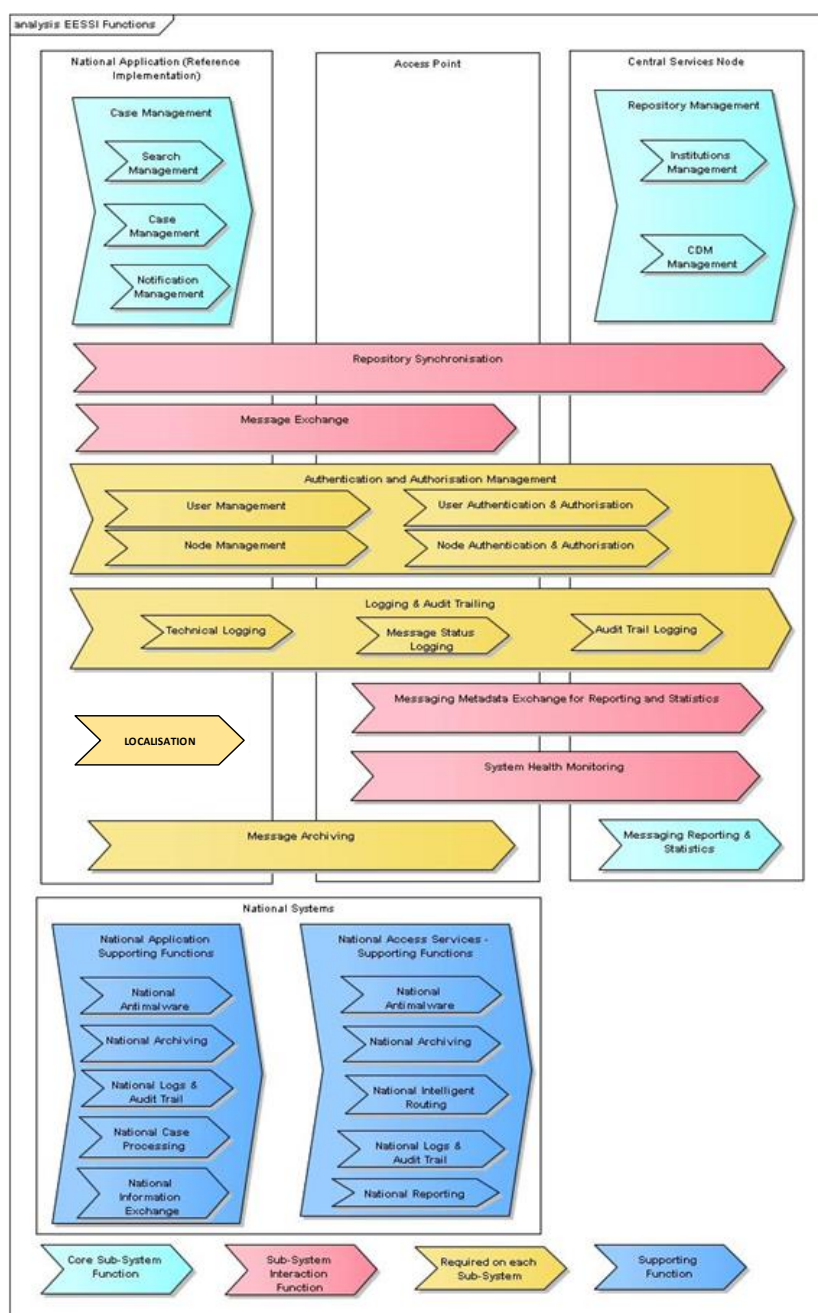


Figure 1: EESSI Core Functionalites & Application Domains

The EESSI application *domains* are basically the three following ones:

- *Central Service Node*;
- *Access Points*;
- *National system implementations* associated to any of the following options:
 - The Reference Implementation for a National Application (RINA) – application based on open source software and using a modular design (reusable components for national systems);
 - National Applications (NA) – applications used by clerks to prepare, send and receive messages according to the agreed business use cases;
 - National Gateways (NG) – optional system that could be used by MS as "gateways" between NAs (or RINA) and AP in order to perform functions as:
 - transformation between national format of the messages to international format;
 - bridging between national communication protocols and the international communication protocols;
 - intelligent routing etc.

RINA, plays a key role within EESSI's national domain, being in charge perform the following main functions, which are further described in the next section.

- Message Exchange;
- Repository Synchronisation;
- Authentication and Authorisation;
- Logging and Audit Trail;
- Case Management;
- Localisation;
- Message Archiving;

These functions will provide for clerks and their organizations the possibility to perform the agreed BUCs in Social Security across Europe.

2.2 RINA Core Functionalities

The following table captures a high-level description of the RINA core functionalities¹. The present document, RINA Architecture Overview, guide the reader on how these set of functionalities have been designed and structured within the RINA logical architecture layers and components.

Function Name	Description
Message Exchange	The Message Exchange function is the top level function of the EESSI system. It describes the sum of actions that allow National Applications, in this case RINA, to send and receive business messages according to the EESSI data exchange contracts, in a secure and reliable way. The function covers the whole chain of message exchange from the sending National Application, through the Access Points and to the receiving National Application(s). The function is split into 3 main functionalities: sending, receiving and processing of messages. The message exchange function provides a way for the users to send and receive business messages transparently over the EESSI system using the ebMS AS4 profile.
Repository Synchronisation	The Repository Synchronisation function provides the functionality to synchronize data across Central Services Node, Access Point and National Application. The Repository Synchronisation uses the Message Exchange function to exchange data between the Central Service Node Access Points and between the Access Points and National Applications. The Repository Synchronisation could be triggered automatically based on preconfigured parameters or manually by system administrators. It is used to transfer the repository data from the Central Services Node in case of the Access Point and from the Access Point in case of a National Application or, in this case, RINA.
Authentication and Authorisation Management	The Authentication function is responsible for granting permissions to the application users to access it. The Authorisation function is in charge with persisting the policies and the composition of operations and roles at the different levels. It answers the following questions with regards to Authorisation Enforcement and Identity & Authorisation Management at the Institutions/RINA level: • How to manage the Clerks and their authorisations at the Institutions/RINA level? • How to manage the system and/or admin users and their authorisations at each EESSI node level? • How to manage the secure communication between the EESSI nodes (CSN, AP, RINA/NA/NG)? In order to be able to enforce authorisation for Clerks at the Institutions/RINA on a need to know basis, the Institutions/RINA must also be able to identify and manage human or system users and their authorisations. Hence every National Application/RINA will provide the following services:

¹ The description of the complete set of EESSI functionalities is offered in "EESSI - Architecture Overview" document [R19]

	• Register/Manage human actors and their privileges (Roles). This covers the user and role management, and will be handled by the so-called User Authorisation Management function; • Identify & Authenticate (login) and Authorise requests of user actors (credential management). It will be handled by the so-called User Authentication Management function. This function also covers the RINA user authentication and authorisation mechanisms for system administrator's tasks and system operations.
Logging & Audit Trailing	This function allows the system to capture and manage information about message exchanges and other business related actions performed by the users (or the system whenever applicable) into functional logs (only captures events which may have legal implications) together with technical logs at an application level. Logs are the recordings of one or more events occurring on the information sub-system. Depending on the type of information captured logs may be referred to as log files, or audit trails. Logs also cover alerts, alarms and event records. Information captured and maintained by this function is used in debugging and forensics, fault monitoring, performance monitoring, trouble shooting, feature usage, security & incident detection and regulatory and standards compliance.
Case Processing	The Case Processing function provides capabilities on: • Search Management - to find cases and/or Social Security Institutions based on sets of predefined or ad-hoc criteria. • Case Processing - covers all aspects of case management, including creation of cases, creation of documents, management of attachments, management of participant lists, sending of documents, running business process specific actions, listing of cases, documents, attachments or actions and configuration of case manager. This function allows the clerk to track all types of business messages, streamline information entry with auto-population of forms based on the case data collected and tracked in the system, respond to regulatory and corporate body inquiries on the spot by having case details available. • Notification Management - provides case specific notifications and alerts in a citizen/institution centric approach and multiple views for visualizing a single case. Every time an event condition is fulfilled, RINA will have to notify the assigned clerks or supervisors of the case. The notifications' events have associated a type that could be error, warning or information.
Localisation	Localisation plays an important role in defining specific presentation behaviour depending on the place where the EESSI components are deployed (i.e. RINA). EESSI RINA localisation consists in adapting the system to the languages, number formatting, date and time formats, currencies, regional differences and other non-functional requirements of the Member States. The scope of EESSI Localisation Specifications describes the localisation of the following components: • RINA Portal; • RINA Case Processing Service (CPS) • SED forms physical artefacts generated from EESSI SED Logical Data Model - (used by RINA and optionally by NAs).
Message Archiving	The Message Archiving function deals mainly with the configuration of the archiving of the cases and the messages residing in RINA. The purpose is to give the opportunity of automatically archiving in a configurable manner the business message content (closed cases or messages) which

	is no longer necessary in order to store it as future evidence. This is applicable to both cases and messages.
National Application Supporting Functions	These are group of elements provided as options of National systems integrating with RINA to utilise associating functionalities: National Antimalware, National Archiving, National Log and Audit Trail, National Case Processing, National Information Exchange (NIE)

Table 3: RINA core functionalities

3. RINA Layers and Core Interfaces

Part of the National Institutions Domain, RINA consists in a collection of technical and business services which will provide clerks and their organisations the tools to participate into EESSI ecosystem. It is based on 4 layers which are built one on top of the other.

The next diagram shows RINA 4 main layers: Portal (GUI), Case Processing Services (CPS), Business Messaging Services (BMS), Technical Messaging Services (TMS); together with its Central Authentication Service module (CAS).

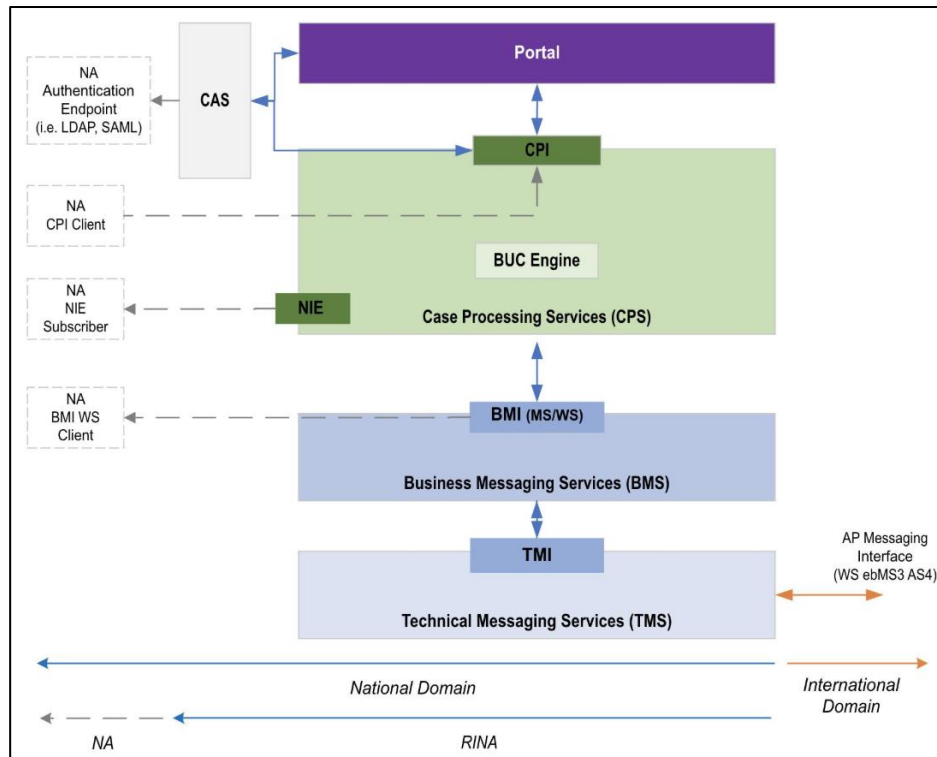


Figure 2: RINA Layers and Core Interfaces

RINA is a modular system and offers different deployment options, based on the needs:

1. RINA (full stack) – includes all of the layers and provides a full-scale solution for national application in the context of EESSI (i.e. UI, stateful case processing services, BMS and TMI). Programmatic integration available options via public interfaces of: CPI, NIE & CAS.
2. RINA BMI WS – removes the stateful case processing layer (CPS/Portal) and offers only the stateless BMS (and included TMS providing the communication within ebMS AS4 protocol) layer(s).

Following sections provide the reader with a high-level description of each them, together with their core interfaces, following a black-box approach.

3.1 RINA Portal

RINA Portal provides the frontend layer of the whole RINA Environment, i.e. the Graphical User Interface (GUI). It does not contain any business logic; however it provides validations and it retrieves and submits data to the backend.

The RINA Portal utilizes the RINA Case Processing Services through the CPI interface and CAS through the CAS protocol, which is used to authenticate users before accessing the portal.

3.2 RINA Case Processing Services (CPS)

RINA Case Processing Services provide the backend layer of the RINA Environment. It includes business logic, which encapsulates the complete stateful case processing functionalities as well as configuration of the RINA instances. It integrates the bellow layers to utilize the connection to the AP through the ebMS3.0 - AS4 protocol.

RINA CPS includes a collection of public interfaces, which are used by either RINA Portal or are integrated within the National Systems. The stateful case processing functionality is leveraged by the integration of the BUC-Engine. RINA CPS layer provides the transactional context for operations. Depending on the particular operation, it encapsulates them within a transaction and handles any possible concurrent modification issues. For more detailed information please refer to "*EESSI - RINA - Case Processing Interface (CPI) [R8]*" document.

The public interfaces provided by this layer include:

- Case Processing Interface (CPI) – exposes the case processing operations through a set of RESTful services and provides following main (among other) modules:
 - Search Management
 - Case Management
 - Configuration Management
 - Notification Management

Please refer to the to "*EESSI - RINA - Case Processing Interface (CPI) [R8]*" document for more detailed information.

- National Information Exchange Interface (NIE) – The National Information Exchange Interface allows RINA to interact with national applications in order to exchange or receive information in the context of case management (e.g. for creating cases, for completing or exchanging SEDs with national systems). The RINA NIE is in charge to call the National Application for all the important events from the case or document lifecycle and notification generation while providing the relevant information (for example the actual SED part of a document), but it is the participating country responsibility to implement the processing (for example store the SED along with other national useful information or give back an updated SED filled in with information that would be otherwise hard or cumbersome for a clerk to fill in manually). Please refer to "*EESSI - RINA - National Information Exchange Interface (NIE) [R10]*" document for more detailed information about NIE.

There are also other non-public interfaces or modules included in the RINA CPS, which interact with other lower layers from the RINA context (e.g. BMS, TMS).

3.3 RINA Business Messaging Services (BMS)

RINA Business Messaging Services (BMS) provide capabilities to convert an ebMS User Message (technical message) to a business message and vice versa and to convert an ebMS error or acknowledgement into a message status update. BMS works on top of Technical Messaging Service (TMS) that is based on protocol ebMS3.0 - AS4 profile. Processing of the messages sent and received is always stateless and they are all handled in the same way, without any specific BUC-related operations, which are left for the integrating systems (e.g. the BUC-Engine in the CPS).

Based on the configuration, each received message and its status changes are passed further to a handler, which decides how to deal with them. For messages being sent, this layer provides a component for consuming them, applying necessary transformation, business signature and validation and passing them to the layer below, i.e. TMS.

The stateless message processing includes the following main (among other) operations:

- Transformation
- Stateless Transaction Validation
- Business Signature

Currently there are two interfaces accessible for integration:

- BMI MicroService (BMI-MS) – is used for the internal integration purposes with upper layers of RINA present (CPS).
- BMI WebService (BMI-WS) – is used for integration through SOAP web-services (with National Applications) and does not require the upper RINA layers present.

Both the interfaces are using the same core underlying business message processing modules. For more detailed information please refer to “*EESSI - RINA - Business Messaging Interface (BMI) [R11]*” document.

3.4 RINA Technical Messaging Services (TMS)

TMS in RINA provides a service for sending/receiving technical messages. It works based on the protocol ebMS3.0 - AS4 profile. National Applications have also the option to reuse Technical Messaging Service (without getting the benefits and functionalities provided by BMS) as described in the specification document of this interface: “*EESSI - AS4 Messaging Profile [R21]*”. The technical messaging interface follows the file drop concept where:

- Files that need to be sent are written in a specific output folder. There are two types of files that are serialized:
 - Configuration (Pmode files and Pulling configuration)
 - Output messages
- Files that are received are read from a specific input folder. There are two types of files that can be de-serialized:
 - Input messages
 - Input status updates

In RINA context, this layer communicates via an internal interface (TMI) with a third-party application - HolodeckB2B engine. It implements the protocol ebMS3.0 - AS4 profile and handles the communication between an AP and RINA.

The only processing of the message on this layer includes the protocol related operations. Any further business operations based on the content of the received/sent message are performed in the upper layers.

For more detailed information please refer to the “*EESSI - RINA - Business Messaging Interface (BMI) [R11]*” document or the *HolodeckB2B documentation [R24]*.

3.5 RINA CAS

RINA CAS provides the authentication module leveraging the CAS 3.0 protocol. The default internal authentication mechanism is integrated between RINA Portal, RINA CPS and RINA CAS. CAS has been chosen for the great versatility of the authentication options it provides.

RINA Portal itself does not provide any authentication mechanism, rather delegated this to the RINA CAS, which cooperates together with RINA CPS to authenticate the users. The IAM module of the RINA CPS provides the functionality to manage the user information, which is used for authentication. This authentication and the authorization settings coming from RINA CPS are used to secure the resources of the CPI interface.

CAS can be configured also to act as the authentication proxy to external authentication providers. In this case, CPS is not consulted to authenticate the credentials, but is the external authentication provider. Nevertheless, authorisation services still reside in Authorisation Services inside CPS.

4. RINA Logical Components View

The following diagram depicts RINA logical architecture, framed into logical packages and components, which are all described in the next subsections.

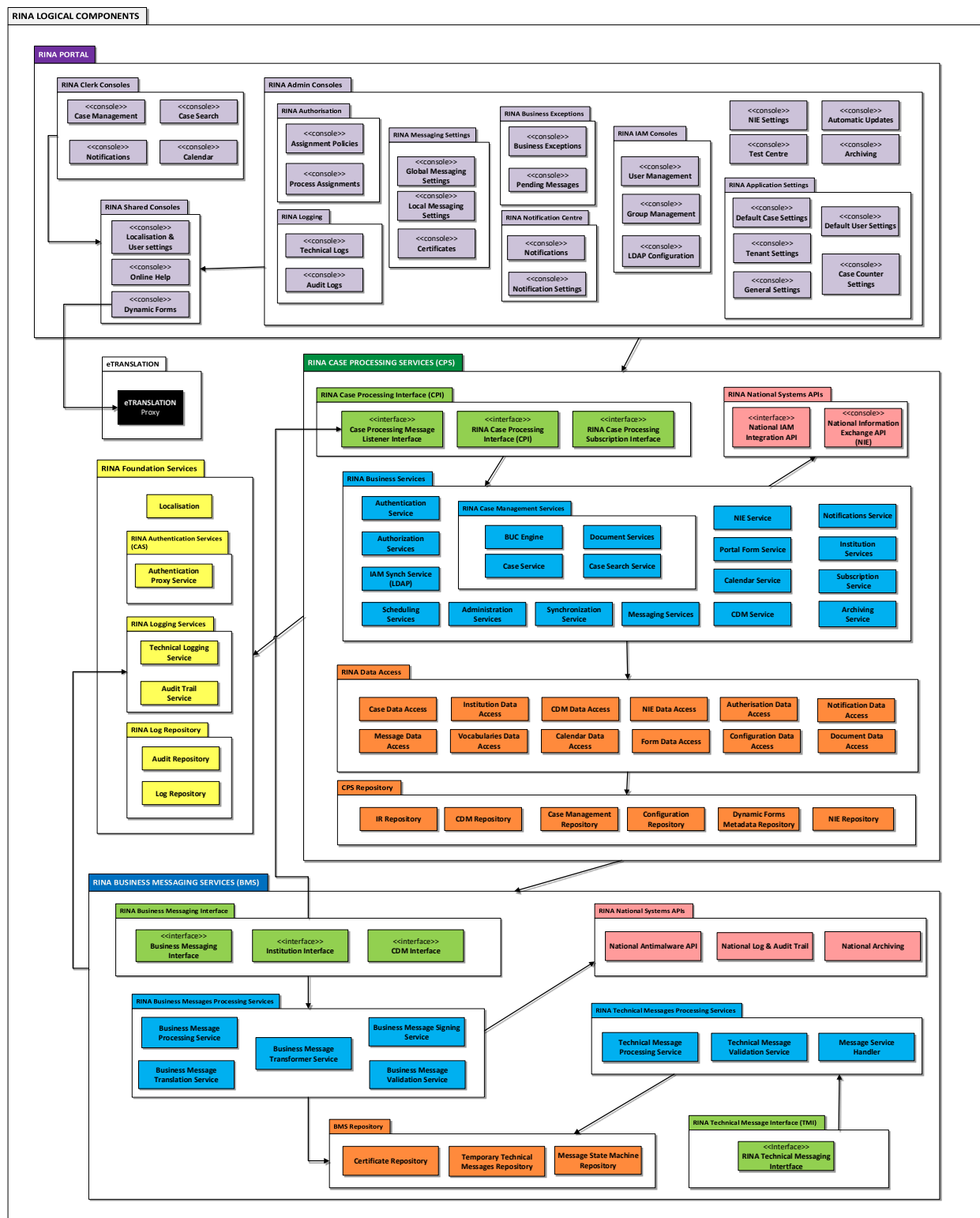


Figure 3: RINA Logical Components View

4.1 RINA PORTAL

This package includes all the graphical user interface (GUI) components used for RINA. It contains two main modules: the Clerk Consoles and the Administration Consoles used by privileged users to administer and configure RINA at different levels. Both modules are utilising the Shared Consoles one for forms rendering, user/localisation settings and online help.

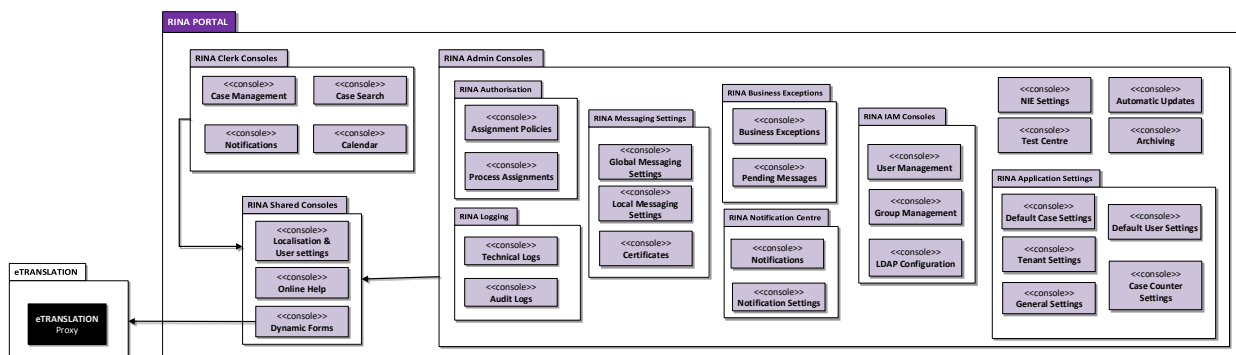


Figure 4: RINA Portal Logical Components

4.1.1 RINA Clerk Consoles

This module enables the clerk to perform all the case management related work. It is a web application designated for clerks in order to exchange in a consistent way the Structured Electronic Documents (SED) which are in the scope of EESSI. This module includes 4 components for searching the cases, notifying the clerk about outstanding cases and tasks, starting of a new case and executing specific human tasks on existing cases.

Component	Description
Case Search	This component allows the users to search the cases they are interested in based on a free text search and a configurable field chooser that gives to the user the option to select, sort and reorder the fields within the search results. It also allows the user to set some predefined searches based on a set of criteria (i.e. case criticality, importance, participants, etc.).
Case Management	This component allows the user to manage the cases. It provides the following functionality: • create/update a business use case within a specific sector • manage the status of the case (open, closed, removed, archived) • manage layout (timeline/classic view) • manage the case participants (counterparties) • manage case assignment (permissions)

	• manage documents (create, save, send, update, delete documents) • filter documents view by document type, document status and document direction • manage comments for cases/documents (create, update, delete comments) • manage attachments for cases/documents (attach, medical information flag, delete attachments) • view document history/versions and conversations • view audit logs for cases/documents • view notifications
Notifications	This component allows the user to monitor and manage any updates for the ongoing cases and documents. It provides the following functionality: • filter notifications by notification severity (error, information, warning) • filter notifications by notification status (read, unread) • filter notifications by notification type (a new case arrived, a new SED arrived, etc.) • search notifications by case id • sort notifications by date and notification type • display the number of notifications per day • manage requests for case assignment (accept, reject) • bulk actions (mark notifications as read/unread)
Calendar	This component allows the user to schedule activities for personal calendar. For each user the calendar allows to schedule activities as a reminders, with actions that should be performed in specific cases when the time comes. Moreover, the calendar displays all the automatically generated notifications by system. The user can view more details about these notifications via the Notifications component.

4.1.2 RINA Administration Consoles

This module includes all components used by privileged users to administer RINA.

4.1.2.1 RINA Application Settings

RINA Application Settings submodule includes five components:

Component	Description
Tenant Settings	This component allows the administrator to configure the Institutions/Liaison Body that are defined as tenants of the specific RINA installation. It provides the following functionality: - manage tenants (add, delete, set default, enable, disable) - view and sort tenants by name, status and is default value
General Settings	This component allows the administrator to configure the general application settings (i.e. the list of languages available to all users, the max idle time, the SED validation mode, the information included in the errors, etc.).
Default Case Settings	This component allows the administrator to configure the general case visualization settings for the users (i.e. Timeline view/Classic view, sort cases by creation date or last update, etc.). These settings can be customized individually by each user through the interface.
Default User Profile	This component allows the administrator to configure the default user profile used by the new users (i.e. language, date format, time zone, etc.). These settings can be customized individually by each user through the interface.
Case Counter Settings	This component allows the administrator to choose and configure the Case Counter type (default, http callback, pattern type) per tenant. Based on these settings RINA will create an additional Business ID for every case.

4.1.2.2 RINA Messaging Settings

RINA Messaging Settings submodule includes three components:

Component	Description
Global Messaging Settings	This component allows the administrator to configure shared messaging settings by all tenants such as the BPM validation mode, the transport mode, retry interval, etc.
Certificates	This component allows the administrator to manage all certificates (TLS, ebMS and Business signatures) for RINA.
Local Messaging Settings	This component allows the administrator to configure the Access Point (AP) data (protocol, IP and port) and specific messaging settings per tenant. The messaging settings

	that the administrator needs to configure depends on the transport mode (pull, push) that was selected previously in Global Messaging Settings component.
--	---

4.1.2.3 NIE Settings

National Information Exchange (NIE) is a RINA stateful interface that allows participant countries to exchange information with their National Applications/Systems. Moreover, it allows RINA to receive documents and/or be notified when configurable events occur during the lifecycle of a case and including its documents, based on subscriptions to specific events during the case handling lifecycle at both case and document levels. To this end, this component allows the administrator to configure specific Case, Document and Notification events that will trigger consequently the appropriate subscriptions.

4.1.2.4 RINA Archiving

RINA is able to archive the closed cases together with documents, comments and attachments. The archiving policy for all of this is configurable and the procedures cover the data archiving of messages. This component allows the administrator to configure the archiving repositories (path, threshold), the policies for archiving messages, the default retention period for messages prior to be removed and the archiving repository that will be used for archiving messages.

4.1.2.5 RINA IAM (Identity and Access Management)

This submodule includes three components:

Component	Description
User Management	This component allows the administrator to manage users and their memberships (group/role) per tenant (add, update, delete).
Group Management	This component allows the administrator to manage groups per tenant (add, update, delete).
LDAP	This component allows the administrator to configure the synchronisation of RINA users and groups (per tenant) with the structure residing at an LDAP server.

4.1.2.6 RINA Authorisation

This submodule includes two components:

Component	Description
Assignment Policies	This component allows the administrator to configure access rights and privileges to perform specific actions on cases by managing assignment policies. There are two different policy types: • The creator policy type that defines the rules according to which the new case creation right is granted to specific user(s)/group(s) • The case policy that defines the conditions under which the case operations (except the case creation) reflected by the selected actors/roles can be performed by the listed user(s) and/or group(s) on received cases or cases created after the activation of the case policy. Moreover, the administrator has the option of grouping policies by creating a group policy that acts as a policy container.
Process Assignments	This component allows the administrator to configure the assigned policies at the level of sectors, processes, application roles and actors. Therefore, the administrator may assign more than one policy to a sector or process manually. Moreover the administrator may perform the same operation automatically by importing a valid process assignment configuration file to the system.

4.1.2.7 RINA Notification Centre

This submodule includes two components:

Component	Description
Notification Settings	This component allows the administrator to configure: • which notifications types will be visible to the users (clerks) • which notifications types will be visible to the administrators • the retention and notification period for business exceptions • the generation of NIE events per notification type
Notifications	This component allows the administrator to track any changes or updates about existing cases. It provides the following functionality:

	• filter notifications by notification severity (error, information, warning) • filter notifications by notification status (read, unread) • filter notifications by notification type (a new case arrived, a new sed arrived etc.) • search notifications by case id • sort results by date and notification type • display the number of notifications per day
--	---

4.1.2.8 RINA Logging

The logging submodule provides the detailed views on the logs that are created at RINA level. It contains two components for accessing audit and technical logs.

Component	Description
Audit Logs	This component gives access to the administrator into valuable information for troubleshooting and review purposes. It provides the following functionality: • define time interval of which the audit logs should be displayed • free text search to limit the returned audit logs • filter audit logs by log type (success, error, unauthorised) • filter audit logs by action type (create, delete, update, execute, read) • filter audit logs by component type (security, notifications, business messaging, technical messaging, etc.) • filter audit logs by participant type (organisation and person) • filter audit logs by participant role (receiver, sender, and subject) • filter audit logs by object type (alarm, comment, attachment, case, policy, document, etc.) • filter audit logs by category type (security, business and messaging) • filter audit logs by event type (submit attachment on document, create document, import subdocument batch, etc.) • sort results by date, event type, action type, etc. • display the number of logs per day/hour

Technical Logs	This component gives access to the administrator into valuable information for troubleshooting. It provides the following functionality: • define time interval of which the technical logs should be displayed • free text search to limit the returned technical logs • filter technical logs by log type BMS (<i>APClient</i>), CPS (<i>REST</i>) • filter technical logs by log level (error, info, debug, etc.) • sort results by date, type, level • display the number of logs per day/hour
-----------------------	--

4.1.2.9 Automatic Updates

The Automatic Updates component allows the administrator to synchronise/configure either IR (Institution repository) or CDM (Common Data Model) retrieved from CSN, through the AP component. In addition, it provides the following functionality:

- free text search to limit the returned resources
- filter resources by resource type (localization, vocabularies, SEDs, etc.)
- filter resources by status (same version, on disk only, installed only, etc.)
- sort resources by type, status, disk version, etc.
- group resources by resource type
- configure settings for automatic updates (i.e. disk resources path on server, shared bmp repository path, etc.)

4.1.2.10 Test Centre

This component allows performing basic testing of RINA's network connectivity to EESSI International Domain. Specifically, there is a sanity test available that can be used to test the status of the Access Point (AP) at which RINA is connected.

4.1.2.11 RINA Business Exceptions

RINA creates messages/exceptions when detects certain issues on incoming messages (i.e. receive SED with invalid signature). The Business Exceptions components allows the administrator to review and act upon such information. It includes two components:

Component	Description
Pending Messages	This component allows the administrator to view the SED details of the message/exception or manually send back to the SED originator a business exception (X050 SED) message. Alternatively, this reject operation will be performed automatically by RINA once the retention period

Business Exceptions	of the corresponding business exception category elapses. Moreover, it provides the following functionality: • free text search to limit the returned messages • filter messages by type (case missing, case removed, etc.) • sort messages by date, type, description • display the number of messages per day • bulk SED rejection This component allows the administrator to view the SED that caused the business exception or even see the Rejection SED generated for this SED. Moreover, it provides the following functionality: • free text search to limit the returned exceptions • filter exceptions by type (case missing, case removed, etc.) • sort exceptions by date, type, description • display the number of exceptions per day
-----------------------------------	--

4.1.3 RINA Shared Consoles

This module includes 3 components used by Clerk and Admin Consoles as well.

Component	Description
Dynamic Forms	Dynamic Forms are used for generating forms on the fly by consuming metadata in JSON format (metadata file). These metadata feed the forms to indicate what will be the fields (i.e. name, type), the values, the order, the validation conditions and other things like placeholders, patterns, etc. RINA integrates the eUI Dynamic Forms ²library for generating dynamically on the runtime the forms (SEDs). Moreover, eUI uses not only a metadata file but also translation files to render dynamically at runtime a HTML form (SED) with inputs to be filled by the user. The metadata file contains information such as: • field/section names • types of the fields (single selection dropdown, multiple selection dropdown, radios etc)

² eUI Framework documentation [R25]

4.2 RINA Case Processing Services (CPS)

This package includes all kind of case management components: interfaces for external use, callback APIs for national application implementation, business services (core components), data access services and repositories that mainly enable the stateful management of BUC Instances (cases).

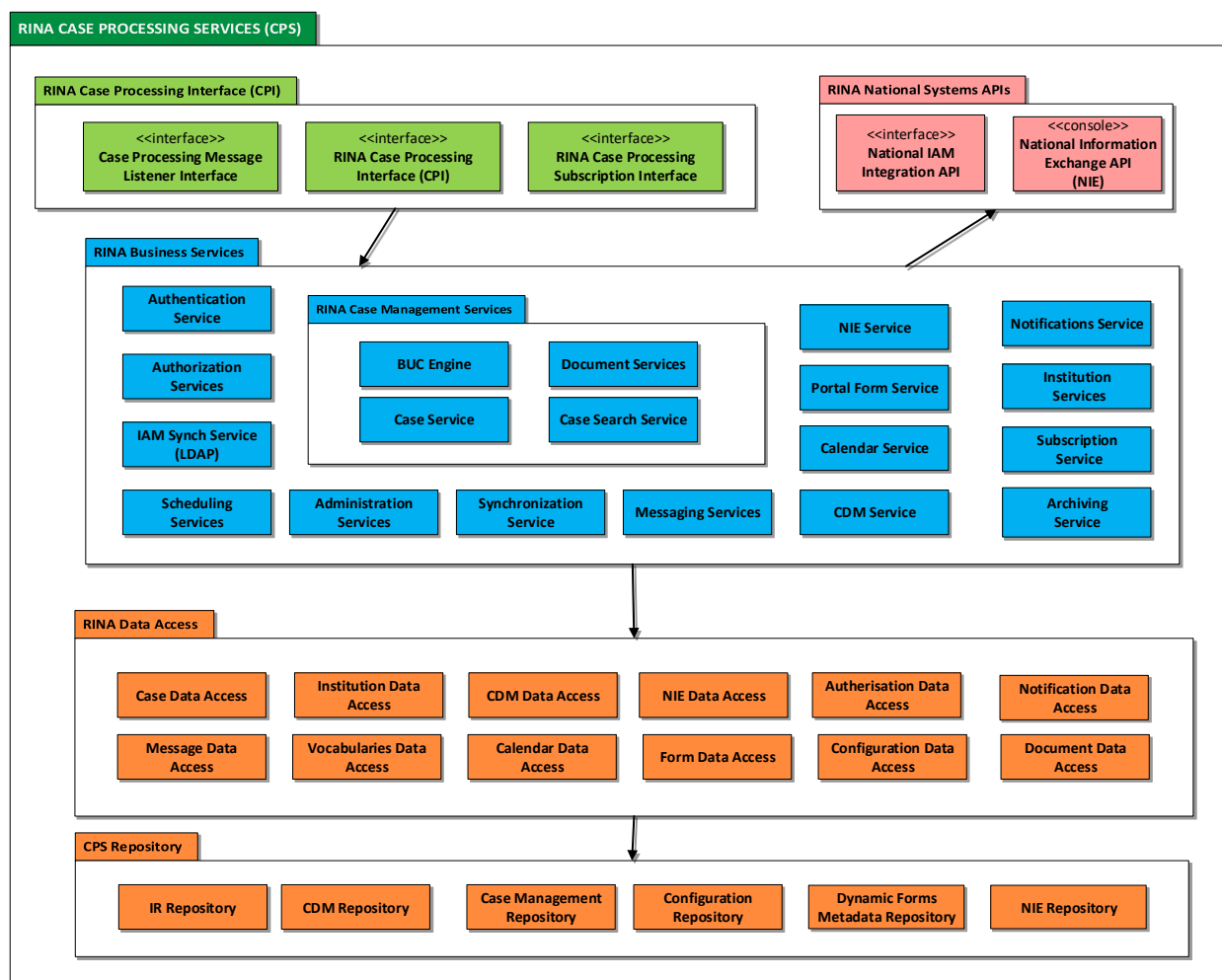


Figure 5: RINA CPS Logical Components

4.2.1 RINA Case Processing Interface (CPI)

This package provides a group of public and internal interfaces and the official communication contracts between upper layers (Portal or any NA client), lower layer components (BMS for the incoming message flow) and the CPS. These interfaces are the following: Case Processing Interface, Case Processing Message Listener Interface, Case Processing Subscription Interface.

Component	Description
RINA Case Processing Interface	This interface, known as RINA CPI provides a RESTful web service that allows RINA portal or national systems to interact with the Case Processing Services. It provides access to different services like the case management services (Case Service, Document Services etc.) notification service, administration and configuration services etc. Clients that use this API can manage cases by creating, updating, deleting or retrieving cases. Also they can manage documents for cases, to publish attachments or comments for both cases and documents. Users can query for notifications if they want to see the outstanding cases and filter them by different categories. Administrators can perform configuration adjustments to the various options of RINA settings.
Case Processing Message Listener Interface	This interface provides a listener to BMS layer in order to receive Business Messages, Status and Signatures by it.
Case Processing Subscription Interface	This interface is used to provide publish-subscribe communication via Websocket with the Portal (or other developed National implementations) for notifying changes in the case state and for new notifications.

4.2.2 RINA National Systems APIs

This package groups all the API callbacks that a national application needs to implement in order to integrate with the corresponding services.

Component	Description
National IAM Integration API	This interface is used to integrate with existing national Identification Access Management (IAM) Systems in order to ensure that RINA remains an extensible platform. It allows RINA to integrate with external identity sources to sync user, groups and role authorisation attributes, in case of external Authentication option.
National Information Exchange API	This National Information Exchange (or NIE) API allows the exchange of events (from/to) and notifications through different subscription services with national systems.

	This interface allows sending and retrieving data to and from the National Applications through programmatic interfaces. RINA Case Processing Interface still needs to be used in order to send messages (and eventually check those received), but the interfaces will automatically communicate the data to the National Applications through this callback interface based on predefined set of events.
--	---

4.2.3 RINA Business Services

RINA Business Services provides the implementation of the group of Case Processing Interfaces, and the necessary services to integrate with the National System APIs, as listed above.

Component	Description
BUC Engine	The BUC Engine, core of the CPS, is in charge of executing the business logic and actions as defined in the EESSI Business Use Cases (business processes). It is an stateful custom made BPM engine, persisting the cases information (BUC instances) in RINA's DB (PostgreSQL) The BUC Engine reacts to the business events (SEDs), is able to determine in a simplified manner what the status of the flow is, which are the already exchanged documents and what are the next potential actions that can be executed. More information on it is available in "<i>EESSI - RINA - Business Use Case Engine Architecture [R13]</i>"
Case Service	This module provides services for: • case management like creating, reading, updating or deleting cases • case assignment • provides also the guidance for the next available actions in the case context, that BUC Engine has computed.
Document Services	The Document Services is the component in charge of the management (storing, validating, retrieving etc.) of documents(SEDs) and attachments. It also acts as the interface of manipulating the executable models (template forms, etc.) and executing the relevant administration actions.

Case Search Service	The Case Search Service is dedicated to the search and retrieval of the list of cases and to manage (edit or apply) search definitions.
Notifications Service	The Notifications Service is handling the management of notifications to the end user. This module takes into consideration several aspects in organising the notified information e.g.: criticality, importance, case type, alarm alerts, new received messages, case related quality of service requirements, deadlines etc.
IAM Synch Service	This service is in charge with the identity and access synchronisation between the external data source (i.e. LDAP) and the local data source the engine is using. Users, groups and roles targeted for this synchronisation.
Authentication Service	This service is in charge of authenticating the users via the internal repository. The verification is performed through both the username and password.
Authorisation Service	This service is in charge of managing (computing, checking etc.) the users permission to access only the intended resources. On the case context level it also manages the case assignments via the assignment policies. There are 7 different case context roles (namely: 'supervisor', 'authorised clerk', 'unauthorised clerk', 'auditor', 'viewer', 'medical user', 'vip user').
Scheduling Services	These services provide scheduling jobs for several functionalities. ▪ Review of the received pending Messages, Signatures, Status (ones that failed to be processed initially): Will trigger periodically the re-processing. • Automatic archiving of cases and messages. • Alarm notifications.
Administration Services	Under this component all the administration, monitoring and configuration services of RINA are logically grouped. That includes management of application, messaging, tenant, resources settings etc.
Messaging Services	These are services responsible to manage messages, on the level of the CPS, i.e. creating, sending,

NIE Service	receiving, persisting pending and processing messages from and to BMS layer. The service is in charge of publishing events, related with cases, documents and notifications, to the National Application NIE Implementations (subscribers).
Portal Form Service	This service is in charge of providing the necessary metadata and translation content files to the Portal for generating the SED forms dynamically.
Calendar Service	This service provides the capacity to the users to manage (create/edit/delete) and review the personalised Calendar content.
Institution Services	This component provides the system the capacity to manage the Institution Repository information that has been synchronised locally and also implements the Institution search endpoint of CPI.
Synchronisation Service	This service is responsible for synchronising and processing locally the IR and CDM Data from the CSN via AP. Those operations are enabled via the sending/receiving of SYNC messages.
CDM Service	This service provides the system the capacity to manage the CDM artefacts that have been synchronised locally.
Subscription Service	This service is responsible for managing the websocket subscriptions from the Portal (or any NA Client) and publishing case and notifications events to it.
Archiving Service	This service provides the capacity to setup archiving configuration for cases or messages, configuration for message retention and also to perform the actual archiving.

4.2.4 RINA Data Access

This package includes all the data access services so that repository information is accessed by the business layer using this higher data layer.

Component	Description
Case Data Access	The Case Data Access component provides data services for accessing the Case Management Repository.

Document Data Access	The Document Data Access component provides data services for accessing the Documents and Attachments inside the Case Management Repository
Institution Data Access	The Institution Data Access component provides data services for accessing the local Institution Repository.
Configuration Data Access	The Configuration Data Access component provides data services for accessing the content associated to the RINA administration module.
Message Data Access	The Message Data Access component provides data services for accessing the received sent or pending messages.
Vocabularies Data Access	The Vocabularies Data Access component provides data services for accessing the vocabularies.
Notification Data Access	The Notification Data Access component provides data services for accessing notifications.
CDM Data Access	The CDM Data Access component provides data services for accessing the Common Data Model content.
Calendar Data Access	The Calendar Data Access component provides data services for accessing the calendar's activities.
Authorisation Data Access	The Authorisation Data Access component provides data services for accessing the relevant content dedicated to the roles assigned of each user (at a case level for business users).
NIE Data Access	The NIE Data Access component provides data services for accessing the content of the NIE events.
Form Data Access	The Form Data Access component provides data services for accessing the forms metadata and translations.

4.2.5 CPS Repository

This package includes the necessary Case Processing Services repositories and also a replica of repositories hosted in the Central Services Node and then replicated at AP (Institution and CDM). This replica is hosted at local level to avoid the need of the RINA to connect continuously to the AP. The local replica must be regularly updated with the information coming from the AP.

Component	Description
Institution Repository	The Institution Repository contains all information of the institutions exchanging messages within EESSI. This information is also replicated at RINA.
CDM Repository	The Core Data Model Repository contains business use case metadata, SED metadata and other common information. The information included in this component must be a replica of the one at Central Services Node level. The information is always updated at the repository in the CSN, published to the repositories at the AP's then synchronised to RINA.
Case Management Repository	This repository persists all the case processing and management related data such as: cases, documents, attachments, comments, business signatures, notifications etc.
Configuration Repository	This repository persists all the configuration related options associated to the operations related to the administration user (through the CPI or Admin portal).
Dynamic Forms Metadata Repository	This repository persists all the related data, such as form structure metadata and translation content for generating localized forms.
NIE Repository	This repository persists all the data related to NIE such as subscriptions, subscribers and listeners.

4.3 RINA Business Messaging Services (BMS)

The RINA Business Messaging Services provide a reusable component for sending business messages to the Access Point using the technical protocol ebMS3.0/AS4. These services ensure the conversion, validation and signing of the business messages for correct exchange with the EESSI environment, and the correct reception of messages from other institutions.

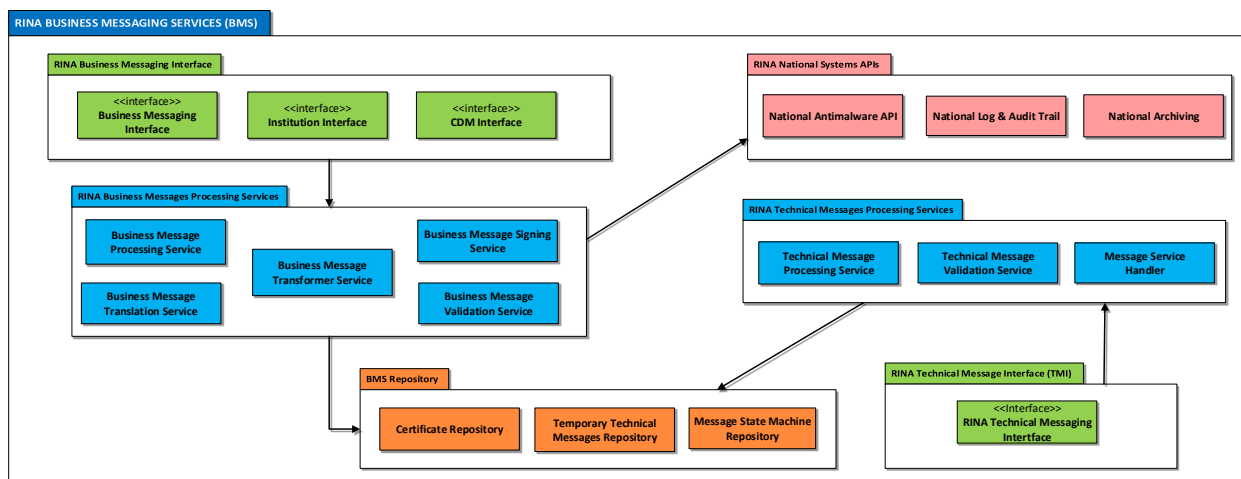


Figure 6: RINA BMS Logical Components

4.3.1 RINA Business Messaging Interface (BMI)

RINA Business Messaging Interface consists in three interfaces: Business Messaging Interface, Institution Interface and CDM Interface.

Component	Description
Business Messaging Interface (BMI)	The RINA Business Messaging Interface provides the following functionality: - Send business messages (user message, either of SED or SYN type) to a preconfigured EESSI Access Point (AP) via (TMS) using the Message Service Handler (MSH); - Notify a registered listener about a received business message (user message, either of SED, SYN or LOG type) from a preconfigured AP using the MSH; - Notify a registered listener about a status update (acknowledgement or delivery failure) of a business message (user message, either of SED or SYN type) from a preconfigured AP using the MSH;

4.3.3 RINA Business Message Processing Services

The RINA Business Processing Services includes all the services performed on a business message.

Component	Description
Business Message Processing Service	The Business Message Processing Service implements the business messaging interface and it does this by orchestrating all the other business message services.
Business Message Translation Service	The Business Message Translation Service makes the conversion between the business domain and the transport domain by translating the business message into a low level ebMS message (technical message) or the other way around (for outgoing and incoming messages respectively).
Business Message Transformer Service	The Business Message Transformer Service makes the conversion of the IDs to GUIDs format for the outgoing and the opposite conversion for the incoming messages.
Business Message Validation Service	The Business Message Validation Service validates the business message structure like the business header, the SED or the attachments.
Business Message Signing Service	The Business Message Validation Service is responsible for signing the SED (outgoing messages).

4.3.4 BMS Repository

This package includes all the repositories used to persist information in this layer.

Component	Description
Certificate Repository	The certificate repository stores the private key data but also a replica of the one at CSN level for the public certificates. The information about public certificates is always updated at the repository in the CSN, published to the repositories at the AP's and then synched with RINA. The management of the certificates (storing, updating) is performed via the CPS layer or any other National Application implementation in case of using only the BMI layer.
Temporary Technical Messages Repository (Queue)	The Temporary Technical Message Repository (Queue) is used to decouple the business domain from the technical domain:

Message State Machine Repository	- all the exchanged messages that are being sent (outgoing) are queued by the Business Processing Services and de-queued by the Technical Processing Services (TMS) - all the exchanged messages that are being received (incoming) are queued by the Technical Messaging Services (TMS) and de-queued by the Business Processing Services. The communication between RINA Business Messages Processing Services and RINA Technical Processing Services is performed through the specific component It is used to store information about incoming messages and status of failing outgoing and incoming messages
--	---

4.3.5 RINA Technical Messaging Services (TMS)

The RINA Technical Messaging Services (TMS) ensure the technical transport of the exchanged messages, including the signing and security measures, their validation and attachment handling.

This corresponds to the technical ebMS client implementation for the interaction with the Access Points.

Component	Description
Technical Message Processing Service	The Technical Processing Service interacts with the Temporary Technical Messages Repository (queue) and other technical message services (described below)
Technical Message Validation Service	The Technical Message Validation Service validates the low level ebMS message
Message Service Handler	The Message Service Handler (Holodeck application) is the ebMS/AS4 implementation that provides all the low level handling at the ebMS/AS4 to exchange ebMS messages with the AP. The main functionalities are: - signing and signing verification (for outgoing and incoming messages respectively) - low level technical error handling and generation exchange of receipts/errors - 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 The following parameters need to be configured: • pmode • certificate stores
--	---

4.3.6 RINA Technical Messaging Interface (TMI)

This interface consists of the transport level services used for the communication with AP.

Component	Description
RINA Technical Messaging Interface (TMI)	This interface is the connecting point between the Technical Messaging Services and the Message Service Handler. It is based on file dropping concept into the file system where files that need to be sent or received are written in specific (output or input respectively) folders. Each MSH endpoint (corresponding to tenant association) contains a dedicated folder for incoming messages.

4.4 RINA Foundation Services

This package contains sub-packages and components that perform logically shared functions related to authentication and policy enforcement, logs and audit trail gathering and messaging supporting services such as connectivity and message assemblers.

It's a core package for each of the subsystems as it contains services critical for the functioning of RINA

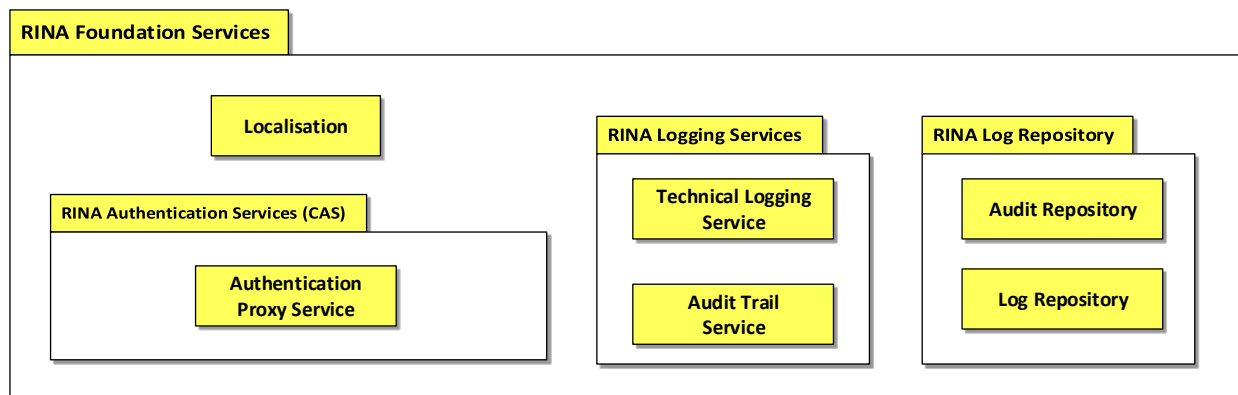


Figure 7: RINA Foundation Services Logical Components

4.4.1 Localisation

As RINA is a software that is deployed and involves business exchanges between several locations, under the Localisation logical component we group all the different services that adapt the RINA software to different languages, number formats, currencies, regional differences and other local non-functional requirements.

4.4.2 RINA Authentication Services (CAS Server)

In RINA, the authentication mechanism is mainly orchestrated by the Central Authentication Server (CAS).

Component	Description
Authentication Proxy Service	RINA CPI uses the CAS 3.0 protocol for authentication. The following parties are present in the scenario: - CAS Authentication Server – the SSO server providing the authentication to users requesting to use a service. - Service – CAS is providing the authentication for registered services. In our case, the registered service is the RINA REST CPI module. - Portal/UI – is the client (or any other client), which wants to use the service and needs to get authenticated by the CAS server The core function of Authentication Proxy Service is to validate the required credentials of users, administrators or

	a number of applications accessing information or protected resources in the subsystem environment, via redirecting the authentication into CPS (internal) or to external providers. Once the login is granted the Login API will create an authenticated session and return a JSESSIONID cookie. This cookie must be sent with every further request to the CPI.
--	---

4.4.3 RINA Logging Services

The RINA Logging Services ensure the correct logging of technical and synchronisation actions within the system, and for the audit trailing data.

Component	Description
Audit Trail Service	This component provides a sequence of chronological records of RINA related events to enable the investigation of different type of incidents. More information can be checked in section 7. RINA Audit Trail
Technical Logging Service	Technical logs are provided at the RINA level. Aggregated technical logging information from various components is aggregated (via Logstash) from the file system and persisted onto the ElasticSearch database. It is accessible through CPI and Portal interfaces. More information can be checked in section 8. RINA Technical Logging

4.4.4 RINA Log Repository

This package includes the two repositories on log and audit trail of RINA.

Component	Description
Audit Repository	This is a logic repository that stores audit records produced by different RINA components
Log Repository	This is a logical repository that stores technical logs records produced by different RINA Components. Technically is mapped to more than one physical locations and Elasticsearch, depending on the configuration.

5. RINA Repositories

RINA is storing all the data needed for its operation in a set of physical repositories. These repositories will be presented in the following section.

5.1 Elastic Search (ES)

The repository of Elastic Search (ES) is currently accepting inserted data only via the logstash component. Its purpose is to store log related data.

All log data are sent to the logstash component via the log4j2 module of RINA. By default log4j2 uses system sockets to push the data to logstash.

Technical logs can be also retrieved from RINA in order to be displayed to the UI. Audit logs which are currently stored to ES are now limited only to those provided by BMI-MS (i.e. the ApClient layer) and they are left there for continuity purposes.

The audit logs of CPI are no longer stored into ES since they have been moved to the SQL DB of RINA for improving data integrity and transactionality.

More information on RINA Audit and Logging architecture can be found in sections 7 and 8 of this document.

5.2 File System

The filesystem repository is used for storing the incoming and outgoing messages as well as the files containing the ebMS private and public keys, the business private and public keys, and the certificates needed for the institutions registered in a specific RINA installation.

The BMI (MS/WS) module accesses the key files in order to be able to sign messages. Furthermore it stores and retrieves messages to and from the filesystem. The incoming messages are sent to CPI or received from CPI.

The file system acts as a persisted queue between the BMI and Message Service Handler (Holodeck). An incoming message is stored to the filesystem to some distinguished location that is specifically assigned to each different tenant of the RINA installation. This incoming message is placed into that location by Holodeck after it is received from the AP. It is then picked from the filesystem by a process, created specifically for that particular tenant, and is executed inside BMI. This process is responsible for passing the message to CPI.

A similar procedure takes place for outgoing messages. The message is sent to BMI from CPI. BMI signs and stores the message into the filesystem. Holodeck picks the message from the filesystem and sends it to AP.

Filesystem is also used for storing specific application node configurations, that are not exposed via the administration services but also for temporary data for Case Processing Services.

5.3 RINA SQL Database

RINA is using SQL database as the main source of truth.

The SQL server that is used is that of PostgreSQL 12. Detailed description of the SQL schema of the RINA DB can be found in the document – *"EESSI - RINA 6.2.1 - SQL DB Schema - Physical Data Model [R22]"* whereas a high definition diagram is also provided for visual inspection of the schema (see *"EESSI - RINA 6.2.1 - SQL DB Schema - Figure [R23]"* image).

The RINA schema serves the purposes of storing all the data related to Case Processing Services (CPS). Data which are related to business entities such cases, document, notifications etc. are stored into SQL tables.

The data which are needed for the configuration of RINA such as certificate management, AP connectivity, organizations, tenants, users and groups, assignment policies and user profiles are stored in this DB too. These data are managed the Administration Services of the UI and CPS, but they are accessed throughout the entire operational cycle of RINA by all its components.

The data which are related to audit logs of RINA have their place inside the database. The same applies for NIE Service related data.

Special care has been taken so that the relations between entities are strongly forced and well defined inside the schema. These relations are imposed using foreign keys for all the table associations so that data integrity is guaranteed. All tables contain a simple surrogate key field or System ID (SID) which is a long integer. This makes the definition of the constraints simple and clean. The SIDs are always given their values from sequence tables defined specifically for each table in an 1-1 fashion.

Most of the tables also contain an ID column used for business Id and/or compatibility with the previous versions of RINA.

Several indices have been defined in to the DB for performance as data integrity when unique constraints are imposed on the data. Mandatory constraints also have been imposed on various fields when required.

5.4 ApClient(BMS) DB

BMI (MS/WS) stores all the essential data into the ApClient DB.

The SQL server that was used is that of PostgreSQL 12. Detailed description of the SQL schema of the RINA DB can be found in the appendix of the *"EESSI - RINA - Business Messaging Interface (BMI) [R11]"* document.

This DB is small and it serves the purpose storing information about incoming messages or failing outgoing and incoming messages. The mechanism is explained in the aforementioned appendix [R11].

5.5 Holodeck DB

Holodeck has its own dedicated instance of PostgreSQL. For more information one could peruse the corresponding documentation of the Message Service Handler within the *Holodeck B2B documentation [R24]*

5.6 RINA Logical Repos mapping

The next table provides the reader with the mapping of the full set of RINA logical repositories (described in section 4), with the physical ones, introduced in this section.

RINA Physical Repository	RINA Logical Repository
Elastic Search	Log Repository Audit Repository (BMS)
File System	Log Repository Audit Repository (BMS) Certificate Repository Temporary Technical Messages Repository
RINA DB	Audit Repository (CPS) IR Repository (local) CDM Repository (local) Case Management Repository Configuration Repository Dynamic Forms Metadata Repository NIE Repository
APClient DB	Configuration Repository Message State Machine Repository
Holodeck DB	Temporary Technical Messages Repository

Table 4: RINA Logical and Physical Repositories

6. RINA Case Processing Services Behaviour

In this section the reader can find more details about the behaviour of RINA different layers and components.

This is achieved by explaining some representative and important business use cases that combine a big part of the functional domains and integration patterns. These are the creation of a Case, the creation and sending of SED/Message as well as the receiving of a message.

The description is not limited to a black-box approach, but attempts to give more fine-grained details about the interaction of different services, zooming into more the CPS related components. Still, for logical simplification, the diagrams are not depicting the full set of internal technical services and calls, but the logical mapping of the services/components is mostly used.

More information about the BUC Engine internal processing but also about the BMS component can be found in the dedicated documentation ("*EESSI - RINA - Business Use Case Engine Architecture [R13]*" & "*EESSI - RINA - Business Messaging Interface (BMI) [R11]*").

Finally, this section is mostly covering the successful outcome of the selected use cases and does not go into more details about the flow interactions and handling in case of errors.

6.1 CPS: Create Case

The creation of a case instance of a specific BUC Type is one of the fundamental functions of RINA. Portal -or any other CPI client- and CPS (including BUC Engine) are involved.

NIE, Audit, Notification and Subscription services are also participating to the orchestration. In this use case more details are provided on how and when they are integrated. In the next chapters/use cases those details will be omitted as they follow very similar behavioural pattern.

The diagram below represents the necessary interactions for the creation of a new case.

In detail,

- The user selects the BUC Type and triggers the creation of a new Case and the portal calls (via REST HTTP request) the relevant CPI Endpoint.
 - *Case Processing Services*, and in particular, *Case Service* proceeds to the following processing calls in a synchronous manner, as part of a single atomic transaction
 - Validates if the user has the available rights to create this case via the *Authorisation Services* and if the user's tenant institution is competent to create this BUC Type, via the *Institution Services*.
 - Persists initial case metadata.
 - Calls *BUC Engine* and BUC Engine executes the internal create case action and updates the case metadata information. BUC Engine also

- computes and persists the next available actions according to the BUC process definition.
- Triggers the *Authorisation Services* to compute the case assignments according to the defined policies and persists them. These are the users and groups assigned to this case with their relevant roles in this specific case context.
 - Calls the *Notifications Services* to compute and create user notifications, according to the case assignments. *Notifications Services* publish the internal events for the new notifications after the transaction is committed.
 - Triggers the *Audit Services* to audit the creation of the new case. If the business operation to create a case will be successful, then a success outcome will be audited, otherwise if there is any error (including the generation and persistence of the audit event) and error outcome will be persisted and the operation will be rolled back.
 - Informs the NIE Service to generate the NIE create case event.
 - Notifies the portal via *Subscription Service* about new case created.
- After the previous transaction is completed successfully, NIE Service will publish the Create Case NIE Event to the subscribed listeners.
 - Portal (and relevant CPI endpoint) retrieves the case and its recomputed actions, after being informed that operation was completed

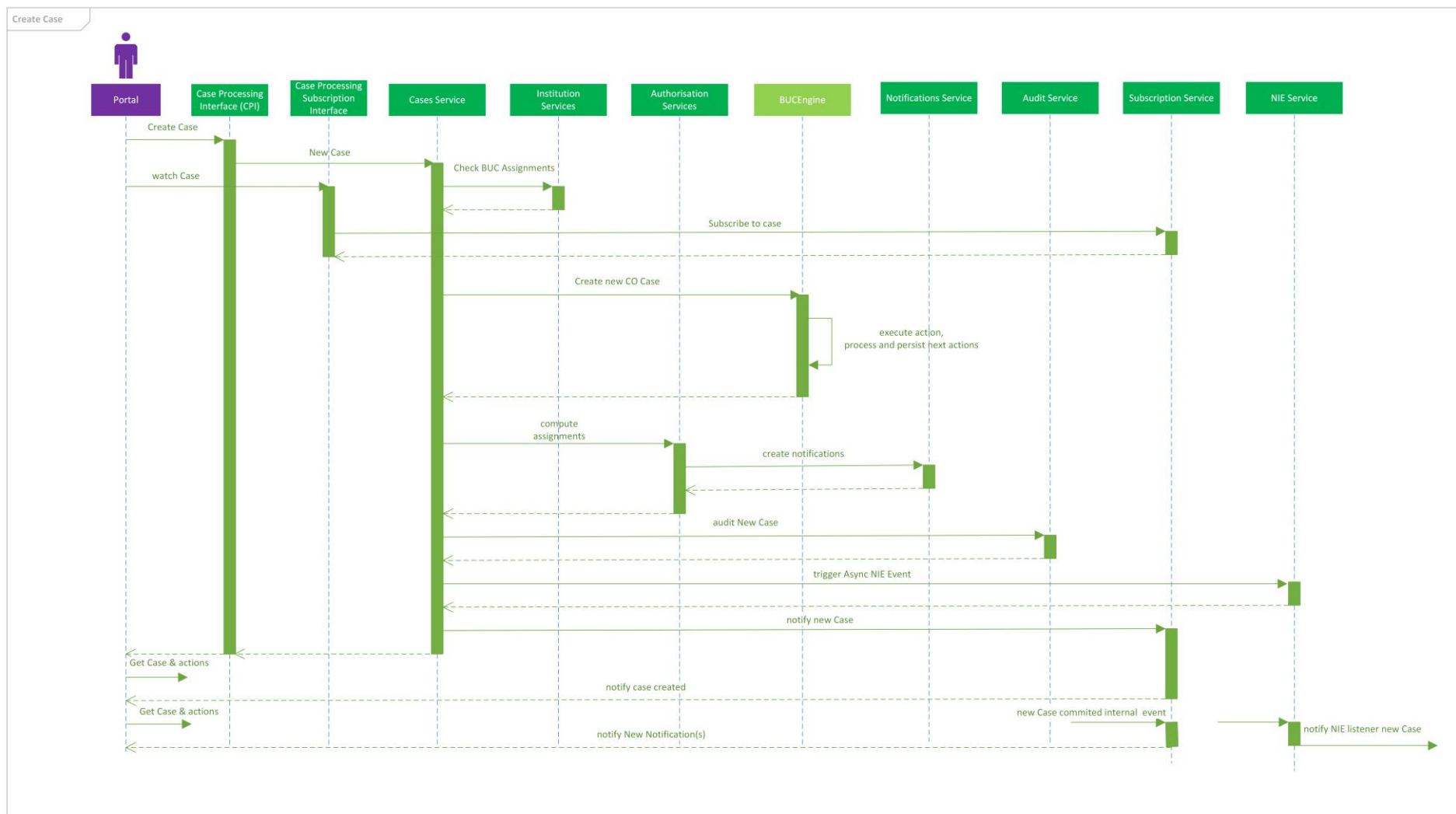


Figure 8: Create Case Sequence Diagram

6.2 CPS: Create and Send Document

Creating and sending a SED to the counterparty Institutions, in the context of a case instance, is maybe the most important operation in RINA and EESSI.

The creation of the SED involves the CPI Client (Portal or other) and the CPS.

In the sending part most of the different layers in EESSI and RINA are involved. In this section we will provide more details on RINA CPS and how they also interact with BMS/TMS.

Of course, for those operations to be performed there is a prerequisite: a case created or received.

The diagram below represents the necessary interactions for the saving of a SED, and sending it enveloped inside a message.

In more detail, regarding the creation of the SED:

- The user selects the SED Type (according to corresponding available create actions) and triggers the creation of a new SED, so as to receive the SED Form. The portal calls (via REST HTTP request) the relevant CPI Endpoint.
 - *Case Service* proceeds to the following processing calls in a synchronous manner, part of a single atomic transaction:
 - Validates if the user has the available rights to view the initial document (SED Form) via the *Authorisation Services*.
 - Calls the *Document Services* to retrieve and prefill the initial document with already present case context data and return it.
 - The initial document is returned to the requester
- *Portal* opens the edit form view of the SED. The Portal is using also the *Portal Form Service* via CPI in order to load the metadata and translations to render the form (not shown in the diagram above).
- The user enters the data in the SED Form, and saves it. Portal is calling the CPI.
 - *Document Services* proceeds to the following processing calls inside the same transaction in a synchronous manner.
 - Validates if the user has the available rights to create the SED via the *Authorisation Services* and that the action is valid.
 - Calls the *BUC Engine* to lock the Create SED action.
 - Validates the document content against corresponding SED XSD and then persists it.
 - Calls the *BUC Engine*, and the *BUC Engine* validates from business point of view the document content, executes the create document action, saves the document metadata and content, generates and persists the new actions and then unlocks the Create SED action.
 - *Document Services* initiates the audit of the transaction, creates any relevant notification and triggers NIE events (*the diagram is simplified in this step, NIE Synchronous events related to the Document Content, are also not described in the previous steps*).

Send SED:

- The user selects to open the Send form in the *Portal*, so that the send participants can be confirmed or modified. The *CPI* will return the initial document of the Send form like described in previous use cases.
- The user triggers the send operation in the *Portal*.
 - *Document Services* proceed to the following processing calls in a synchronous manner, part of a single atomic transaction:
 - Validates the provided Admin Send Action Document Content.
 - Validates if the user has the available rights to send the document via the *Authorisation Services*.
 - Calls the *BUC Engine* to lock the Send SED action.
 - Call the *BUC Engine* and *BUC Engine* validates, executes the send SED action, generates and persists the new actions and user messages, updates the metadata of document and case according to business, and then finally will unlock the Create SED action
 - *BUC Engine* calls the *Messaging Services*, and the *Messaging Services* trigger internal sending event for one or more messages (one for each participant).
 - Initiate the audit of the send action, create any relevant notification and trigger NIE events (*the diagram and description is simplified in this step, more details can be found in Create Case section, as the behaviour is similar*)
- After the previous transaction is completed, the *Messaging Services* send the Business Message(s) to *BMS-MS*.

○

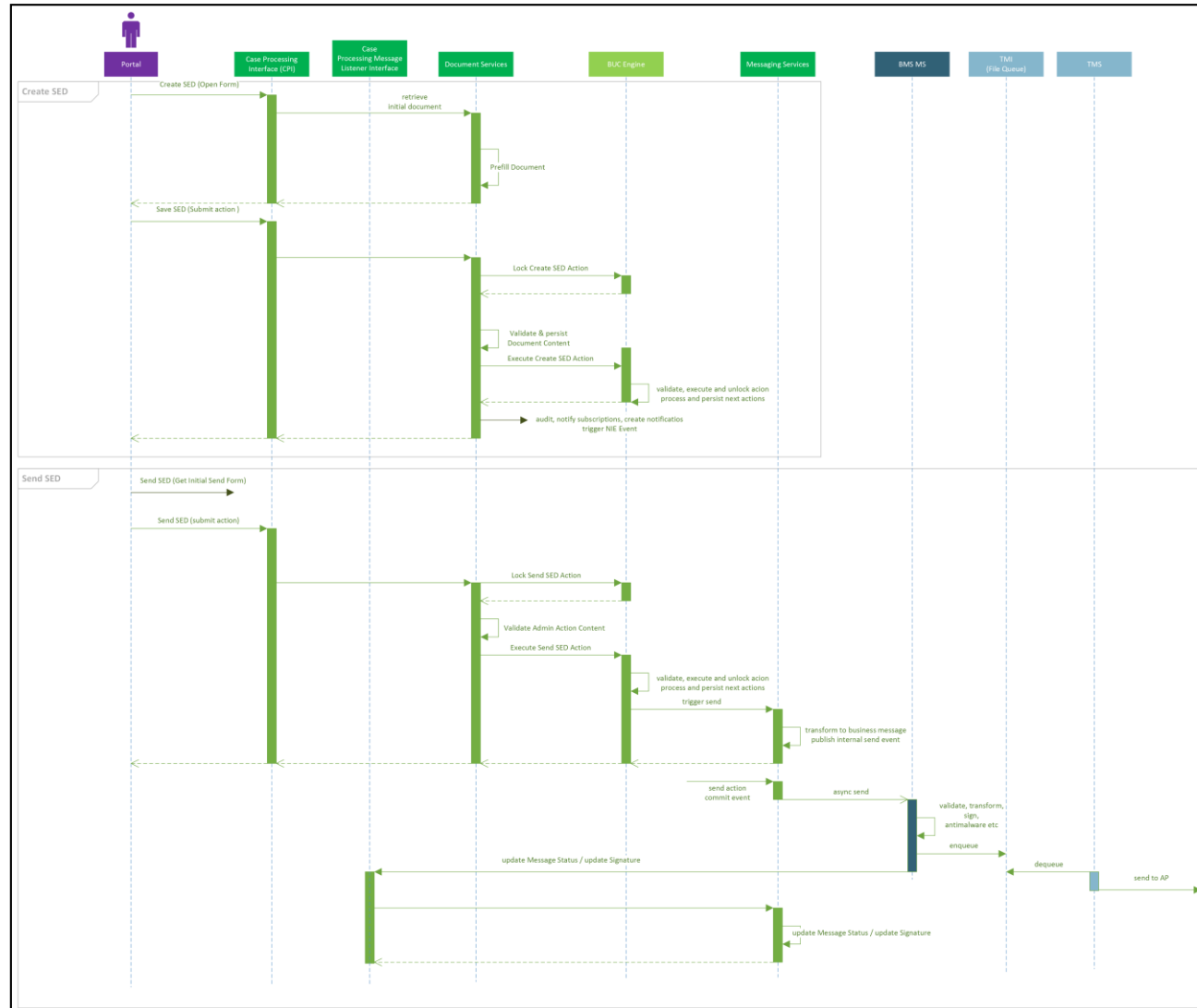


Figure 9: Create and Send Document Sequence Diagram

- *BMS MS* performs asynchronously the following actions:
 - validates the message(s) against Transaction and SED XSDs, apply transformation, perform antimalware check, apply business signature and enqueue the message(s) for *TMS (Holodeck)*, that will finally send the message(s) to the *Access Point*.
(More details can be found under the *RINA - Business Messaging Interface (BMI) [R11]* documentation).
 - Registered *CPIAListener* notifies the *Case Processing Message Listener Interface* (on CPS) about the message status (SENT/ERROR).
 - If the original message is already persisted, *CPS Messaging Services* update its message status. Otherwise a pending message status is persisted. Periodically *Scheduling Services* will check if the message finally exists and will reprocess the status of the message.
 - Registered *CPIAListener* notifies the *Case Processing Message Listener Interface* (on CPS) to update the message signature.
 - If the original message is already persisted, *CPS Messaging Services* update the message signature. Otherwise a pending message signature is persisted. Periodically *Scheduling Services* will check again if the message exists and if this is true will reprocess and persist the signature of the message.

6.3 CPS: Receive Message

The reception of a business message is also a crucial EESSI and RINA functionality.

It again involves almost all EESSI and RINA components. Like before, this section is focusing on the CPS layer, but also showcases how the lower layers BMS/TMS will interact with it in order to make the reception of a business message possible.

There are different types of received messages, according the case context:

- RECEIVED that are messages that include new or updated SEDs.
- START or START_FORWARD messages that contain received SEDs but will also result in the creation of a counterparty case.
- There are also SYNC type of messages, that contain the IR or CDM Synchronisation SEDs and data for which we do not include more details in this section.

The diagram below represents the necessary interactions for the reception of a message (that contains the SED). Of course, prior to the receiving, the creation of a case, creation and sending of the SED are prerequisite steps.

In more detail:

- *TMS* receives a new message from AP. If the message is accepted, *TMS* enqueues the file in the configured file message IN directory.
- *BMS MS* is informed by the new message (by a File watcher) and
 - dequeues and processes it. Messages, at *BMS* layer, have an internal state which can be *PROCESSING* or *PROCESSED*. This state is persisted in ApClient DB to avoid race conditions or duplicate messages. If for a

- message there is no state in DB, the message id will be persisted with initial state *PROCESSING* (more details can be found under "*EESSI - RINA - Business Messaging Interface (BMI) [R11]*").
- *BMS MS* dispatches the message to the *CPI AP listener* (that will send it to CPS component layer, see below step) and the *CPI AP Listener (BMS)* delivers the message to the *Case Processing Message Listener Interface (on CPS)* through secured REST calls. Contents (*SED* and attachments) are saved in new files to be available for the CPS.
 - *Messaging Services (on CPS)* map the payload in a pending message, persist it in the DB and trigger the async processing in the CPS layer and return. The pending message is locked by *Messaging Services (on CPS)* to avoid parallel processing by the *Scheduling Services (on CPS)*.
 - *BMS MS* marks the message as *PROCESSED* and the relevant file is moved out from the IN Queue so that is not reprocessed.
 - After the dispatching of the message from BMS to CPS is completed, *Messaging services (on CPS)* performs the actual processing of the received message, in a new transaction.
 - According to the Message case action type, the message will be validated, and a specific handler is assigned to handle it.
 - If the message is of type *SYNC*, the *Synchronization Services* will handle the message.
 - If the message is of type *Start* or *StartForward*, a new case will be created. In a similar manner with create case, *Authorisation Services* is triggered to compute and persist the users and groups assigned to this case with their relevant roles.
 - If the message is not *SYNC*, a new document metadata will be created (or updated if already exists and type is *UPDATE*) and the new content is persisted on a new document version.
 - Message is dispatched forward to *BUC Engine*. *BUC Engine* executes the specific action according to the message type, the user message is persisted, the case and document metadata are updated, the attachment metadata (if any) is created (or updated) and the new available actions on the case are computed and persisted according to definition of the BUC process.
 - *Messaging Services* store the attachments content, if any, in the configured directory, and calls the relevant services to audit, generate the notifications and NIE events for received message.
 - If the processing is successful, the pending message will be deleted and also the temporary files.
 - If during the processing an error will occur, the processing operation and only will be rolled back. The pending message will be then reprocessed by the *Scheduling services* according to parameters

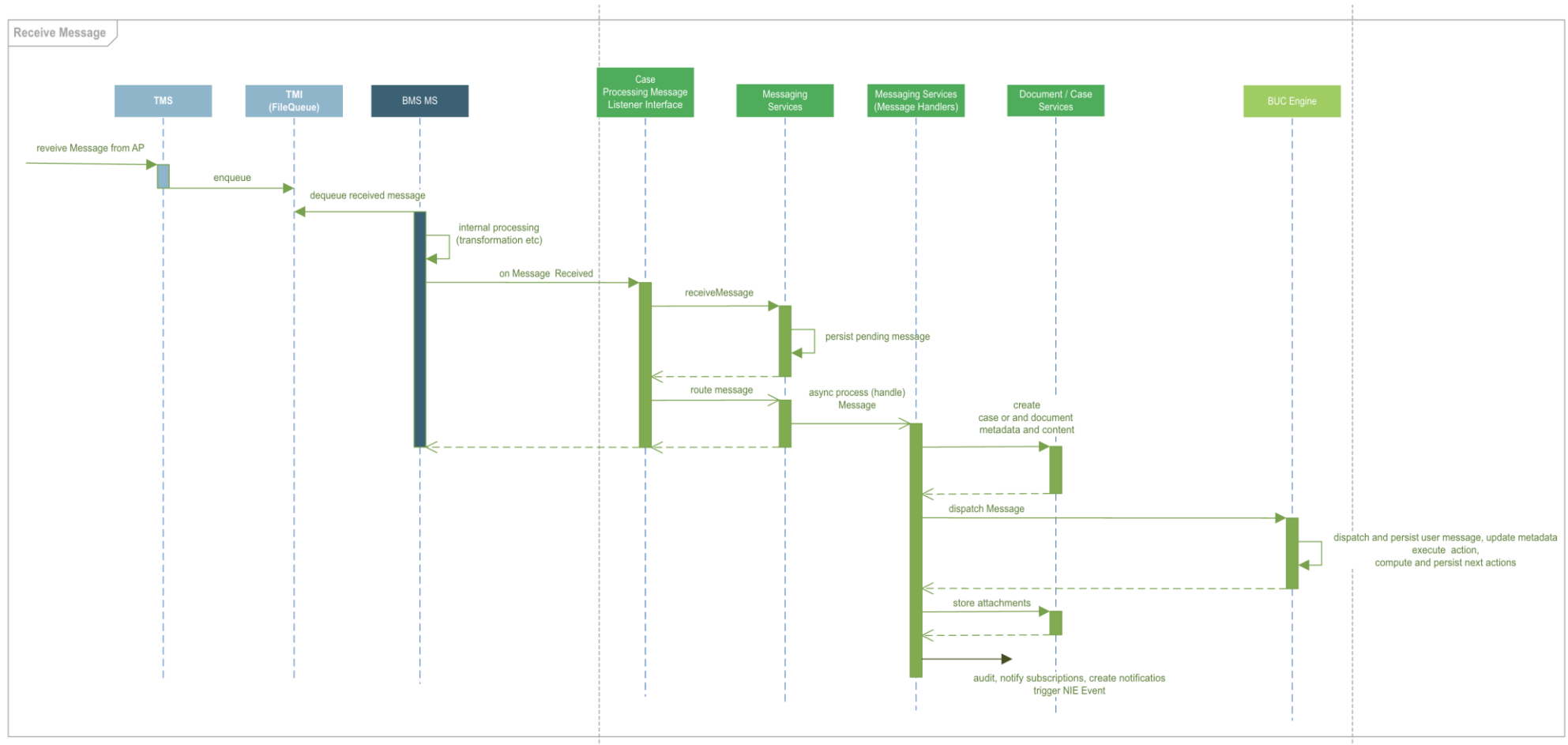


Figure 10: Receive Message Sequence Diagram

7. RINA Audit Trail

The audit trail is a sequence of chronological records of EESSI RINA system or user related events to enable the investigation of different type of incidents. These (business) events belong to the case management, message exchange and the 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 principles/rules applied in RINA to capture and handle audit trail data, the audit trail architecture and its components, as well as an overview on the data model.

7.1 Audit Trail principles

This section encompasses the auditing principles framing RINA's audit trail architecture :

- The audit trail must include the full extent of business auditing (information of adequate context related to business purposes).
- Auditing operations are transactional, and does not allow completion of other operations if auditing fails (for critical areas).
- The Auditing data is persisted, being stored in DB.

All RINA audit trail architecture fully complies with these 3 principles.

7.2 Audit Trail architecture

In the multi layered RINA, CPS and BMS components are participating in the overall audit trail architecture, as they are involved in the most business significant operations. Both are sharing a common audit model, being responsible for different areas and type of events.

The diagram below provides the audit trail mechanism of those components.

A high level description is following to present the reader with the different ways the audit trail is persisted and can be retrieved per different component.

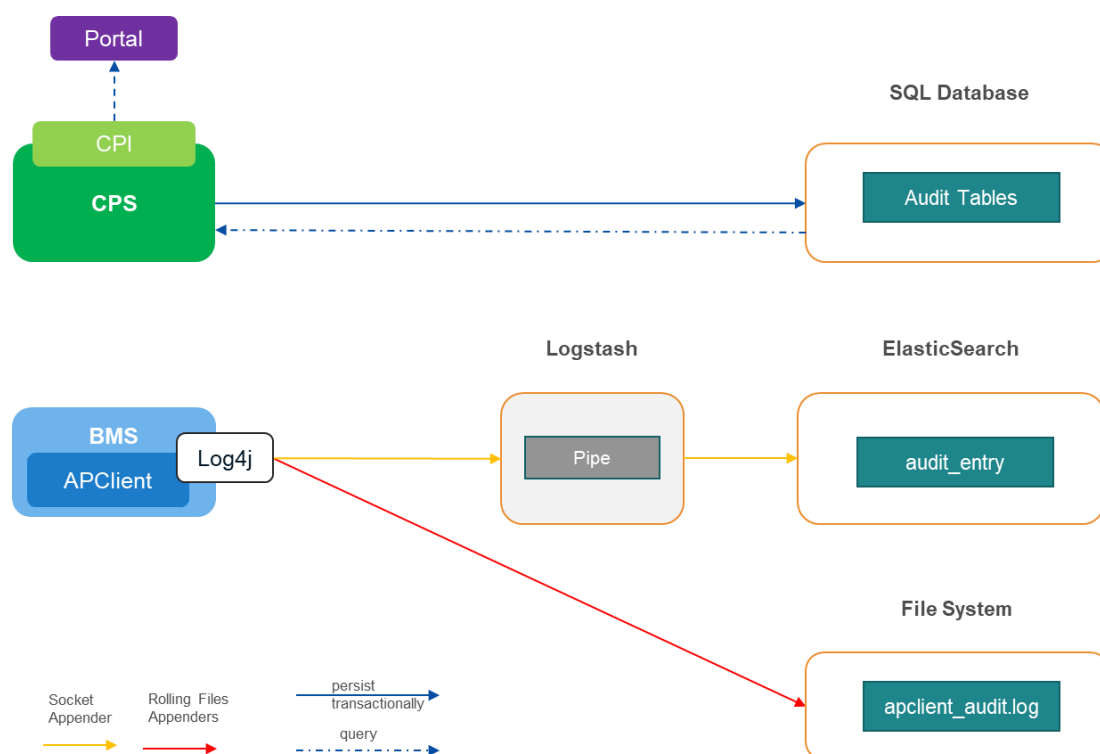


Figure 11: RINA Audit Trail architecture

7.2.1 CPS and Audit Trail

Case Processing Services, as the component responsible for the most user business operations, produce and store the audit logs (events) into the PostgreSQL Database ("*EESSI - RINA 6.2.1 - SQL DB Schema - Physical Data Model [R22]*").

The generation of the audit records is performed as part of the same business transaction with the triggering operation. This ensures that every auditable operation will always be audited either this operation has a successful outcome or not.

The audit events of CPS can be retrieved by CPI or Portal. Admin users have access to all the events produced by CPS whereas normal business users have access to events that are related to the resources they has sufficient audit access (cases, documents etc.).

More details about the CPI endpoints can be found in the "*EESSI - RINA - Case Processing Interface (CPI) - Reference [R12]*".

7.2.2 BMS and Audit Trail

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 [R17]*").

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

7.3 Audit Trail Data Model

CPS and BMS, like mentioned before, use the same Audit Trail Data model to enable consistency and easier aggregation.

Those components serve separate functions or the same functions on different layers, and therefore they end up populating audit records with different values (always of the same model).

In this section we aim to provide the full set for both components, whereas in CPI and BMI documentation the specific distinction will be highlighted

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.

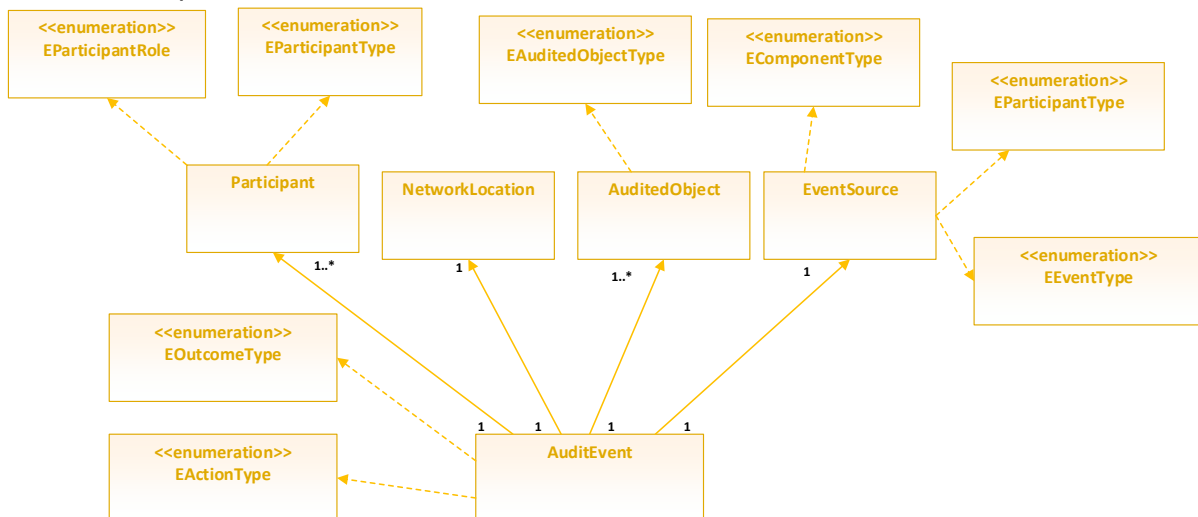


Figure 12: RINA Audit Data Model

Regarding the different domains of the Audit Model, there are two different physical representations

- SQL Tables for CPS in PostgreSQL.
- JSON Document for CPI (CPS component layer), BMS (ApClient component) in Elasticsearch/Audit File.

7.3.1 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
outcomeDetails	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 SQL for CPS component layer, under File/ES for BMS component layer) – hence not available through the CPI or Admin Portal.

7.3.2 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	In case of error the object triggered by REST shall be recorded here	String

7.3.3 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

7.3.4 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

7.3.5 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
Role	The role of participant in a triggered event/audited object (i.e. sender, receiver, subject)	EParticipantRole

7.3.6 EActionType enumeration values and description

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)

Following values are supported for the action type and can be used to filter the messages by the CPI Endpoint or Portal.

- Read
- Create
- Update
- Delete
- Execute

7.3.7 EAuditedObjectType enumeration

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

In case of a triggered event which involves actions on more types for each audited object type an audited object will be appended to the audited object list (e.g. for an event which will consist in submitting and attachment to a case, there will be two audited objects: one for the attachment and one for the case).

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
attch	ATTACHMENT	Used when the triggered event involves an action pertaining to an attachment (i.e. submit/ delete/ retrieve an attachment to a case or document)
cas	CASE	Used when the triggered event involves also an action pertaining to a case
cmt	COMMENT	Used when the triggered event involves an action pertaining to a comment (i.e. submit or delete a comment to/from a case or document)
alarm	ALARM	Used when the triggered event involves an action regarding an alarm (e.g. setting or clearing and alarm)

doc	DOCUMENT	Used when the triggered event involves an action pertaining to a document/SED (e.g. actions like submit/retrieve documents, actions like submitting/ deleting/ etc. attachments or comments to a document, etc.)
sdoc	SUBDOCUMENT	Used when the triggered event involves an action pertaining to a subdocument (e.g. actions like submit/retrieve subdocuments, actions like submitting/ deleting/ etc...)
notf	NOTIFICATION	Used when the triggered event involves an action pertaining to a notification (i.e. retrieve notification details/ summary of notifications)
srch	SEARCHDEFINITION	Used when the triggered event involves an action pertaining to a search definition (i.e. create, update, delete, or use of a search definition when search case by search definition)
usgr	USER_GROUP	Used when the triggered event involves an action pertaining to a user or group (i.e. user/group creation / deletion)
uspr	USER_PROFILE	Used when the triggered event involves an action pertaining to a user profile (i.e. user profile update)
appr	APPLICATION_PROFILE	Used when the triggered event involves an action pertaining to the application profile
cred	CREDENTIAL	Used when the triggered event involves a login or logout actions
poly	POLICY	Used when the triggered event involves an action pertaining to an authorisation policy (e.g. actor assignment)
bmsg	BUSINESS_MESSAGE	Used when the triggered event involves an action pertaining to a business message
tmsg	TECHNICAL_MESSAGE	Used when the triggered event involves an action pertaining to a technical message (ebMS 3 received or sent user message)
ackm	BUSINESS_ACK	Used when the triggered event involves an action pertaining to an acknowledgement technical message (ebMS3 received or sent signal acknowledgement message)
errm	BUSINESS_ERROR	Used when the triggered event involves an action pertaining to an error technical message (ebMS3 received or sent signal error message)

Following values are supported for the object type and can be used to filter the messages by the CPI Endpoint or Portal.

- Notification
- Notification Type
- Credential
- Attachment
- Comment
- Alarm
- Subdocument
- Business Message
- Search Definition
- Case
- User Profile
- Policy
- Document
- Application Profile
- User Group

7.3.8 ECategoryType enumeration values and description

The category type is used in order to specify the category of an event source. This can be one of the following: messaging, security or business and are corresponding to RINA layers. The presentation layer is out of scope for the audit trail.

Attribute	Name	Description
mess	MESSAGING	Used for event source when the triggered action involves messaging events/messaging layer
sec	SECURITY	Used for event source when the triggered action involves security events (e.g. actions impacting the users, group, credentials, policies etc.)
bus	BUSINESS	Used for event source when the triggered action involves business events/business layer

Following values are supported for the category types and can be used for filtering messages by the CPI Endpoint or Portal.

- Security
- Business
- Messaging

7.3.9 EComponentType enumeration values and description

Defines the component types relevant for an event source.

Value	Name	Description
Attch	ATTACHMENTS	Will be chosen if the triggered event involves the attachments component
Cas	CASES	Will be chosen if the triggered event involves the cases component
cmt	COMMENTS	Will be chosen if the triggered event involves the comments component
doc	DOCUMENTS	Will be chosen if the triggered event involves the documents component
notf	NOTIFICATIONS	Will be chosen if the triggered event involves the notifications component
srch	SEARCH_DEFINITIONS	Will be chosen if the triggered event involves the search definition component
Sec	SECURITY	Will be chosen if the triggered event involves the security component
admin	ADMINISTRATION	Will be chosen if the triggered event involves the administration component
bmsg	BUSINESS_MESSAGING	Will be chosen if the triggered event involves the business messaging component
tmsg	TECHNICAL_MESSAGING	Will be chosen if the triggered event involves the technical messaging component

Following values are supported for the component types and can be used for filtering messages by the CPI Endpoint or Portal.

- Notifications
- Security
- Search Definitions
- Cases
- Documents
- Administration
- Attachments
- Comments

7.3.10 EEventType values and description

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

Value	Name	Description
sub_attch_on_cas	SUBMIT_ATTACHMENT_ON_CASE	An event generated during submit attachment on a case
del_attch_frm_cas	DELETE_ATTACHMENT_ON_CASE	An event generated during delete attachment of a case
retr_attch_of_cas	RETRIEVE_ATTACHMENT_ON_CASE	An event generated during retrieve attachment of a case
sub_attch_on_doc	SUBMIT_ATTACHMENT_ON_DOCUMENT	An event generated during submit attachment on a document
retr_attch_of_doc	RETRIEVE_ATTACHMENT_ON_DOCUMENT	An event generated during retrieve attachment of a document
del_attch_of_doc	DELETE_ATTACHMENT_ON_DOCUMENT	An event generated during delete attachment of a document
srch_cas_by_def_and_or_txt	SEARCH_CASES_BY_SEARCH_DEFINITION_AND_OR_FREE_TEXT	An event generated during search cases based on definition and/or free text
cre_new_cas	CREATE_NEW_CASE	An event generated during create new case
retr_cas_by_id	RETRIEVE_CASE_BY_ID	An event generated during get a case by its local Case id
retr_cas_by_international_id	RETRIEVE_CASE_BY_INTERNATIONAL_ID	An event generated during get the details of a case by its international id
retr_cas_id_by_international_id	RETRIEVE_CASE_ID_BY_INTERNATIONAL_ID	An event generated during get a local case id by its international id
set_alarm	SET_ALARM	An event generated when a new alarm is created
clr_alarm	CLEAR_ALARM	An event is generated when an alarm is cleared

retr_cas_by_busi ness_id	RETRIEVE_CASE_BY_BUSINESS_ID	An event generated during get the details of a case or the local Case Id by its business id
assig_cas	ASSIGN_CASE	An event generated during assign a case
ar_un_cas	ARCHIVE_UNARCHIVE_CASE	An event generated when archive or unarchive
upd_cas_sensitive	UPDATE_CASE_SENSITIVE	An event generated when the 'sensitive' flag in the case metadata has been updated.
retr_cas_hash_by _id	RETRIEVE_CASE_HASHCODE_BY_ID	An event generated during get the case hash code of the case by the case id
imp_sdoc	IMPORT_BATCH	An event generated during import of the subdocuments
exp_sdoc	EXPORT_BATCH	An event generated during exporting the subdocuments
retr_cas_assig	RETRIEVE_CASE_ASSIGNMENTS	An event generated during get case assignment
sub_new_cmt_on _cas	SUBMIT_COMMENT_ON_CASE	An event generated during submit a new comment on a case
del_cmt_of_cas	DELETE_COMMENT_ON_CASE	An event generated during delete a comment of a case
sub_new_cmt_on _doc	SUBMIT_COMMENT_ON_DOCUMENT	An event generated during submit a new comment on a document
del_cmt_of_doc	DELETE_COMMENT_ON_DOCUMENT	An event generated during delete a comment of a document
retr_init_doc_of_ cas	RETRIEVE_INITIAL_DOCUMENT	An event generated during retrieve initial Document of an action. An event when retrieving

		the Initial Document Content or the Admin Document Content. That covers admin case operations i.e.: <i>Close, Reopen, Delete case, Local Close / Local Reopen, Send Participants, Select Participants, Read Participants, Update Participants, Request Approval, Remove Subdocument</i>
cre_doc	CREATE_DOCUMENT	An event generated when submitting the Document first time
upd_doc	UPDATE_DOCUMENT	An event generated when submitting the Document subsequently
del_doc	DELETE_DOCUMENT	An event generated when deleting the Document
snd_doc	SEND_DOCUMENT	An event generated when sending the Document to counterparties
sub_doc	SUBMIT_DOCUMENT	An event when submitting the Document Content or the Admin Document Content. That covers admin case operations i.e. <i>Close, Reopen, Delete case, Local Close, Local Reopen, Send Participants, Select Participants, Read Participants, Update Participants, Request Approval, Remove Subdocument</i>
retr_doc	RETRIEVE_DOCUMENT	An event generated during retrieve Document

retr_thumb	RETRIEVE_THUMBNAIL	An event generated during retrieve thumbnail
login	LOG_IN	An event generated during login
logout	LOG_OUT	An event generated during logout
retr_notf_details	RETRIEVE_NOTIFICATIONS_DETAILS	An event generated during retrieve the notifications details
retr_cnsld_summ_notf_usr	RETRIEVE_NOTIFICATIONS_CONSOLIDATED_SUMMARY	An event generated during retrieve the consolidated summary of notifications for a user (errors, warnings, information)
retr_summ_notf_usr	RETRIEVE_NOTIFICATION_SUMMARY	An event generated during retrieve the summary of notifications for a user
retr_tsno_and_ntf_on_ts	RETRIEVE_NOTIFICATION_TIME_SLOTS	An event generated during retrieve the time slots and the number of notifications on each time slot
upd_notf	UPDATE_NOTIFICATION	An event generated during update a notification
cre_new_srch_def	CREATE_SEARCH_DEFINITION	An event generated during create a new Search Definition
upd_spec_srch_def	UPDATE_SEARCH_DEFINITION	An event generated during update a specific Search Definition
del_spec_srch_def	DELETE_SEARCH_DEFINITION	An event generated during delete a specific Search Definition
upd_usr_prof	UPDATE_USER_PROFILE	An event generated during update the User Profile

upd_app_prof	UPDATE_APPLICATION_PROFILE	An event generated during update the Application Profile
usr_gr_cre	CREATE_USER_GROUP	An event generated during user/group creation
usr_gr_del	DELETE_USER_GROUP	An event generated during user/group deletion
auth_pol_chg	CHANGE_AUTHORIZATION_POLICY	An event generated during authorisation policy change
send_bmsg	SEND_BUSINESS_MESSAGE	An event generated during 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 generated during notify a register listener about a received business message This record exists in BMS ApClient audit log.
notif_reg_lsn_status_bmsg	NOTIFY_ABOUT_STATUS_UPDATE	An event generated during notify a register listener about a status update of a business message (ack or error) This record exists in BMS ApClient audit log.
send_tmsg	SEND_TECHNICAL_MESSAGE	An event generated during send a technical message to a preconfigured EESSI access point using the Holodeck AS4 (user message, pull request, ack., error)
rec_tmsg	RECEIVE_TECHNICAL_MESSAGE	An event generated during receive a technical

		message from a preconfigured EESSI access Point using the Holodeck AS4 (user message, Ack., error)
app_start	APPLICATION_START	An event generated during the start of the application
app_stop	APPLICATION_END	An event generated during the stop of the application
rec_new_cas	RECEIVE_NEW_CASE	An event generated when a new Case is received
rec_bmsg	RECEIVE_BUSINESS_MESSAGE	An event is generated when new Business Message is received on CPS component layer
pr_rec_bmsg	PROCESS_RECEIVED_MESSAGE	An event is generated when a received Business Message is processed on CPS component layer

Following values are supported for the event types and can be used for filtering messages, by the CPI Endpoint or Portal.

- Submit Attachment On Document
- Create Document
- Import Subdocument Batch
- Create Search Definition
- Logout
- Delete Attachment On Document
- Export Subdocument Batch
- Application Start
- Submit Document
- Change Authorization Policy
- Retrieve Notifications Consolidated Summary
- Update Document
- Send Document
- Submit Document
- Update Search Definition
- Retrieve Initial Document
- Set Alarm
- Delete Attachment on Case

- Retrieve Case By Id
- Create User Or Group
- Send Business Message
- Retrieve Attachment On Document
- Submit Attachment on Case
- Retrieve Case Hash Code By Id
- Delete Comment On Document
- Retrieve Notification Time Slots
- Retrieve Case Assignments
- Login
- Submit Comment On Document
- Update User Profile
- Retrieve Attachment on Case
- Clear Alarm
- Update Application Profile
- Delete Document
- Retrieve Notification Summary
- Retrieve Document
- Delete Comment On Case
- Notify About Received Business Message
- Submit Comment On Case
- Assign Case
- Send Technical Message
- Retrieve Notification Details
- Delete User Or Group
- Notify About Status Update
- Create New Case
- Retrieve Thumbnail
- Application End
- Delete Search Definition
- Search Cases By Search Definition And Or FreeText
- Update Notification
- Receive New Case
- Receive Business Message (only CPI)
- Process Received Message (only CPI)

7.3.11 EOutcome Type enumeration values and description

The outcome type is an audit event attribute which specifies 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 HTTP 500 type of exception are occurring
rjct	UNAUTHORIZED	Used mainly when HTTP 401 & HTTP 403 types of exceptions are occurring

7.3.12 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

Following values are supported for the participant role and can be used for filtering the messages:

- Receiver
- Sender
- Subject

7.3.13 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	ORGANISATION	Used if the audited event participant is non-person.

Following values are supported for the participant type and can be used for filtering the messages:

- Organization
- Person

7.3.14 Audit events types and audited Objects

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

Category	Component	Event	Audited Objects
Business	#ATTACHMENTS: operations on Attachments	SUBMIT_ATTACHMENT_ON_CASE	➤ Attachment ➤ Case
		DELETE_ATTACHMENT_ON_CASE	➤ Attachment ➤ Case
		RETRIEVE_ATTACHMENT_ON_CASE	➤ Attachment ➤ Case
		SUBMIT_ATTACHMENT_ON_DOCUMENT	➤ Attachment ➤ Case ➤ Document
		RETRIEVE_ATTACHMENT_ON_DOCUMENT	➤ Attachment ➤ Case ➤ Document
		DELETE_ATTACHMENT_ON_CASE	➤ Attachment ➤ Case ➤ Document
		DELETE_ATTACHMENT_ON_CASE	➤ Attachment ➤ Case ➤ Document
Business	#CASES: operations on Cases	SEARCH_CASES_BY_SEARCH_DEFINITION_AND_OR_FREE_TEXT	➤ Search text ➤ SearchDefinition Id (Optional)
		CREATE_NEW_CASE	➤ Case
		RECEIVE_NEW_CASE	➤ Case
		RETRIEVE_CASE_BY_ID	➤ Case
		UPDATE_CASE_SENSITIVE	➤ Case
		ASSIGN_CASE	➤ Case
		RETRIEVE_CASE_HASHCODE_BY_ID	➤ Case
		RETRIEVE_CASE_ASSIGNMENTS	➤ Case
Business	#COMMENTS: operations on Comments	SUBMIT_COMMENT_ON_CASE	➤ Case ➤ Comment
		DELETE_COMMENT_ON_CASE	➤ Case ➤ Comment
		SUBMIT_COMMENT_ON_DOCUMENT	➤ Case ➤ Document ➤ Comment
		DELETE_COMMENT_ON_DOCUMENT	➤ Case ➤ Document ➤ Comment
		DELETE_COMMENT_ON_DOCUMENT	➤ Case ➤ Document ➤ Comment

Business	# DOCUMENTS (incl. thumbnail): operations on Documents	RETRIEVE_INITIAL_DOCUMENT	➤ Case
			➤ Action (id and name)
			➤ Document (Optional)
		SUBMIT_DOCUMENT	➤ Case
			➤ Action (id and name)
			➤ Document
		RETRIEVE_DOCUMENT	➤ Case
			➤ Document
		RETRIEVE_THUMBNAIL	➤ Case
			➤ Document
Business	# MESSAGING Operations on Messages	RECEIVE_BUSINESS_MESSAGE	➤ Case (internationalId)
		PROCESS_RECEIVED_BUSINESS_MESSAGE	➤ Document
		SEND_TECHNICAL_MESSAGE	➤ BusinessMessage
			➤ Case
			➤ Document
		NOTIFY_ABOUT_RECEIVED_BUSINESS_MESSAGE	➤ BusinessMessage
			➤ Case
			➤ Document
		NOTIFY_ABOUT_STATUS_UPDATE	➤ BusinessMessage
			➤ Case
			➤ Document
Security	# SECURITY: operations on login	LOG_IN	➤ Credential
		LOG_OUT	➤ Credential
		CREATE_USER_GROUP	➤ Credential
		DELETE_USER_GROUP	➤ Credential
		UPDATE_USER_GROUP	➤ Credential
		CHANGE_AUTHORIZATION_POLICY	➤ Policy
Business	# NOTIFICATIONS: operations on Notifications	RETRIEVE_NOTIFICATIONS_DETAILS	➤ Notification (Ids)
			➤ Case (Optional)
		RETRIEVE_NOTIFICATIONS_CONSOLIDATED_SUMMARY	➤ N/A
		RETRIEVE_NOTIFICATION_SUMMARY	➤ N/A
		RETRIEVE_NOTIFICATION_TIME_SLOTS	➤ N/A

		UPDATE_NOTIFICATION	➤ Notification
Business	#	CREATE_SEARCH_DEFINITION	➤ SearchDefinition
	SEARCH_DEFINITIONS: operations on SearchDefinitions	UPDATE_SEARCH_DEFINITION	➤ SearchDefinition
		DELETE_SEARCH_DEFINITION	➤ SearchDefinition
Business	# admin: operations on USER_PROFILE and APPLICATION_PROFILE	UPDATE_USER_PROFILE	➤ UserProfile
		UPDATE_APPLICATION_PROFILE	➤ ApplicationProfile
Security	#Admin	APPLICATION_START	➤ ApplicationProfile
		APPLICATION_STOP	➤ ApplicationProfile

Notes: The Id of the Participant in case of role Subject and type Person takes the value of the PIN of the case subject person, and if this does not exist the First Name and Last Name.

7.4 Audit model CS DB Domain representation

This section aims to provide a very short view of the CPS Repository (PostgreSQL) schema part that is responsible to keep the audit events in CPS. The purpose is to highlight that the same audit model is used, but with the necessary changes that the SQL physical representation entails.

The schema consists of three tables:

- *audit_event*
- *audit_object*
- *audit_participant*.

Each row of *audit_event* table has one to many relationships to the other two tables. No foreign key relationships are used to any other CPS data, to ensure that audit records remain immutable to any future data change.

The mapping with the logical data model is self explanatory by the name of each column. Note that *created_at* column of *audit_event* table represents the date of the audit event. Furthermore, the SQL Domain contains also the information about the tenant Id that is related to the audited action.

The possible values are the same like described in the previous sections. Note that the enumerations are persisted with their name, rather than their value. (i.e. *SUCCESS* for *outcome_type*)

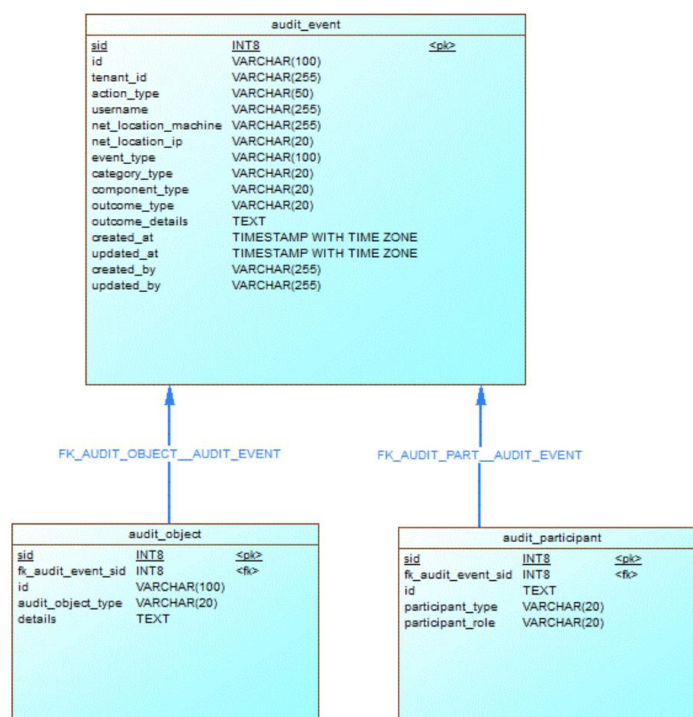


Figure 13: RINA Audit Model CS DB Domain Representation

7.5 NA and Audit Trail

National Applications could also provide their own audit services to unlock custom processing of audit trail according to national specific needs.

Such systems could query and aggregate the audit events from CPI (or directly PostgreSQL) and Elasticsearch (or log files).

Audit trail model, as explained above, is mostly focused to provide references to the audited resources and objects.

If any more information is required by the National Implementation Services, those references shall suffice to use other RINA Interfaces or directly repositories in order to retrieve more detailed data. It is not suggested that technical logs are used for this purpose, as they have different scope and also do not have a strict message data structure and contract.

8. RINA Technical Logging

The technical logging is another fundamental part of RINA, particularly important for assisting with information in case of technical flaws, performance issues or security violation purposes.

This section aims to provide the reader with an overview of the distributed logging architecture of RINA, depicting all the components generating logs and how they are handled within the system.

It also contains the principles/rules (including the logging levels) applied in RINA to capture and handle technical logging data as well as an overview of logging level context.

Finally, it lists all the log files generated per component together with their location in the file system.

8.1 Logging architecture (default)

The RINA Technical logging is leveraging the log4j infrastructure as the baseline for the generation, configuration and persistence of technical logs.

All different component under CPS, BMS layers (including CAS module) are generating log data and participate in the logging architecture. Namely:

- CPS
- BUC Engine
- Logstash
- BMI
- ApClient
- CAS
- Holodeck

In addition, there are other 2 RINA components which also generate log information, but using their own logging functionality:

- Load Balancer
- PostgreSQL
- ElasticSearch

By default the persistence of the logs for each different component is performed in files via log4j FileAppenders under separate dedicated locations. (dedicated repository for each component).

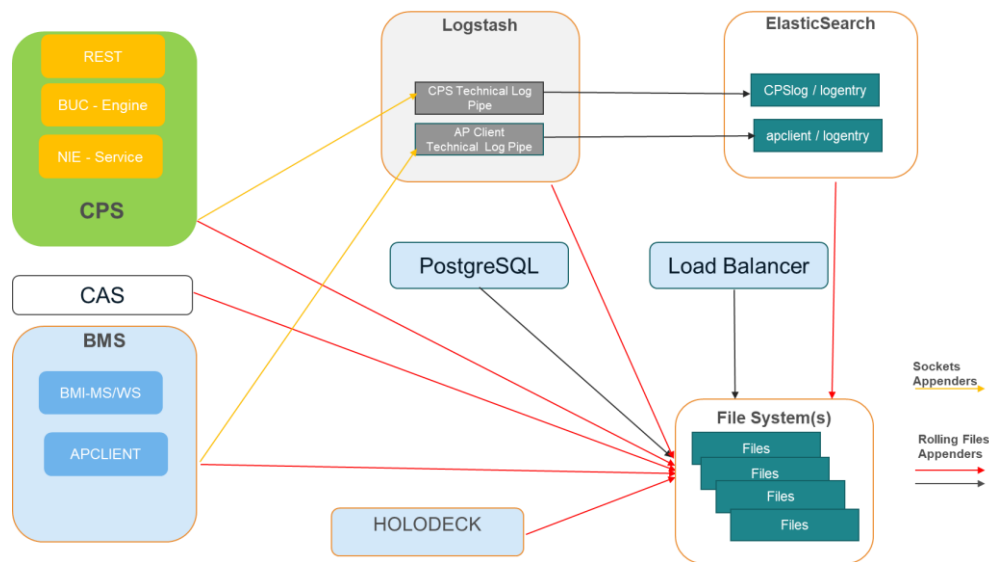


Figure 14: RINA Logging Architecture

In addition, RINA is providing by default, an aggregation configuration of the most important component logs. This is used by log4j socket appenders, Logstash and Elasticsearch database.

In more detail, components deployed under CPS and BMS, are additionally redirecting the log data via sockets to Logstash that will finally persist them in the Elasticsearch.

Logstash is transforming the technical logs in such a way that they are persisted in Elasticsearch using the same data model that is Syslog compatible.

That enables CPS to be able to retrieve all the technical logs aggregated in Elasticsearch and provide query and filter capabilities via the CPI - and Portal -. (there is a dedicated endpoint associated to the admin user, as indicated in "EESSI - RINA - Case Processing Interface (CPI) - Reference [R12]" document).

Finally, log4j provides the capability to administrators to alter the default configuration and architecture. "EESSI - RINA - Operations Manual [R17]" provides information about the Log4j2 configuration.

8.2 Log Records context and content

Each log entry is consisted by several context information and the message content.

The context contains (among others):

- Log level (i.e. all, trace, debug, info, warn, error)
- Timestamp (date/time/timezone) that the log was created.
- Package, class name, method name, line number, thread name (or ID);
- User Id that triggered the operation if applicable (CPS component only)
- Tenant Id that triggered the operation if applicable (CPS component only)

Regarding the message content, this can vary, depending on the different triggering operations and their outcome, but certain principles are followed through RINA components to provide a more consistent approach to the log messages.

Last but not least, sensitive information (personal identification information (PII) falling under GDPR and passwords) is never stored in RINA log files.

More details on both logging & exception management principles and log trace structure can be found in the ANNEXES of this document.

8.3 Logging levels

A key element of the logging implementation is the definition of the different logging levels. The following table represents the ones used in the RINA application.

Logging level	Description
Trace	This is the fine-grained diagnostic level, serving for events which indicate internal state transitions in full detail. This serves the frequently repeatable actions such as in the cases of nested loops. It is only for development purposes.
Debug	Detailed information for development and occasionally for troubleshooting purposes. This is fine-grained diagnostic level.
Info	Used restrictively, and at key places in the code, e.g. beginning and end of major operations (method entry, method exit). It should log action related information and be accompanied - whenever applicable - by an ID of an associated action or business entity. This level serves for events which constitute major state changes within a software component -- such as initialization, shutdown, persistent resource allocation, etc. -- which are part of normal operations. Events on this level are typically reported for non-library components. Each software component should log at least four events on this level: • when it starts its initialization, • when it becomes operational, • when it starts orderly shutdown, • just before it terminates normally.
Warn	Applicable for potential problems. Occurs also when an exception is caught and handled gracefully without re-throwing another exception.
Error	Error not stopping application. Unexpected situations. They should be logged only once and only in the final place of catching

the exception. Usually occurs when exceptions cannot be handled gracefully.

Table 5: RINA Logging Levels

The criteria to choose between *info*, *debug* and *trace* log levels is the performance of the system and the readability of the log file. Very frequent logging may slow down the application and result in a bloated log file.

Calls of *warn* and *error* log levels are mostly found in conditionals or in the catch section of try-catch blocks, which restricts the frequency of their activation.

8.4 RINA log files

The following table provides an overview of the list of the generated log files in RINA, that comes with the default configuration, along with their description mapped with the corresponding component and server location:

Component	Log file name	Description	Location (Windows/Linux)
APClient	apclient.log	Contains dedicated logging about activities for the APClient	Windows - C:\EESSI\Tomcat\logs Linux - /eessi/tomcat/logs
CAS	cas.log	Contains dedicated logging about the activity for CAS	Win - C:\EESSI\Tomcat\cas\logs Linux - /eessi/tomcat/cas/logs
CPS	rest-api.log	Contains all the dedicated logging about the activity of the CPS layer including the CPI, BUC Engine and NIE.	Win - C:\EESSI\Tomcat\logs Linux - /eessi/tomcat/logs
BMIWS	aplistener.log rest-api.log	Contains all the dedicated logging about the BMIWS	Win - C:\EESSI\Tomcat\logs Linux - /eessi/tomcat/logs
LOGSTASH	restlog.log apclientlog.log audit.log	Contains all the dedicated logging activities for logstash	Win - C:\EESSI\Logstash\logs Linux - /eessi/logstash/logs
HOLODECK	holodeckb2b.log ebms_errors.log soap_in.log soap_out.log	Contains all the dedicated logging activities for Hollodeck	Win - C:\EESSI\HolodeckB2B\logs Linux - /eessi/holodeckb2b/logs
POSTGRES	postgresql.log	Contains all the dedicated logging activities for Postgres	Win - C:\EESSI\PostgreSQL\12\data\log Linux - /var/log/postgresql

LOAD BALANCER	access.log error.log	Contains all the Win - dedicated logging C:\EESSI\Apache\logs activities for the Linux - LoadBalancer /var/log/apache2
Tomcat	catalina.out	Contains generic Win - logging of the C:\EESSI\tomcat\logs application Linux - server/service /eessi/tomcat/logs

Table 6: RINA Log Files (with component and location mapping)

Additional clarifications regarding the Logging configuration can be obtained in the relevant sections of the "EESSI - RINA - Operations Manual [R17]".

ANNEX A. Technical Logging and Exception Management Principles in RINA

This section encompasses the principles framing RINA's technical logging architecture, to which all the components are compliant:

- The Technical content is at least persisted in the file system.
- The sensitive information (personal identification information (PII) falling under GDPR and passwords) must not be stored in log files.

i. Logging principles overview

General logging principles are as follows:

- The logger object is declared static and final in each class to ensure that every instance of a class shares a common logger object.
- The right level of logging all shall be used (i.e. all, trace, debug, info, warn, error)
- Use meaningful log messages that are relevant to the context (containing following aspects), such as:
 - IDs;
 - Business related information should be omitted (or obfuscated if really needed but falling under GDPR restrictions).
- Information logged should not contain verbose volume of content and should be based as much as possible on self-explanatory data.
- Log format should contain:
 - Timestamp (date/time/timezone) should be included (preferably in ISO format);
 - Package (in the format of an acronym), class name, method name, line number, thread name (or ID);
 - User and tenant if defined in the session context;
 - Exception cause;
 - Exception stack (if in DEBUG level).
- Log at least the following events (also applicable at the level of relevance [public vs private method] and logging level):
 - Method entry (optionally with the method's input parameter values;
 - Method exit;
 - Root cause message of exceptions that are handled at the exception's origin point;
- Avoid logging at every place where a custom exception is thrown. Instead log the custom exception's message in its ultimate handler.

ii. Exception management principles

The exception management principles applicable for the business management interface are the following

- Do not catch generic "Exception" class (Exception, Throwable) and catch just the exceptions that you can handle;
- Never swallow exceptions in catch block. Log in this case an event in WARN level;
- Do not keep print stack trace in code;
- Do not handle exceptions inside loops;
- Never throw exception from finally block;
- Always wrap exception in custom exception;
- Avoid too many checked exception in method declaration; Use them preferably only for procedural algorithms and for static methods of utility classes.
- Never throw exceptions at the GUI level. Instead use exception handler to throw API errors:
 - Provide references to the log if possible. As a reference, the technical error shown in the GUI should contain the same IDs as the errors persisted in the logs.

iii. Exception logging principles

The exception management principles applicable for the business management interface are the following:

- Handle Exceptions close to its origin (Log and handle exception at the same level)
 - Does NOT mean — catch and swallow (i.e. suppress or ignore exceptions);
 - Use the specific exception type to differentiate exceptions and handle exceptions in some explicit manner.
- Log and throw the exception up the method call stack using a custom exception relevant to that source layer and let it be handled later by a method up the call stack:
 - Allow the creation of groups of exceptions and handling exceptions in a generic manner;
 - When catching an exception and throwing it using an exception relevant to that source layer, make sure to use the construct that passes the original exception's cause. This will help preserve the original root cause of the exception;
- Log Exceptions just once and log it at the ultimate place of 'catch':
 - Logging the same exception stack trace more than once can confuse the programmer examining the stack trace about the original source of exception;
 - There is an exception to this rule, in case of existing code that may not have logged the exception details at its origin. In such cases, it would be required to log the exception details in the first method up the call stack that handles that exception. But care should be taken NOT to COMPLETELY overwrite the original exception's message with some other message when logging.

ANNEX B. RINA default logging format

RINA has two log4j.xml files defining the data contracts to the file systems using appenders and three connectors to Logstash.

The first one declares the contract for the Rest-Api, APClient and the Audit log whereas the second one is used to define the contract for CAS.

It uses the log4j's RollingFile appenders together with a nested PatternLayout appender and PatterLayout appender to define these contracts. These appenders use the default settings of the TimeBasedTriggeringPolicy that causes a rollover once the date/time pattern no longer applies to the active file.

These appenders are depicted below:

- a) The first appender declare the contract for the Rest API. The file pattern and pattern Layout are declared as follow:

- filePattern="\${sys:catalina.home}/logs/rest-api-%d{MM-dd-yyyy}.log"
- pattern="%d{yyyy/MM/dd HH:mm:ss,SSS} [%-6p] [%t] %c{1.}.%M(%F:%L) [username=%X{username}, tenant=%X{tenant}] - %m%n"

where %d{yyyy/MM/dd HH:mm:ss,SSS} represents the date and time of the log

%-6p the level

%c is the class where the logger is invoked

And m* in the pattern represent the application supplied message.

Below is a snippet of the daily reviews log:

```
2020/11/27 23:45:00,076 [INFO ] [MessageBroker-3]
e.e.d.e.r.s.t.i.DailyReviewsServiceTx.archiveMessages(DailyReviewsServiceTx.java
:256) [username=supervisor, tenant=PLNA14] - Start archiving messages...
```

Note: The **username** and **tenant** log elements listed above have been introduced as a reference to logging that involves persisted content of operations associated to such material (i.e. *when user performs activities in the system, when a message is received by the system*). When applicable, they will be associated with the corresponding values; otherwise, they will remain empty.

- b) The second appender is used to define the contract for the APClient and file pattern and pattern layout are declared as follow:

- filePattern="\${sys:catalina.home}/logs/apclient-%d{MM-dd-yyyy}.log"
- pattern="%d{yyyy/MM/dd HH:mm:ss,SSS} [%-6p] [%t] %c{1.}.%M(%F:%L) - %msg%n"

- c) The last appender is in turn used to declare the contract for Auditlog which uses the same type of pattern

- filePattern="\${sys:catalina.home}/logs/apclient-audit-%d{MM-dd-yyyy}.log"

- `pattern="%d{yyyy/MM/dd HH:mm:ss,SSS} [%-6p] [%t] %c{1.}.%M(%F:%L)
- %msg%n"`

The connectors use exactly the same file pattern but using the RFC5424 Layout instead of the pattern layout with key pair value for the priority, stack trace, thread and logger name.

The second log4j file declares the contract for CAS and uses similar appenders but with size based triggering policy of 10MB