domibusConnector-4.0.0-RELEASE - Administration Guide

Table of contents

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 ”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.

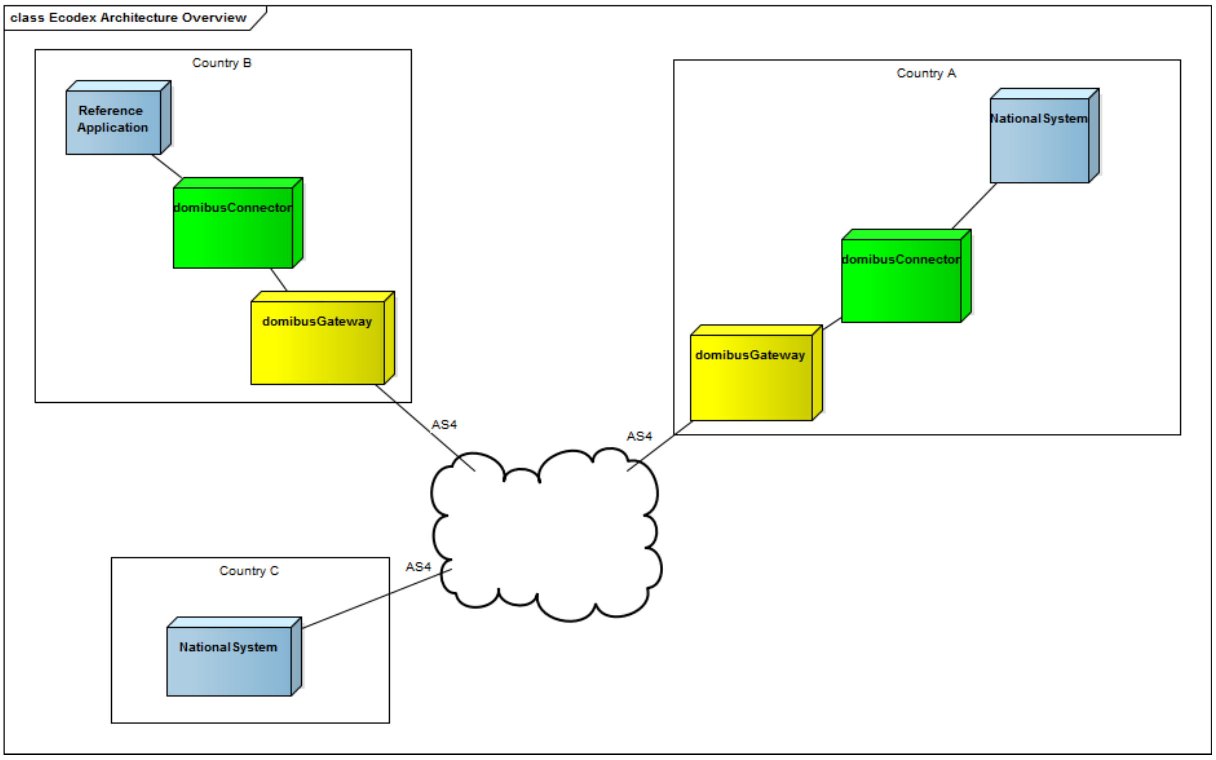


Figure 1: e-CODEX environment overview

2.2. Connector in the national environment

The connector is a web application. The connector has two sides. A national side, connector-client or connector-backend side, and the gateway side. Messages are always delivered from the gateway to the connectorClient. In the other direction the messages are submitted.

As Figure 2 shows the connector clients may use the domibusConnectorClient[1] to establish a connection to the connector or they can directly use the provided web service interfaces of the domibusConnector. The connector supports multiple clients.[2]

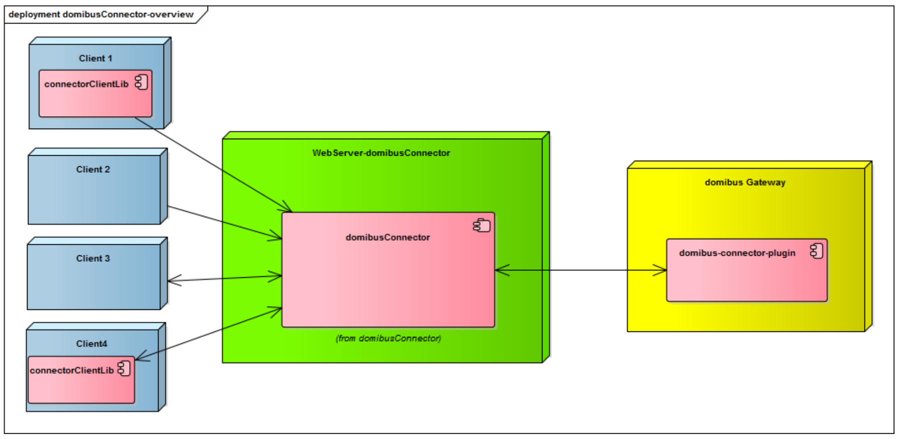


Figure 2: Connector in the national environment

The NationalSystem in this example uses the domibusConnectorClient-library to establish the connection to the domibusConnector. The domibusConnectorClient-library is included as a jar dependency inside the national implementation.

The ConnectorClient of this example addresses the web 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.

2.2.1. Connector 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 server[3]. 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.

2.2.1.1. DomibusConnectorGatewayDeliveryWebService

The connector implements this interface for message delivery from the gateway to the connector. This interface is implemented by the connectorGatewayLink (see chapter 2.3.10) module.

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

[2] Figure 5 shows how the message routing is done

[3] https://secure.e-codex.eu/nexus/content/repositories/releases/eu/domibus/connector/domibusConnectorAPI/4.0.0-RELEASE/domibusConnectorAPI-4.0.0-RELEASE-wsdl.zip

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

2.2.1.2. DomibusConnectorBackendWebService

The DomibusConnectorBackendWebService interface is implemented by the domibusConnectorBackendLink module (see chapter 2.3.9) 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

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

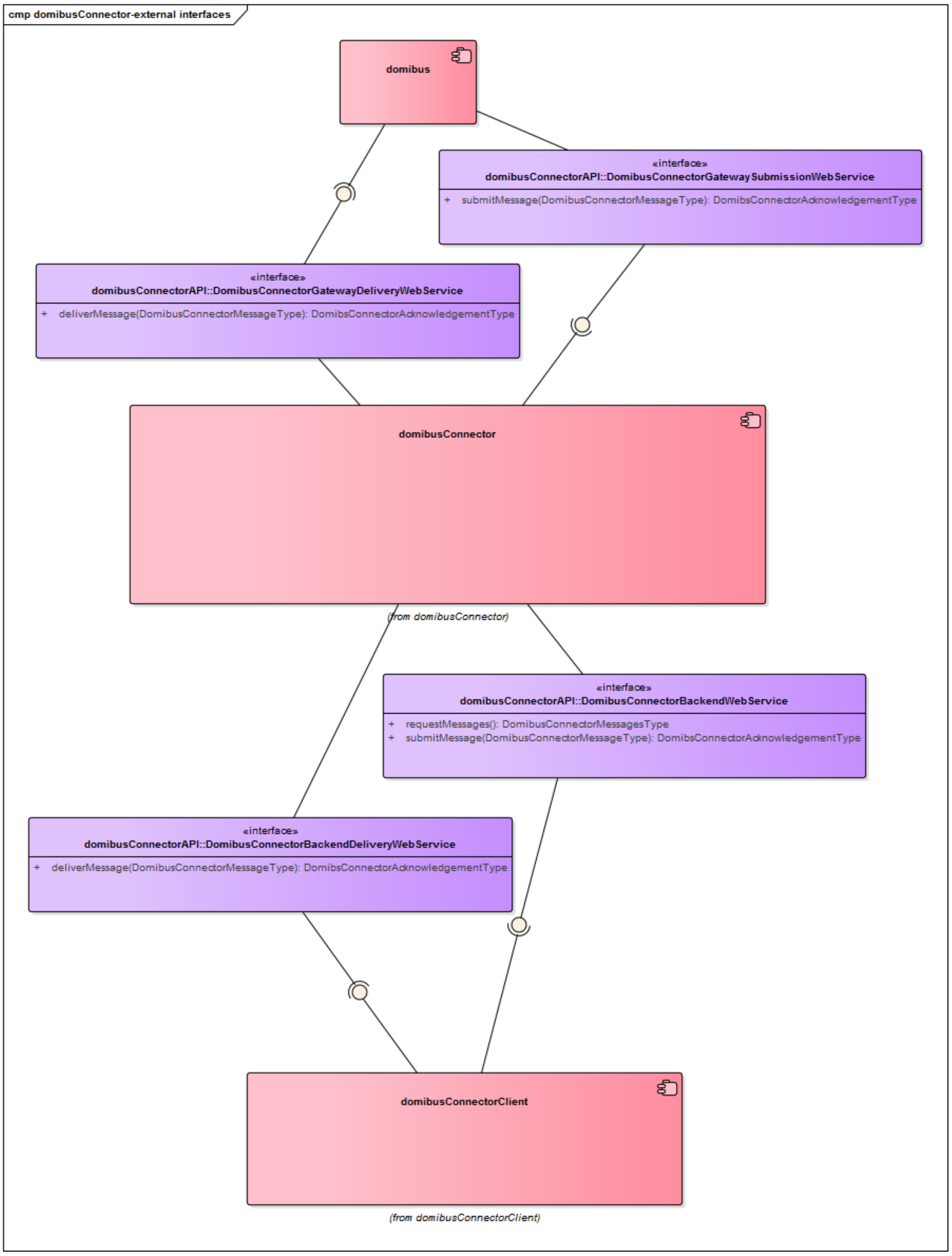
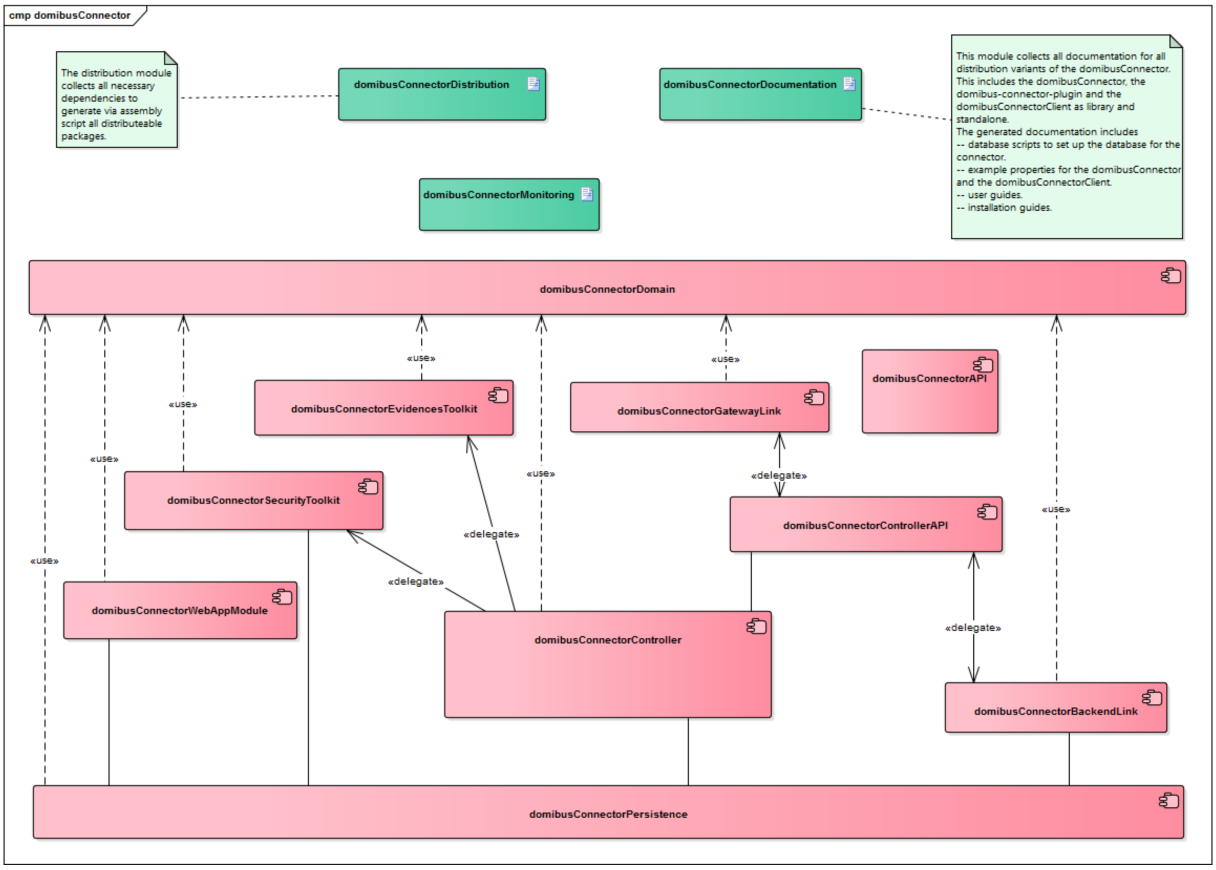

cmp domibusConnector-external interfaces
domibus
«interface»
domibusConnectorAPI::DomibusConnectorGatewaySubmissionWebService
+ submitMessage(DomibusConnectorMessageType): DomibsConnectorAcknowledgementType
«interface»
domibusConnectorAPI::DomibusConnectorGatewayDeliveryWebService
+ deliverMessage(DomibusConnectorMessageType): DomibsConnectorAcknowledgementType
domibusConnector
(from domibusConnector)
«interface»
domibusConnectorAPI::DomibusConnectorBackendWebService
+ requestMessages(): DomibusConnectorMessagesType
+ submitMessage(DomibusConnectorMessageType): DomibsConnectorAcknowledgementType
«interface»
domibusConnectorAPI::DomibusConnectorBackendDeliveryWebService
+ deliverMessage(DomibusConnectorMessageType): DomibsConnectorAcknowledgementType
domibusConnectorClient
(from domibusConnectorClient)

Figure 3: Connector Web Service Interfaces

2.3. Connector Components

The connector 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: Connector 4.0 component overviewFigure 4). Each module has a specific purpose.


cmp domibusConnector
domibusConnectorDistribution
domibusConnectorDocumentation
domibusConnectorMonitoring
domibusConnectorDomain
«use»
domibusConnectorEvidencesToolkit
domibusConnectorGatewayLink
domibusConnectorAPI
«delegate»
domibusConnectorSecurityToolkit
domibusConnectorControllerAPI
domibusConnectorWebAppModule
domibusConnectorController
domibusConnectorBackendLink
domibusConnectorPersistence

The distribution module collects all necessary dependencies to generate via assembly script all distributeable packages.

This module collects all documentation for all distribution variants of the domibusConnector. This includes the domibusConnector, the domibus-connector-plugin and the domibusConnectorClient as library and standalone.
The generated documentation includes
-- database scripts to set up the database for the connector.
-- example properties for the domibusConnector and the domibusConnectorClient.
-- user guides.
-- installation guides.

Figure 4: Connector 4.0 component overview

2.3.1. domibuConnectorAPI

Holds the public interface descriptions, the wsdl files and the web service policy.

2.3.2. domibusConnectorDomain

Contains the domain model, only used by the connector internally.

2.3.3. domibusConnectorControllerAPI

Provides an internal API for connectorController and the link modules.

2.3.4. domibusConnectorController

This module contains all the business logic. It uses other modules for resolving the asic-s container, sending messages to the gateway or persisting messages.

2.3.5. domibusConnectorEvidencesToolkit

This module is responsible for creating the evidences. It is called by the connectorController.

2.3.6. domibusConnectorWebApp

This module includes the other modules and also contains a web application for managing the connector. Also the deployable ware is generated from this module.

2.3.7. 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.

2.3.8. domibusConnectorPersistence

This module is responsible for persisting messages, confirmations and message states. Message content is only persisted between the message has been received by the gateway/connectorClient and will be removed after the message has been delivered to the client/submitted to the gateway.

2.3.9. 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 push client the web service of the client is called. For pull clients the messages are stored until the client asks for them.

This module also determines the correct backend where the message is delivered to. Figure 5: BackendLink message routing shows an activity diagram how this is done.

2.3.10. 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.

2.3.11. Other modules

The maven project contains some more modules. Those modules are used for creating the distribution, holding the documentation and providing test data and executing tests. Those modules are not part of this description because they are only used during development.

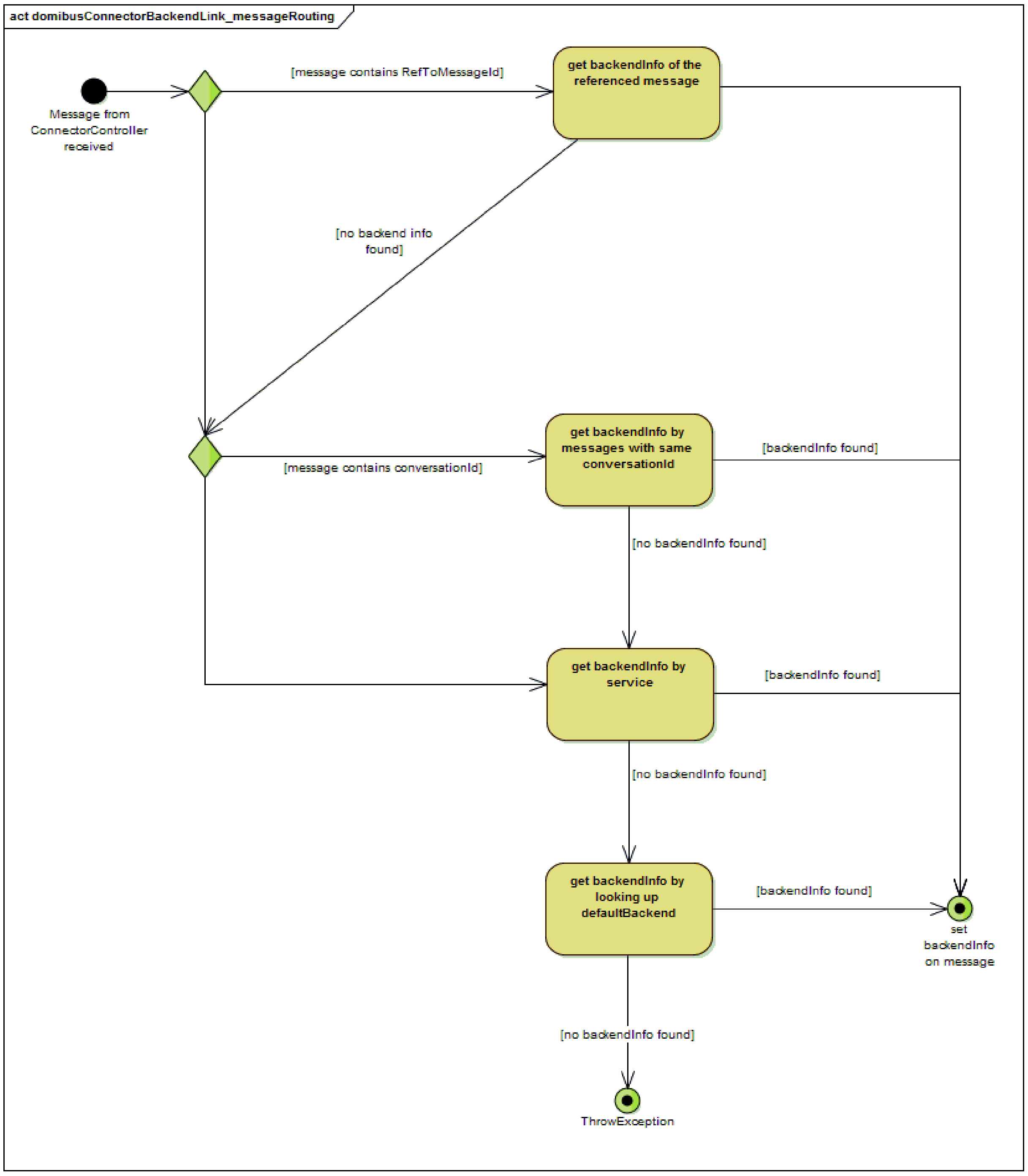

act domibusConnectorBackendLink_messageRouting
Message from ConnectorController received
[message contains RefToMessageId]
get backendInfo of the referenced message
[no backend info found]
[message contains conversationId]
get backendInfo by messages with same conversationId
[backendInfo found]
[no backendInfo found]
get backendInfo by service
[backendInfo found]
[no backendInfo found]
get backendInfo by looking up defaultBackend
[backendInfo found]
set backendInfo on message
[no backendInfo found]
ThrowException

Figure 5: BackendLink message routing

2.4. Internal Queues

The connector uses an internal queueing system for decoupling the connector 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 connector uses spring boot to start its own message broker (ActiveMQ) for these queues. So all properties used in spring boot can be used to configure this internal message broker. It is also possibly 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[4]: *tcp://remotehost:61616* 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 4.0 release uses 3 queues (see Figure 6):

- gatewayToControllerQueue
- backendToControllerQueue
- backendWaitQueue

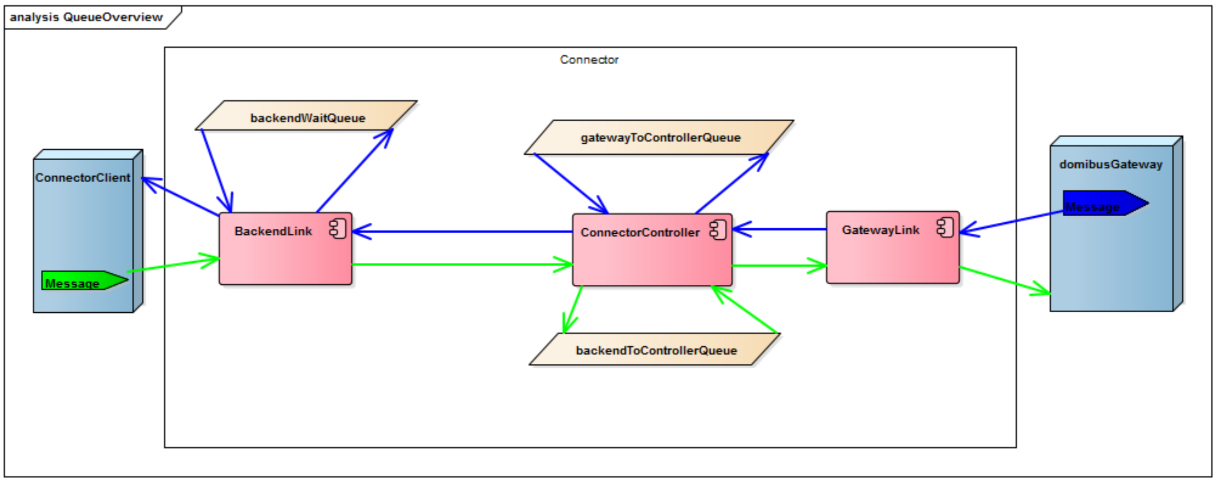


Figure 6: Connector internal queues overview and message flow

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.

[4] http://activemq.apache.org/uri-protocols.html

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 connector has multiple configuration options. This chapter contains a list of configuration properties. Some of those properties are for more experienced users. Those properties are highlighted with blue colour.

The properties are grouped by the connector modules:

3.1. domibusConnectorController

Property name	Description
connector.controller.evidence.timeout-active	This property configures if timeouts for message confirmations should be checked!
connector.controller.evidence.check-timeout	This property defines how often the timeouts for the evidences should be checked the default value is 5 minutes.
connector.controller.evidence.delivery-timeout	This property defines the timeout how long the connector should wait for a DELIVERY confirmation message after a message has been successfully submitted to the gateway. The default value is 24 hours. If the timeout limit is reached the timeoutProcessor is started and a NON_DELIVERY confirmation is created for the according business message.
connector.controller.evidence.relayREMMD-timeout	This property defines how long the connector should wait for a relayREMMD evidence message after a message has been successfully submitted to the gateway. The default value is 24 hours. If the timeout exceeds the according timeoutProcessor is started and a NON_DELIVERY confirmation is created.
connector.controller.evidence.retrieval-timeout	This property defines how long the connector should wait for a retrieval evidence message after a message has been successfully submitted to the gateway. The default value is 24 hours. If the timeout exceeds a NON_DELIVERY confirmation is created for the according message.
domibus.connector.internal.gateway.to.controller.queue	This property defines the name of the internal gateway to controller queue. For details see chapter 2.4.2.1 page 12. The default name for the queue is " gatewayToControllerQueue”
domibus.connector.internal.backend.to.controller.queue	This property defines the name of the internal backend to controller queue. For details see chapter 2.4.2.2 page 13. The default value for this property is “backendToControllerQueue”.

Table 2: domibusConnectorController configuration properties

3.2. domibusConnectorGatewayLink

Property name	Description
connector.gatewaylink.ws.address	Configures the address part for the deliver message service. The service url is defined relative to the url of the CXF-Servlet (usually configured under /services), the defined url is appended. So the default url for pushing messages from domibus gateway to connector will be “${SERVER_CONTEXT}/services/ domibusConnectorDeliveryWebservice” Default value is “/domibusConnectorDeliveryWebservice”.
connector.gatewaylink.ws.submission-endpoint-address	Configures the URL for the submitting messages service to the Gateway. Must be configured so the connector can submit messages to the gateway.
connector.gatewaylink.ws.tls-key-store.password	This key/trust-store related properties are described in the Key/TrustStore Guide. [3]
connector.gatewaylink.ws.tls-key-store.path	
connector.gatewaylink.ws.tls-key.alias	
connector.gatewaylink.ws.tls-key.password	
connector.gatewaylink.ws.tls-trust-store.password	
connector.gatewaylink.ws.tls-trust-store.path	

Table 3: domibusConnectorGatewayLink configuration properties

3.3. domibusConnectorBackendLink

Property name	Description
connector.backend.internal.wait-queue.name	See chapter 2.4.2.3 on page 13
connector.backend.internal.wait-queue.receive-timeout	Configures how long the polling command should wait for messages. The queue is polled if a pull client asks for messages.
connector.backend.ws.backend-publish-address	Specifies the address where the Backend WebService should be published the path specefied here is added to the path of the CXF-Servlet (which is per default

	configured as /service - this leads to the default URL of "/services/backend" The default value is "/backend"
connector.backend.ws.ws-policy	Contains the path to the wsdl policy which is used communicating between connectorBackendClient and connectorBackend Default value is "classpath:wsdl/backend.policy.xml"
connector.backend.ws.key.store.path	This key/trust-store related properties are described in the Key/TrustStore Guide. [3]
connector.backend.ws.key.store.password	
connector.backend.ws.key.key.alias	
connector.backend.ws.key.key.password	
connector.backend.ws.trust.loadCaCerts	
connector.backend.ws.trust.store.path	
connector.backend.ws.trust.store.password	

Table 4: domibusConnectorBackendLink configuration properties

3.4. domibusConnectorSecurityToolkit

Property name	Description
connector.security.ojstore.path	Those key/trust-store related properties are described in the Key/TrustStore Guide. [3]
connector.security.ojstore.password	
connector.security.trustStore.path	
connector.security.trustStore.password	

Table 5: domibusConnectorSecurityToolkit configuration properties

3.5. domibusConnectorEvidencesToolkit

Property name	Description
connector.evidences.keystore.path	Those key/trust-store related properties are described in the Key/TrustStore Guide. [3]
connector.evidences.keystore.password	
connector.evidences.key.alias	
connector.evidences.key.password	
connector.evidences.hashAlgorithm	The hashAlgorithm used for signing the evidences. Possibly are: MD5, SHA1, SHA256, SHA512 The default value is SHA256.

Table 6: domibusConnectorEvidencesToolkit configuration properties

4. Logging

The connector uses log4j2 logging framework for logging. The logging system supports different logging levels and different logging for different components of the connector. For further information consult the log4j2 [4] documentation.

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 log4j.properties which are provided by the distribution package of the domibusConnector.

5. Graphical User Interface

The domibusConnector 4.0 is a web application which already contains the domibusWebAdmin. The domibusWebAdmin makes it possible to have a quick overview over the sent messages and makes it possible to upload the pModes for configuration purposes.

This chapter contains informations about the user interface features. It also describes how to use them.

5.1. Login

When you deployed the domibusWebAdmin 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.

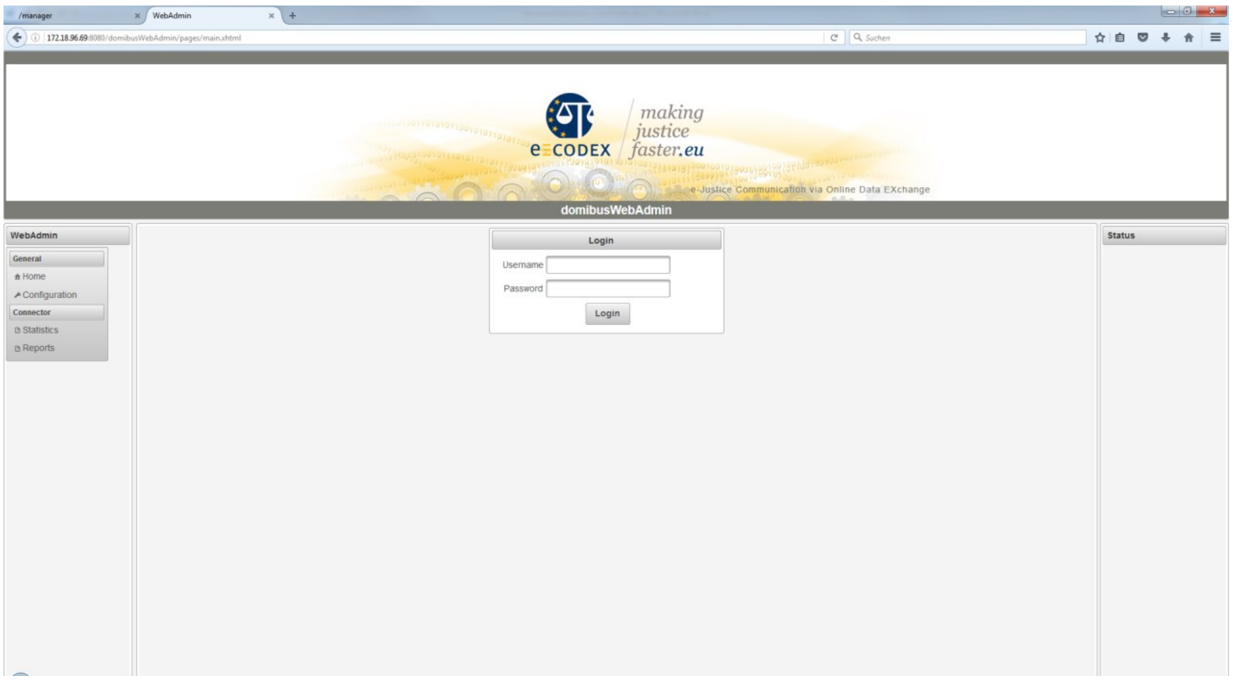


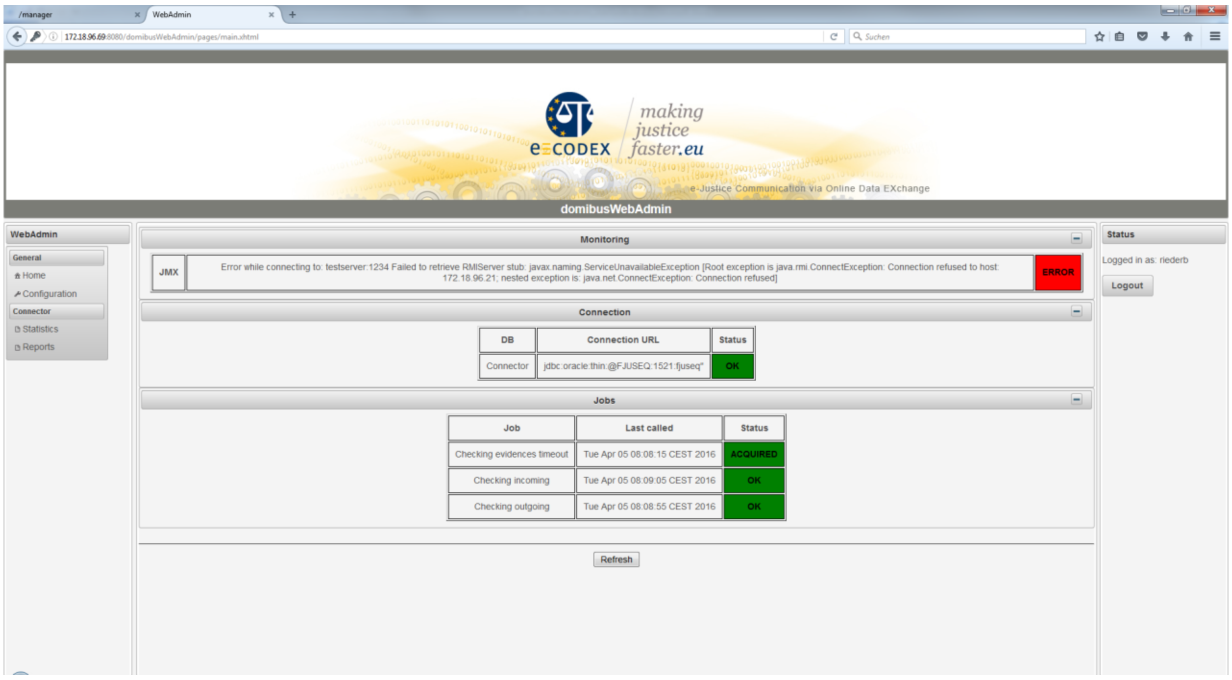
Figure 7: Screenshot login screen

When the provided database scripts of the domibusWebAdmin were used to install the database then a first default user was created. This is “admin” with the password “admin”. Later, in the “Configuration” section of the domibusWebAdmin you can create, delete and edit users.

!!! It is highly recommended to change the password of the “admin” user to grant restricted access to the domibusWebAdmin. !!!

5.2. Main Page

After login, you should find yourself on the main page, which shows the navigation menu on the left side and the summarized monitoring information in the center. The monitoring information is generally concluded with a state in the form OK (Green color), WARNING (Yellow) or ERROR (Red).



Monitoring

JMX	Error while connecting to: testserver:1234 Failed to retrieve RMIServer stub: javax.naming.ServiceUnavailableException [Root exception is java.rmi.ConnectException: Connection refused to host: 172.18.96.21; nested exception is: java.net.ConnectException: Connection refused]	ERROR

Connection

DB	Connection URL	Status
Connector	jdbc:oracle:thin:@FJUSEQ:1521:fjuseq"	OK

Jobs

Job	Last called	Status
Checking evidences timeout	Tue Apr 05 08:08:15 CEST 2016	ACQUIRED
Checking incoming	Tue Apr 05 08:09:05 CEST 2016	OK
Checking outgoing	Tue Apr 05 08:08:55 CEST 2016	OK

Figure 8: Screenshot main page

5.2.1. Monitoring panel

Depending on the configured monitoring type the monitoring panel is displayed first.

The domibusConnector framework offers different interfaces to expose its monitoring information, namely JMX and REST. The monitoring panel shows the state of the selected monitoring type.

An ERROR is displayed, when the selected monitoring type is not reachable.

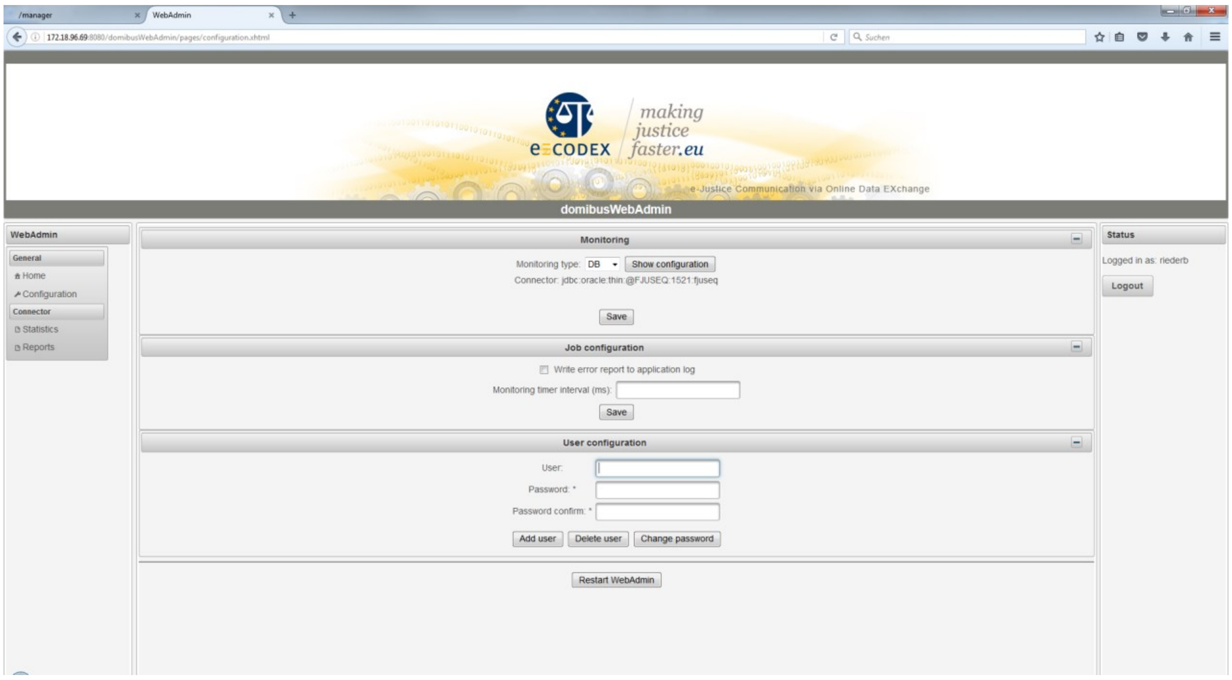
Note: When no monitoring interface is selected (by default monitoring type “DB”, the panel is not shown at all.

5.2.2. Connection panel

This panel shows the information of the connection to the configured database.

5.3. Configuration Page

The configuration page shows three panels for configuring different parts:



172.18.96.69:8080/domibusWebAdmin/pages/configuration.xhtml

Monitoring type: DB

Connector: jdbc:oracle:thin:@FJUSEQ:1521:fjuseq

Logged in as: riederb

Figure 9: Screenshot configuration page

5.3.1. Monitoring panel

The monitoring panel is deprecated and not working in the current release. It will be replaced by a new user interface. The recommended way of monitoring the connector is monitoring the logging output.

5.3.2. Job configuration panel

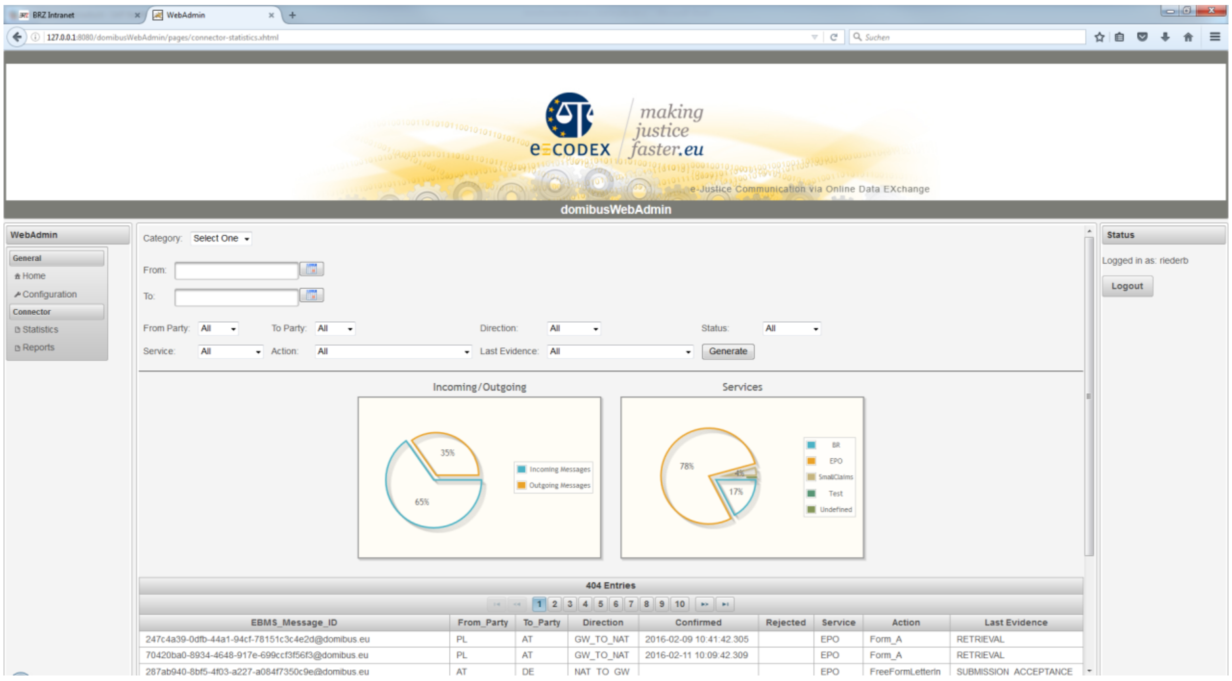
The job configuration panel is deprecated and will be replaced by a new user interface. Job configuration is done by configuring the connector.properties file.

5.3.3. User configuration panel

This is a panel to add and delete users. Also the password for a user can be changed here. To add a user you must first enter the new User and the password. To delete a user or change its password the correct User must be entered with its correct password.

5.3.4. Statistics

The statistics are shown out of the database of the connected domibusConnector database:



Category: Select One

From:

To:

From Party: All | To Party: All | Direction: All | Status: All

Service: All | Action: All | Last Evidence: All

404 Entries

EBMS_Message_ID	From_Party	To_Party	Direction	Confirmed	Rejected	Service	Action	Last Evidence
247c4a39-0dfb-44a1-94cf-78151c3c4e2d@domibus.eu	PL	AT	GW_TO_NAT	2016-02-09 10:41:42.305		EPO	Form_A	RETRIEVAL
70420ba0-8934-4648-917e-699ccf3f56f3@domibus.eu	PL	AT	GW_TO_NAT	2016-02-11 10:09:42.309		EPO	Form_A	RETRIEVAL
287ab940-8bf5-4f03-a227-a084f7350c9e@domibus.eu	AT	DE	NAT_TO_GW			EPO	FreeFormLetterIn	SUBMISSION_ACCEPTANCE

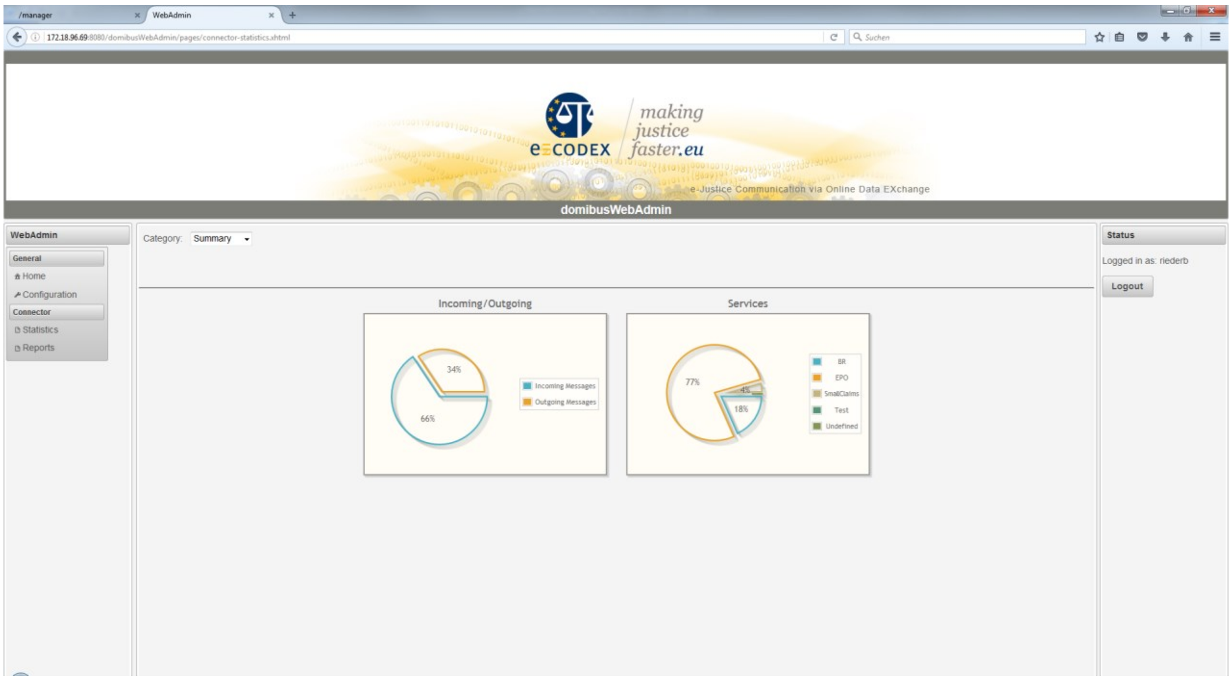
Figure 10: Screenshot statistic page - details

When entering the page it first shows an overall statistic as a summary and a listing.

The first box “Category” lets you choose between two options:

<u>Summary:</u>

This page shows simple diagrams for overall data. At the actual state it shows the amount of incoming/outgoing messages and the type of service.

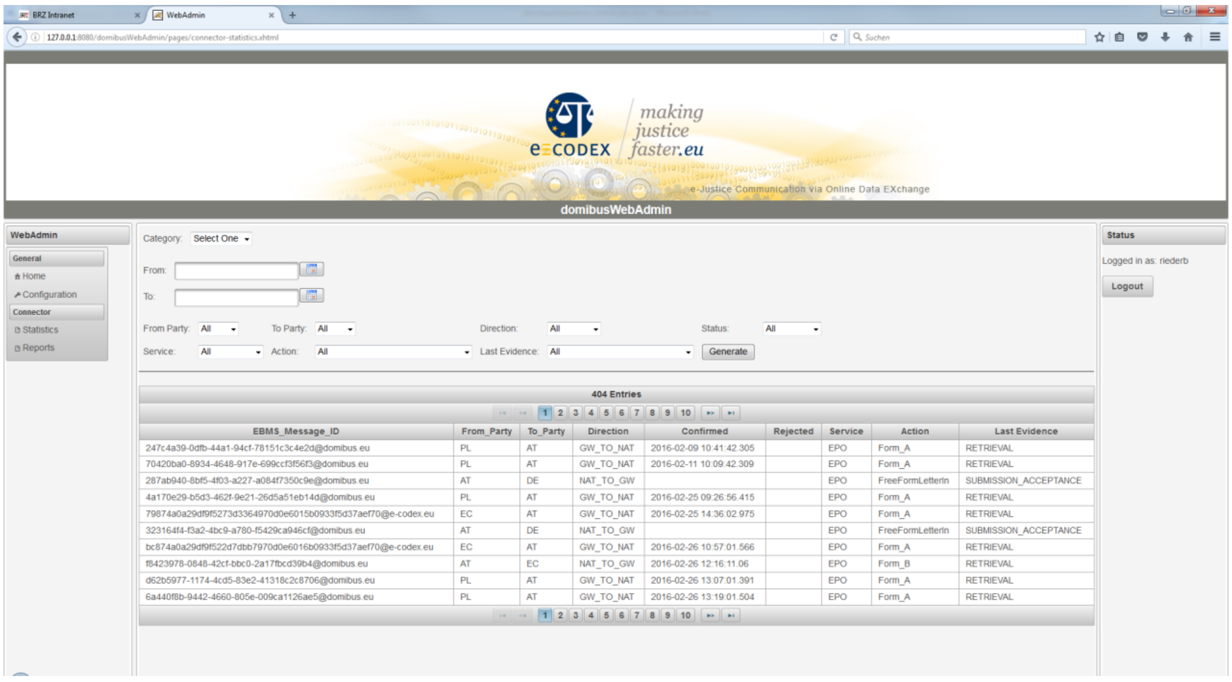


Category: Summary

Figure 11: Screenshot statistic page - summary

Custom:

Here, the connector messages can be filtered by different parameters. By hitting the generate button, the result is displayed in the table below.



Category: Select One

From:

To:

From Party: All | To Party: All | Direction: All | Status: All

Service: All | Action: All | Last Evidence: All | Generate

404 Entries

EBMS_Message_ID	From_Party	To_Party	Direction	Confirmed	Rejected	Service	Action	Last Evidence
247c4a39-0dfb-44a1-94cf-78151c3c4e2d@domibus.eu	PL	AT	GW_TO_NAT	2016-02-09 10:41:42.305		EPO	Form_A	RETRIEVAL
70420ba0-8934-4648-917e-699ccf3f56f3@domibus.eu	PL	AT	GW_TO_NAT	2016-02-11 10:09:42.309		EPO	Form_A	RETRIEVAL
287ab940-8bf5-4f03-a227-a084f7350c9e@domibus.eu	AT	DE	NAT_TO_GW			EPO	FreeFormLetterIn	SUBMISSION_ACCEPTANCE
4a170e29-b5d3-462f-9e21-26d5a51eb14d@domibus.eu	PL	AT	GW_TO_NAT	2016-02-25 09:26:56.415		EPO	Form_A	RETRIEVAL
79874a0a29df9f5273d3364970d0e6015b0933f5d37aef70@e-codex.eu	EC	AT	GW_TO_NAT	2016-02-25 14:36:02.975		EPO	Form_A	RETRIEVAL
323164f4-f3a2-4bc9-a780-f5429ca946cf@domibus.eu	AT	DE	NAT_TO_GW			EPO	FreeFormLetterIn	SUBMISSION_ACCEPTANCE
bc874a0a29df9f522d7dbb7970d0e6016b0933f5d37aef70@e-codex.eu	EC	AT	GW_TO_NAT	2016-02-26 10:57:01.566		EPO	Form_A	RETRIEVAL
f8423978-0848-42cf-bbc0-2a17fbcd39b4@domibus.eu	AT	EC	NAT_TO_GW	2016-02-26 12:16:11.06		EPO	Form_B	RETRIEVAL
d62b5977-1174-4cd5-83e2-41318c2c8706@domibus.eu	PL	AT	GW_TO_NAT	2016-02-26 13:07:01.391		EPO	Form_A	RETRIEVAL
6a440f8b-9442-4660-805e-009ca1126ae5@domibus.eu	PL	AT	GW_TO_NAT	2016-02-26 13:19:01.504		EPO	Form_A	RETRIEVAL

Figure 12: Screenshot statistics - custom

5.3.5. Reports

The domibusWebAdmin can also show you reports from the connector database in a formatted way.

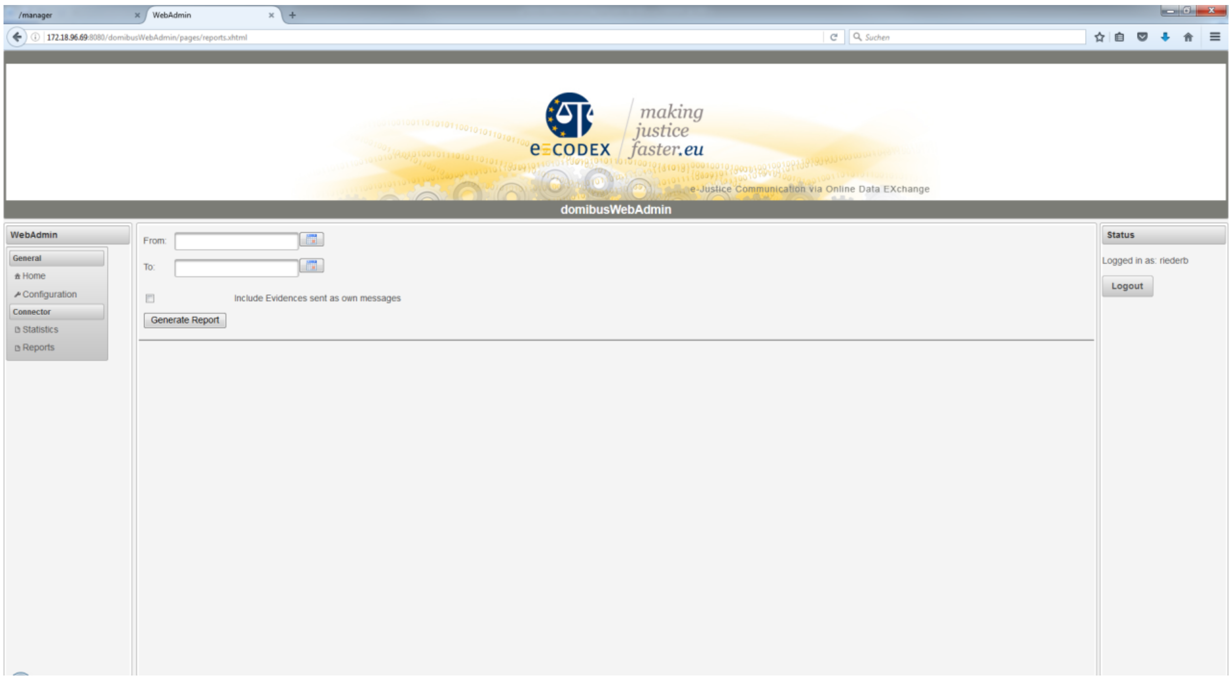
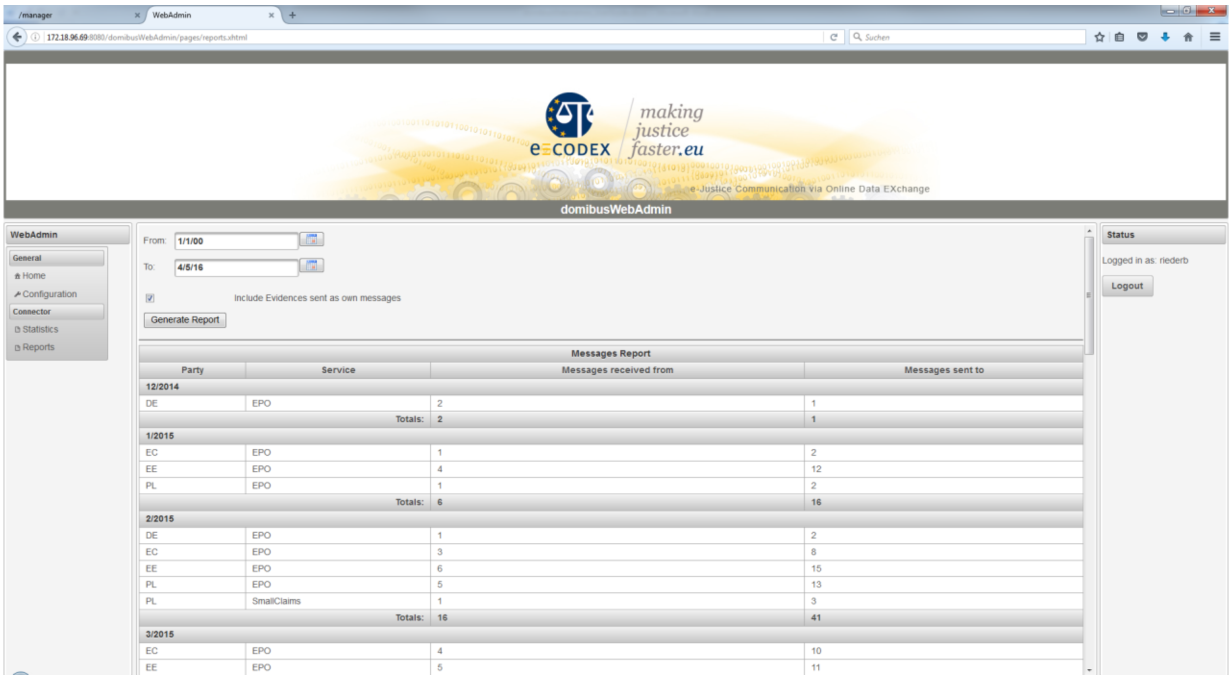


Figure 13: Screenshot report page

The search form just requests the time period you want the reports generated from. The best way is to use the calendar help besides the fields as the date format that has to be entered is platform dependent. If no period is entered (both fields left empty) the domibusWebAdmin generates reports from the 1st of January 2000 to the current date.

You can also choose whether to include evidences that are sent as own messages (without the first evidence that comes together with the message itself). Please note that the evidence messages are always sent the other direction as the message (as they are responses to the message).

When the reports are generated, they are displayed like this:



From: 1/1/00

To: 4/5/16

☑ Include Evidences sent as own messages

Messages Report

Party	Service	Messages received from	Messages sent to
12/2014			
DE	EPO	2	1
	Totals:	**2**	**1**
1/2015			
EC	EPO	1	2
EE	EPO	4	12
PL	EPO	1	2
	Totals:	**6**	**16**
2/2015			
DE	EPO	1	2
EC	EPO	3	8
EE	EPO	6	15
PL	EPO	5	13
PL	SmallClaims	1	3
	Totals:	**16**	**41**
3/2015			
EC	EPO	4	10
EE	EPO	5	11

Logged in as: riederb

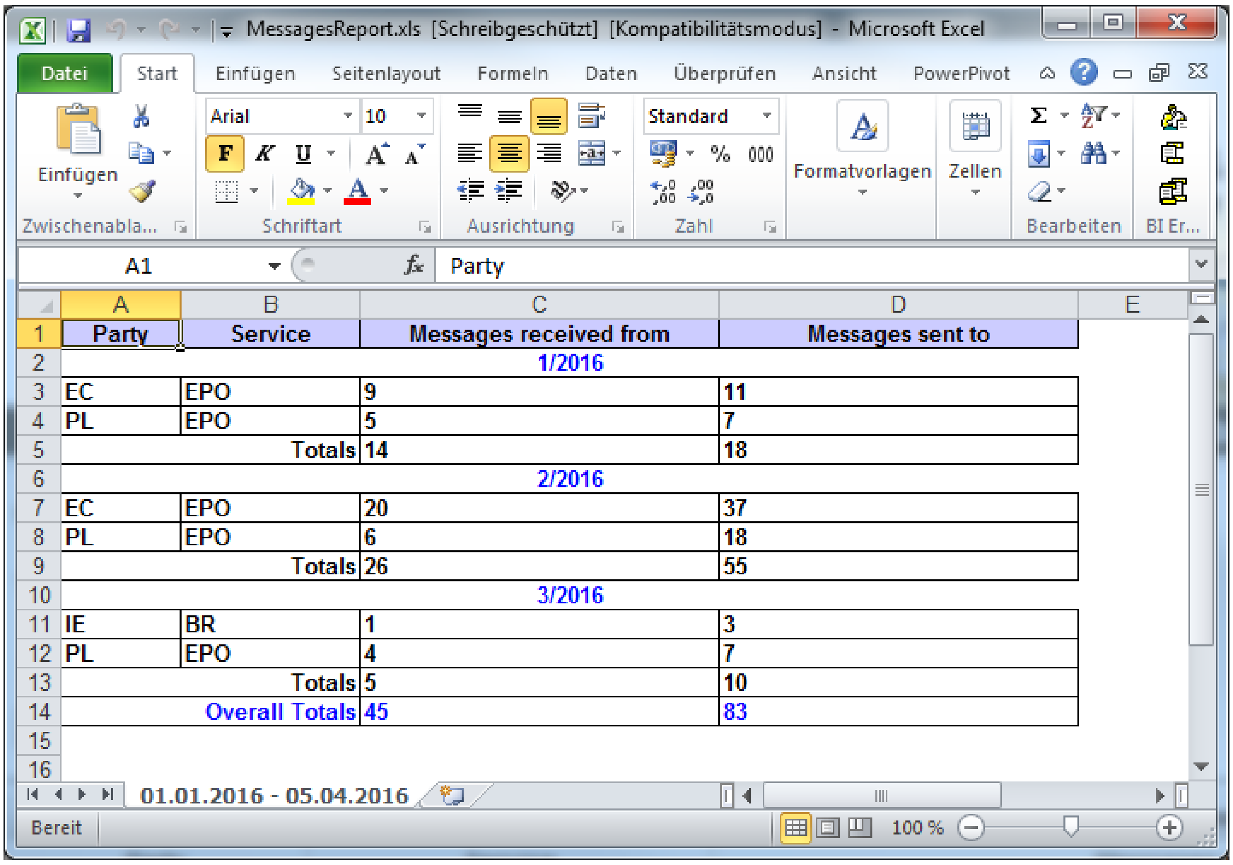
Figure 14: Screenshot report page - messages report

The reports show the Party to/from which the messages are sent/come from. Then the service, the amount of messages received and the amount of the messages sent per Party and Service.

The results are grouped monthly and the summarized amounts per month are also displayed.

__**Export:**__

The reports shown as the result can also be exported in XLS (MS Excel) format for further processing. The XLS is pre-formatted like this:



Party	Service	Messages received from	Messages sent to
		1/2016	
EC	EPO	9	11
PL	EPO	5	7
	Totals	14	18
		2/2016	
EC	EPO	20	37
PL	EPO	6	18
	Totals	26	55
		3/2016	
IE	BR	1	3
PL	EPO	4	7
	Totals	5	10
	Overall Totals	45	83

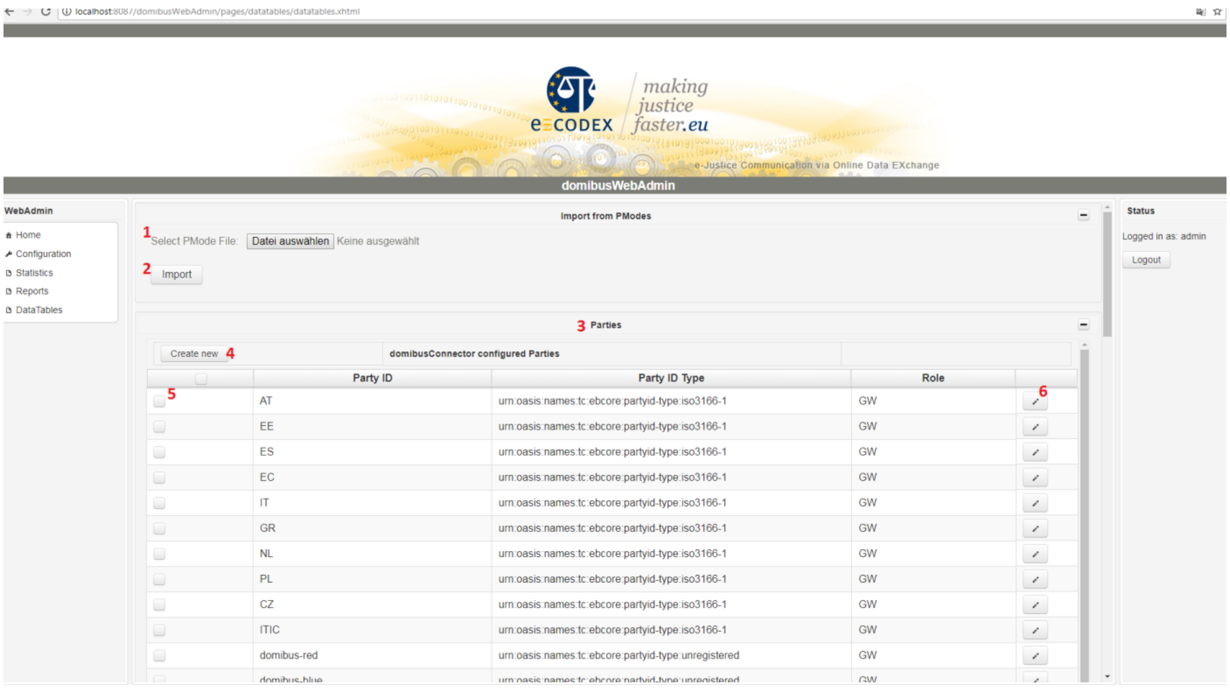
Figure 15: Screenshot of an exported report opened with Excel

In the sheet-tab the time period of the selection can be seen in case the reports need to be persisted regularly.

The data itself is formatted hardly different from the view on the page with one extension. It also shows overall totals that summarizes the amount of messages over the whole period.

5.4. Data Tables

The domibusWebAdmin provides also an overview over configured parties, actions and services. This view is reachable over the menu entry “DataTables”.



localhost:8087/domibusWebAdmin/pages/datatables/datatables.xhtml

e-Justice Communication via Online Data EXchange

domibusWebAdmin

WebAdmin
- Home
- Configuration
- Statistics
- Reports
- DataTables

Import from PModes

1 Select PMode File: Datei auswählen Keine ausgewählt

2 Import

3 Parties

Create new 4 domibusConnector configured Parties

	Party ID	Party ID Type	Role	
5	AT	urn:oasis:names:tc:ebcore:partyid-type:iso3166-1	GW	6
	EE	urn:oasis:names:tc:ebcore:partyid-type:iso3166-1	GW	
	ES	urn:oasis:names:tc:ebcore:partyid-type:iso3166-1	GW	
	EC	urn:oasis:names:tc:ebcore:partyid-type:iso3166-1	GW	
	IT	urn:oasis:names:tc:ebcore:partyid-type:iso3166-1	GW	
	GR	urn:oasis:names:tc:ebcore:partyid-type:iso3166-1	GW	
	NL	urn:oasis:names:tc:ebcore:partyid-type:iso3166-1	GW	
	PL	urn:oasis:names:tc:ebcore:partyid-type:iso3166-1	GW	
	CZ	urn:oasis:names:tc:ebcore:partyid-type:iso3166-1	GW	
	ITIC	urn:oasis:names:tc:ebcore:partyid-type:iso3166-1	GW	
	domibus-red	urn:oasis:names:tc:ebcore:partyid-type:unregistered	GW	
	domibus-blue	urn:oasis:names:tc:ebcore:partyid-type:unregistered	GW	

Status

Logged in as: admin

Logout

Figure 16: Screenshot of DataTables view

5.4.1. Import from PModes

The „Import from PModes“ function allows you to upload a pMode-xml file. The file is used to create parties, actions and services which aren’t yet configured. So this process will only create new items. There won’t be any modifications on existing items (services, actions or parties).

Import a pmode-File:

1. select a pModeFile (see Figure 16, red number 1)
2. press the import button (see Figure 16, red number 2)

If there are any failures maybe your pmode file is corrupt.

5.4.2. Parties

This table shows all which shows all configured parties. You can create new parties, modify the partyIdType of existing ones and also delete parties. Deletion is only possibly if there weren’t any messages processed according to that party.

5.4.2.1. Create a new party:

1. Press the “Create new” button (see Figure 16, red number 4)
2. A dialog appears. Now you can type in Party ID, Party ID Type and Role of the party. Be sure that the combination of Party ID and Role is unique.
3. Press “Create” to save the new Party OR “Cancel” to abort

5.4.2.2. Modify a party:

Each row contains a button marked with a pencil symbol. Click at the pencil symbol (see Figure 16, red number 6). You can only change the Party ID Type of the party. Press “save” to save the changes OR “Cancel” to abort the changes.

5.4.2.3. Delete a party:

To delete a party select the party by ticking the checkbox in the first column (see Figure 16, red number 5). After selecting all parties you can start the deletion process by hitting the “Delete” button at the bottom of the table.
You can only delete parties which have not received or sent to any messages.

5.4.3. Actions

This table shows all configured actions. You can create new actions, modify existing ones and also delete actions. Deletion is only possibly if there weren’t any messages processed according to that action. You can only change if the action requires a pdf.

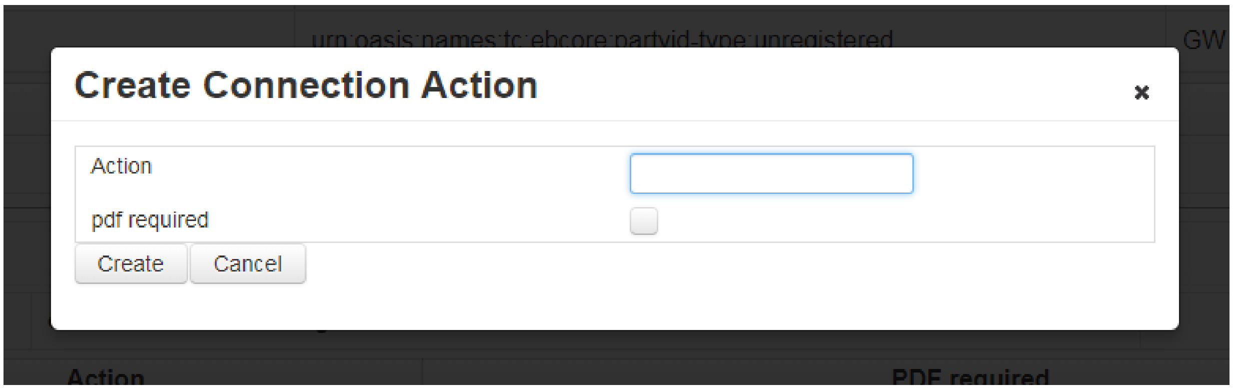


Figure 17: Create Connection Action Dialog

5.4.3.1. Create a new action

1. Press the “Create new” button at the actions table.
2. A Dialog appears (see Figure 17: Create Connection Action Dialog). Now you can type in the action and you can use the checkbox to determine if the action requires a pdf. Be aware that the action must be unique.
3. Press “Create” to save the new action OR “Cancel” to abort

5.4.3.2. Modify an existing action

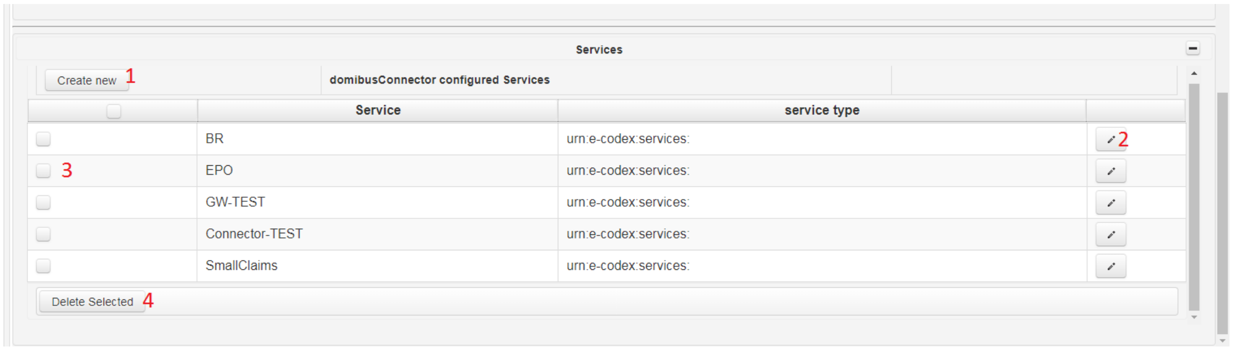
Each action row contains a button with a pencil symbol. Press this button to open the edit dialog. You can only change if the action requires a pdf. At the edit dialog use “Save” to save the change and “Cancel” to abort the modification.

5.4.3.3. Delete an action

To delete an action you have to select the action by ticking the checkbox in the first column. After that you can delete all marked actions by hitting the “Delete Selected” button at the end of the table.

5.4.4. Services

This table shows all configured services. You can create new services, modify the service type of existing ones and also delete services. Deletion is only possibly if there weren't any messages processed according to that service.



	Service	service type
	BR	urn:e-codex:services:
	EPO	urn:e-codex:services:
	GW-TEST	urn:e-codex:services:
	Connector-TEST	urn:e-codex:services:
	SmallClaims	urn:e-codex:services:

Figure 18: Screenshot of the services table only

5.4.4.1. Create a new service

1. Press the "Create new" button at the services table
2. A Dialog appears. Now you can type in the service and the service type. Be aware the service must be unique.
3. Press "Create" to save the new service or "Cancel" to abort.

5.4.4.2. Modify service

1. Press the button with the pencil symbol of the row you want to edit (see Figure 18: Screenshot of the services table only, red 2).
2. A dialog opens, now you can edit the service type
3. Press "Save" to save the change OR "Cancel" to abort

5.4.4.3. Delete service

To delete a service mark it in the first row by ticking the checkbox. After that you can delete all marked services by hitting the "Delete Selected" button (see Figure 18: Screenshot of the services table only, red 4)

5.4.5. Extra information

This create, modify and delete functions cannot modify an item (party, action or service) if there are any references pointing to it. This means if the connector has already processed messages using an item.

So if you really want to delete an already used service, action or party you have to do it by yourself in your database and you should know what you are doing!

6. Tasks involving database access

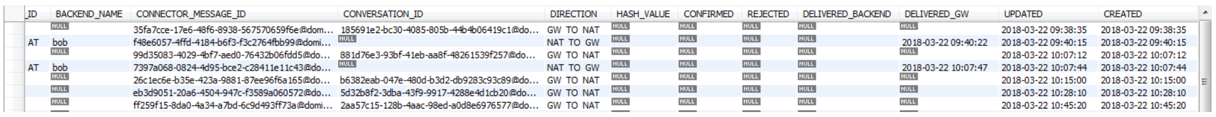
This chapter describes typical problems or tasks which are not supported by the user interface yet. So the main objective is to give an overview of important tables and columns in the database and their meaning.

6.1. Finding a message

Sometimes the question appears what happened to a specific message. It is also possible to use the user interface for finding a message (see chapter 5.3.4, page 21). First of all you need to know something about the message you are looking for. Good identifiers for the message are

- EBMS Message ID (assigned by the Gateway)
- Backend Message ID (assigned by the national system, or backend application)
- Connector Message ID (assigned by the connector, is not communicated to any other system)

After you gathered the information open the connector database. The connector contains a table named DOMIBUS_CONNECTOR_MESSAGE (Figure 19 shows an example of the table). This table contains all business messages received by the connector. You can search the message by the EBMS_MESSAGE_ID, BACKEND_MESSAGE_ID or also by the CONVERSATION_ID. The column DIRECTION tells use if the message was submitted to the gateway or was received by the gateway.



_ID	BACKEND_NAME	CONNECTOR_MESSAGE_ID	CONVERSATION_ID	DIRECTION	HASH_VALUE	CONFIRMED	REJECTED	DELIVERED_BACKEND	DELIVERED_GW	UPDATED	CREATED
	NULL	35fa7cce-17e6-48f6-8938-567570659f6e@dom...	185691e2-bc30-4085-805b-44b4b06419c1@do...	GW TO NAT	NULL	NULL	NULL	NULL	NULL	2018-03-22 09:38:35	2018-03-22 09:38:35
AT	bob	f48e6057-4ffd-4184-b6f3-f3c2764fbb99@domi...	NULL	NAT TO GW	NULL	NULL	NULL	NULL	2018-03-22 09:40:22	2018-03-22 09:40:15	2018-03-22 09:40:15
	NULL	99d35083-4029-4bf7-aed0-76432b06fdd5@do...	881d76e3-93bf-41eb-aa8f-48261539f257@do...	GW TO NAT	NULL	NULL	NULL	NULL	NULL	2018-03-22 10:07:12	2018-03-22 10:07:12
AT	bob	7397a068-0824-4d95-bce2-c28411e11c43@do...	NULL	NAT TO GW	NULL	NULL	NULL	NULL	2018-03-22 10:07:47	2018-03-22 10:07:44	2018-03-22 10:07:44
	NULL	26c1ec6e-b35e-423a-9881-87ee96f6a165@do...	b6382eab-047e-480d-b3d2-db9283c93c89@do...	GW TO NAT	NULL	NULL	NULL	NULL	NULL	2018-03-22 10:15:00	2018-03-22 10:15:00
	NULL	eb3d9051-20a6-4504-947c-f3589a060572@do...	5d32b8f2-3dba-43f9-9917-4288e4d1cb20@do...	GW TO NAT	NULL	NULL	NULL	NULL	NULL	2018-03-22 10:28:10	2018-03-22 10:28:10
	NULL	ff259f15-8da0-4a34-a7bd-6c9d493ff73a@domi...	2aa57c15-128b-4aac-98ed-a0d8e6976577@do...	GW TO NAT	NULL	NULL	NULL	NULL	NULL	2018-03-22 10:45:20	2018-03-22 10:45:20

Figure 19: Connector 4.0 – example of DOMIBUS_CONNECTOR_MESSAGE table

6.2. Message State

After you have found your message in the DOMIBUS_CONNECTOR_MESSAGE table you are interested to determine the state of your message. For this purpose there are 4 interesting columns

- CONFIRMED – Is null by default, will be set to the timestamp of the confirmation
- REJECTED – Is null by default, will be set to the timestamp of the rejection
- DELIVERED_BACKEND – is null by default, will be set to the delivery time
- DELIVERED_GW – is null by default, will be set to the delivery time

6.2.1. Column CONFIRMED description

Is null by default and will be set to the timestamp of the confirmation. The message is confirmed if and only if the message has not been rejected yet and the according DELIVERY confirmation is received by the connector.

6.2.2. Column REJECTED description

Is null by default and will be set to the timestamp of the rejection. The message is rejected if there is a NON_DELIVERY received for this message. Or the evidence timeout is reached, then a NON_DELIVERY is created and the message is marked as rejected.

6.2.3. Column DELIVERED_BACKEND description

Is null by default and will be set to the timestamp of the delivery to the backend/national system. Depending on the backend type the behaviour when this timestamp is set differs slightly.

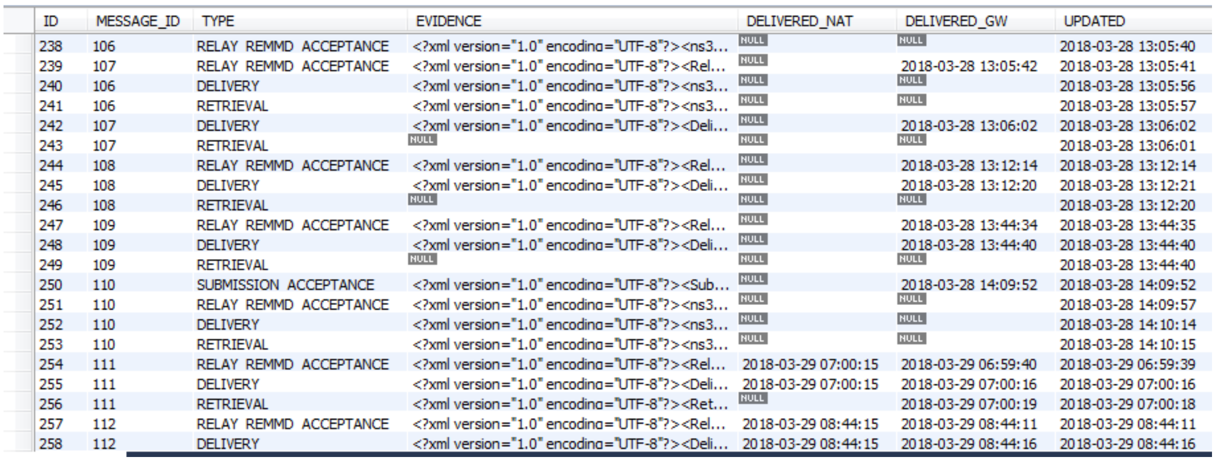
6.2.4. Column DELIVERED_GW description

Is null by default and will be set to the timestamp of the delivery to the gateway. This timestamp is set after the gateway has accepted the message.

6.3. Looking for Evidences / Confirmations

The message confirmations or ETSI-REM [5] Evidences are stored in a different table. This confirmation messages are providing proof that a message has been accepted, relayed, delivered, rejected. The evidences are created by the connector.

The evidences are stored in the DOMIBUS_CONNECTOR_EVIDENCE table and the MESSAGE_ID is referencing the message.



ID	MESSAGE_ID	TYPE	EVIDENCE	DELIVERED_NAT	DELIVERED_GW	UPDATED
238	106	RELAY REMMD ACCEPTANCE	<?xml version="1.0" encoding="UTF-8"?><ns3...	NULL	NULL	2018-03-28 13:05:40
239	107	RELAY REMMD ACCEPTANCE	<?xml version="1.0" encoding="UTF-8"?><Rel...	NULL	2018-03-28 13:05:42	2018-03-28 13:05:41
240	106	DELIVERY	<?xml version="1.0" encoding="UTF-8"?><ns3...	NULL	NULL	2018-03-28 13:05:56
241	106	RETRIEVAL	<?xml version="1.0" encoding="UTF-8"?><ns3...	NULL	NULL	2018-03-28 13:05:57
242	107	DELIVERY	<?xml version="1.0" encoding="UTF-8"?><Deli...	NULL	2018-03-28 13:06:02	2018-03-28 13:06:02
243	107	RETRIEVAL	NULL	NULL	NULL	2018-03-28 13:06:01
244	108	RELAY REMMD ACCEPTANCE	<?xml version="1.0" encoding="UTF-8"?><Rel...	NULL	2018-03-28 13:12:14	2018-03-28 13:12:14
245	108	DELIVERY	<?xml version="1.0" encoding="UTF-8"?><Deli...	NULL	2018-03-28 13:12:20	2018-03-28 13:12:21
246	108	RETRIEVAL	NULL	NULL	NULL	2018-03-28 13:12:20
247	109	RELAY REMMD ACCEPTANCE	<?xml version="1.0" encoding="UTF-8"?><Rel...	NULL	2018-03-28 13:44:34	2018-03-28 13:44:35
248	109	DELIVERY	<?xml version="1.0" encoding="UTF-8"?><Deli...	NULL	2018-03-28 13:44:40	2018-03-28 13:44:40
249	109	RETRIEVAL	NULL	NULL	NULL	2018-03-28 13:44:40
250	110	SUBMISSION ACCEPTANCE	<?xml version="1.0" encoding="UTF-8"?><Sub...	NULL	2018-03-28 14:09:52	2018-03-28 14:09:52
251	110	RELAY REMMD ACCEPTANCE	<?xml version="1.0" encoding="UTF-8"?><ns3...	NULL	NULL	2018-03-28 14:09:57
252	110	DELIVERY	<?xml version="1.0" encoding="UTF-8"?><ns3...	NULL	NULL	2018-03-28 14:10:14
253	110	RETRIEVAL	<?xml version="1.0" encoding="UTF-8"?><ns3...	NULL	NULL	2018-03-28 14:10:15
254	111	RELAY REMMD ACCEPTANCE	<?xml version="1.0" encoding="UTF-8"?><Rel...	2018-03-29 07:00:15	2018-03-29 06:59:40	2018-03-29 06:59:39
255	111	DELIVERY	<?xml version="1.0" encoding="UTF-8"?><Deli...	2018-03-29 07:00:15	2018-03-29 07:00:16	2018-03-29 07:00:16
256	111	RETRIEVAL	<?xml version="1.0" encoding="UTF-8"?><Ret...	NULL	2018-03-29 07:00:19	2018-03-29 07:00:18
257	112	RELAY REMMD ACCEPTANCE	<?xml version="1.0" encoding="UTF-8"?><Rel...	2018-03-29 08:44:15	2018-03-29 08:44:11	2018-03-29 08:44:11
258	112	DELIVERY	<?xml version="1.0" encoding="UTF-8"?><Deli...	2018-03-29 08:44:15	2018-03-29 08:44:16	2018-03-29 08:44:16

Figure 20: Connector 4.0 – example of DOMIBUS_CONNECTOR_EVIDENCE table

Two columns are helping us to inspect the transmission state of the confirmation message:

- DELIVERED_NAT
- DELIVERED_GW

6.3.1. Column DELIVERED_NAT description

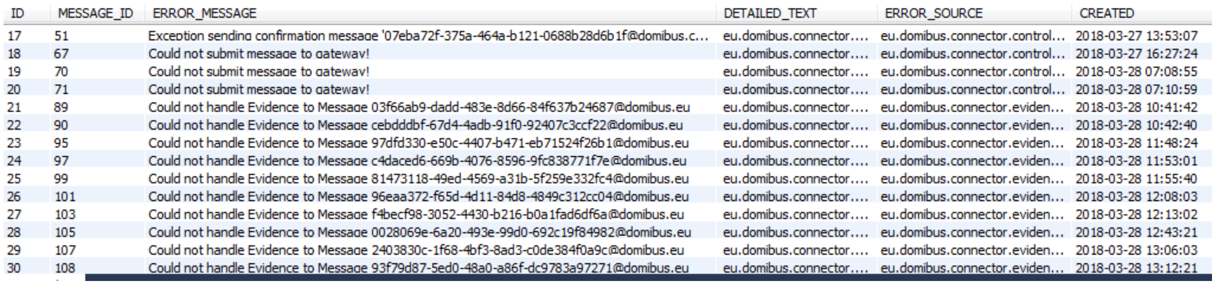
The DELIVERED_NAT column is null by default and will be set to the timestamp when the message is delivered to the national system/connector client. The behaviour of setting this timestamp can be slightly different between the different backends (push/push, push/pull).

6.3.2. Column DELIVERED_GW description

The DELIVERED_GW column is null by default and will be set with the current time when the confirmation message is successfully delivered to the gateway.

6.4. 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 21) 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 gateway!	eu.domibus.connector....	eu.domibus.connector.control...	2018-03-27 16:27:24
19	70	Could not submit message to gateway!	eu.domibus.connector....	eu.domibus.connector.control...	2018-03-28 07:08:55
20	71	Could not submit message to gateway!	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 21: 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 6.5).

6.5. Following the message processing

When the connector is handling a message it assigns a unique connector message id. This message id is also printed out during logging. So this id can be used to follow the message process.

As an example:

The following command on a unix system will show us all log entries related to the message with the connector message id: 03f66ab9-dadd-483e-8d66-84f637b24687@domibus.eu

```
$> cat connector.log | grep msgid=03f66ab9-dadd-483e-8d66-84f637b24687@domibus.eu
```

I. LIST of FIGURES

II. LIST of TABLES

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.