

domibusConnector-4.1.0-RELEASE - Administration Guide

Table of contents

TABLE OF CONTENTS	2
1. INTRODUCTION	4
1.1. SCOPE AND OBJECTIVE OF THIS DOCUMENT	4
2. ARCHITECTURAL OVERVIEW	5
2.1. CONNECTOR IN THE E-CODEX ENVIRONMENT	5
2.2. CONNECTOR IN THE NATIONAL ENVIRONMENT	6
2.2.1. DOMIBUSCONNECTOR INTERFACES.....	6
2.3. DOMIBUSCONNECTOR COMPONENTS	8
2.3.1. DOMIBUCONNECTORAPI.....	9
2.3.2. DOMIBUSCONNECTORBACKENDLINK	9
2.3.3. DOMIBUSCONNECTORGATEWAYLINK	9
2.3.4. DOMIBUSCONNECTORCONTROLLER	9
2.3.5. DOMIBUSCONNECTOREVIDENCES TOOLKIT.....	10
2.3.6. DOMIBUSCONNECTORSECURITY TOOLKIT.....	10
2.3.7. DOMIBUSCONNECTORWEB.....	10
2.3.8. DOMIBUSCONNECTORSTARTER	10
2.4. INTERNAL QUEUES	10
2.4.1. CONFIGURING THE INTERNAL MESSAGE BROKER	11
2.4.2. QUEUE DESCRIPTION	11
3. CONFIGURATION.....	13
3.1. CONNECTOR.PROPERTIES.....	13
3.2. PROPERTY CONFIGURATION IN THE DOMIBUSCONNECTOR DATABASE	13
4. LOGGING	14
4.1. LOGGING LEVEL	14
4.2. LOGGER NAMES	14
4.3. BUSINESS LOGGING	14
4.4. FOLLOW LOGS OF A MESSAGE PROCESS	15
5. DOMIBUSCONNECTOR ADMINISTRATION UI.....	16
5.1. LOGIN	16
5.1.1. PRECONFIGURED USERS	16
5.2. STATIC INFORMATION AND MAIN MENU.....	17
5.2.1. USER INFORMATION	17
5.2.2. MAIN MENU	17
5.3. MESSAGES SECTION.....	17
5.3.1. MESSAGE SEARCH.....	18
5.3.2. ALL MESSAGES	18

5.3.3.	MESSAGE DETAILS	19
5.3.4.	MESSAGES REPORTS.....	20
5.4.	PMODES SECTION	21
5.5.	CONFIGURATION SECTION	22
5.6.	USERS SECTION.....	24
5.6.1.	ALL USERS	25
5.6.2.	USER DETAILS.....	25
5.6.3.	ADD NEW USER	26
6.	TASKS INVOLVING DATABASE ACCESS.....	27
6.1.	SEARCH FOR ERRORS.....	27
I.	LIST OF FIGURES	28
II.	LIST OF TABLES.....	29
III.	LIST OF REFERENCES.....	30

1. Introduction

1.1. Scope and Objective of this document

This document should give an introduction into administrating the connector and also give a short overview of the internals of the connector.

The document starts with the architectural overview which provides a brief overview of the connector internals. It also covers where the connector should be placed in an e-Codex environment.

Further on the administration guide is split into two parts. In the first part the possibilities of the web user interface are described. The next part covers tasks that need interaction with the connector database.

This guide does not cover things like configuring the database connection, the web application server or the gateway or connector client connections. For this information we refer to the document "domibusConnector_InstallationGuide" distributed together with this document.

2. Architectural Overview

2.1. Connector in the e-codex environment

In the e-Codex world the messages are transported as AS4 messages. The payload of this AS4 message are a business document (xml) and an ASIC-S container which contains all the other related documents (pdf, xml, ...). Also a trust token is generated which hides the national trust system for the other participants in the e-Codex environment. During the message transport ETSI-REM confirmations are generated. This confirmations or evidences are the proof that a message has been processed by a specific system.

The transportation and handling of AS4 messages are done by an AS4 gateway. In the e-Codex world usually the DOMIBUS gateway is used for that purpose.

The ASIC-S container handling, Trust-Token generation and the confirmation evidence handling are done by the connector. So the domibusConnector is a part of the toolchain to connect the national system with the international e-Codex world.

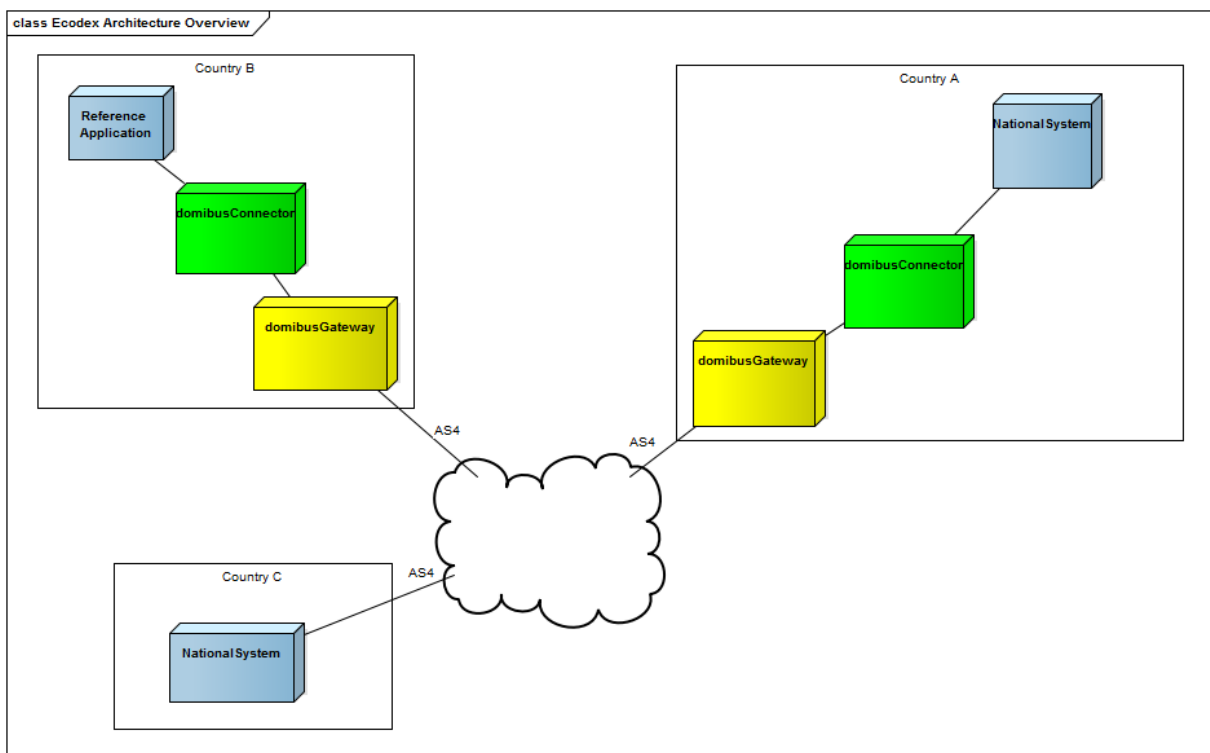


Figure 1: e-CODEX environment overview

2.2. Connector in the national environment

The connector is a web application. The connector has two sides. A backend side which is a national solution or a domibusConnectorClient, and the gateway side. Messages are always delivered from the gateway to the connector's backend. In the other direction the messages are submitted.

As Figure 2 shows the connector clients may use the domibusConnectorClient¹ to establish a connection to the connector or they can directly use the provided web service interfaces of the domibusConnector. The domibusConnector supports multiple clients.²

¹ The domibusConnectorClient is also available over nexus [7] and contains a detailed documentation

1.1.1. ² domibusConnectorEvidencesToolkit

This module is responsible for creating the ETSI-REM evidences. It is called by the domibusConnectorController.

1.1.2. domibusConnectorSecurityToolkit

This module is responsible for validation and creation of the ASIC-S container. It does all the work related to the ASIC-S container. For this purpose it makes use of the dss security library.

1.1.3. domibusConnectorWeb

This module holds the domibusConnector Administration UI.

1.1.4. domibusConnectorStarter

The starter has dependencies to all other modules. It provides the deployable web application (WAR) and controls the starting routine of the domibusConnector.

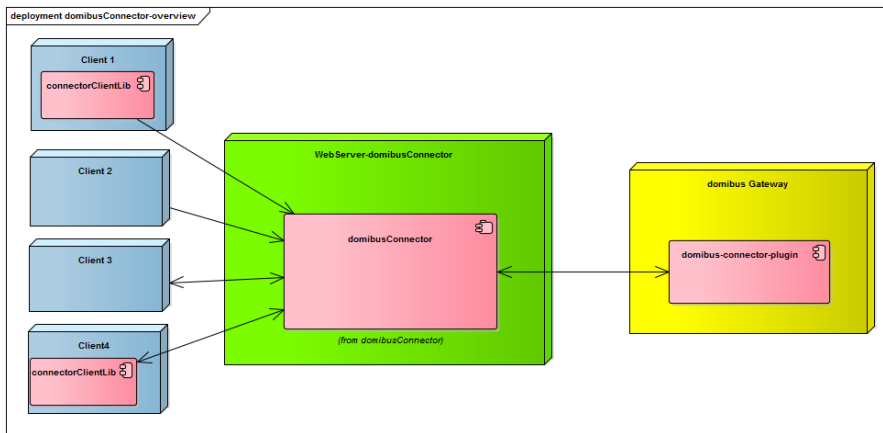


Figure 2: domibusConnector in the national environment

The Client 1 and Client 4 in this example use the domibusConnectorClient-library to establish the connection to the domibusConnector. The domibusConnectorClient-library is included as a jar

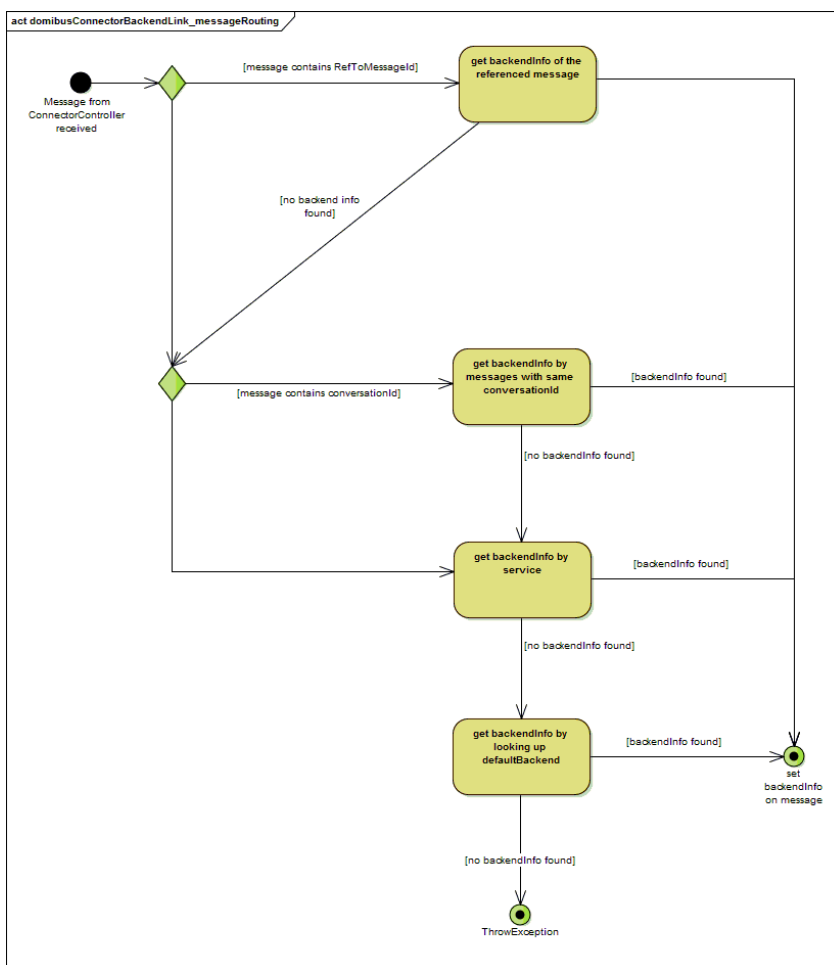


Figure 5 shows how the message routing is done

dependency inside the client's implementation. For further information refer to the distributed documentation of the `domibusConnectorClient`.

The Client 2 and Client 3 of this example address the web service interfaces of the `domibusConnector` directly, so there is no need for including any `domibusConnectorClient` jar dependency.

A `domibus-connector-plugin` installed at the DOMIBUS gateway is responsible for the communication between the `domibusConnector` and the gateway. This plugin is also available as a distribution on the Nexus and also contains documentation on usage and installation.

2.2.1. `domibusConnector` interfaces

The `domibusConnector` provides web service interfaces on both sides. These interfaces are specified using the WSDL-language. The specification can be downloaded over the e-Codex Nexus repository server³. Figure 3 gives an overview of these public interfaces. On the client side there are two variants possible, the push/push or the push/pull interface. The push/push interface also requires the client to provide a web service so the connector can push messages to the client.

³ <https://secure.e-codex.eu/nexus/content/repositories/releases/eu/domibus/connector/domibusConnectorAPI/4.1.0-RELEASE/domibusConnectorAPI-4.1.0-RELEASE-wsdl.zip>

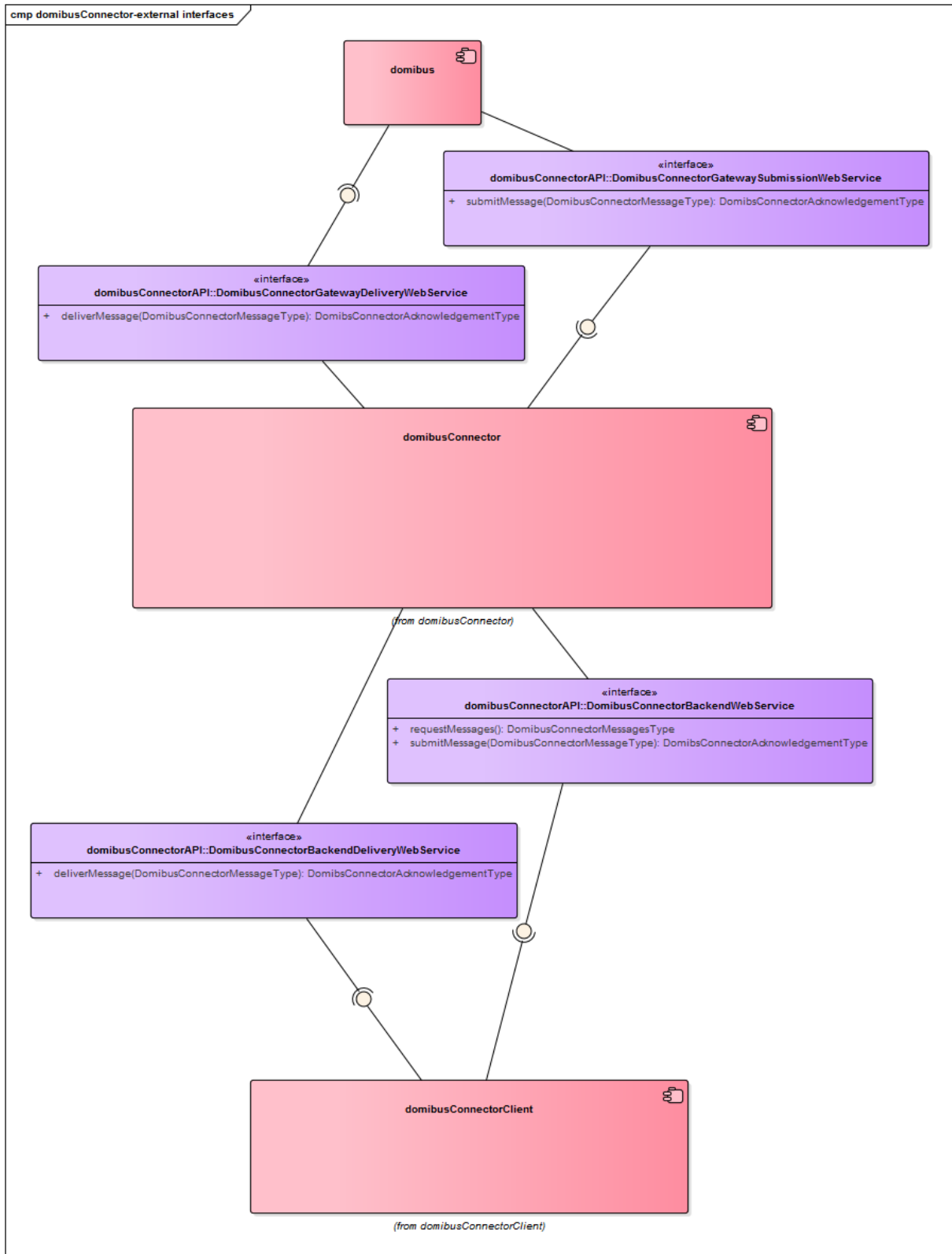


Figure 3: Connector Web Service Interfaces

2.2.1.1. DomibusConnectorGatewayDeliveryWebService

The domibusConnector implements this interface for message delivery from the gateway to the connector. This interface is implemented by the domibusConnectorGatewayLink (see chapter 2.3.3) module.

2.2.1.2. DomibusConnectorGatewaySubmissionWebService

The DomibusConnectorGatewaySubmissionWebService interface is the counterpart. It is implemented by the domibus-connector-plugin installed at the DOMIBUS gateway and allows the connector to push messages to the gateway.

2.2.1.3. DomibusConnectorBackendWebService

The DomibusConnectorBackendWebService interface is implemented by the domibusConnectorBackendLink module (see chapter 2.3.2) and provides two methods:

- requestMessages: allows the client to pull pending messages
- submitMessage: is called to submit a message from the client to the connector

2.2.1.4. DomibusConnectorBackendDeliveryWebService

For pushing messages to the client, the client has to implement the DomibusConnectorBackendDeliveryWebService which allows the connector to push messages to the client.

2.3. domibusConnector Components

The domibusConnector itself is a java web application. The application is developed as a multi module maven project. The project is separated into multiple modules (Figure 4: domibusConnector 4.1.0 simplified component overview). Each module has a specific purpose.

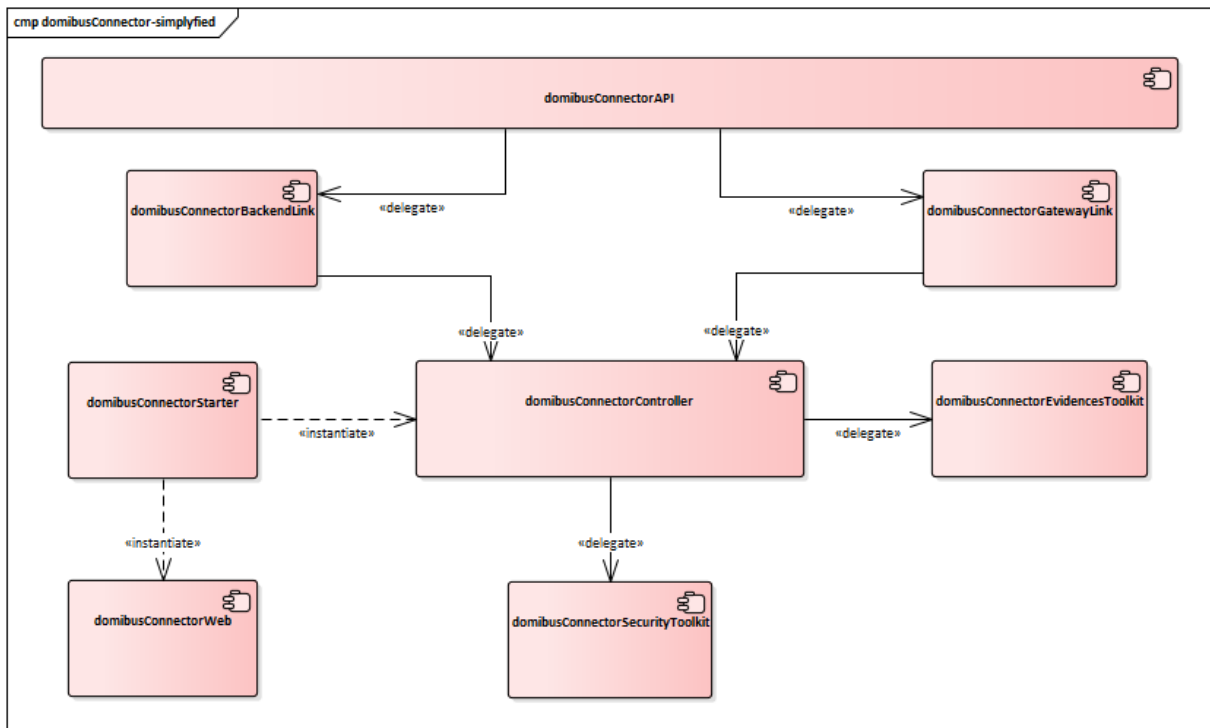


Figure 4: domibusConnector 4.1.0 simplified component overview

2.3.1. domibuConnectorAPI

Holds the public interface descriptions, the wsdl files, the web service policy and the Message structure object and definition.

2.3.2. domibusConnectorBackendLink

This module is responsible for the communication with the connectorClients (national system, backend, connectorClient). For this purpose a web service is created. For delivering messages a web service client calls the backend's web service and pushes the messages to the client. For pull clients the messages are stored until the client asks for them.

2.3.3. This module also determines the correct backend where the message is delivered to. domibusConnectorEvidencesToolkit

This module is responsible for creating the ETSI-REM evidences. It is called by the domibusConnectorController.

2.3.4. domibusConnectorSecurityToolkit

This module is responsible for validation and creation of the ASIC-S container. It does all the work related to the ASIC-S container. For this purpose it makes use of the dss security library.

2.3.5. domibusConnectorWeb

This module holds the domibusConnector Administration UI.

2.3.6. domibusConnectorStarter

The starter has dependencies to all other modules. It provides the deployable web application (WAR) and controls the starting routine of the domibusConnector.

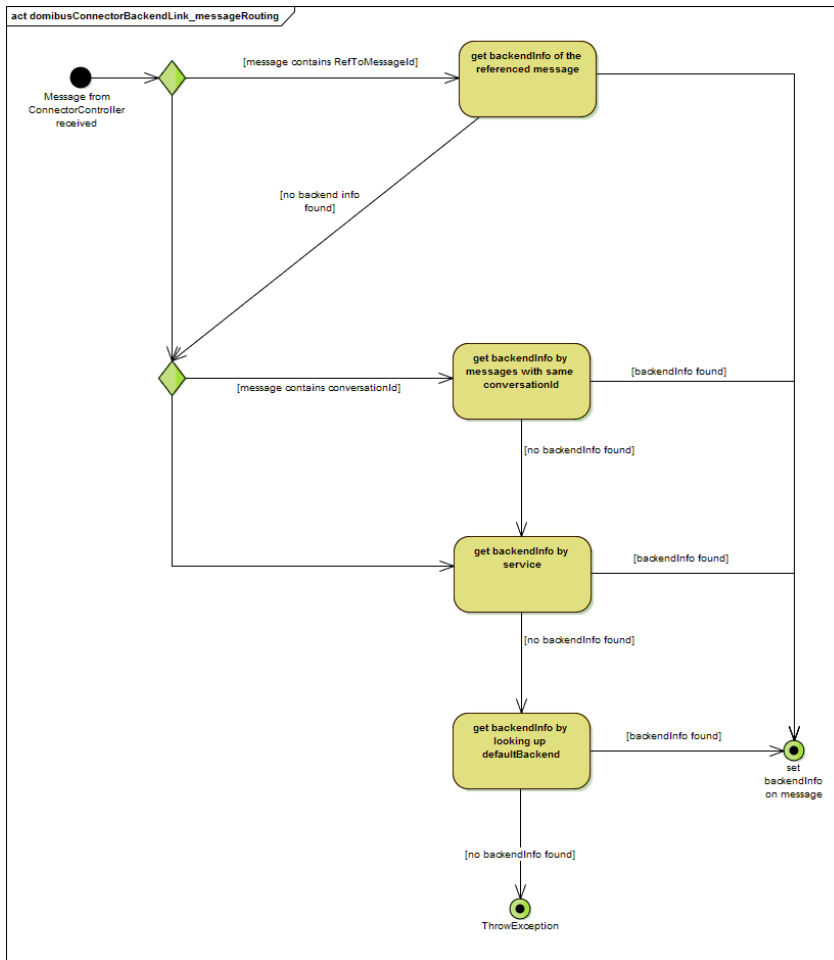


Figure 5: domibusConnectorBackendLink message routing shows an activity diagram how this is done.

2.3.7. domibusConnectorGatewayLink

This module is responsible for providing the web service for the gateway communication. It also submits the messages to the gateway for the connectorController. The current release only supports web services for transport, later releases may also support JMS.

2.3.8. domibusConnectorController

This module controls the message's workflow. It uses other modules for resolving the asic-s container, sending messages to the gateway or persisting messages. It is also responsible for time triggered jobs to cleanup the message storage and calculates evidence timeouts.

2.3.9. domibusConnectorEvidencesToolkit

This module is responsible for creating the ETSI-REM evidences. It is called by the domibusConnectorController.

2.3.10. domibusConnectorSecurityToolkit

This module is responsible for validation and creation of the ASIC-S container. It does all the work related to the ASIC-S container. For this purpose it makes use of the dss security library.

2.3.11. domibusConnectorWeb

This module holds the domibusConnector Administration UI.

2.3.12. domibusConnectorStarter

The starter has dependencies to all other modules. It provides the deployable web application (WAR) and controls the starting routine of the domibusConnector.

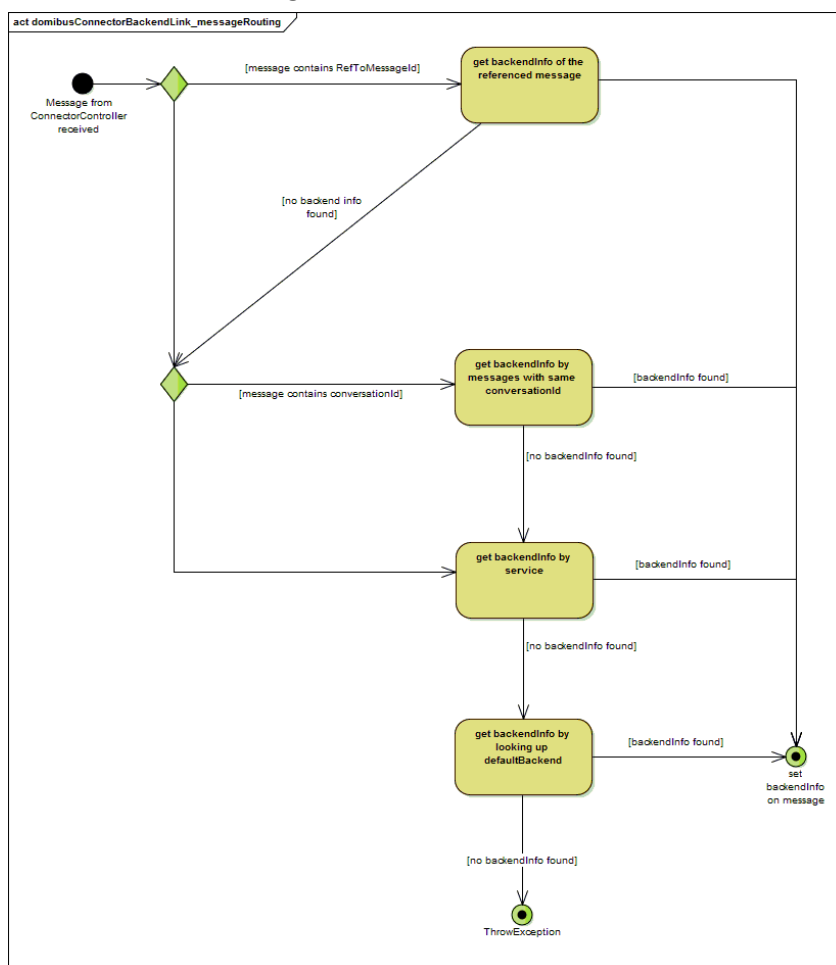


Figure 5: domibusConnectorBackendLink message routing

2.4. Internal Queues

The domibusConnector uses an internal queueing system for decoupling its components. This queueing system should also allow a vertical scalability and to improve failover in future releases. These queues are for internal connector use only and must not be used by anything else.

2.4.1. Configuring the internal message broker

The domibusConnector uses spring boot ⁴to start its own ActiveMQ⁵ message broker for these queues. So all properties used in spring boot can be used to configure this internal message broker. It is also possible to deactivate the internal message broker and configure an external broker.

spring.activemq.broker-url	URL of the ActiveMQ broker. Auto-generated by default. For a remote broker connection consider this example ⁶ : <i>tcp://remotehost:61616</i> For a more complex configuration (failover,tls,..) consult the activemq documentation. [1]
spring.activemq.password	Password for the broker connection
spring.activemq.user	Username for the connection

Table 1: internal message broker configuration properties example

A more detailed description is provided by the spring project [2].

2.4.2. Queue Description

The current version 4.1.0-RELEASE uses 3 queues (see Figure 6):

- gatewayToControllerQueue
- backendToControllerQueue
- backendWaitQueue

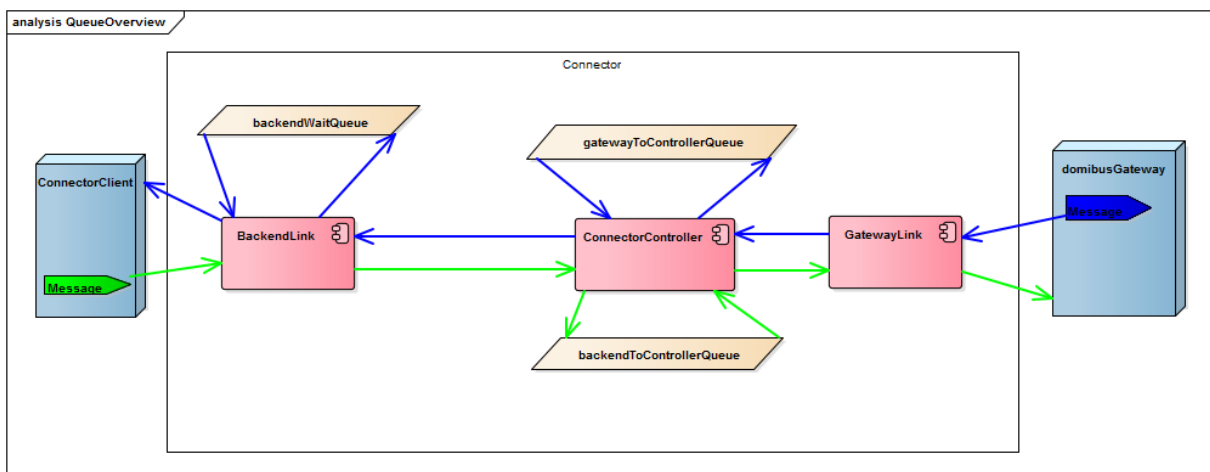


Figure 6: Connector internal queues overview and message flow

⁴ <http://spring.io/projects/spring-boot>

⁵ <http://activemq.apache.org/>

⁶ <http://activemq.apache.org/uri-protocols.html>

2.4.2.1. GatewayToControllerQueue

Messages are put on this queue immediately after they have been successfully received from the gateway and stored to the connector database. The controller is fetching messages from this queue and processes (ASICS-S container handling, message confirmation handling) them.

2.4.2.2. backendToControllerQueue

Messages are put on this queue immediately after they have been successfully received from the backend/connectorClient and stored to the connector database. The controller is fetching messages from this queue and processes (creating ASIC-S container, creating confirmation/evidence message) them.

2.4.2.3. backendWaitQueue

Messages processed by the backendLink are put on the backendWaitQueue. On this queue the messages are waiting for being fetched by a pull client or fetched by the backendLink push for pushing them to the push client. For all messages on this queue the correct backend is already determined.

3. Configuration

The domibusConnector needs external configuration to be able to process properly. There are different ways to provide the properties's values.

3.1. connector.properties

The starting point to configure the domibusConnector is the "connector.properties". Within the documentation of the distribution package an example properties file is placed. This example properties file shows the application specific properties that need to be set. In the example properties file every property is described properly via comments. When installing and preparing the domibusConnector this property file needs to be configured and placed. A detailed description of this procedure is given in the document "domibusConnector_InstallationGuide".

3.2. Property configuration in the domibusConnector database

In the database of the domibusConnector the table "DOMIBUS_CONNECTOR_PROPERTY" can also hold configuration properties and their values. If the configuration is done via domibusConnector-Administration-UI as described in chapter [Configuration section](#) the configured and stored configuration is persisted into that table.

When starting, the domibusConnector connects to its database schema and checks if there are properties configured in the table. Those property values will have higher priority than those of the "connector.properties" file if properties are configured in both places.

4. Logging

The connector uses log4j2 logging framework for logging. The logging system supports different logging levels for different components of the connector. The basics logging flow of log4j2 is that a logger collects log messages and hands them over to the configured log appenders. The Log appenders are responsible for formatting and writing them to a file, sending them over a network or writing to the console.

For further information and the capabilities consult the log4j2 [4] documentation:

<https://logging.apache.org/log4j/2.x/>

4.1. Logging Level

The used logging levels are listed by increasing criticality:

- TRACE – logs almost everything (do not use that in production!)
- DEBUG – for DEBUG purposes, logs almost everything
- INFO – for Information, perfect log level to put this log messages into an audit system
- WARN – for Warnings, something might have been gone wrong, the administrator should take a look on it, but the system was able to move on
- ERROR – for Errors, where something happened and the system was not able to move on, maybe a message ended up in error state

4.2. Logger Names

The connector uses different logger names, which can be configured with different logging levels. For the description of the logger names consult the example log4j2.xml which are provided by the distribution package of the domibusConnector.

The logger names can be used to increase the logging of the orm mapper components to show the executed sql queries. The in Table 2: Example Logger Config configured logger increases the logging level of all org.hibernate components to DEBUG, so the database queries will be logged to the Appenders Console and File.

Table 2: Example Logger Config

```
<Logger name="org.hibernate" level="DEBUG" additivity="false">
  <AppenderRef ref="Console" />
  <AppenderRef ref="File" />
</Logger>
```

4.3. Business Logging

Additional to the different logger names the software also using logging marks to mark messages which are relevant for business. In the default configuration all log messages marked with a business flag are written into a different log file. Currently the connector uses following business log flags:

- BUSINESS
 - BUSINESS_CONTENT used for log the deletion of business content, or other important messages when processing attachments or business xml.

- BUSINESS_EVIDENCE used for log messages concerning evidences, like creation
- BUSINESS_CERT used for logging certificate related messages, like certificates which will outdate or are already outdated.

The parent Log Mark also contains all child log marks. So filtering for the BUSINESS log mark will also contain the child log marks.

The default logging configuration uses a logging marker to separate with BUSINESS marked logging messages and writes them into a business log file. The RollingFile appender will fill up the current file to 5 Megabytes. After this size limit is reached it will create a new file.

```
<RollingFile name="Business" fileName="${basePath}/business.log"
filePattern="${basePath}/business-%d{yyyy-MM-dd}.log">
    <MarkerFilter marker="BUSINESS" onMatch="ACCEPT" onMismatch="DENY"/>
    <PatternLayout pattern="${sys:CONN_FILE_LOG_PATTERN}" />
    <Policies>
        <SizeBasedTriggeringPolicy size="5M" />
    </Policies>
</RollingFile>
```

Table 3: Example Log Appender

4.4. Follow logs of a message process

Sometimes it is necessary to reproduce the way of a message through the connector. For this purpose the connector sets on every log message related to a message process a so called MDC key with the name "messageid".

In the default logging configuration this will be printed out in the field [tid=04eb2c2c-b3ce-4b29-b7c7-cc06c99c4265@domibus.eu]. With the tools cat and grep the log can be reduced to the messages related to this message transport process.

```
cat connector.log | grep 04eb2c2c-b3ce-4b29-b7c7-cc06c99c4265@domibus.eu
```

Be aware that the result will only contain log messages where the connector already knows the message. So if the message is currently being received or loaded from the database the message process id is currently unknown.

The message process id matches the CONNECTOR_MESSAGE_ID in the database.

5. domibusConnector Administration UI

The domibusConnector-4.1.0-RELEASE is a web application which already contains the domibusConnector Administration UI (User Interface). This domibusConnector Administration UI gives information over processed messages and makes it possible to do configuration over a graphical user interface.

This chapter contains information about the domibusConnector Administration UI features. It also describes how to use them.

5.1. Login

When you deployed the domibusConnector on your application server it is placed at the relative path `http://[your servers ip or name and port]/[web application context]/admin`. When entering the site it first wants you to login with a valid user and password.



domibusConnector - Administration



Username

Password

[Login](#)
[Change Password](#)

Figure 7: Screenshot login screen

5.1.1. Preconfigured users

When the provided database scripts of the domibusConnector have been used to install and prepare or migrate the database of the domibusConnector then two default users were created.

Username	Initial Password	Role	Description
admin	admin	ADMIN	Default administration user
user	user	USER	Default user read-only user.

Table 4 Preconfigured users from the Administration UI

The role concept of the domibusConnector Administration UI foresees 2 Roles:

- USER: This role contains privileges to search messages, create reports and read configuration.
- ADMIN: In addition to the USER role, users of this role can edit configuration, upload P-Modes and create or edit other users.

!!! When first logging in with one of the predefined users, the domibusConnector Administration UI immediately forces a password change. !!!

5.2. Static information and main menu

After login, you should find yourself on the “Info” page which shows some useful static information about the domibusConnector instance like version and connected database.

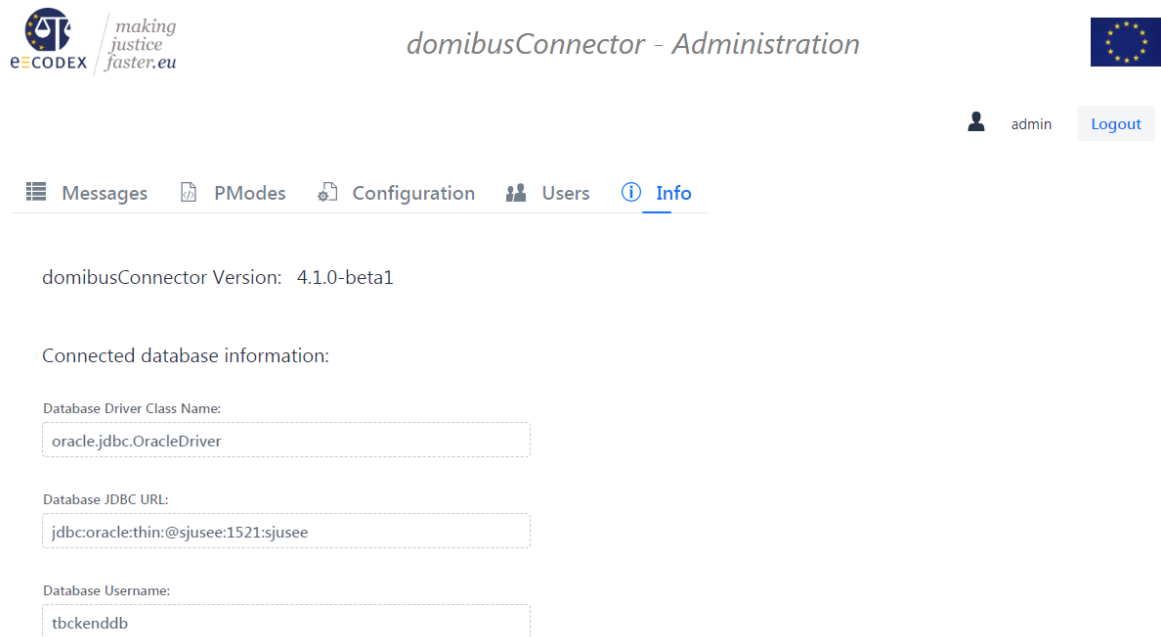


Figure 8: Screenshot Info page

5.2.1. User Information

Between the header of the Administration UI and the main menu, the logged in user can be seen. There is also a “Logout” button that should be used when work has been done.

5.2.2. Main menu

The main menu holds the main sections of the Administration UI. The section chosen can be seen as it is highlighted in blue and underlined.

5.3. Messages section

The “Messages” section shows all information from the database of the domibusConnector to processed messages:

[Messages](#) [PModes](#) [Configuration](#) [Users](#) [Info](#)

[Message Search](#) [All Messages](#) [Message Details](#) [Messages Reports](#)

Search by Connector Message ID

Search by EBMS Message ID

Search by Backend Message ID

Search by Conversation ID

From Date

To Date

Figure 9: Screenshot Messages section

5.3.1. Message Search

If a certain message is searched, where particular message information is known, this message can be searched here providing the particular information.

Also, if this particular information finds multiple results, like with the ConversationID, or with a certain time period, a list of results is displayed.

Once a single result selection is chosen, or, if the search result is a single result, the “Message Details” described in [Message Details](#) are shown.

5.3.2. All Messages

This sub-section provides a complete list of all messages processed by this instance of the domibusConnector.

Message Search [All Messages](#) Message Details Messages Reports

Filter by From Party ID Filter by To Party ID Filter by Service Filter by Action Filter by backend [Clear Filter](#)

Details	Connector Message ID	From Party ID	To Party ID	Service	Action	Created	Delivered Backend	Delivered Gateway	Backend Client
	e3d143ef-9a52-4498-aca9-1a26c7e3be13@domibus.connector.eu	EXECGW0	EXECGW1	EXECSERVICE	EXECAction1	2018-10-19 10:51:01.			cn=connector-client
	69527874-67f9-4309-b32a-ad0c7d3cbf89@domibus.connector.eu	EXECGW0	EXECGW1	EXECSERVICE	EXECAction1	2018-10-19 10:59:41.		2018-10-19 10:59:44.	cn=connector-client
	26e61c2e-919d-4076-823d-3a37b11d8b14@domibus.connector.eu	EXECGW0	EXECGW1	EXECSERVICE	EXECAction1	2018-10-19 11:59:21.		2018-10-19 11:59:27.	cn=connector-client
	7d1ad529-cbd3-406b-9e81-b04919b3e68f@domibus.connector.eu	EXECGW0	EXECGW1	EXECSERVICE	EXECAction1	2018-10-30 08:59:45.			connector-client
	122b39d1-30d2-4ab0-9a4d-48bbe7dab71f@domibus.connector.eu	EXECGW0	EXECGW1	EXECSERVICE	EXECAction1	2018-10-30 09:26:45.		2018-10-30 09:26:54.	connector-client
	4babc7dd-8be2-45d4-ac00-5d45221b3f3f@domibus.connector.eu	EXECGW0	EXECGW1	EXECSERVICE	EXECAction1	2018-10-19 11:02:21.		2018-10-19 11:02:25.	cn=connector-client
	8b10c755-1947-40c2-8e99-b028731827c8@domibus.connector.eu	EXECGW0	EXECGW1	EXECSERVICE	EXECAction1	2018-10-19 11:03:21.		2018-10-19 11:03:24.	cn=connector-client

Figure 10 Screenshot All Messages sub-section

This list can be filtered using provided filter criteria. Those filter criteria can be combined and take effect as soon as edited. The “Clear Filter” button clears all criteria and the full list will be shown again.

This table, as well as any table within the Administration UI, can be sorted by clicking the column header in any direction. Also combined sorting of multiple columns is possible.

The currently shown message list (with or without filters) can be exported and downloaded as a Microsoft Excel file for further usage.

5.3.3. Message Details

Once a single message is chosen, whether this is done by selecting a message from a list or searching for single resulting criteria, this message is then shown with the “Message Details”.

Message Search [All Messages](#) [Message Details](#) Messages Reports

Connector Message ID	26e61c2e-919d-4076-823d-3a37b11d8b14@domibus.connector.eu
Backend Message ID	20181019115909970_EXECGW\$(exec.workshop.user.number)
EBMS Message ID	e63efc3b-3484-4407-bbbe-190c9acb3c34@domibus.eu
Conversation ID	
Original Sender	Test1

...

Message created at 2018-10-30 08:59:45.69

Evidences:

Evidence Type ▾	Delivered to Gateway ▾	Delivered to Backend ▾
SUBMISSION_REJECTION		2018-10-30 09:00:00.251012
SUBMISSION_ACCEPTANCE		


Those message details contain ALL available information to this particular message, including all ETSI-REM evidences created or received for this message.

5.3.4. Messages Reports


Other than the lists of messages shown before, this sub-section only focuses on numbers.

[Message Search](#)
[All Messages](#)
[Message Details](#)
[Messages Reports](#)

From Date



To Date



☐ Include sent evidences as messages

[Generate Report](#)

Figure 11: Screenshot Messages Reports – search criteria

For first a period has to be selected that define the time range of the reports that have to be generated. By using the “calendar” symbols and picking the date there is the most comfortable way to not run into date format issues.

The checkbox “Include sent evidences as messages” indicates, if ETSI-REM evidences that are sent or received for messages as own messages should be counted as such.

Once criteria are selected, the “Generate Report” Button must be clicked to generate and show the reports.

From Date
 ✕ 📅

To Date
 ✕ 📅

☒ Include sent evidences as messages

[Generate Report](#)

9/2018			
Party	Service	Messages received from	Messages sent to
LV	BC-009MLA	23	15
Total:		23	15

10/2018			
Party	Service	Messages received from	Messages sent to
ZC	BC-009MLA	5	5
Total:		58	41



Figure 12 Screenshot Message Reports results

As seen in Figure 12 the results are separated by month. The generated reports can be downloaded as an XLS file for further usage.

5.4. PModes section

The “p-modes” that are distributed by the configuration management for the DOMIBUS gateway can also be used for the domibusConnector database to have necessary data to support business use cases in your domibusConnector database.

[Messages](#)
[PModes](#)
[Configuration](#)
[Users](#)
[Info](#)

[Import PModes](#)
[Data Tables](#)

[Select File...](#)
📁 Drop file here

Figure 13: Screenshot PModes section






The p-mode file from the configuration management can be selected here and will be uploaded into the database of the domibusConnector.

Every ACTION, PARTY and SERVICE that is found in the p-modes and which do not exist in the database already, will be created. The domibusConnector will neither change any existing database items (service, action or party), nor will it delete any of those. If the p-mode-file cannot be processed an error message will appear.






After importing PModes, or if there are already imported PModes you can see and edit the results from this import in sub-section “Data Tables”

Import PModes [Data Tables](#)

DOMIBUS_CONNECTOR_PARTY:

Party ID	Party ID Type	Party Role
 EXECGW0	urn:oasis:names:tc:ebcore:partyid-type:unregistered	GW
 EXECGW1	urn:oasis:names:tc:ebcore:partyid-type:unregistered	GW
 EXECGW2	urn:oasis:names:tc:ebcore:partyid-type:unregistered	GW
 EXECGW3	urn:oasis:names:tc:ebcore:partyid-type:unregistered	GW
 EXECGW4	urn:oasis:names:tc:ebcore:partyid-type:unregistered	GW

DOMIBUS_CONNECTOR_ACTION:

Action
 SubmissionAcceptanceRejection
 RelayREMMDAcceptanceRejection
 DeliveryNonDeliveryToRecipient
 RetrievalNonRetrievalToRecipient
 EXFCAction1

DOMIBUS_CONNECTOR_SERVICE:

Service	Service Type
---------	--------------

Figure 14 Screenshot Data Tables sub-section

5.5. Configuration section

All properties described in chapter [Configuration](#) can also be viewed and edited within the domibusConnector Administration UI.

Messages
PModes
Configuration
Users
Info

Discard Changes
Save Changes
Reload Configuration

Environment Configuration
Security Toolkit Configuration
Gateway Configuration
Backend Configuration
Evidences Configuration

Message storage - filesystem path

C:\Entwicklung\Test\connector\fs-storage ⓘ

☒ Create directories ⓘ

Service for Connector 2 Connector Tests

EXECService x ⓘ

Action for Connector 2 Connector Tests

EXECAction2 x ⓘ

Proxy Configuration:

☒ Use an HTTP Proxy ⓘ

Figure 15 Screenshot Configuration section

Be aware that edited and saved configuration properties will NOT be written into the properties file they initially need to be set in, but will be stored into the domibusConnector database.

The three possibilities of handling the configuration edited within the UI are given by the buttons in the button bar. Each of them will open an explanatory dialog before taking action.

Discard Changes

All changes since the last time of saving will be discarded.

Confirm Cancel

Figure 16 Screenshot Discard Changes Dialog

Save Changes

All changed configuration properties will be saved into the database table DOMIBUS_CONNECTOR_PROPERTIES.

Be aware that changes to the configuration except backend client configuration will only take effect after restart of the domibusConnector.

Also take note that the configured properties in the property files will NOT be changed!

Confirm Cancel

Figure 17 Screenshot Save Changes Dialog



Figure 18 Screenshot Reload Configuration Dialog

This whole section is readable only for users of role USER.

The different properties are grouped thematically into sub-sections.

- Environment Configuration
- Security Toolkit Configuration
- Gateway Configuration
- Backend Configuration
- Evidences Configuration

Each configuration item has the same structure:

- Name of the configuration item.
- Value of the configuration item. This can be text, select box or check box.
- Information on the configuration item.

When opening the information on a configuration item a dialog appears

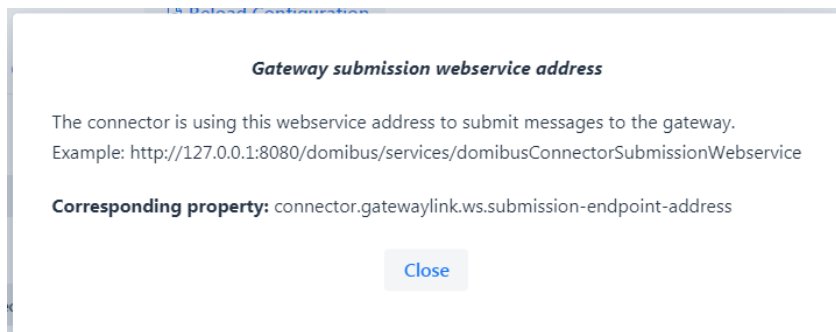


Figure 19 Screenshot Information on a configuration item

This dialog displays the name of the configuration item, an explanation what it means, mostly an example and the corresponding property name. That way, every configuration item can be found either in the database or in the property file.

5.6. Users section

The “Users” section gives users of the role ADMIN the possibility to create new users, edit existing users or deactivate/delete users.

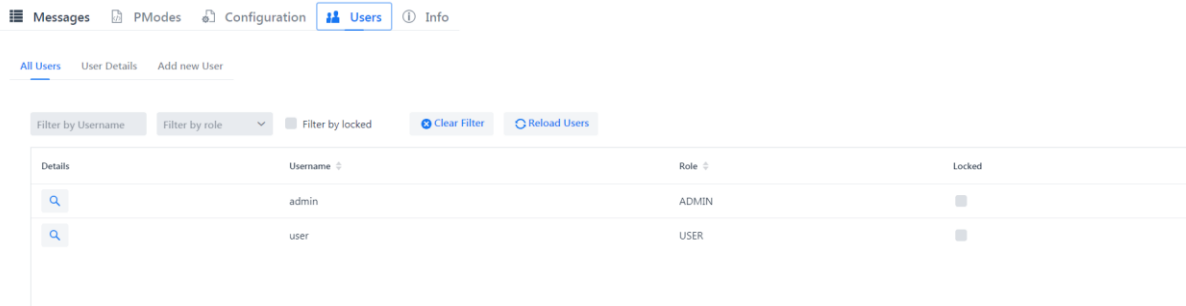


Figure 20 Screenshot Users section

5.6.1. All Users

The first displayed sub-section is “All Users”. It gives an overview of existing users, their roles and the status of a user.

This list can also be filtered.

5.6.2. User Details

When selecting a user, the “User Details” will show.

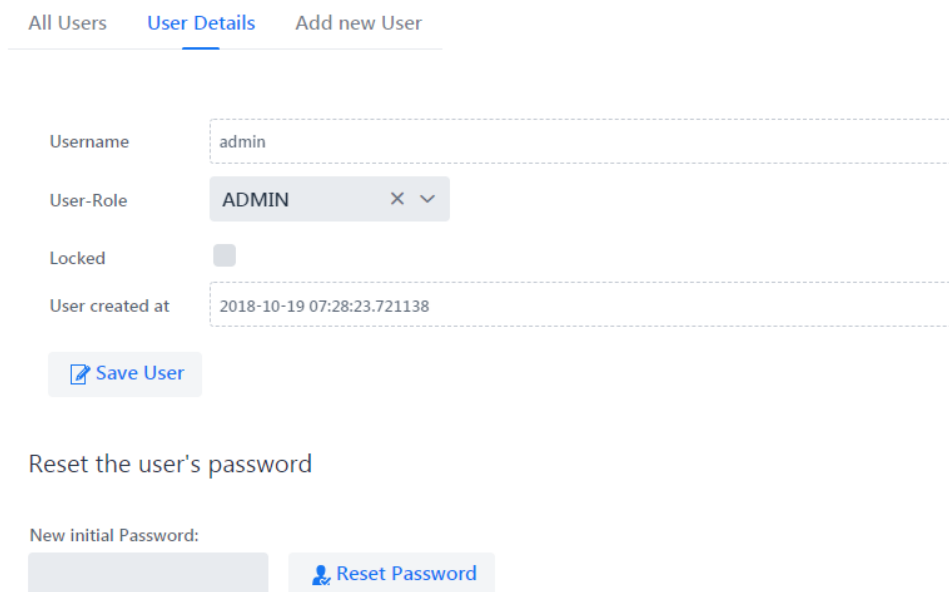


Figure 21 Screenshot User Details

Some of the user’s details must not be edited. Like the Username. This is static and cannot be changed.

What can be changed is the role of that user. Also the user can be locked. If a user is locked already because of failed login trials, it can be unlocked here. This should always be combined with reset of the user’s password. The new password that can be given here is NOT a persistent one. This is a new initial one for the user. The next time the user will login with this given initial password, it is forced to change the password immediately.

5.6.3. Add new User

Also new users can be created.

All Users User Details [Add new User](#)

Username

Role

Initial password


 [Create User](#)

Figure 22 Screenshot Add new User

All that needs to be given is a Username (static once the user is created), the role the user attends to and an initial password.

6. Tasks involving database access

This chapter describes typical problems or tasks which are not supported by the user interface yet. So the main objective is to give an overview of important tables and columns in the database and their meaning.

6.1. Search for errors

Every error which can be related to a message is appended to the message and stored into the database. The table DOMIBUS_CONNECTOR_MSG_ERROR (Figure 23) holds these errors.

ID	MESSAGE_ID	ERROR_MESSAGE	DETAILED_TEXT	ERROR_SOURCE	CREATED
17	51	Exception sending confirmation message '07eba72f-375a-464a-b121-0688b28d6b1f@domibus.c...	eu.domibus.connector....	eu.domibus.connector.control...	2018-03-27 13:53:07
18	67	Could not submit message to oatewav!	eu.domibus.connector....	eu.domibus.connector.control...	2018-03-27 16:27:24
19	70	Could not submit message to oatewav!	eu.domibus.connector....	eu.domibus.connector.control...	2018-03-28 07:08:55
20	71	Could not submit message to oatewav!	eu.domibus.connector....	eu.domibus.connector.control...	2018-03-28 07:10:59
21	89	Could not handle Evidence to Message 03f66ab9-dadd-483e-8d66-84f637b24687@domibus.eu	eu.domibus.connector....	eu.domibus.connector.eviden...	2018-03-28 10:41:42
22	90	Could not handle Evidence to Message cebdddbf-67d4-4adb-91f0-92407c3ccf22@domibus.eu	eu.domibus.connector....	eu.domibus.connector.eviden...	2018-03-28 10:42:40
23	95	Could not handle Evidence to Message 97dfd330-e50c-4407-b471-eb71524f26b1@domibus.eu	eu.domibus.connector....	eu.domibus.connector.eviden...	2018-03-28 11:48:24
24	97	Could not handle Evidence to Message c4daced6-669b-4076-8596-9fc838771f7e@domibus.eu	eu.domibus.connector....	eu.domibus.connector.eviden...	2018-03-28 11:53:01
25	99	Could not handle Evidence to Message 81473118-49ed-4569-a31b-5f259e332fc4@domibus.eu	eu.domibus.connector....	eu.domibus.connector.eviden...	2018-03-28 11:55:40
26	101	Could not handle Evidence to Message 96eaa372-f65d-4d11-84d8-4849c312cc04@domibus.eu	eu.domibus.connector....	eu.domibus.connector.eviden...	2018-03-28 12:08:03
27	103	Could not handle Evidence to Message f4becf98-3052-4430-b216-b0a1fad6df6a@domibus.eu	eu.domibus.connector....	eu.domibus.connector.eviden...	2018-03-28 12:13:02
28	105	Could not handle Evidence to Message 0028069e-6a20-493e-99d0-692c19f84982@domibus.eu	eu.domibus.connector....	eu.domibus.connector.eviden...	2018-03-28 12:43:21
29	107	Could not handle Evidence to Message 2403830c-1f68-4bf3-8ad3-c0de384f0a9c@domibus.eu	eu.domibus.connector....	eu.domibus.connector.eviden...	2018-03-28 13:06:03
30	108	Could not handle Evidence to Message 93f79d87-5ed0-48a0-a86f-dc9783a97271@domibus.eu	eu.domibus.connector....	eu.domibus.connector.eviden...	2018-03-28 13:12:21

Figure 23: Connector 4.0 – example of DOMIBUS_CONNECTOR_MSG_ERROR table

The column ERROR_MESSAGE contains a description of the occurred error. The whole stacktrace is stored in the DETAILED_TEXT column. The connector is also printing the connector message id to the logging output, with this information you can follow the message process (see **Fehler! Verweisquelle konnte nicht gefunden werden.**).

I. LIST of FIGURES

Figure 1: e-CODEX environment overview	5
Figure 2: domibusConnector in the national environment.....	6
Figure 3: Connector Web Service Interfaces	7
Figure 4: domibusConnector 4.1.0 simplyfied component overview	9
Figure 5: domibusConnectorBackendLink message routing	10
Figure 6: Connector internal queues overview and message flow	11
Figure 7: Screenshot login screen	16
Figure 8: Screenshot Info page.....	17
Figure 9: Screenshot Messages section	18
Figure 10 Screenshot All Messages sub-section.....	19
Figure 11: Screenshot Messages Reports – search criteria.....	20
Figure 12 Screenshot Message Reports results	21
Figure 13: Screenshot PModes section	21
Figure 14 Screenshot Data Tables sub-section	22
Figure 15 Screenshot Configuration section	23
Figure 16 Screenshot Discard Changes Dialog	23
Figure 17 Screenshot Save Changes Dialog.....	23
Figure 18 Screenshot Reload Configuration Dialog.....	24
Figure 19 Screenshot Information on a configuration item.....	24
Figure 20 Screenshot Users section	25
Figure 21 Screenshot User Details	25
Figure 22 Screenshot Add new User	26
Figure 23: Connector 4.0 – example of DOMIBUS_CONNECTOR_MESSAGE tableFehler! Textmarke nicht definiert.	
Figure 24: Connector 4.0 – example of DOMIBUS_CONNECTOR_EVIDENCE tableFehler! Textmarke nicht definiert.	
Figure 25: Connector 4.0 – example of DOMIBUS_CONNECTOR_MSG_ERROR table.....	27

II. LIST of TABLES

Table 1: internal message broker configuration properties example.....	11
Table 2 Preconfigured users from the Administration UI	16

III. LIST of REFERENCES

- [1 „ActiveMQ,“ [Online]. Available: <http://activemq.apache.org/>.
]
- [2 „Spring Commona Application Properties,“ [Online]. Available: <https://docs.spring.io/spring-boot/docs/1.5.8.RELEASE/reference/html/common-application-properties.html>.
]
- [3 *Required certificates, key- and truststores in the e-Codex environment*, 2018/05.
]
- [4 “Apache Log4j 2,” [Online]. Available: <https://logging.apache.org/log4j/2.x/>.
]
- [5 [Online]. Available:
] http://www.etsi.org/deliver/etsi_ts/102600_102699/10264002/02.01.01_60/ts_10264002v020101p.pdf.
- [6 „domibusConnectorAPI 4.0.0 RELEASE WSDL,“ [Online]. Available: <https://secure.e-codex.eu/nexus/content/repositories/releases/eu/domibus/connector/domibusConnectorAPI/4.0.0-RELEASE/domibusConnectorAPI-4.0.0-RELEASE-wsdl.zip>.
]
- [7 „nexus repositroy,“ [Online]. Available: <https://secure.e-codex.eu/nexus/content/repositories/releases/eu/domibus>.
]