Documentation for Domibus 3.0.1-FINAL 2015-12-07

Christian Koch, IT.NRW
Stefan Müller, IT.NRW

Documentation for Domibus 3.0.1-FINAL 2015-12-07

by Christian Koch and Stefan Müller

Publication date 2015-12-07

Abstract

This is the technical documentation of the Domibus ebMS3/AS4 gateway version (v.3.0.1-FINAL).

Please send comments on this specification to `<christian.koch@it.nrw.de>`

Table of Contents

List of Figures

List of Tables

Glossary

$domibus.local.context	The local, unproxied gateway URL (defaults to `http://localhost:8080/domibus`)
$domibus.config.location	The absolute path of the domibus configuration folder.
$domibus.install	The absolute path of the domibus installation folder.
$RDBMS	The name of the used database server, i.e. *MySQL*
e-CODEX	e-Justice Communication via Online Data EXchange
e-SENS	Electronic Simple European Networked Services

Chapter 1. Introduction

This document is intended for server administrators who are tasked with installing the gateway software. Basic knowledge of operating systems, proxy servers, servlet containers and networking is assumed. This manual is neither intended to be a guide to e-CODEX [http://www.e-codex.eu] or e-SENS [http://www.esens.eu] in general, nor does it contain the preliminary steps required to participate in the e-CODEX or e-SENS project. If you are looking for this kind of information please have a look at the quick start guide of the project you are interested in

1.1. Domibus

Domibus is an ebMS3 gateway which is fully compliant with the [AS4-Profile] based [D6.2] e-SENS profile. A complete feature matrix can be found inAppendix B, *Feature List*.

Chapter 2. Installation preperation

This chapter describes the and required as well as how to obtain the software and what artifact to use.

2.1. System Requirements

Java Runtime	A Java runtime environment version 7 or higher is required.
Servlet Container	A Java Servlet container supporting Servlet v3.0 or higher, Apache Tomcat 8.0 [http://tomcat.apache.org/] is recommended.
Proxy Server	Due to the fact that Domibus exposes multiple webservices and administration interfaces by default the usage of a proxy server is highly recommended. Please see Section 4.2, “Proxy Configuration” for details.
RDBMS	Supported databases are Oracle 10g and higher as well as MySQL 5.5 and higher.
x.509 v3 Certificate with private key	While not required for basic use, a certificate with a public/private keypair is required to support any of the security options regarding encryption and/or signing. Additional requirements i.e. allowed issuers are subject of business agreements.
Hardware Requirements	The required hardware depends on your expected message load. See Appendix D, *Performance Data* for details.

2.2. Software Packages

There are different release packagings available for the e-CODEX gateway:

Table 2.1. Software packages

Filename: domibus-3.0.1-FINAL	**Gateway WAR-file**	**Fully configured Tomcat 8 Server**	**Configuration files**	**SQL-scripts for database generation / updates**
.war	✓	✗	✗	✗
-distribution-war.zip	✓	✗	✓	✓
-tomcat-full.zip	✓	✓	✓	✓
-sql-scripts.zip	✗	✗	✗	✓
-configuration.zip	✗	✗	✓	✗

Important

While the download packages contain database scripts for Apache Derby use of this database engine is only recommended for testing purposes and not for productive use.

2.2.1. domibus-3.0.1-FINAL.war

This package contains only the web application. The usage of this package is only recommended for development and testing purposes

2.2.2. domibus-3.0.1-FINAL-distribution-war.zip

This package contains the web application, tomcat configuration files and database scripts. Use this package if you are performing an update of an existing installation or if you want to deploy the software to a servlet container different from the one that is available in the tomcat-full package.

2.2.3. domibus-3.0.1-FINAL-tomcat-full.zip

This package contains a Tomcat 8 server with a preinstalled Domibus gateway. This package is the easiest way to deploy the Domibus gateway into a testing or production environment. Usage of this package is recommended, and this manual assumes that this package is used for deployment.

2.2.4. domibus-3.0.1-FINAL-tomcat-configuration.zip

This package only contains the configuration files for the application.

2.2.5. domibus-3.0.1-FINAL-sql-scripts.zip

This package only contains initial and delta SQL scripts for MySQL and Oracle database engines.

2.3. Obtaining the software

As of now all Domibus software can be downloaded from the projects repository [https://secure.e-codex.eu/nexus]. Required credentials will be provided to you by the local person responsible for the e-CODEX or e-SENS project. The artifacts can be directly downloaded from here [https://secure.e-codex.eu/nexus/content/groups/public/eu/domibus/domibus/]

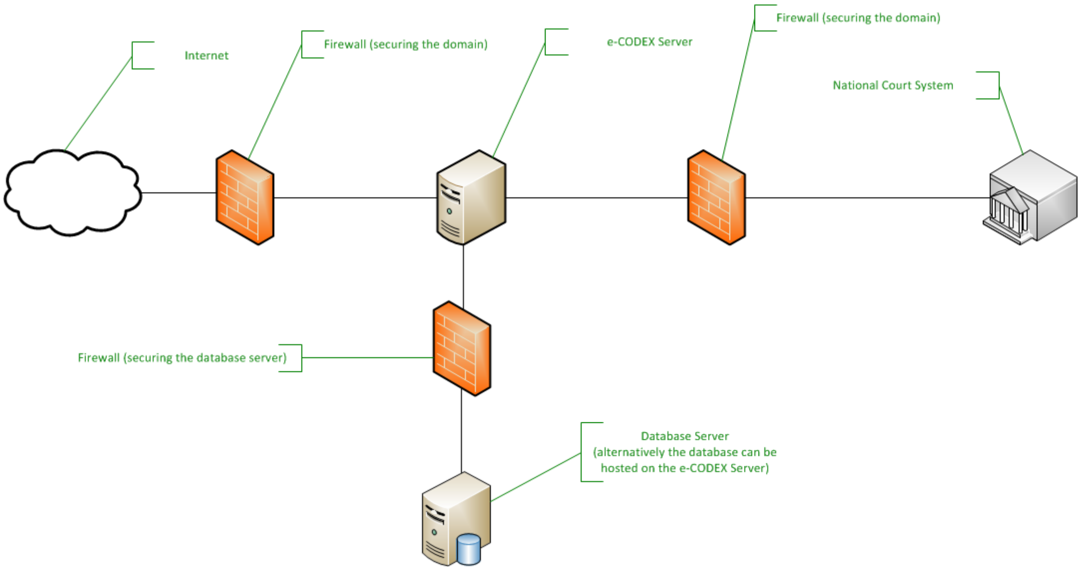
Future planning

It is expected that the final version of Domibus 3 will be hosted on joinup [https://joinup.ec.europa.eu/] by the European Commission.

2.4. System Overview

The Domibus gateway must be installed on a server in a secure networking area behind a firewall and a proxy server (not shown). The database server used by the application can either be deployed on the same server as the gateway or a dedicated database server.

Figure 2.1. System overview


Internet
Firewall (securing the domain)
e-CODEX Server
Firewall (securing the domain)
National Court System
Firewall (securing the database server)
Database Server
(alternatively the database can be
hosted on the e-CODEX Server)

Important

If the Domibus gateway is to deployed as a solution the use of a dedicated database server is required.

Chapter 3. Domibus Deployment

This chapter describes the deployment of Domibus from scratch. If you are performing an update instead please refer to the `upgrade-info.txt` file from your downloaded artifact.

Note

There is no upgrade path from Domibus/Holodeck 2.x and earlier. Domibus 3 requires a fresh install.

3.1. Single server deployment

Prerequisites

a. A running MySQL or Oracle database server

b. A Domibus artifact containing the database scripts

c. A Domibus artifact containing the `.war` and `config` files or the `-tomcat-full` artifact

1. **Prepare the Database**

   a. Create a domibus user and schema. Required userprivileges are *SELECT*, *CREATE_TEMPORARY_TABLE*, *CREATE_ROUTINE*, *INSERT*, *DELETE* and *UPDATE*

   b. Apply the `$RDBMS-initial.ddl` script from the `sql-scripts/` folder to your database.

   c. Apply the `$RDBMS-quartz.ddl` script from the `sql-scripts/` folder to your database.

   For additional information on how to perform these tasks please consult your RDBMS documentation or database administrator.

2. **Deploy the Domibus artifact**

   - **Recommended**

     copy the whole `/domibus` folder from the `tomcat-full` artifact to your server.

   - **Alternative**

     a. deploy the `domibus.war` file to your servlet container. Consult your container vendors documentation on how to perform this.

     b. copy the `conf/` folder from the Domibus artifact to an appropiate location on your hard drive

3. **Set up the environment variables**

   Domibus expects a single JVM-parameter `$domibus.config.location`, pointing towards the `conf/domibus/` folder copied in the previous step. If using the Tomcat distribution of Domibus documentation on how to do this can be found here [https://tomcat.apache.org/tomcat-8.0-doc/RUNNING.txt] (3.3).

4. a. **Configure the container**

   Edit the `$domibus.config.location/domibus-configuration.xml` according to Chapter 4, *Domibus Gateway Configuration*

   b. **Deploy the database driver**

      Make sure the container has the required jdbc compliant database driver available. If you are using the Tomcat distribution of Domibus this is done by copying the `.jar` file containing the driver to the `lib` directory of the container. If you are unsure what driver to use, please consult your database administrator.

5. **Configure the proxy server**

   Configure your proxy server according to Section 4.2, “Proxy Configuration”

6. **Start the container**

   To start the Tomcat container execute the `$domibus.install/bin/startup.sh` (on Linux) or `$domibus.install/bin/startup.bat` (on Windows) script.

7. **Check the administration dashboard**

   If the deployment has been successful you should be able to reach the administrative dashboard described inChapter 5, *Administration Dashboard*.

You now should have a working Domibus deployment. Before you can start exchanging messages you have to provide buisness case specific processing modes (PModes) to the gateway. As those are deployment dependant those PModes are not a part of the Domibus distribution. See Chapter 6, *Processing Mode Configuration* on how to generate your own PMode set.

3.2. Cluster deployment

Chapter 4. Domibus Gateway Configuration

4.1. domibus-configuration.xml

The `domibus-configuration.xml` file located in `$domibus.conflig.location` is the central configuration file for the Domibus gateway.

```
        <?xml version="1.0" encoding="UTF-8"?>
<!--
  ~ Copyright 2015 e-CODEX Project
  ~
  ~ Licensed under the EUPL, Version 1.1 or – as soon they
  ~ will be approved by the European Commission - subsequent
  ~ versions of the EUPL (the "Licence");
  ~ You may not use this work except in compliance with the
  ~ Licence.
  ~ You may obtain a copy of the Licence at:
  ~ http://ec.europa.eu/idabc/eupl5
  ~ Unless required by applicable law or agreed to in
  ~ writing, software distributed under the Licence is
  ~ distributed on an "AS IS" basis,
  ~ WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
  ~ express or implied.
  ~ See the Licence for the specific language governing
  ~ permissions and limitations under the Licence.
  -->

<beans xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:util="http://www.springframework.org/schema/util"
       xmlns:sec="http://www.springframework.org/schema/security"
       xmlns="http://www.springframework.org/schema/beans"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd
    http://www.springframework.org/schema/util
    http://www.springframework.org/schema/util/spring-util.xsd
    http://www.springframework.org/schema/security
    http://www.springframework.org/schema/security/spring-security-3.2.xsd">

    <!-- database configuration -->
    <bean id="entityManagerFactory"
          class="org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean">
        <!-- the datasource to be used by domibus -->
        <property name="dataSource" ref="dataSource"/>
        <!-- package to scan for autodetection of entity classes
        This value should not be changed -->
        <property name="packagesToScan" value="eu.domibus"/>
        <!-- This adapter allows to plug in vendor specific behaviour into the
        Spring EntityManager -->
        <property name="jpaVendorAdapter">
            <!-- By default a HibernateJpaVendorAdapter is used -->
            <bean class="org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter"/>
        </property>
        <!-- The validation mode to be used by the provider for the persistence unit  -->
        <property name="validationMode" value="AUTO"/>
        <!-- Properties for the jpa configuration -->
        <property name="jpaProperties">
            <props>
                <!-- Automatic validation of the database schema when session
                factory is created -could be disabled by setting to "none" -->
                <prop key="hibernate.hbm2ddl.auto">validate</prop>
                <!-- database dialect to use -->
                <prop key="hibernate.dialect">
                    org.hibernate.dialect.MySQL5InnoDBDialect
                </prop>
                <!-- Log all SQL statements to console - FOR DEBUGGING PURPOSES -->
                <prop key="hibernate.show_sql">false</prop>
                <!-- format SQL statements before logging - only used when
                hibernate.show_sql=true -->
                <prop key="hibernate.format_sql">true</prop>
                <!-- generate comments inside SQL statements - only used when
                hibernate.show_sql=true -->
                <prop key="hibernate.use_sql_comments">true</prop>
            </props>
        </property>
    </bean>

    <!-- database pooling provider -->
    <bean id="dataSource" class = "com.mchange.v2.c3p0.ComboPooledDataSource" destroy-
method="close">
        <property name="driverClass" value="com.mysql.jdbc.Driver" />
        <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/domibus"/>
        <property name="user" value="domibus" />
        <property name="password" value="changeit" />

        <!-- these are C3P0 properties -->
        <property name="acquireIncrement" value="1" />
        <property name="minPoolSize" value="3" />
        <property name="maxPoolSize" value="10" />
        <property name="maxIdleTime" value="20000" />
    </bean>



    <!-- Passwordcallback for private key access. By default a simple
         implementation which stores the password in plaintext is used.
         THIS IS NOT SAFE FOR PRODUCTION PURPOSES please switch to
         your own implementation
    -->
    <bean id="keystorePasswordCallback"
          class="eu.domibus.ebms3.security.SimpleKeystorePasswordCallback">
        <!-- Map with "alias" as key and "password" as value.
             This map will be used by the passwordcallback to
             retrieve the private key password for a given alias -->
        <property name="passwordStore">
            <util:map>
                <entry key="blue_gw_sigkey" value="test123456"/>
            </util:map>
        </property>
    </bean>

    <!-- Properties for keystore with private key -->
    <util:properties id="keystoreProperties">
        <!-- The crypto provider to be used -->
        <prop key="org.apache.ws.security.crypto.provider">
            org.apache.ws.security.components.crypto.Merlin
        </prop>
        <!-- Type of the used keystore -->
        <prop key="org.apache.ws.security.crypto.merlin.keystore.type">jks
        </prop>
        <!-- The password used to load the keystore -->
        <prop key="org.apache.ws.security.crypto.merlin.keystore.password">
            test123
        </prop>
        <!-- The keystore alias to use for decryption and signing. -->
        <prop key="org.apache.ws.security.crypto.merlin.keystore.alias">
            blue_gw_sigkey
        </prop>
        <!-- The location of the keystore -->
        <prop key="org.apache.ws.security.crypto.merlin.file">
            ${domibus.config.location}/gateway_keystore_blue.jks
        </prop>
    </util:properties>

    <!-- Properties for truststore with public keys for the partners -->
    <util:properties id="truststoreProperties">
        <!-- The crypto provider to be used -->
        <prop key="org.apache.ws.security.crypto.provider">
            org.apache.ws.security.components.crypto.Merlin
        </prop>
        <!-- Type of the used keystore -->
        <prop key="org.apache.ws.security.crypto.merlin.keystore.type">jks
        </prop>
        <!-- The password used to load the keystore -->
        <prop key="org.apache.ws.security.crypto.merlin.keystore.password">
            test123
        </prop>
        <!-- The location of the keystore -->
        <prop key="org.apache.ws.security.crypto.merlin.file">
            ${domibus.config.location}/gateway_truststore_blue.jks
        </prop>
    </util:properties>

    <!-- Internal configuration provider. Do not change unless you have special requirements
 (i.e. dynamic
    discovery injection of PModes)-->
    <bean id="pModeProvider"
          class="eu.domibus.ebms3.common.dao.PModeDao"
          init-method="init"/>

    <!-- General domibus properties -->
    <util:properties id="domibusProperties">
        <!-- The suffix of the messageId generated by this instance of domibus.
        Schema is:  ${UUID}@${SUFFIX} -->
        <prop key="domibus.msh.messageid.suffix">domibus.eu</prop>
        <!-- Sender Worker execution interval as a cron expression -->
        <prop key="domibus.msh.sender.cron">0/5 * * * * ?</prop>
        <!-- Should unrecoverable errors should be retried or not -->
        <prop key="domibus.dispatch.ebms.error.unrecoverable.retry">true</prop>
    </util:properties>

    <!-- Mimetypes listed in this map will not be compressed (in outgoing messages)
    even if compression is turned on for the given message. -->
    <util:list id="compressionBlacklist" value-type="java.lang.String">
        <value>application/vnd.etsi.asic-s+zip</value>
        <value>image/jpeg</value>
    </util:list>

    <!-- message sender that is being executed by the senderworker -->
    <bean id="messageSender" class="eu.domibus.ebms3.sender.MessageSender"/>

    <!-- quartz job for the  -->
    <bean id="senderWorkerJob"
          class="org.springframework.scheduling.quartz.JobDetailFactoryBean">
        <property name="jobClass" value="eu.domibus.ebms3.sender.SenderWorker"/>
        <property name="durability" value="true"/>
    </bean>

    <bean id="senderWorkerTrigger"
          class="org.springframework.scheduling.quartz.CronTriggerFactoryBean">
        <property name="jobDetail" ref="senderWorkerJob"/>
        <property name="cronExpression" value="${domibus.msh.sender.cron}"/>
    </bean>

    <!-- customizable list of triggers to be executed by quartz -->
    <bean id="userdefinedTriggerList"
          class="org.springframework.beans.factory.config.ListFactoryBean">
        <property name="sourceList">
            <list>
                <ref bean="senderWorkerTrigger"/>
            </list>
        </property>
    </bean>

    <!-- Administration GUI user credentials-->
    <bean name="bcryptEncoder"
          class="org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder"/>

    <sec:authentication-manager>
        <sec:authentication-provider>
            <sec:password-encoder ref="bcryptEncoder"/>
            <sec:user-service>
                <sec:user name="user" password="$2a
$10$HApapHvDStTEwjjneMCvxuqUKVyycXZRfXMwjU0rRmaWMsjWQp/Zu"
                          authorities="ROLE_USER"/>
                <sec:user name="admin" password="$2a
$10$5uKS72xK2ArGDgb2CwjYnOzQcOmB7CPxK6fz2MGcDBM9vJ4rUql36"
                          authorities="ROLE_USER, ROLE_ADMIN"/>
            </sec:user-service>
        </sec:authentication-provider>
    </sec:authentication-manager>


</beans>
```

Table 4.1. Required configuration

Element id	Propery Key	Propery Value
entityManagerFactory	jpaProperties : hibernate.dialect	Dialect of the employed database. Valid values can be found here [https://docs.jboss.org/hibernate/orm/4.3/javadocs/]

Element id	Propery Key	Propery Value
dataSource	driverClassName	jdbc compliant database driver class
dataSource	class	If not using Tomcat, change to your vendors implementation
dataSource	url	Database URL
dataSource	user	Database user
dataSource	password	Database password
keystoreProperties	org.apache.ws.security. crypto.merlin.file	Filename of the keystore containing the private keypair for signing / decryption of messages.
keystoreProperties	org.apache.ws.security. crypto.merlin.keystore.alias	Alias of the key to be used for signing / decryption of messages.
keystoreProperties	org.apache.ws.security. crypto.merlin.keystore.password	Password for the keystore
keystorePasswordCallback	class	An implementation of `javax.security.auth.callback.CallbackHandler` providing the private key password. [a]
truststoreProperties	org.apache.ws.security.crypto. merlin.file	Filename of the keystore containing the public keys for signature verification / encryption of messages.
truststoreProperties	org.apache.ws.security.crypto. merlin.keystore.password	Password for the keystore containing the public keys for signature verification / encryption of messages.
domibusProperties	domibus.msh.messageid.suffix	suffix to be added to ebMS3 message ids (the resulting format will be `UUID@$domibus.msh.messageid.suffix`).
authentication-manager	authentication-provider : user-service : user : password	bcrypt hashed passwords for users of the administrative dashboard (seeChapter 5, *Administration Dashboard*). Defaults to `123456`

[a] The default implementation might be too insecure for your environment. If you decide to use it (i.e. for testing purposes) make sure to set the "passwordStore" entry in the "keystorePasswordCallback" bean to the correct value.

All other properties should be left at the default value.

4.2. Proxy Configuration

The Domibus gateway exposes a number of different services listed in the table below. Configure your proxy server in a way that only authorized clients can reach the services intended for them (i.e. if you are using an Apache2 webserver as proxy use `ProxyPass` and `Allow` directives for the configuration, see the Apache documentation [http://httpd.apache.org/docs/2.2/mod/mod_proxy.html]). The standard server distributed in the tomcat-full package listens on $domibus.local.context [http://localhost:8080/domibus]

Warning

IF YOU DO NOT SECURE THE ACCESS TO THE ADMINISTRATIVE SERVICES THE CONFIDENTIALITY AND INTEGRITY OF YOUR DATA IS AT RISK!

Table 4.2. Exposed interfaces

URL: $domibus.local.context	**Exposed to**	**Description**
/services/msh	business partners	This is the URL of your gateway endpoint.
/services/backend	backend application	This is the URL of the backend webservice delivered with domibus out of the box. If you use a custom interface please make sure it is only exposed to your backend application.
/services	service overview	This is the URL of an overview of all deployed services. For security reasons this should not be exposed to the public.
/home	internal	This is the URL of the administrative dashboard(seeChapter 5, *Administration Dashboard*). Only authorized users/admins should have access to it.

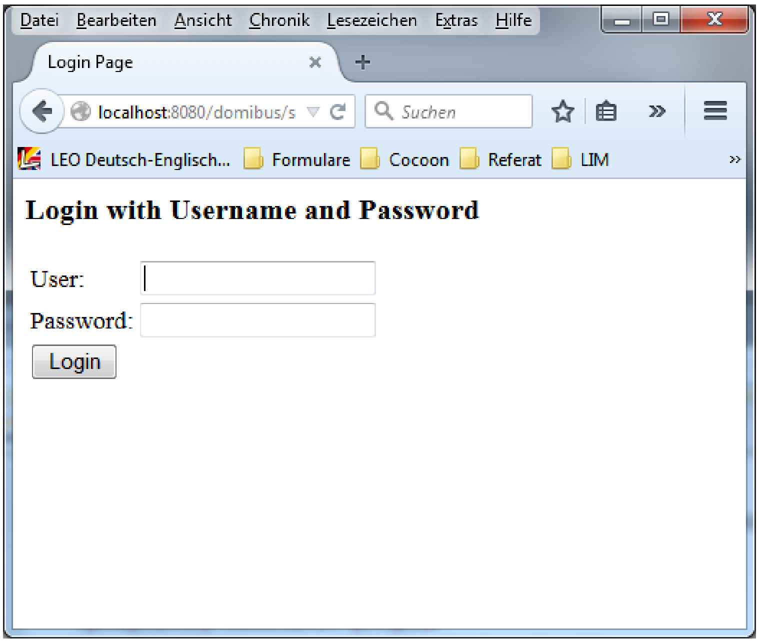
Chapter 5. Administration Dashboard

The Domibus gateway comes with a web based administration interface with the following funtionalities:

- a fully filterable message logging view
- a fully filterable error logging view
- deployment of processing modes (users with adminstrative privileges only)

The login page of the administration dashboard can be found at: $domibus.local.contex/home (e.g. `http://localhost:8080/domibus/home`)

Figure 5.1. Login page of adminstration dashboard



Login with Username and Password

User:

Password:

Note

The default password for both users (`user, admin`) is`123456`.

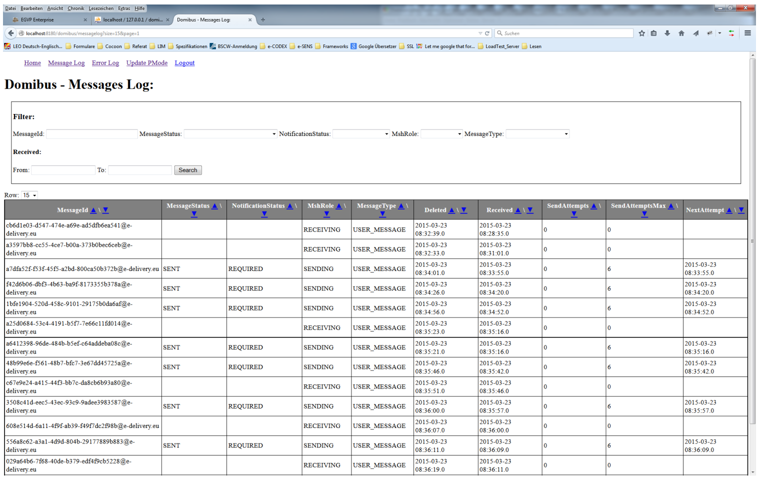
Important

Change the password for both users before starting the gateway the first time. In addition your proxy configuration should already restrict access to the administration dashboard. If not see Section 4.2, “Proxy Configuration”

5.1. Message Logging

The message logging page contains a table, showing all the messages processed by your gateway (sending and receiving). This table is sortable and filterable by each column. Below you can find a description of the table columns.

Figure 5.2. Message logging page of adminstration dashboard


Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
EGVP Enterprise
localhost / 127.0.0.1 / domi...
Domibus - Messages Log:
localhost:8180/domibus/messagelog?size=15&page=1
LEO Deutsch-Englisch... Formulare Cocoon Referat LIM Spezifikationen BSCW-Anmeldung e-CODEX e-SENS Frameworks Google Übersetzer
Home Message Log Error Log Update PMode Logout
Search

Domibus - Messages Log:

Filter:

MessageId: MessageStatus: NotificationStatus: MshRole: MessageType:

Received:

From: To:

Row: 15

MessageId	MessageStatus	NotificationStatus	MshRole	MessageType	Deleted	Received	SendAttempts	SendAttemptsMax	NextAttempt
cb6d1e03-d547-474e-a69e-ad5dfb6ea541@e-delivery.eu			RECEIVING	USER_MESSAGE	2015-03-23 08:32:39.0	2015-03-23 08:28:35.0	0	0	
a3597bb8-cc55-4ce7-b00a-373b0bec6ceb@e-delivery.eu			RECEIVING	USER_MESSAGE	2015-03-23 08:32:33.0	2015-03-23 08:31:01.0	0	0	
a7dfa52f-f53f-45f5-a2bd-800ca50b372b@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-03-23 08:34:01.0	2015-03-23 08:33:55.0	0	6	2015-03-23 08:33:55.0
f42d6b06-dbf3-4b63-ba9f-8173355b378a@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-03-23 08:34:26.0	2015-03-23 08:34:20.0	0	6	2015-03-23 08:34:20.0
1bfe1904-520d-458c-9101-29175b0da6af@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-03-23 08:34:56.0	2015-03-23 08:34:52.0	0	6	2015-03-23 08:34:52.0
a25d0684-53c4-4191-b5f7-7e66c11fd014@e-delivery.eu			RECEIVING	USER_MESSAGE	2015-03-23 08:35:23.0	2015-03-23 08:35:16.0	0	0	
a6412398-96de-484b-b5ef-c64addeba08c@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-03-23 08:35:21.0	2015-03-23 08:35:16.0	0	6	2015-03-23 08:35:16.0
48b99e6e-f561-48b7-bfc7-3e67dd45725a@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-03-23 08:35:46.0	2015-03-23 08:35:42.0	0	6	2015-03-23 08:35:42.0
c67e9e24-a415-44f3-bb7c-da8cb6b93a80@e-delivery.eu			RECEIVING	USER_MESSAGE	2015-03-23 08:35:51.0	2015-03-23 08:35:46.0	0	0	
3508c41d-eec5-43ec-93c9-9adee3983587@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-03-23 08:36:00.0	2015-03-23 08:35:57.0	0	6	2015-03-23 08:35:57.0
608e514d-6a11-4f9f-ab39-f49f7dc2f98b@e-delivery.eu			RECEIVING	USER_MESSAGE	2015-03-23 08:36:07.0	2015-03-23 08:36:00.0	0	0	
556a8c62-a3a1-4d9d-804b-29177889b883@e-delivery.eu	SENT	REQUIRED	SENDING	USER_MESSAGE	2015-03-23 08:36:11.0	2015-03-23 08:36:09.0	0	6	2015-03-23 08:36:09.0
029a64b6-7f68-40de-b379-edf4f9cb5228@e-delivery.eu			RECEIVING	USER_MESSAGE	2015-03-23 08:36:19.0	2015-03-23 08:36:11.0	0	0	

MessageId	The ebms message id
MessageStatus	Status of this message. Possible values are: SENT, FAILED, IN_PROGRESS, SENT_WITH_WARNINGS, RECEIVED, RECEIVED_WITH_WARNINGS
NotificationStatus	When producer/consumer have to be notified about errors the value is REQUIRED, if not the value is NOT_REQUIRED. When producer/consumer already have been notified the value is NOTIFIED
MshRole	The status of this message. Possible values are: SENDING, RECEIVING
MessageType	The (ebms) type of the message. Possible value are: SIGNAL_MESSAGE, USER_MESSAGE
Deleted	The date when this message was deleted. In case of sending: • An outgoing message has been sent without error eb:Error/@severity failure failure, and an AS4 receipt has been received • An outgoing message has been sent without error eb:Error/@severity failure, and AS4 is disabled • An outgoing message could not be sent and the final AS4 retry has passed • An outgoing message could not be sent and AS4 is disabled (eb:Error/@severity failure, [CORE 6.2.5]) In case of a received message, the message is deleted after it has been downloaded from a backend consumer.
Received	The date when a message was received either by the backend (sending) or another gateway (receiving).
SendAttempts	The (current) number of attempts that have been made in order to send the message.
SendAttemptsMax	The maximum number of attempts configured (via PMode) for this message transmission.

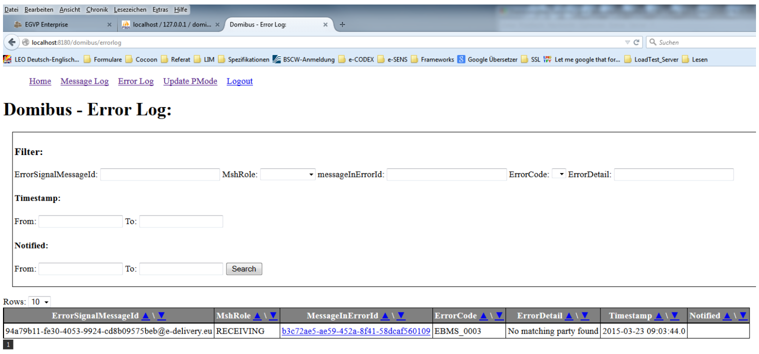
NextAttempt	The date when the next attempt takes place.

5.2. Error Logging

This page contains a table with ebms3 error that have been received by another gateway or created by this gateway while sending a message.

ErrorSignalMessageId	The ebms message id of the signal message which contained the ebms3 error.
MshRole	The status of this message. Possible values are: SENDING, RECEIVING;
MessageInErrorId	The id of the message which caused this error.
ErrorCode	The error code of this ebms error.
ErrorDetail	A short description of this error.
Timestamp	The date when this error was received (in case of receiving) or created (in case of sending).
Notified	The date when a producer/consumer was notified about this error.

Figure 5.3. Error logging page of adminstration dashboard


Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
EGVP Enterprise
localhost / 127.0.0.1 / domi...
Domibus - Error Log:
localhost:8180/domibus/errorlog
Suchen
LEO Deutsch-Englisch... Formulare Cocoon Referat LIM Spezifikationen BSCW-Anmeldung e-CODEX e-SENS Frameworks Google Übersetzer SSL Let me google that for... LoadTest_Server Lesen
Home Message Log Error Log Update PMode Logout
Search
1

Domibus - Error Log:

Filter:

ErrorSignalMessageId: MshRole: messageInErrorId: ErrorCode: ErrorDetail:

Timestamp:

From: To:

Notified:

From: To:

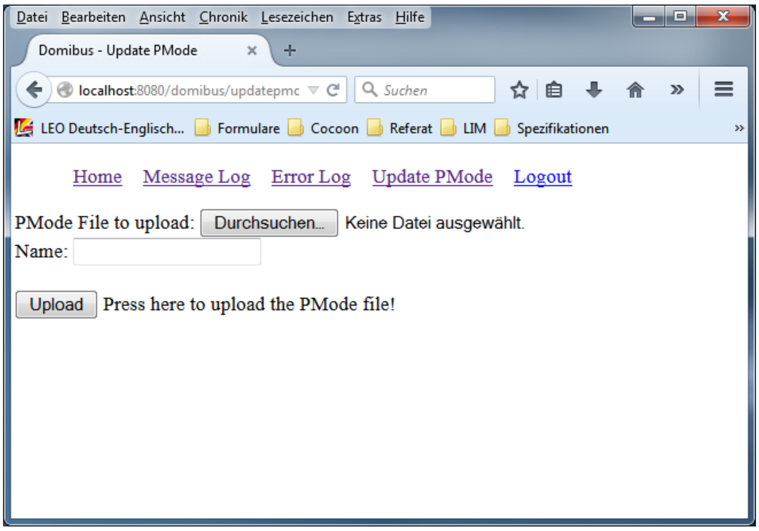
Rows: 10

ErrorSignalMessageId ▲ \ ▼	MshRole ▲ \ ▼	MessageInErrorId ▲ \ ▼	ErrorCode ▲ \ ▼	ErrorDetail ▲ \ ▼	Timestamp ▲ \ ▼	Notified ▲ \ ▼
94a79b11-fe30-4053-9924-cd8b09575beb@e-delivery.eu	RECEIVING	b3c72ae5-ae59-452a-8f41-58dcaf560109	EBMS_0003	No matching party found	2015-03-23 09:03:44.0	

5.3. PMode Upload

This section is only accesible by administrators and provides a pmode upload functionality. You can choose a pmode configuration (xml) on your local machine and apply it to the gateway. The currenty pmode configuration is then replaced.

Figure 5.4. PMode Upload


Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
Domibus - Update PMode
localhost:8080/domibus/updatepmc
Suchen
LEO Deutsch-Englisch... Formulare Cocoon Referat LIM Spezifikationen
Home Message Log Error Log Update PMode Logout
Durchsuchen...
Upload

PMode File to upload: Keine Datei ausgewählt.
Name:

Press here to upload the PMode file!

Chapter 6. Processing Mode Configuration

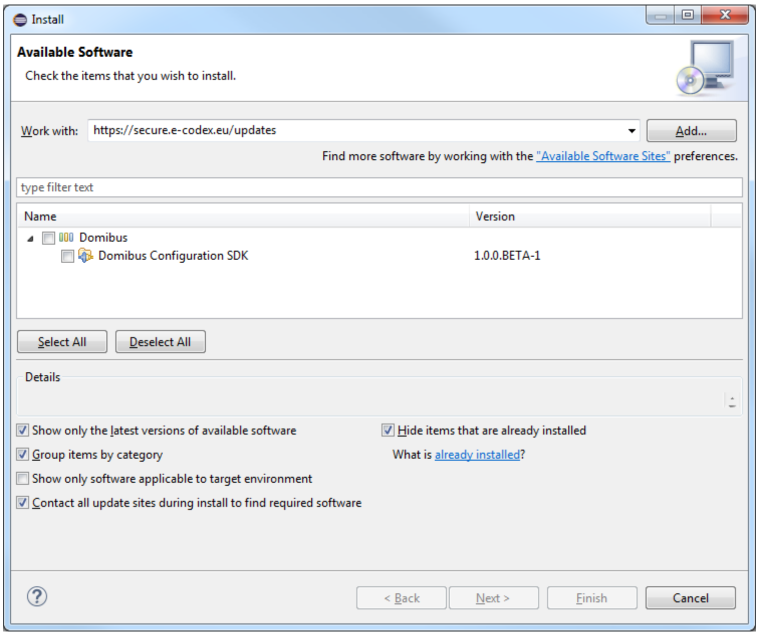
ebMS3 relies on so-called processing modes (PModes) to describe how messages between business partners are to be exchanged.

6.1. Generating Processing modes

6.1.1. Setting up the environment

Domibus provides a plugin for the Eclipse IDE which can be used to genetate configuration XML files. Eclipse can be obtained for free here [http://www.eclipse.org/]. The plugin has been tested with Eclipse 4.4, but should work for all recent versions.

To install the plugin use `https://secure.e-codex.eu/updates` as updatesite.


Install
Available Software
Check the items that you wish to install.
Add...
Find more software by working with the "Available Software Sites" preferences.
type filter text
Name
Version
Select All
Deselect All
Details
What is already installed?
< Back
Next >
Finish
Cancel

Work with: https://secure.e-codex.eu/updates

Name	Version
☐ Domibus	
☐ Domibus Configuration SDK	1.0.0.BETA-1

☑ Show only the latest versions of available software
☑ Group items by category
☐ Show only software applicable to target environment
☑ Contact all update sites during install to find required software
☑ Hide items that are already installed

This plugin requires the Xtext SDK as dependency. It should be installed automatically. If that is not the case you can install it manually from the default Eclipse update sites.

To start writing the configuration create a new Eclipse general project. Create a file with a `.pconf` extension. The plugin adds the required project configuration automatically. Syntax highlighting and code completion are supported.

Important

The `.pconf` file is used to generate the concrete Domibus PMode XML configuration files. It is not understood directly by Domibus.

6.1.2. Writing the configuration

The configutration is written in a domain specific language which is then transformed to XML. An example file looks like this:

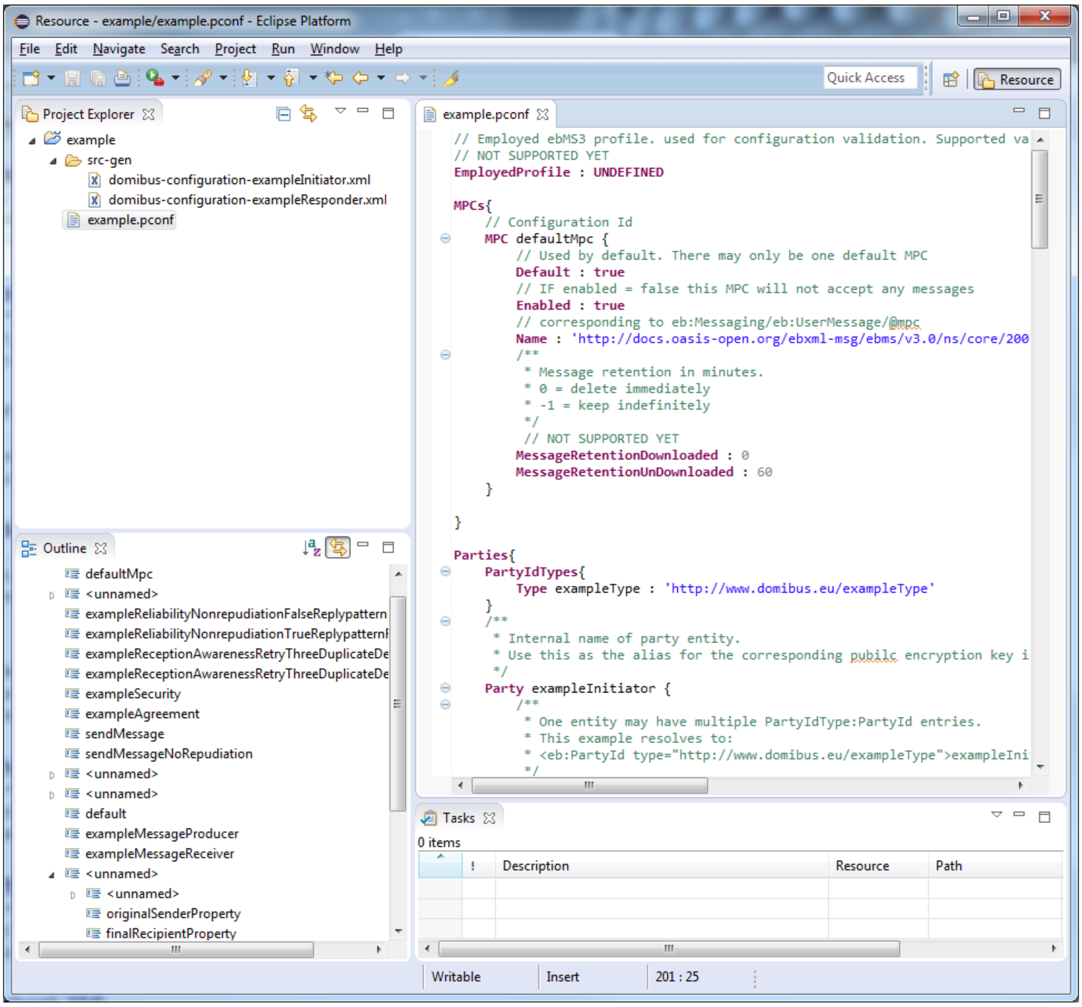
```
                // Employed ebMS3 profile. used for configuration validation. Supported values
 are: AS4, ESENS, UNDEFINED
// NOT SUPPORTED YET
EmployedProfile : UNDEFINED
```

```
MPCs{
 // Configuration Id
 MPC defaultMpc {
  // Used by default. There may only be one default MPC
  Default : true
  // IF enabled = false this MPC will not accept any messages
  Enabled : true
  // corresponding to eb:Messaging/eb:UserMessage/@mpc
  Name : 'http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/defaultMPC'
  /**
   * Message retention in minutes.
   * 0 = delete immediately
   * -1 = keep indefinitely
   */
  MessageRetentionDownloaded : 0
  MessageRetentionUnDownloaded : 60
 }

}

Parties{
 PartyIdTypes{
  Type exampleType : 'http://www.domibus.eu/exampleType'
 }
 /**
  * Internal name of party entity.
  * Use this as the alias for the corresponding pubilc encryption key in your keystore.
  */
 Party exampleInitiator {
  /**
   * One entity may have multiple PartyIdType:PartyId entries.
   * This example resolves to:
   * <eb:PartyId type="http://www.domibus.eu/exampleType">exampleInitiatorId</eb:PartyId>
   */
  exampleType : 'exampleInitiatorId'
  Endpoint : 'http://localhost:8080/domibus/services/msh'
  /**
   * This value is interpreted by the employed submission module,
   * if an errorHandling with enabled process error notifications is used.
   * The format is dependant on the submission module. I.e. a JMS module might expect a
 queuename
   * while a webservice module might expect an URL. As the default submission module has no
   * back channel to the backend this value is ignored.
   */
  SenderErrorsTo : 'http://localhost:9090/errorhandling'
  ReceiverErrorsTo : 'http://localhost:9090/errorhandling'
 }
 Party exampleResponder {
  exampleType : 'exampleResponderId'
  Endpoint : 'http://localhost:8180/domibus/services/msh'
  SenderErrorsTo : 'http://localhost:9190/errorhandling'
  ReceiverErrorsTo : 'http://localhost:9190/errorhandling'
 }

}

AS4{
 // AS4 reliability configuration.
 // REPLY PATTERN CALLBACK IS NOT SUPPORTED YET
 Reliability exampleReliabilityNonrepudiationFalseReplypatternResponse : Nonrepudiation  = false
 Replypattern = response
 Reliability exampleReliabilityNonrepudiationTrueReplypatternResponse : Nonrepudiation  = true
 Replypattern = response
 /**
  * AS4 reception awareness configuration.
  * retryTimeout in minutes
  * only supported strategy is CONSTANT
  * duplicateDetection checks the full TB_MESSAGE_LOG table for duplicates
  */
 ReceptionAwareness exampleReceptionAwarenessRetryThreeDuplicateDetectionFalse : retryTimeout =
 1 retryCount = 6 strategy = CONSTANT duplicateDetection = false
 ReceptionAwareness exampleReceptionAwarenessRetryThreeDuplicateDetectionTrue : retryTimeout = 1
 retryCount = 6 strategy = CONSTANT duplicateDetection = true
}

Securities{
 Security exampleSecurity {
  // WS-SecurityPolicy file in ${domibus.config.location}
  Policy : 'exampleSecurityPolicy.xml'
  // Allowed values are: RSA_SHA256, RSA_SHA1
  SignatureMethod : RSA_SHA256
 }
```

```
}

BusinessProcessConfiguration{

 Agreements{
  // resolves to: <eb:AgreementRef>http://domibus.eu/agreement</eb:AgreementRef>
  Agreement exampleAgreement : 'http://domibus.eu/agreement'
 }

 Actions{
  // resolves to: <eb:Action>SendMessage</eb:Action>
  Action sendMessage : 'SendMessage'
  Action sendMessageNoRepudiation : 'SendMessageNoRepudiation'
 }

 Services{
  ServiceTypes {
   Type exampleServiceType : 'exampleService'
  }
  // resolves to: <eb:Service type="exampleService">AS4</eb:Service>
  Service as4Service : 'AS4' type = exampleServiceType Actions {sendMessage
 sendMessageNoRepudiation}
 }

 ExchangePatterns{
  Patterns{
   MEP oneway : 'http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/oneWay' Legs = 1
   // NOT SUPPORTED YET
   MEP twoway : 'http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/twoWay' Legs = 2
  }
  Bindings{
   Binding push : 'http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/push'
   // NOT SUPPORTED YET
   Binding pull : 'http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/pull'
   Binding sync : 'http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/sync'
  }
 }

 Roles{
  // resolves to: <eb:Role>http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/
defaultRole</eb:Role>
  Role default : 'http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/defaultRole'
  Role exampleMessageProducer : 'exampleMessageProducer'
  Role exampleMessageReceiver : 'exampleMessageReceiver'
 }

 Processes{
  PayloadProfiles{
   Payloads{
    Payload asicsPayload : cid = 'asic-s@e-codex.eu' mime = 'application/vnd.etsi.asic-s+zip'
 required = true
    Payload businessContentInBody : cid = '' mime = 'text/xml' inBody = true required = true
    Payload businessContentAsAttachment : cid = 'businessDocement@e-codex.eu' mime = 'text/xml'
 required = true
    Payload submissionAcceptanceEvidence : cid = 'submissionAcceptance@e-codex.eu' mime = 'text/
xml' required = true
   }
   Profile ecodexMessageProfile : asicsPayload businessContentInBody
 submissionAcceptanceEvidence maxSize=1024
  }

  PropertySets{
   Properties{
    Property originalSenderProperty : key=originalSender type = string required= true
    Property finalRecipientProperty : key=finalRecipient type = string required= true
   }
   Set ecodexPropertySet: finalRecipientProperty originalSenderProperty
  }

  ErrorHandlings{
   ErrorHandling ecodexErrorHandling {
    ErrorAsResponse : true
    ProcessErrorNotifyProducer : true
    // Note that only errorneous messages with a valid pmode can be reported
    ProcessErrorNotifyConsumer : true
    DeliveryFailureNotifyProducer : true
   }
  }

  Legs{
   Leg examplePushLegOne {
    ReceptionAwareness : exampleReceptionAwarenessRetryThreeDuplicateDetectionTrue
    PayloadProfile : ecodexMessageProfile
    ProperySet : ecodexPropertySet
    Service : as4Service
    Action : sendMessage
    DefaultMpc : defaultMpc
    mpc exampleInitiator : defaultMpc
    mpc exampleResponder : defaultMpc
    Security : exampleSecurity
    Reliability : exampleReliabilityNonrepudiationTrueReplypatternResponse
    ErrorHandling : ecodexErrorHandling
    CompressPayloads : false
   }
   Leg examplePushLegOneNoNonRepudiation {
    ReceptionAwareness : exampleReceptionAwarenessRetryThreeDuplicateDetectionTrue
    PayloadProfile : ecodexMessageProfile
    ProperySet : ecodexPropertySet
    Service : as4Service
    Action : sendMessageNoRepudiation
    DefaultMpc : defaultMpc
    mpc exampleInitiator : defaultMpc
    mpc exampleResponder : defaultMpc
    Security : exampleSecurity
    Reliability : exampleReliabilityNonrepudiationFalseReplypatternResponse
    ErrorHandling : ecodexErrorHandling
    CompressPayloads : false
   }
   Leg examplePushLegOneNoNonRepudiationCompressionTrue {
    ReceptionAwareness : exampleReceptionAwarenessRetryThreeDuplicateDetectionTrue
    PayloadProfile : ecodexMessageProfile
    ProperySet : ecodexPropertySet
    Service : as4Service
    Action : sendMessageNoRepudiation
    DefaultMpc : defaultMpc
    mpc exampleInitiator : defaultMpc
    mpc exampleResponder : defaultMpc
    Security : exampleSecurity
    Reliability : exampleReliabilityNonrepudiationFalseReplypatternResponse
    ErrorHandling : ecodexErrorHandling
    CompressPayloads : true
   }

  }

  Process as4exampleProcess{
   Agreement : exampleAgreement
   Mep : oneway
   Binding : push
   InitiatorRole : exampleMessageProducer
   ResponderRole : exampleMessageReceiver
   // all parties that are allowed to send a message as initiator
   InitiatorParties : exampleInitiator, exampleResponder
   // all parties that are allowed to send a message as responder
   ResponderParties : exampleResponder, exampleInitiator
   Legs : examplePushLegOne
  }

  Process as4exampleNoNonRepudiation{
   Agreement : exampleAgreement
   Mep : oneway
   Binding : push
   InitiatorRole : exampleMessageProducer
   ResponderRole : exampleMessageReceiver
   InitiatorParties : exampleInitiator
   ResponderParties : exampleResponder
   Legs : examplePushLegOneNoNonRepudiation
  }
  Process as4exampleNoNonRepudiationCompressionTrue{
   Agreement : exampleAgreement
   Mep : oneway
   Binding : push
   InitiatorRole : exampleMessageProducer
   ResponderRole : exampleMessageReceiver
   InitiatorParties : exampleInitiator
   ResponderParties : exampleResponder
   Legs : examplePushLegOneNoNonRepudiationCompressionTrue
  }
 }
}
```

Note

If the file is valid the plugin generates one xml file per `Party` on the fly. Those can be found in the `src-gen` folder of the project.


Resource - example/example.pconf - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Quick Access
Resource
Project Explorer
example
src-gen
domibus-configuration-exampleInitiator.xml
domibus-configuration-exampleResponder.xml
example.pconf
Outline
defaultMpc
<unnamed>
exampleReliabilityNonrepudiationFalseReplypattern
exampleReliabilityNonrepudiationTrueReplypatternf
exampleReceptionAwarenessRetryThreeDuplicateDe
exampleReceptionAwarenessRetryThreeDuplicateDe
exampleSecurity
exampleAgreement
sendMessage
sendMessageNoRepudiation
<unnamed>
<unnamed>
default
exampleMessageProducer
exampleMessageReceiver
<unnamed>
<unnamed>
originalSenderProperty
finalRecipientProperty
example.pconf
Tasks
0 items
!
Description
Resource
Path
Writable
Insert
201 : 25

```
// Employed ebMS3 profile. used for configuration validation. Supported va
// NOT SUPPORTED YET
EmployedProfile : UNDEFINED

MPCs{
    // Configuration Id
    MPC defaultMpc {
        // Used by default. There may only be one default MPC
        Default : true
        // IF enabled = false this MPC will not accept any messages
        Enabled : true
        // corresponding to eb:Messaging/eb:UserMessage/@mpc
        Name : 'http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200
        /**
         * Message retention in minutes.
         * 0 = delete immediately
         * -1 = keep indefinitely
         */
         // NOT SUPPORTED YET
        MessageRetentionDownloaded : 0
        MessageRetentionUnDownloaded : 60
    }

}

Parties{
    PartyIdTypes{
        Type exampleType : 'http://www.domibus.eu/exampleType'
    }
    /**
     * Internal name of party entity.
     * Use this as the alias for the corresponding pubilc encryption key i
     */
    Party exampleInitiator {
        /**
         * One entity may have multiple PartyIdType:PartyId entries.
         * This example resolves to:
         * <eb:PartyId type="http://www.domibus.eu/exampleType">exampleIni
         */
```

6.1.3. Installing the configuration

As of Domibus 3.0 Beta 2 it is possible to upload pmodes configuration files via the administration dashboard. Please refer to Section 5.3, “PMode Upload”

Appendix A. Revision History

Revision History

Revision 0.1	19.02.2015	Christian Koch
Initial draft		
Revision 0.2	03.03.2015	Christian Koch
Improved formatting		

Appendix B. Feature List

Appendix C. Backend Integration

C.1. General Overview

To allow integration into any desired infrastructure, Domibus provides a flexible plugin system, supporting arbitrary transport protocols, formats and securities.

C.2. Plugin Structure

Each plugin must contain implementations of `eu.domibus.submission.BackendConnector<U,T>`, `eu.domibus.submission.transformer.MessageSubmissionTransformer<U>` and `eu.domibus.submission.transformer.MessageRetrievalTransformer<T>`.

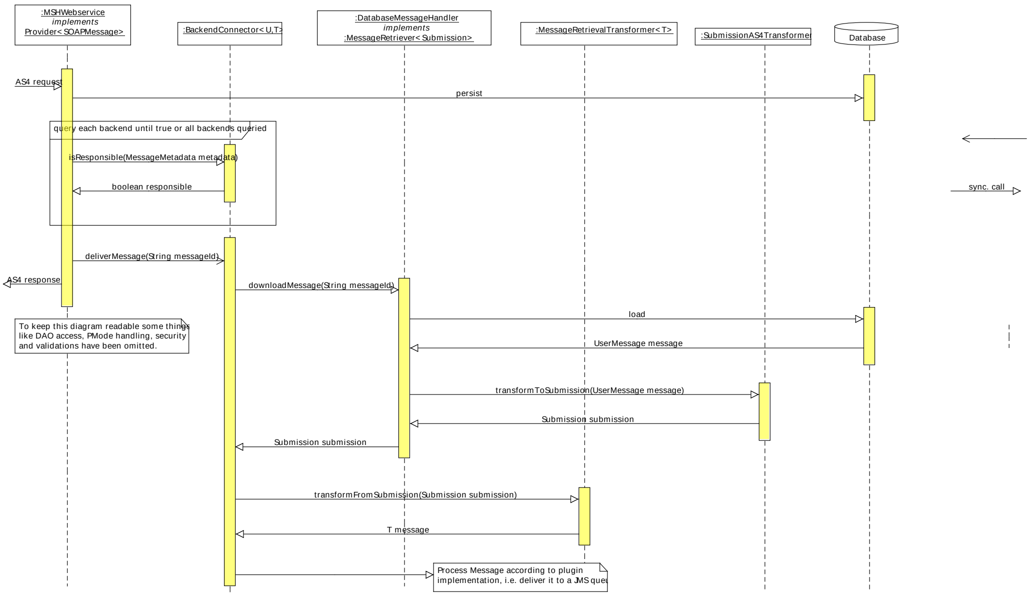
The `BackendConnector<U,T>` implementation is responsible for the connection to the backend system, receiving messages in the `U` format and delivering messages in the `T` format. The `Message*Transformer` classes are responsible for the transformation between`U`, `eu.domibus.submission.Submission` (the Domibus 3 internal data format, for documentation refer to the according JavaDoc) and `T`.

This way multiple plugins can share the same data formats while using different transport protocols or enforcing different security policies. It also is possible to implement transport handlers for protocols while keeping the actual data format pluggable.

C.2.1. Message flow

Figure C.1. Message submission from the backend

Figure C.2. Message reception by the backend and delivery to the plugin



:MSHWebservice
implements
Provider< SOAPMessage>
:BackendConnector< U,T>
:DatabaseMessageHandler
implements
:MessageRetriever< Submission>
:MessageRetrievalTransformer< T>
:SubmissionAS4Transformer
Database
AS4 request
persist
query each backend until true or all backends queried
isResponsible(MessageMetadata metadata)
boolean responsible
sync. call
deliverMessage(String messageId)
AS4 response
downloadMessage(String messageId)
load
UserMessage message
transformToSubmission(UserMessage message)
Submission submission
Submission submission
transformFromSubmission(Submission submission)
T message

To keep this diagram readable some things like DAO access, PMode handling, security and validations have been omitted.

Process Message according to plugin implementation, i.e. deliver it to a JMS queu

C.2.2. Code examples

C.2.2.1. eu.domibus.submission.BackendConnector<U,T>

It is highly recommended to extend `eu.domibus.submission.AbstractBackendConnector<U,T>` for this implementation.

Important

It is vital to still use the `implements` clause as otherwise the Spring framework will produce hard to debug errors.

```
                    package eu.domibus.submission;

import eu.domibus.submission.transformer.MessageRetrievalTransformer;
import eu.domibus.submission.transformer.MessageSubmissionTransformer;
import eu.domibus.submission.transformer.exception.TransformationException;
import eu.domibus.submission.validation.exception.ValidationException;
import org.springframework.scheduling.annotation.Async;
import org.springframework.scheduling.annotation.AsyncResult;

import java.util.concurrent.Future;

public class ExampleBackendConnector extends AbstractBackendConnector<MyObjectIn, MyObjectOut>
                                     implements BackendConnector<MyObjectIn, MyObjectOut>{
                                    //The implements clause is important as Spring got some
 issues with abstract classes

    private MessageSubmissionTransformer<MyObjectIn> messageSubmissionTransformer;
    private MessageRetrievalTransformer<MyObjectOut> messageRetrievalTransformer;

    @Override
    public MessageSubmissionTransformer<MyObjectIn> getMessageSubmissionTransformer() {
        return this.getMessageSubmissionTransformer();
    }

    public void setMessageSubmissionTransformer(MessageSubmissionTransformer<MyObjectIn>
 messageSubmissionTransformer) {
        this.messageSubmissionTransformer = messageSubmissionTransformer;
    }

    @Override
    public MessageRetrievalTransformer<MyObjectOut> getMessageRetrievalTransformer() {
        return this.getMessageRetrievalTransformer();
    }

    public void setMessageRetrievalTransformer(MessageRetrievalTransformer<MyObjectOut>
 messageRetrievalTransformer) {
        this.messageRetrievalTransformer = messageRetrievalTransformer;
    }

    @Override
    public boolean isResponsible(MessageMetadata metadata) {
        /*
        This is the most simple case. If you have only one active
        plugin it automatically is responsible for all messages.
        */
        return true;
    }

    @Override
    @Async
    public Future<Boolean> deliverMessage(final MessageMetadata metadata) {
        //example of how to handle async calls with spring
        return new AsyncResult<>(this.submitToBackend(metadata.getMessageId()));
    }

    private boolean submitToBackend(String messageId) {
        /*
        Here you could put the message on a queue, call an external webservice or store it on
 the file system.
         */

        throw new UnsupportedOperationException("Needs to be implemented");
    }

    /*
    This is an example of the method that is called from outside. In a JMS implementation it
 would most likely be
    called onMessage(Message message), while in a webservice plugin it would have the name of
 the webservice operation.
    */
    public String handleMessageFromBackend(MyObjectIn messageFromBackend) throws
 ValidationException, TransformationException {
        /*
         Handle stuff specific to your implementation, i.e. get message from JMS or webservice,
 check security etc.
         */
        return submit(messageFromBackend);
    }

}
```

C.2.2.2. eu.domibus.submission.transformer.Message*Transformer<U|T>

The transformer classes are responsible for the transformation from/to a `Submission` object. The difficulty of the concrete implementation is dependant on the data model used for communication between the backend and the plugin. It can range from calling a bunch of setters to implementing some dedicated dynamic discovery logic.

Note

While it is a goal of Domibus 3 to make plugin development as easy as possible, some basic AS4 knowledge is required to successfuly implement a working transformer.

A good example for a transformer implementation is `eu.domibus.submission.transformer.impl.JMSMessageTransformer` which transforms `MapMessage` objects conforming to the e-CODEX JMS specification to Submission objects.

```
                    /*
 * Copyright 2015 e-CODEX Project
 *
 * Licensed under the EUPL, Version 1.1 or – as soon they
 * will be approved by the European Commission - subsequent
 * versions of the EUPL (the "Licence");
 * You may not use this work except in compliance with the
 * Licence.
 * You may obtain a copy of the Licence at:
 * http://ec.europa.eu/idabc/eupl5
 * Unless required by applicable law or agreed to in
 * writing, software distributed under the Licence is
 * distributed on an "AS IS" basis,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied.
 * See the Licence for the specific language governing
 * permissions and limitations under the Licence.
 */

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

package eu.domibus.submission.transformer.impl;


import eu.domibus.common.model.org.oasis_open.docs.ebxml_msg.ebms.v3_0.ns.core._200704.Property;
import eu.domibus.submission.Submission;
import eu.domibus.submission.transformer.MessageRetrievalTransformer;
import eu.domibus.submission.transformer.MessageSubmissionTransformer;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.springframework.stereotype.Service;

import javax.jms.JMSException;
import javax.jms.MapMessage;
import java.text.MessageFormat;
import java.util.Map;
import java.util.Properties;


/**
 * This class is responsible for transformations from {@link javax.jms.MapMessage} to {@link
 eu.domibus.submission.Submission} and vice versa
 *
 * @author Padraig
 */

@Service
public class JMSMessageTransformer
        implements MessageRetrievalTransformer<MapMessage>,
 MessageSubmissionTransformer<MapMessage> {
    public static final String SUBMISSION_JMS_MAPMESSAGE_ACTION = "Action";
    public static final String SUBMISSION_JMS_MAPMESSAGE_SERVICE = "Service";
    public static final String SUBMISSION_JMS_MAPMESSAGE_SERVICE_TYPE = "serviceType";
    public static final String SUBMISSION_JMS_MAPMESSAGE_CONVERSATION_ID = "ConversationID";
    public static final String SUBMISSION_JMS_MAPMESSAGE_AGREEMENT_REF = "AgreementRef";
    public static final String SUBMISSION_JMS_MAPMESSAGE_REF_TO_MESSAGE_ID = "refToMessageId";

    public static final String SUBMISSION_JMS_MAPMESSAGE_FROM_PARTY_ID = "fromPartyID";
    public static final String SUBMISSION_JMS_MAPMESSAGE_FROM_PARTY_TYPE = "fromPartyType";
    public static final String SUBMISSION_JMS_MAPMESSAGE_FROM_ROLE = "fromRole";

    public static final String SUBMISSION_JMS_MAPMESSAGE_TO_PARTY_ID = "toPartyID";
    public static final String SUBMISSION_JMS_MAPMESSAGE_TO_PARTY_TYPE = "toPartyType";
    public static final String SUBMISSION_JMS_MAPMESSAGE_TO_ROLE = "toRole";

    public static final String SUBMISSION_JMS_MAPMESSAGE_PROPERTY_ORIGINAL_SENDER
 = "originalSender";
    public static final String SUBMISSION_JMS_MAPMESSAGE_PROPERTY_FINAL_RECIPIENT
 = "finalRecipient";
    public static final String SUBMISSION_JMS_MAPMESSAGE_PROPERTY_ENDPOINT = "endPointAddress";

    public static final String SUBMISSION_JMS_MAPMESSAGE_PROTOCOL = "protocol";
    public static final String SUBMISSION_JMS_MAPMESSAGE_TOTAL_NUMBER_OF_PAYLOADS
 = "totalNumberOfPayloads";
    public static final String PAYLOAD_FILE_NAME_FORMAT = "payload_{0}.bin";
    public static final String BODYLOAD_FILE_NAME_FORMAT = "bodyload.bin";
    public static final String MESSAGING_FILE_NAME = "messaging.xml";
    public static final String METADATA_ARTIFACT_NAME = "metadata.xml";
    private static final String SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_NAME_PREFIX = "payload-";
    public static final String SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_NAME_FORMAT =
 JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_NAME_PREFIX + "{0}";
    private static final String SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_DESCRIPTION_SUFFIX = "-
description";
    public static final String SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_DESCRIPTION_FORMAT
 = JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_NAME_FORMAT +
 JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_DESCRIPTION_SUFFIX;
    private static final String SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_TYPE_SUFFIX = "-
MimeType";
    public static final String SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_TYPE_FORMAT
 = JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_NAME_FORMAT +
 JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_TYPE_SUFFIX;
    private static final String SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_CONTENT_ID_SUFFIX = "-
MimeContentID";
    public static final String SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_CONTENT_ID_FORMAT
 = JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_NAME_FORMAT +
 JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_CONTENT_ID_SUFFIX;
    private static final Log LOG = LogFactory.getLog(JMSMessageTransformer.class);


    /**
     * Transforms {@link eu.domibus.submission.Submission} to {@link javax.jms.MapMessage}
     *
     * @param submission the message to be transformed     *
     * @return result of the transformation as {@link javax.jms.MapMessage}
     */
    @Override
    public MapMessage transformFromSubmission(final Submission submission, final MapMessage
 messageOut) {


        try {
            messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_ACTION,
 submission.getAction());

 messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_SERVICE,
 submission.getService());
```

```
messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_CONVERSATION_ID,
submission.getConversationId());

           for (final Submission.Party fromParty : submission.getFromParties()) {

messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_FROM_PARTY_ID,
fromParty.getPartyId());

messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_FROM_PARTY_TYPE,
fromParty.getPartyIdType());
           }

           for (final Submission.Party toParty : submission.getToParties()) {

messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_TO_PARTY_ID,
toParty.getPartyId());

messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_TO_PARTY_TYPE,
toParty.getPartyIdType());
           }


messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_FROM_ROLE,
submission.getFromRole());

messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_TO_ROLE,
submission.getToRole());

           for (final Map.Entry p : submission.getMessageProperties().entrySet()) {
               if
(p.getKey().equals(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROPERTY_ORIGINAL_SENDER)) {

messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROPERTY_ORIGINAL_SENDER,
p.getValue().toString());
               }

               if
(p.getKey().equals(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROPERTY_ENDPOINT)) {

messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROPERTY_ENDPOINT,
p.getValue().toString());
               }

               if
(p.getKey().equals(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROPERTY_FINAL_RECIPIENT)) {

messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROPERTY_FINAL_RECIPIENT,
p.getValue().toString());
               }
           }


messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROTOCOL, "AS4");

messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_AGREEMENT_REF,
submission.getAgreementRef());

messageOut.setStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_REF_TO_MESSAGE_ID,
submission.getRefToMessageId());

           int counter = 2;

           for (final Submission.Payload p : submission.getPayloads()) {

               if (p.isInBody()) {

messageOut.setBytes(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_NAME_FORMAT,
1), p.getPayloadData());

messageOut.setStringProperty(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_T
1), p.getPayloadProperties().getProperty(Property.MIME_TYPE));

messageOut.setStringProperty(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_C
1), p.getContentId());
                   if (p.getDescription() != null) {

messageOut.setStringProperty(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_DESCRI
1), p.getDescription());
                   }
               } else {
                   final String payContID =
String.valueOf(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_CONTENT_ID_FORM
counter));
                   final String payDescrip =
String.valueOf(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_DESCRIPTION_FORMAT,
counter));
                   final String propPayload =
String.valueOf(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_NAME_FORMAT,
counter));
                   final String payMimeTypeProp =
String.valueOf(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_TYPE_FORMAT,
counter));
                   messageOut.setBytes(propPayload, p.getPayloadData());
                   messageOut.setStringProperty(payMimeTypeProp,
p.getPayloadProperties().getProperty(Property.MIME_TYPE));
                   messageOut.setStringProperty(payContID, p.getContentId());

                   if (p.getDescription() != null) {
                       messageOut.setStringProperty(payDescrip, p.getDescription());
                   }
                   counter++;
               }
           }

messageOut.setInt(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_TOTAL_NUMBER_OF_PAYLOADS,
submission.getPayloads().size());
       } catch (final JMSException ex) {
           JMSMessageTransformer.LOG.error("Error while filling the MapMessage", ex);
       }

       return messageOut;
   }

   /**
    * Transforms {@link javax.jms.MapMessage} to {@link eu.domibus.submission.Submission}
    *
    * @param messageIn the message ({@link javax.jms.MapMessage}) to be tranformed
    * @return the result of the transformation as {@link eu.domibus.submission.Submission}
    */
   @Override
   public Submission transformToSubmission(final MapMessage messageIn) {

       final Submission target = new Submission();

       try {


target.setAction(messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_ACTION));

target.setService(messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_SERVICE));

target.setServiceType(messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_SERVICE_TYPE))

target.setConversationId(messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_CONVERSATIO
           final String fromPartyID =
messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_FROM_PARTY_ID);
           final String fromPartyType =
messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_FROM_PARTY_TYPE);
           target.addFromParty(fromPartyID, fromPartyType);

target.setFromRole(messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_FROM_ROLE));
           final String toPartyID =
messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_TO_PARTY_ID);
           final String toPartyType =
messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_TO_PARTY_TYPE);
           target.addToParty(toPartyID, toPartyType);

target.setToRole(messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_TO_ROLE));

target.addMessageProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROPERTY_ORIGINAL_SENDER,
messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROPERTY_ORIGINAL_SENDER));

target.addMessageProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROPERTY_FINAL_RECIPIENT,
messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PROPERTY_FINAL_RECIPIENT));

target.setRefToMessageId(messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_REF_TO_MESS

target.setAgreementRef(messageIn.getStringProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_AGREEMENT_REF
           final int numPayloads =
messageIn.getIntProperty(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_TOTAL_NUMBER_OF_PAYLOADS);


           for (int i = 1; i <= numPayloads; i++) {
                final String propPayload =
 String.valueOf(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_NAME_FORMAT,
 i));

                final String bodyloadFileName = JMSMessageTransformer.BODYLOAD_FILE_NAME_FORMAT;

                final String contentId;
                final String mimeType;
                String description = null;
                final byte[] payloadData;
                payloadData = messageIn.getBytes(propPayload);
                final String payMimeTypeProp =
 String.valueOf(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_TYPE_FORMAT,
 i));
                mimeType = messageIn.getStringProperty(payMimeTypeProp);
                final String payDescrip =
 String.valueOf(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_DESCRIPTION_FORMAT,
 i));

                if (messageIn.getStringProperty(payDescrip) != null) {
                    description = messageIn.getStringProperty(payDescrip);
                }

                final String payContID =
 String.valueOf(MessageFormat.format(JMSMessageTransformer.SUBMISSION_JMS_MAPMESSAGE_PAYLOAD_MIME_CONTENT_ID_FORM
 i));

                contentId = messageIn.getStringProperty(payContID);

                final Properties partProperties = new Properties();
                if (mimeType != null && !mimeType.trim().equals("")) {
                    partProperties.setProperty(Property.MIME_TYPE, mimeType);
                }

                target.addPayload(contentId, payloadData, partProperties, i == 1, description,
 null);
            }


        } catch (final JMSException ex) {
            JMSMessageTransformer.LOG.error("Error while getting properties from MapMessage",
 ex);
            throw new RuntimeException(ex);
        }

        return target;

    }


}
```

C.2.3. Plugin configuration

All plugins are configured in the `${domibus.config.location}/submission-config.xml` file.

```
                <?xml version="1.0" encoding="UTF-8"?>
<beans xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:util="http://www.springframework.org/schema/util"
       xmlns="http://www.springframework.org/schema/beans"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://
www.springframework.org/schema/beans/spring-beans.xsd   http://www.springframework.org/schema/
util http://www.springframework.org/schema/util/spring-util.xsd">

    <!--
        All backends must be listed here to allow message delivery
    -->
    <util:list id="backends"
               value-type="eu.domibus.submission.BackendConnector">
        <ref bean="exampleBackendConnector"/>
    </util:list>

    <bean id="messageTransformer"
          class="eu.domibus.submission.transformer.impl.ExampleMessageTransformer">
    </bean>

    <bean id="exampleBackendConnector"
```

```
          class="eu.domibus.submission.ExampleBackendConnector">
        <property name="messageRetrievalTransformer" ref="messageTransformer"/>
        <property name="messageSubmissionTransformer" ref="messageTransformer"/>
    </bean>
</beans>
```

This is the basic configuration. Real implementations might require more complicated setups, as the next, real-world example shows.

```
                <?xml version="1.0" encoding="UTF-8"?>
<!--
  ~ Copyright 2015 e-CODEX Project
  ~
  ~ Licensed under the EUPL, Version 1.1 or – as soon they
  ~ will be approved by the European Commission - subsequent
  ~ versions of the EUPL (the "Licence");
  ~ You may not use this work except in compliance with the
  ~ Licence.
  ~ You may obtain a copy of the Licence at:
  ~ http://ec.europa.eu/idabc/eupl5
  ~ Unless required by applicable law or agreed to in
  ~ writing, software distributed under the Licence is
  ~ distributed on an "AS IS" basis,
  ~ WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
  ~ express or implied.
  ~ See the Licence for the specific language governing
  ~ permissions and limitations under the Licence.
  -->

<beans xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:jaxws="http://cxf.apache.org/jaxws"
       xmlns:util="http://www.springframework.org/schema/util"
       xmlns="http://www.springframework.org/schema/beans"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://
www.springframework.org/schema/beans/spring-beans.xsd http://cxf.apache.org/jaxws http://
cxf.apache.org/schemas/jaxws.xsd http://www.springframework.org/schema/util http://
www.springframework.org/schema/util/spring-util.xsd">


    <util:list id="backends"
               value-type="eu.domibus.submission.BackendConnector">
        <ref bean="backendWebservice"/>
        <ref bean="messageListener"/>
    </util:list>

    <!-- Webservice plugin configuration -->
    <bean id="backendWebservice"
          class="eu.domibus.submission.webService.impl.BackendWebServiceImpl"/>
    <jaxws:endpoint id="backendInterfaceEndpoint"
                    implementor="#backendWebservice" address="/backend"/>


    <!-- JMS plugin configuration -->
    <bean id="messageListener" class="eu.domibus.submission.jms.BackendJMSImpl">
        <property name="connectionFactory" ref="connectionFactory"/>
        <property name="outQueue" ref="outQueue"/>
        <property name="receivingQueue" ref="revievingQueue"/>
    </bean>

    <bean id="jmsContainer"
          class="org.springframework.jms.listener.DefaultMessageListenerContainer">
        <property name="messageListener" ref="messageListener"/>
        <property name="connectionFactory" ref="connectionFactory"/>
        <property name="destination" ref="inQueue"/>
    </bean>

    <bean id="amqConnectionFactory"
          class="org.apache.activemq.ActiveMQConnectionFactory">
        <constructor-arg index="0" value="tcp://127.0.0.1:61616"/>
    </bean>

    <bean id="connectionFactory"
          class="org.springframework.jms.connection.CachingConnectionFactory">
        <constructor-arg ref="amqConnectionFactory"/>
    </bean>

    <bean id="inQueue" class="org.apache.activemq.command.ActiveMQQueue">
        <constructor-arg index="0" value="queue/inQueue"/>
    </bean>
    <bean id="outQueue" class="org.apache.activemq.command.ActiveMQQueue">
        <constructor-arg index="0" value="queue/outQueue"/>
    </bean>

    <bean id="revievingQueue" class="org.apache.activemq.command.ActiveMQQueue">
        <constructor-arg index="0" value="queue/receivingQueue"/>
    </bean>

</beans>
```

This example shows a webservice and a JMS plugin working in parallel. Both plugins use Spring annotations to wire their transformers, so this configuration is not needed in the XML file.

Important

The order of the plugins in the "`backends`" list is important as the `isResponsible(MessageMetadata metadata)` methods of the plugins are called in that order.

Appendix D. Performance Data

References

[ebMS3CORE] *OASIS ebXML Messaging Services Version 3.0: Part 1, Core Features*. 1 October 2007. OASIS Standard. http://http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/os/ebms_core-3.0-spec-os.html .

[AS4-Profile] *AS4 Profile of ebMS 3.0 Version 1.0*. 23 January 2013. OASIS Standard. http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/os/AS4-profile-v1.0-os.html .

[D6.2] *e-SENS ebMS3 Profile*. 2014. e-SENS Project. http://wiki.ds.unipi.gr/display/ESENS/ABB+-+Message+Exchange+Protocol .

Index