Submitted to the EC on 26/02/2013

COMPETITIVENESS AND INNOVATION FRAMEWORK PROGRAMME

ICT Policy Support Programme (ICT PSP)

e-CODEX

e-Justice Communication via Online Data Exchange

ICT PSP call identifier: CIP-ICT-PSP-2009-4

ICT PSP main Theme identifier: CIP ICT PSP 2010 5.2 3: E-JUSTICE SERVICES

Project full title: e-Justice Communication via Online Data Exchange

Grant agreement n°: 270968

Deliverable 5.4 Developed Modules

Deliverable ID:	D5.4
Deliverable Name :	Concept of Implementation
Status :draft	V1.0
Dissemination Level :	e-CODEX consortium
Due date of deliverable :	30.11.2012
Actual submission date :	26.02.2013
Work Package :	e-CODEX WP5
Organisation name of lead partner for this deliverable :	Bundesministerium für Justiz, Ministerio de la Justicia
Author(s):	AT,DE and ES
Partner(s) contributing :	AT,DE and ES

The main goal of this document is to give an overview about the two main deliverables of WP5, which are the Gateway and the Generic Connector Framework. It describes shortly the structure of this deliverables and the dependencies to other deliverable.

Restricted to e-CODEX consortium

Table of Contents

LIST OF ABBREVIATIONS 7

EXECUTIVE SUMMARY 11

1. INTRODUCTION 12

1.1. SCOPE AND OBJECTIVE OF DELIVERABLE 12

1.2. WP5 GENERAL OBJECTIVES AND VISION 12

1.3. RELATIONS TO INTERNAL E-CODEX ENVIRONMENT 12

1.4. RELATIONS TO EXTERNAL E-CODEX ENVIRONMENT 13

1.5. QUALITY MANAGEMENT 13

1.6. RISK MANAGEMENT 13

1.7. STRUCTURE OF THE DOCUMENT 13

2. SOFTWARE MODULE GATEWAY 15

2.1. OVERVIEW AND FUNCTIONALITY 15

2.2. COMPONENT STRUCTURE 15

2.2.1. EBMS MODULE 16

2.2.2. RELIABILITY MODULE 19

2.2.3. SECURITY MODULE 20

2.2.4. LOGGING 20

2.2.4.1. Scope and Objective of the Module 20

2.2.4.1.1. Event Logging 22

2.2.4.1.2. Message Logging 22

2.2.4.2. Configuration 23

2.2.4.3. Function call 25

2.2.5. BACKEND WEB SERVICE 26

2.2.5.1. Subsystem Definition 26

2.2.5.1.1. Subsystem Description 26

2.2.5.1.2. Subsystem Scope. 26

2.2.5.1.3. Technological Environment Description 27

2.2.5.1.4. Principal Users Identification 27

2.2.5.2. Logical Model 27

2.2.5.2.1. SendMessage 28

2.2.5.2.2. SendMessageWithReference 29

2.2.5.2.3. DownloadMessage 30

2.2.5.2.1. listPendingsMessages 31

2.2.5.3. Use Case Analysis 31

2.2.5.4. Error Codes 32

2.2.6. P-MODES CONFIGURATION 32

Restricted to e-CODEX consortium

2.2.7. DB SCHEMAS 36
3. SW MODULE GENERIC CONNECTOR FRAMEWORK 72
3.1. WORKFLOW 72
3.1.1. CONFIGURATION 76
3.2. COMPONENT STRUCTURE 78
3.2.1. ECODEXCONNECTORCOMMON 78
3.2.2. ECODEXCONNECTORNATIONALBACKENDCLIENT 78
3.2.3. ECODEXCONNECTOREVIDENCESTOOLKIT 80
3.2.4. ECODEXCONNECTORSECURITYTOOLKIT 80
3.2.5. ECODEXCONNECTORCONTENTMAPPER 80
3.2.6. ECODEXCONNECTORCONTROLLER 81
3.2.7. ECODEXCONNECTORGATEWAYWEBSERVICECLIENT 81

Restricted to e-CODEX consortium

List of Figures

Figure 1. Component Structure 15
Figure 2. Internal Structure of Holodeck 16
Figure 3 e-CODEX Holodeck gateway modules 21
Figure 4 – Appender configuration 24
Figure 5 – Logger configuration 24
Figure 6 – Logger declaration 25
Figure 7 – Logger call 25
Figure 8 WSDL Diagram 26
Figure 9 Send Message Diagram 28
Figure 10 Send Message with Reference Diagram 29
Figure 11 Download Message Diagram 30
Figure 12 List Pending Message Diagram 31
Figure 13 Use Cases Backend Web Service 32
Figure 14 DB Schema Gateway 70
Figure 15 Incoming Workflow 73
Figure 16 Outgoing Workflow 75
Figure 17 e-CODEX Connector Structure 78

Restricted to e-CODEX consortium

List of Tables

Table 1 History 6
Table 2 Risks 13
Table 3 Document Structure 14
Table 4 – Standard Apache Log4j log level 22
Table 5 – e-CODEX specific Message log level 23
Table 6 – Message log level fields 23
Table 7 – e-CODEX Gateway Log4j logging configuration 23

Restricted to e-CODEX consortium

History

Version	*Date*	*Changes made*	*Modified by*
0.1	09.10.2012	Template for directory structure	AT
0.2	01.12.2012	Insert Description of SW module Gateway	ES
0.5	20.12.2012	Insert Description of SW module generic connector framework and compilation for first review	AT
0.6	05.02.2013	Modifications to the whole document because the quality review made by the Auditing Service	ES
1.0	26.02.2013	V 1.0 released	

Table 1 History

Restricted to e-CODEX consortium

List of Abbreviations

Acronym	Explanation
ACL	Access Control List
ADM	Architecture Development Method
AICPA	The American Institute of Certified Public Accountants (AICPA)
API	Application programming interface
APDU	Application Protocol Data Unit
AS	Applicability Statement AS1[1], AS2[2], AS3[3] and AS4[4] are a family of protocols specifying how to transport data securely and reliably over the Internet.
AT	Austria
BC	Business Collaboration,
BD	Business Document
BE	Belgium
BT	Business Transaction
BusDox	Business Document Exchange Network (PEPPOL)
CA	Certification Authority
CAdES	CMS Advanced Electronic Signatures, published by ETSI as TS 101 733
CBPKI	Cross-border Public Key Infrastructure (Estonia)
CISA	Certified Information Systems Auditor
CMS	Cryptographic Message Syntax, see "CAdES"-Description
COM	Commission
C-PEPS	Citizen Country PEPS Request Invocation Method
CRL	Certificate Revocation List, see "RFC 5280"; http://www.ietf.org/rfc/rfc5280.txt
CROBIES	Cross-Border Interoperability of eSignatures
CZ	CzechRepublic
D.I.M	Distributed Identity Management; Technical framework for dealing with identities in the context of web service
DA	Delivery Agent
DAC	Discretionary Access Control
DB	Data Base
DE	Germany
DES	Data Encryption Standard
DERBY (Apache Derby)	Is an open source relational database implemented entirely in Java (http://db.apache.org/derby) that comes with the Holodeck package
DGP	Delivery Gateway Protocol
DGJUST	Directorate General for Justice
DNIe	Documento Nacional de Identidade Electrónico (National ID card / Spain)
DPC	Data Protection and Confidentiality
Driver	Software allowing computer programs to interact with a hardware device

[1]AS1 specification, RFC 3335, http://www.ietf.org/rfc/rfc3335.txt

[2] AS2 specification, RFC 4130, http://www.ietf.org/rfc/rfc4130.txt

[3]AS3 specification, RFC 4823, http://tools.ietf.org/html/rfc4823

[4]AS4 conformance profile,

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/csprd03/AS4-profile-v1.0-csprd03.odt

Restricted to eCODEX consortium

DSP	Delivery Service Provider
DSS	Digital signature Standard (NIST)
E2EE	End-to-End Encryption
ebBP	ebXML Business Process, part of ebXML stack
ebMS	ebXML Messaging Services
ebXML	Electronic Business using eXtensible Markup Language, commonly known as e-business XML
EC	European Commission
ECC	Elliptic curve cryptography (NIST)
eCM	e-CODEX member
e-CODEX	e-Justice Communication via Online Data Exchange
ED-GW	Electronic Delivery Gateway
EDI VAN	EDI[5]: Electronic Data Interchange, VAN: Value Added Network
EE	Estonia
eID	Electronic Identity
eIDM	Electronic Identity Management
EIF	European Interoperability Framework
EOPIC	Equivalent of a personal identification code
EPO	European Payment Order
ER	Entity-Relationship. It's an abstract way to describe a Data Base
ES	Spain
ETSI	European Telecommunications Standards Institute
EU	European Union
FR	France
GR	Greece
GUI	Graphical User Interface
GW	Gateway
Holodeck	Is a free open-source software for b2b messaging, based on the Oasis specifications for ebXML Messaging version 3 and AS4 profile (http://holodeck-b2b.org)
HPC	Health Professional Card
HU	Hungary
ICT	Information and Communications Technology
ID / eID	Identity Document / electronic Identity Document
IDABC	Interoperable Delivery of European eGovernment Services to public Administrations, Businesses and Citizens
IEEE	Institute of Electrical and Electronic Engineers
IEEE 830	IEEE Software Requirements Specification
IOP	Interoperability
ISSP	Information System Security Policy
IT	Italy
JHA	Justice and Home Affairs Council
LDAP	Lightweight Directory Access Protocol – RFC 4511
Log4J	Logging library for Java, widely adopted (http://logging.apache.org/log4j/1.2)
LSP	Large Scale Pilot
LTV	Long Time Validity
MAC	Mandatory Access Control
MEP	Message Exchange Pattern. It's a definition of how two systems interchange messages, one-direction, bidirectional, robust exchange,…

[5]EDI standards include: UN/EDIFACT, ANSI ASC X12, TRADACOMS, ODETTE

Restricted to e-CODEX consortium

MIME	Multipurpose Internet Mail Extensions[6]
MS	EU MemberState
MS-CAPI	Cryptographic Application Programming Interface (Microsoft)
MT	Malta
MSH	Message Service Handler (MSH) or message gateway that provides a standardized, reliable, and secure infrastructure for enterprises to exchange business documents. It is in compliance with the OASIS ebXML Message Service (http://freebxml.org/msh.htm)
mTAN	Mobile Transaction Authentication Number
MTOM	Message Transmission Optimization Mechanism (SOAP)
MW	Middleware
NCP	National Contact Points
NIF	National Interoperability Framework
NL	The Netherlands
OCSP	Online Certificate Status Protocol, see "RFC 2560" http://www.ietf.org/rfc/rfc2560.txt
OpenSC	Set of software tools and libraries to work with smart cards with cryptographic capabilities http://www.opensc-project.org
OS	Operating System
PAdES	PDF Advanced Electronic Signature, published by ETSI as TS 102 778
PDP	Policy Decision Point
PEGS	Pan-European e-Government Services
PEPPOL	Pan-European Public Procurement Online (http://www.peppol.eu/)
PEPS	Pan-European Proxy Services (STORK)
PERMIS	PrivilEge and Role Management Infrastructure Standards http://www.openpermis.org
PET	Privacy Enhancing Technologies
PIC	Personal Identification Code
PKCS	Public-key cryptography standards (PKCS #1 - PKCS #15)
PMI	Permission Management Infrastructure
P-Mode	Processing Mode
PT	Portugal
PVP	"Portalverbundprotokoll" (Austrian solution for connecting public authorities)
QAA	Quality Authentication Assurance
QC	Qualified Certificate
RBAC	Role-Based Access Control
REM	Registered E-mail (ETSI TS 102 640)
RFC 3161	RFC 3161 timestamp protocol (The RFC 3161 builds on the IETF standard "Cryptographic Message Syntax", published as the RFC 2630.)
RFC 3161 (GT)	Guardtime RFC 3161 timestamp (GuardTime timestamping protocol and timestamp format are based on the IETF standard "Internet X.509 Public Key Infrastructure Time-Stamp Protocol", published as the RFC 3161.)
RO	Romania
S.A.F.E.	Secure Access to Federated e-Justice / e-Government (German eID Solution)
SAM card	Secure Authentication Module card

[6] MIME Part One: http://tools.ietf.org/html/rfc2045. Links to additional parts of the specification are given therein.

Restricted to CEDEX consortium

SAML 2.0	Security Assertion Markup Language v2.0, authentication request and response format (http://www.oasisopen.org/specs/index.php#saml)
SHA	Secure Hash Algorithm (NIST)
SOAP	Simple Object Access Protocol
SP	Security Policy
S-PEPS	Service Provider PEPS (STORK)
SP-MW	Middleware Service Provider (STORK)
SPOCS	Simple Procedures Online for Cross- Border Services (http://www.eu-spocs.eu/)
SSCD	Secure Signature Creation Device
SSL V3+	Secure Sockets Layer v3
SSO	Single Sign-On Profile
STORK	Secure idenTity acrOss boRders linked (https://www.eID-stork.eu/)
TAN	Transaction Authentication Number
Time Mark	Timestamp alternative defined in XAdES specification
TLS	Transport Layer Security
TLS 1.0+.	Transport Layer Security Version 1.0 + (RFC 2246, http://tools.ietf.org/html/rfc5246)
TOGAF	The Open Group Architecture Framework
Token	Physical device that an authorized user of computer services is given to ease authentication.
TR	Turkey
tScheme	tScheme is the independent, industry-led, self-regulatory scheme set up to create strict assessment criteria, against which it will approve Trust Services. (http://www.tscheme.org/)
TSL	Trust-service Status List, published by ETSI as TS 102 231
TSP	Trusted Service Provider
TTP	Trusted Third Party
UC	Use Case
UN/CEFACT	United Nations Centre for Trade Facilitation and Electronic Business
VIdP	Virtual Identity Providers
VIdP	Virtual IDP. A system component helping to abstract Pan-European eID interoperability. It either serves as a delegation component between the SP-MW or S-PEPS and the needed SPware (appropriate MW server Component) or enables SP-MW to communicate with other C-PEPS.
WP	Work Package
WP29	Article 29 Data Protection Working Party
WP4	Work Package 4 of the e-CODEX project, Identity (eID for natural and legal persons, roles, mandates and rights) and eSignatures
WSDL	Web Services Description Language
WS-I	Web Services Interoperability[7]
W3C	World Wide Web Consortium
XACML	eXtensible Access Control Markup Language http://saml.xml.org/xacml-oasis-standard
XAdES	XML Advanced Digital signatures, published by ETSI as TS 101 903

[7] http://www.oasis-ws-i.org/

Restricted to e-CODEX consortium

Executive Summary

The **e-**Justice **C**ommunication via **O**nline **D**ata **EX**change (e-CODEX) project aims to improve the access of citizens and business to legal means cross border in Europe as well as to improve interoperability between legal authorities within the EU. The goal is to achieve this objective with ideally no impact to the existing national ICT solutions.

In this context transport of data and documents is a key piece of the solution. Any functionality to be developed for a cross-border e-Justice service will necessarily mean transport of information from one country to another also including communication between the e-Justice Portal and some national solution. For this reason a work package explicitly dedicated to transport of data and documents has been defined within e-CODEX: Exchange of documents/data, e-Filing and e-Payment (WP5).

The objective of this document is to describe the official Software deliverables of WP5. There are two main deliverables which are the Gateway (based on the open source project Holodeck) and the generic connector framework handling the e-CODEX specific functionalities and components. For both deliverables a detailed description of the workflows, components, configuration and database layout is given.

In order to be as efficient as possible and, consequently, reduce the time and the cost of the development, a deep analysis of the building blocks that were available was made at the beginning of the project. Holodeck was chosen as the core of the gateway, and its "missing features" were also identified. The features 'REM ETSI evidence generator', 'enhanced logging' and 'web service interface that supersedes the file system based approach' were identified, analyzed and developed by the WP 5. The 'back end web service' and 'logging module' were introduced in the gateway itself and the evidence generator was moved to the National Connector.

The logging module was based on the standard log4j implementation, recording the logged activity to a data base instead of a variety of flat text files.

The back end web service is an additional interface to the ebMS gateway. It enables a way to send messages from the web services perspective without access to the gateway file system, reinforcing the security of the system and enabling a way to generate the initial SubMissionAcceptanceRejection evidence.

The 'REM ETSI evidence generator' is a module integrated as part of the National Connector and it is responsible of the generation of the selected evidences and its releasing through the gateway connection or to the national system.

Once the Gateways and National Connectors are up and running they will provide the e-Delivery functionalities required to put the pilots into service. Usual tasks following a software implementation are maintenance and upgrading. For WP5 maintenance will mainly mean to solve failings of the Gateways. Upgrading of Gateways contemplate to include dynamic discovery, … (to be completed).

The next steps of the project from a WP 5 perspective will be defined in a Follow Up meeting to be celebrated in February 2013.

Restricted to e-CODEX consortium

1. Introduction

1.1. Scope and Objective of Deliverable

This document is the deliverable D5.4 "Developed Modules" of Work Package (WP5) of the e-CODEX project.

The document is stored on the BSCW https://www.jol.nrw.de/pub/bscw.cgi/ /index-de.html. The Software Deliverables itself are also stored on the BSCW (https://www.jol.nrw.de/pub/bscw.cgi/ /index-de.html) and are available upon request.

1.2. WP5 General Objectives and Vision

e-CODEX is a Large Scale Project in the domain of e-Justice that aims to provide to citizens, enterprises and legal professionals an easier access to justice in cross border procedures and to make cross border collaboration of courts and authorities easier and more efficient by creating interoperability of the existing national ICT solutions.

When structuring the work of the e-CODEX project, various considerations were followed to find an optimal organizational structuring. The project aims to develop the interoperability building blocks for e-Justice services in Europe that address the horizontal issues between Member States. Furthermore, these building blocks will need to be proven in real e-Justice services in the countries involved. The project organization will thus need to support these goals properly to ensure that they can also be achieved from a managerial perspective.

Based on the initial building block breakdown[8] for the large scale pilot implementation candidates, WP5 aims to deliver the capability to bind together documents and data that need to be routed or exchanged to enable European cross-border processes in e-Justice. As an example of this reusing strategy e-CODEX signing, signature validation and ASIC creation features are provided through the Security Library based on the DSS tool; the EvidenceBuilder library has been derived from the SPOCS code.

1.3. Relations to Internal e-CODEX Environment

It is clear that there are dependencies between the different WP's in e-CODEX context. The WP5 is strongly linked to WP6 that enhances the overall functionality for e-Justice Services with the "content" of the documents. Another link is to WP7 that provides the IT-groundwork and architecture for interoperability between the systems to be connected, including the security and legal aspects. Beyond that WP4 delivered the identification and electronic signature building blocks used by WP5 software modules. WP3 is defining the underlying business processes of the judicial proceedings considered within e-CODEX. Requirements resulting from these business processes have been considered for the transport infrastructure implemented by WP5.

restricted to e-CODEX consortium

[8] For a deeper view you can read "Deliverable 5.2 Reusable Assets", chapter 4, "Transportation Building Block"

1.4. Relations to External e-CODEX Environment

WP5 has a strong relation to all other LSPs (especially to SPOCS and PEPPOL) with regards to the transport infrastructure developed within these projects. The results, documents and expertise gained by SPOCS and PEPPOL have been considered from the very beginning. Parts of their software modules (e.g. the evidence builder) have been reused by e-CODEX.

1.5. Quality Management

Deliverable 5.4 has been provided as a first draft (version 0.5) for a commentary review 2 months before the final delivery was planned. The review participants are all work package partners plus the External Quality Manager. The review comments gained are collected by the work package leaders and processed for the updated version (0.9.) which is delivered 2 weeks before the deadline of the final delivery. A second commentary review will be done using this version 0.9 of D5.4. The participants are again all work package partners plus the External Quality Manager.

The processing of all review comments is documented in the inspection report, which lists the review comments line by line including a statement how the respective review comment has been processed. The inspection report is published together with the update document of D5.4.

1.6. Risk Management

The risks as identified in the course of the creation of deliverable D5.4 and their probability and possible impact are as follows:

ID	Description	Probability		Impact		Expected value		Response	Owner
		inherent	residual	inherent	residual	inherent	residual		
1	The MS cannot use the SW modules due to MS specific requirements or restrictions.	medium	low	high	high	high	high	reduce	WP5, All pilotin g MS
2	The needed sub deliverables from other WP's are not mature enough or not ready.	medium	medium	high	high	high	high	accept	WP4, WP6, WP5
3	The SW deliverables are not available in time.	medium	medium	high	high	high	high	accept	WP5
4	Some Functionality is missing	medium	low	high	high	high	high	reduce	WP5

Table 2 Risks

1.7. Structure of the Document

The document is structured as follows:

Restricted to e-CODEX consortium

Chapter	Description
1. Introduction	Present the document and describe the work done
2. Gateway	Description of SW Module Gateway
3. Generic Connector Framework	Description of SW Module Generic Connector Framework

Table 3 Document Structure

Restricted to e-CODEX consortium

2. Software Module Gateway

2.1. Overview and Functionality

The Gateway of the e-CODEX project it is a piece of software that is responsible of several tasks. Some of the features are provided by the ebMS implementation (Holodeck) on which the Gateway is based:

- Transforms the National Message Format injected from the National Connector to the standard ebMS message format.
- Signs and encrypts the communication between the different Gateways.
- Implements Reliability and Quality of Service configurable behavior. These features will be reinforced in the next version because some failures need fixing.

Additionally, the project has complemented some extra features:

- Logging module that updates a Data Base in order to facilitate auditing tasks.
- The interface based on the file system scrutiny has been changed and implemented a web service interface (Back end Interface)
- New Clean up service that purges the messages that couldn`t been downloaded at an specified time.

2.2. Component Structure

The Gateway is a tightly coupled system based on the Open Source ebMS 3 implementation, Holodeck[9]

Figure 1. Component Structure

[9] http://holodeck-b2b.org

Holodeck is deployed within Axis2 stack as a set of modules (inside axis2 processing pipeline) as illustrated in the picture below:

Figure 2. Internal Structure of Holodeck

We will examine each module in further detail[10].

2.2.1. ebMS Module

The "ebMS Module" is the folder called "holodeck-ebms3" located under the directory "holodeck/msh/WEB-INF/modules/". The configuration file is located in "holodeck/msh/WEB-INF/modules/holodeck-ebms3/META-INF/module.xml". This configuration file is shown below:

```
<module name="holodeck-ebms3" class="org.holodeck.ebms3.module.Ebms3Module">
 <parameter name="PersistenceUnit">ebms3-derby</parameter>
 <parameter name="PModesDir" locked="false">../../../../config/pmodes</parameter>
 <parameter name="Modules">holodeck-ebms3, holodeck-reliability, holodeck-security, rampart</parameter>
 <parameter name="GATEWAY_CONFIG_FILE">../../../../config/gateway.xml</parameter>
 <parameter name="MSHAddress" locked="false">http://localhost:8080/holodeck/services/msh</parameter>

 // ==============================================================================
    The name of the folder in which submitted messages will be stored.
    It could be an absolute path to some folder on the file system, or
    just the name of a folder (in which case it will be automatically
    created under WEB-INF directory).
 // ==============================================================================
 <parameter name="SubmittedMessagesFolder">../../../../store/send</parameter>
 // ==============================================================================
    The value of this parameter is a comma-separated list of possible names
    (or IP addresses this local machine, that is the machine running this MSH)
    may have. It helps the MSH by looking at a given URL to determine if
    it is the URL of this MSH or the URL of another remote MSH by just
    comparing the machine name in the URL to the values of this parameter.
 // ==============================================================================
```

[10] The description of the different modules has been obtained from http://holodeck-b2b.org/docs/developer/architecture.html

```
<parameter name="LocalMachine">localhost,127.0.0.1,holodeck,192.240.13.76</parameter>

// ==============================================================================
   If true, the MSH, when writing a message to database, it will also
   persist all the attachments the message may have to database. This
   would work for small to medium attachments (may be up 1 Giga per
   attachment). However, if an attachment is way bigger than 1 Giga, the
   database columns may be limited in storage space, and in this case it would
   be better if the MSH persist the attachment on the file system where
   there is no limit on storage (except for the disk capacity of course).
   Make this parameter false if you are planning on having attachment very
   large that may exceed 1 GB (a giga bytes) or whatever limit your database is set to
// ==============================================================================
<parameter name="StoreAttachmentsInDB">false</parameter>
// ==============================================================================
   The name of the folder in which the attachments of the arriving messages
   will be stored.
   It could be an absolute path to some folder on the file system, or
   just the name of a folder (in which case it will be automatically
   created under WEB-INF\modules\holodeck-ebms3 directory).
// ==============================================================================
<parameter name="ReceivedMessagesFolder">../../../../store/receive</parameter>
<parameter name="Workers">../../../../config/workers.xml</parameter>

<InFlow>
  <handler name="PModeFinder" class="org.holodeck.ebms3.handlers.PModeFinder">
    <order phase="ebms3InPhase" phaseFirst="true"/>
  </handler>
  <handler name="ErrorLogger" class="org.holodeck.ebms3.handlers.ErrorLogger">
    <order phase="ebms3InPhase" after="PModeFinder"/>
  </handler>
  <handler name="Validation" class="org.holodeck.ebms3.handlers.Validation">
    <order phase="ebms3InPhase" after="ErrorLogger"/>
  </handler>
  <handler name="ReceiptProcessor" class="org.holodeck.ebms3.handlers.ReceiptProcessor">
  </handler>
  <handler name="ReceiptGen" class="org.holodeck.ebms3.handlers.ReceiptGen">
    <order phase="ebms3InPhase" after="ReceiptProcessor"/>
  </handler>
  <handler name="PullProcessor" class="org.holodeck.ebms3.handlers.PullProcessor">
    <order phase="ebms3InPhase" after="ReceiptGen"/>
  </handler>
  <handler name="RespPackager" class="org.holodeck.ebms3.handlers.ResponsePackager">
    <order phase="ebms3InPhase" after="PullProcessor"/>
  </handler>
  <handler name="ReceivedUserMsg" class="org.holodeck.ebms3.handlers.ReceivedUserMsgHandler">
    <order phase="ebms3InPhase" after="RespPackager"/>
  </handler>
  <handler name="InvokeCallback" class="org.holodeck.ebms3.handlers.InvokeCallback">
    <order phase="ebms3InPhase" after="ReceivedUserMsg"/>
  </handler>
  <handler name="HeaderDetacher" class="org.holodeck.ebms3.handlers.HeaderDetacher">
    <order phase="ebms3InPhase" after="InvokeCallback"/>
  </handler>
</InFlow>

<OutFlow>
```

```
  <handler name="Packager" class="org.holodeck.ebms3.handlers.Packager">
    <order phase="ebms3OutPhase" phaseFirst="true"/>
  </handler>
  <handler name="CallbackPersist"
        class="org.holodeck.ebms3.handlers.CallbackPersist">
    <order phase="ebms3OutPhase" after="Packager"/>
  </handler>
  <handler name="RespPackager" class="org.holodeck.ebms3.handlers.ResponsePackager">
    <order phase="ebms3OutPhase" after="CallbackPersist"/>
  </handler>
  <handler name="PullProcessor" class="org.holodeck.ebms3.handlers.PullProcessor">
    <order phase="ebms3OutPhase" after="RespPackager"/>
  </handler>
  <handler name="TrackReceipt" class="org.holodeck.ebms3.handlers.TrackReceipt">
    <order phase="ebms3OutPhase" after="PullProcessor"/>
  </handler>
  <handler name="ReceiptAppender" class="org.holodeck.ebms3.handlers.ReceiptAppender">
    <order phase="ebms3OutPhase" after="TrackReceipt"/>
  </handler>
 </OutFlow>

 <INFaultFlow/>
 <OutFaultFlow/>
</module>
```

PersistenceUnit: the name of the JPA persistence unit. Persistence units are defined in the configuration file "holodeck/msh/WEB-INF/classes/META-INF/persistence.xml". The default persistence unit for the ebMS module is to use derby as embedded. If you want to use another database, you can change the persistence unit here (but you have to create the database and tables from ddl scripts provided in the directory "holodeck/store/derby/").

PModesDir: this is the directory where holodeck stores the P-Mode documents ("holodeck/config/pmodes")

Modules: this is the list of the modules that an outgoing message should go through

GATEWAY_CONFIG_FILE: this is the configuration file of Holodeck Gateway (holodeck service that dispatches messages to various consumers). The default location is "holodeck/config/gateway.xml".

MSHAddress: this is the address of this MSH. You should replace "localhost" in the URL by the name (or domain name) of the machine on which holodeck is running. Why holodeck needs to know its own address? Well, this is used to determine the sender from the receiver when reading a P-Mode document. By knowing its own address, this helps holodeck which figuring out which role (sender or receiver) it plays within a given P-Mode document.

SubmittedMessagesFolder: this is the folder where a user (person or application) drops payloads (data files) in there so that holodeck picks them up and either push them out in a message or make them in a message ready to be pulled when a PullRequest arrives for them. The default location is "holodeck/store/send/"

LocalMachine: the value of this parameter is a list of all the names for the local machine on which holodeck is running (be it ip address, name, domain name, etc...). This parameter helps holodeck determining which P-Mode is being used when it receives a message (not all messages contain the "P-Mode" attribute in their SOAP header, and when such information is missing, holodeck makes extra work to figure out what is the name of the P-Mode to associate with the message being received. Knowing all the names of the local machine, helps determining the P-Mode name.

StoreAttachmentsInDB: this parameter is not used anymore. It used to be an option to choose whether storing attachment in database (when persisting a message to database). Holodeck does not

store attachments in database (it stores them as files on the hard disk), but only stores a reference to the attachments. This is more practical because if the attachments are really huge, the database tables cannot hold them, and that's why attachments are stored as files on the disk).

ReceivedMessagesFolder: this is the folder where holodeck stores received messages. Actually when the consumer called "org.holodeck.ebms3.consumers.impl.SaveToFolder" gets the message from holodeck gateway, it stores the message there.

Workers: this is the configuration file that contains the workers (background tasks executed periodically). The default location of this file is "holodeck/config/workers.xml".

2.2.2. Reliability Module

The "Reliability Module" is the folder called "holodeck-reliability" located under the directory "holodeck/msh/WEB-INF/modules/". The configuration file for this module which is "holodeck/msh/WEB-INF/modules/holodeck-reliability/META-INF/module.xml". This configuration file is shown below (showing only the parameters):

```
<module name="holodeck-reliability"
        class="org.holodeck.reliability.module.ReliabilityModule">

 // ---------------------------------------------------------------------------
 // The name of the folder in which attachments will be stored.
 // It could be an absolute path to some folder on the file system, or
 // just the name of a folder (in which case it will be automatically
 // created under WEB-INF\modules\holodeck-reliability directory).
 // ---------------------------------------------------------------------------
 <parameter name="MessageStorageFolder">Storage_Folder</parameter>
 <parameter name="RELIABILITY_CONFIG_FILE">../../../../config/reliability-config.xml</parameter>
 <parameter name="PersistenceUnit">wsrm-derby</parameter>
 <parameter name="MessageLifetime" locked="false">P29D</parameter>
 <parameter name="ResendFromDBAtStartup">false</parameter>
 ...
 </module>
```

MessageStorageFolder: this is the folder where the reliability stores the attachments of an outgoing message (reliable outgoing messages have to be stored in the database prior to sending them, because in case no acknowledgment is received for them, they would have to be resent by the reliability module). This details should not concern the user (the user should not care where the reliability module stores the attachments).

RELIABILITY_CONFIG_FILE: this is the configuration file that contains quality of service definitions for reliability. The default file is "holodeck/config/reliability-config.xml".

PersistenceUnit: this is the JPA persistence unit used by the reliability module to persist outgoing messages (the default is embedded Derby).

MessageLifetime: the value of this parameter specifies the time interval after which a message expires in the database.

ResendFromDBAtStartup: the value of this parameter controls whether the "re-sender" background task (in the reliability module) should execute when starting up holodeck. When starting holodeck (and therefore the reliability module as well), there could be some messages stored in the database

of the reliability module that have been sent already in the past but not acknowledged yet. So this parameter controls whether these messages should be resent or not as holodeck starts up.

2.2.3. Security Module

The "Security Module" is the folder called "holodeck-security" located under the directory "holodeck/msh/WEB-INF/modules/". The configuration file for this module is "holodeck/msh/WEB-INF/modules/holodeck-security/META-INF/module.xml". This configuration file is shown below (showing only the parameters):

```
<module name="holodeck-security" class="org.holodeck.security.module.SecurityModule">
  <parameter name="SecurityConfigFile">../../../../config/security-config.xml</parameter>
  <parameter name="PoliciesFolder">../../../../config/policies</parameter>
  <parameter name="KeysFolder">../../../../config/keys</parameter>
  ...
</module>
```

SecurityConfigFile: this is the configuration file that tells the security module what keystore, encryption user, callback class (for passwords), and the WSS policy to use.

PoliciesFolder: this is the folder where holodeck stores WSS policies.

KeysFolder: this is the folder where holodeck stores security keys and keystores.

2.2.4. Logging

2.2.4.1. Scope and Objective of the Module

The Logging Module of the e-CODEX Gateway provides basic logging functionality for the Gateway application.

It is based on the Apache Log4j Standard in version 1.2.15[11] and extends this standard with a message logging functionality which is needed to fulfil the requirements of the e-CODEX Project.

Due to the fact that the e-CODEX Gateway implementation based on the Open Source implementation Holodeck, the logging module fits other requirements which cover expandability of the implementation.

It provides the functionality of message and event logging to the other gateway communication modules using standardized interfaces.

Restricted to e-CODEX consortium

[11] Log4j project website (12-10-2012): http://logging.apache.org/log4j/1.2/

Figure 3 e-CODEX Holodeck gateway modules

Restricted to e-CODEX consortium

2.2.4.1.1. Event Logging

The basic event logging functionality is part of the Apache Log4j [12]implementation which the logging module is based on. It provides eight different log levels for the different types of a logging event.

Name	Description
ALL	The ALL has the lowest possible rank and is intended to turn on all logging
DEBUG	The DEBUG Level designates fine-grained informational events that are most useful to debug an application.
ERROR	The ERROR level designates error events that might still allow the application to continue running.
FATAL	The FATAL level designates very severe error events that will presumably lead the application to abort.
INFO	The INFO level designates informational messages that highlight the progress of the application at coarse-grained level.
OFF	The OFF has the highest possible rank and is intended to turn off logging.
TRACE	The TRACE Level designates finer-grained informational events than the DEBUG.
WARN	The WARN level designates potentially harmful situations.

Table 4 – Standard Apache Log4j log level

For these levels the logging module is using the standard interfaces described in the Apache Log4j documentation[13].

It has been selected the most useful log levels because the rest (ALL, OFF) mentioned in the standard[14] aren't very common. The ALL level, instructs to the logger system that all the messages will be recorded, the OFF level turn off the logger system.

2.2.4.1.2. Message Logging

To cover the requirement WP7-RQ-NF-021[15] for the message logging in the judicial area which is described in D7.2, the logging module of the e-CODEX gateway implementation must provide a basic message logging functionality. This message logging will be used in the different communication interfaces and their calls in the gateway communication modules.

To reuse the existing logging functions provided by the Apache Log4j logging library, the e-CODEX gateway logging module extend this library by adding a special log level in the log level hierarchy.

Name	Description

[12] Log4j project website (12-10-2012): http://logging.apache.org/log4j/1.2/

[13] Log4j online documentation (12-10-2012): http://logging.apache.org/log4j/1.2/manual.html

[14] http://logging.apache.org/log4j/1.2/apidocs/org/apache/log4j/Level.html

[15] D7.2 Requirements Finalisation & D3.2 Described Test Scenarios. 10.Appendix III: Non-functional Requirements. Page 97.

Restricted to e-CODEX consortium

MESSAGE	The MESSAGE level is a holodeck specific log level for the message event logging.

Table 5 – e-CODEX specific Message log level

For the message logging other and specific attributes are necessary. These fields are derived from the message information fields of the ebMS message. They are **not** containing personal information of the sender or recipient.

Type	**Name**	**Description**
String	messageId	Message ID of the current message
String	sender	Sender ID or sender address
String	fromRole	The fromRole of the ebMS sender
String	recipient	recipient ID or recipient address
String	toRole	The toRole of the ebMS recipient
String	service	Service parameter of the ebMS message flow
String	action	Action parameter of the ebMS message flow
String	conversationId	ConversationID which the current message belongs to
String	pmode	P-Mode which is used for the message transport
String	status	Current status of the message (incoming/outgoing)
Date	timestamp	Timestamp of the message logging event. Is set by the constructor

Table 6 – Message log level fields

2.2.4.2. Configuration

The configuration of the logging module will be done in the log4j.xml file in the directory *msh\WEB-INF\classes* of the Holodeck installation directory.

This file used the standard log4j XML-schema to configure the different appenders for the log level and modules.

The standard configuration contains the following appenders:

Name	Type	Log level
Axis2FILE	File Appender	DEBUG
Ebms3FILE	File Appender	DEBUG
HibFILE	File Appender	INFO
LoggingFILE	File Appender	DEBUG
WsrmFILE	File Appender	DEBUG
SecurityFILE	File Appender	DEBUG
BackendFILE	File Appender	DEBUG
MessageHolodeckAppender	Holodeck Appender	Message
EventHolodeckAppender	Holodeck Appender	Error

Table 7 – e-CODEX Gateway Log4j logging configuration

Restricted to e-CODEX consortium

Other appender could be configured by adding the following configuration to the file.

```
<appender class="[appender class as absolute package name]" name="[appender name]">
[parameter for File appender configuration
    <param value="true" name="Append"/>
    <param value="[path to file]" name="File"/>
    <param value="5120KB" name="MaxFileSize"/>
    <layout class="org.apache.log4j.PatternLayout">
        <param value="%d %-5p %c %x - %m%n" name="ConversionPattern"/>
    </layout>
]
[parameter for the database appender
    <param name="Threshold" value="[log level]"/>
]
</appender>
```

Figure 4 – Appender configuration

```
<logger additivity="false" name="[package path as root path for the specified appender]">
    <appender-ref ref="[appender reference]"/>
</logger>

<logger additivity="false" name="[package path as root path for the specified appender]">
    <level value="[log level]"/>
    <appender-ref ref="[appender reference]"/>
</logger>
```

Figure 5 – Logger configuration

2.2.4.3. Function call

To use the logging functionality in the gateway implementation, the standard Log4j[16] function call must be used.

For each class in which the logging module should be used, a separate instance of the logger. An instance will be provided by the getLogger method, which organizes a pool of logger instances.

```
private static final Logger log =
Logger.getLogger([Classname].class.getName());
```

Figure 6 – Logger declaration

After a logger instance is available the standard logger methods should be used.

For each log level several logging methods are available.

The way to call the message logging is shown in Figure 7.

```
[Message logger call]
log.log(org.holodeck.logging.level.Message.MESSAGE,
org.holodeck.logging.persistent.MessageInfo);
```

Figure 7 – Logger call

Restricted to e-CODE consortium

[16] Apache Log4j Java doc (12-10-2012): https://logging.apache.org/log4j/1.2/apidocs/index.html

2.2.5. Backend Web Service

2.2.5.1. Subsystem Definition.

Backend Web Service is a module added to the original Holodeck server. This module or extension provides a way of communicating with the original server using Web Services instead of file submission.

2.2.5.1.1. Subsystem Description

Holodeck b2b is a free open-source software for b2b messaging, based on the Oasis specifications for ebXML Messaging version 3 and aligned with AS4 profile[17] as has been stated before.

Messages can be submitted to Holodeck, either to be sent out (i.e. pushed) or to be stored so that they can be pulled by other B2B partners[18]. However, nowadays, this message submission can only be done in one way, manually, creating an empty folder and dropping payloads in it, Holodeck will then pick up these payloads along metadata in an xml file to create a message that will either be pushed (sent out) or stored to be pulled by others.

Therefore, Backend Web Service module will provide a way of submitting messages using a Web Service including the operations of sending, downloading and listing pending messages.

2.2.5.1.2. Subsystem Scope.

The main functions of the Backend Web Service are:

- Sending messages using the Holodeck API to another Gateway.
- Validating the EBMS Header parameters and the existence of payload data.
- Mapping P-Mode using some parameters of the EBMS Header.
- Downloading pending messages of the Gateway.
- Listing pending messages of the Gateway.
- Invoking the Logging module.
- Deleting expired messages periodically.

The web service has four operations:

- **sendMessage**: send a message to another Gateway using the data of the SOAP request.
- **sendMessageWithReference**: send a message to another Gateway using the data downloaded from the URI of the SOAP request.
- **downloadMessage**: download a message from the Gateway server.
- **listPendingMessages**: list the messages which are pending in the Gateway server.

Figure 8 WSDL Diagram

[17] http://holodeck-b2b.org/docs/index.html

[18] http://holodeck-b2b.org/docs/holodeck-features.html

2.2.5.1.3. Technological Environment Description

The hardware and software requirements are identical as in the previous modules.

2.2.5.1.4. Principal Users Identification

Backend Web Services cannot differentiate between the clients who are invoking the web service. Therefore, there is only one type of user:

- **National Connector**: every Gateway is expected to be only invoked by its own National Connector using the Backend Web Service.

The control of the access to this interface will be made by the security mechanisms of the underlying infrastructure (provided by the different MS) and is out of the scope of this project. Nevertheless, the gateway only accepts SSL connections.

2.2.5.2. Logical Model

The business process of Backend Web Service is divided into the web service operations:

- **sendMessage**
- **sendMessageWithReference**
- **downloadMessage**
- **listPendingMessages**

Restricted to e-CODEX consortium

2.2.5.2.1. SendMessage

The objective of this operation is to send data from the Gateway A (origin where it has been invoked) to the Gateway B (target server).

Technically, it differs from the common web services operation because it receives many of its parameters using the SOAP Header. Therefore, the arguments of the operations are divided in two groups:

- **EBMS header**: this element is based on the Messaging header of the ebMS-3 standard. It includes the different parameters which will define the communication between the Gateway A and the Gateway B.
- **Data payloads**: this element includes the binary data codified in Base64 of the different payloads.

The return value is the following one:

- **messageID**: it is the ID of the EBMS3 message sent.

This workflow of the operation is the following one:

- For the Gateway A:
  1. The Message is received.
  2. The Back end module validates the parameters.
  3. According to the parameters received, the EMBS module is invoked on order to send the message to the Gateway B.
  4. The Back end module waits until the message has been sent or failed.
  5. The operation returns the Id of the message sent.
- For the Gateway B:
  1. The EBMS module receives the message from the Gateway A and invokes the Back end consumer.
  2. The Back end module copies the message to the hard disk into a temporary folder and inserts its information into the DB.
  3. The EBMS module informs the Gateway A that the message has been sent successfully.

This is the sequence diagram of the sendMessage operation:

Figure 9 Send Message Diagram

2.2.5.2.2. SendMessageWithReference

The objective of this operation is to send data from the Gateway A (origin where it has been invoked) to the Gateway B (target server).

It differs from the previous operation because it receives URLs where the data must be downloaded. The arguments of the operations are also divided in two groups:

- **EBMS header**: this element is based on the Messaging header of the ebMS-3 standard. It includes the different parameters which will define the communication between the Gateway A and the Gateway B.
- **Data payloads**: this element includes the URLs where the payload data must be downloaded.

The return value is the following one:

- **messageID**: it is the EBMS ID of the EBMS3 message sent.

This workflow of the operation is the following one:

- For the Gateway A:
  1. The Message is received.
  2. The Back end module validates the parameters
  3. The payloads data is downloaded.
  4. According to the parameters, the EMBS module is invoked in order to send the message to the Gateway B.
  5. The Back end module waits until the message has been sent or failed.
  6. The operation returns the Id of the message sent.
- For the Gateway B:
  1. The EBMS module receives the message from the Gateway A and invokes the Back end consumer.
  2. The Back end module copies the message to the hard disk into a temporary folde and inserts its information into the DB.
  3. The EBMS module informs the Gateway A that the message has been sent successfully.

This is the sequence diagram of the sendMessageWithReference operation:

Figure 10 Send Message with Reference Diagram

to e-CODEX consortium

2.2.5.2.3. DownloadMessage

The objective of this operation is to retrieve the message data.

The argument of the operations is the following one:

- **messageID**: this is the Back end ID of the message to download.

The return values are the following ones:

- **EBMS header**: this element is based on the Message header of the ebMS-3 standard. It includes the different parameters of the EBMS shipment.
- **Data payloads**: this element includes the binary data codified in Base64 of the different payloads.

This workflow of the operation is the following one:

1. The request is received.
2. The Back end module validates the parameters.
3. The Back end module loads the Message from the hard disk and DB.
4. The operation returns the message.

This is the sequence diagram of the downloadMessage operation:

Figure 11 Download Message Diagram

2.2.5.2.1. listPendingsMessages

The objective of this operation is to retrieve a list of messages which can be downloaded.

The operation does not receive any argument.

The return values are the following ones:

- **messagesID**: the list of ID of the messages which can be downloaded.

This workflow of the operation is the following one:

1. The request is received.
2. The Back end looks in the DB for the messages which can be downloaded.
3. The operation returns the list.

This is the sequence diagram of the listPendingsMessages operation:

Figure 12 List Pending Message Diagram

2.2.5.3. Use Case Analysis

There are four use cases identified in the Backend Web Service which matches with the service operations:

- **Send message**
- **Send message with reference**
- **Download message**
- **List pending messages**

This is the Use Case Diagram:

Figure 13 Use Cases Backend Web Service

2.2.5.4. Error Codes

2.2.6. P-Modes Configuration

A P-Mode is a concept introduced in the ebMS3 specification that is used for managing the way of sending and processing ebXML messages between two MSHs.

In this P-Mode, a set of parameters, like reliability, security, sign, encryption, MEP... are defined for establishing the sending.

The syntax of a P-Mode file is

- The <Producer> node indicates the entity that generates User Messages. An entity is a person, an application, a corporation... A User Message is a SOAP message that contains payload data. It is represented by the xml element <eb:From> within the SOAP header in the ebXML specification.

- The <UserService> indicates the destination of the User Message. It is represented by the xml elements <eb:To>, <eb:CollaborationInfo>, <eb:PayloadInfo> and <eb:MessageProperties> in the ebXML specification.

Restricted to e-COD

- The <Binding> element contains information about the transport of the message (URL address of the destination, the SOAP version, SOAP action, and MEP (Message Exchange Pattern) to use) and about detailed information of each leg of the MEP being used (for example, the reliability and the security that are applied to each leg of the MEP).

These parameters are defined in the OASIS ebXML Messaging Services Version 3.0[19]:

- PMode.ID: (optional)
  Is the identifier for the P-Mode. This identifier is user-defined and optional, for the convenience of P-Mode management. It must uniquely identify the P-Mode among all P-Modes deployed on the same MSH, and may be absent if the P-Mode is identified by other means, e.g. embedded in a larger structure that is itself identified, or has parameter values distinct from other P-Modes used on the same MSH. If the ID is specified, the AgreementRef/@pmode attribute value is also expected to be set in associated messages.

- PMode.MEP
  The type of ebMS MEP associated with this P-Mode. The value must be a URI, e.g: http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/oneWay.

- PMode.MEPbinding:
  The transport channel binding assigned to the MEP (push, pull, sync, push-and-push, push-and-pull, pull-and-push, pull-and-pull…). The value must be a URI, e.g: http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/push.

- PMode.Initiator.Party:
  (PMode.Initiator and its sub-elements are optional if PMode.Responder is present.) Qualifies the party initiating the MEP (see Section 2.2.3). A user message initiating an MEP instance under this P-Mode must have its eb:Messaging/eb:UserMessage/eb:PartyInfo/eb:From element contain the same PartyId elements as the PartyId elements defined in this parameter. Any user message sent to the initiator must have its eb:PartyInfo/eb:To map to or be compatible with this parameter.

- PMode.Initiator.Role:
  Name of the role assumed by the party sending the first message of this MEP. Either the message element eb:Messaging/eb:UserMessage/eb:PartyInfo/eb:From/eb:Role or the element eb:Messaging/eb:UserMessage/eb:PartyInfo/eb:To/eb:Role of each message in this MEP must have this value, depending on the direction of message transfer.

- PMode.Responder.Party:
  (PMode.Responder and its sub-elements are optional if PMode.Initiator is present.) Qualifies the party responding to the initiator party in this MEP. Any user message sent to the responder must have its eb:Messaging/eb:UserMessage/eb:PartyInfo/eb:To element contain the same PartyId elements as the PartyId elements defined in this parameter.

- PMode.Responder.Role:
  Name of the role assumed by the party receiving the first message of this MEP. Either the message element eb:Messaging/eb:UserMessage/eb:PartyInfo/eb:From/eb:Roleor the

[19] OASIS ebXML Messaging Services Version 3.0: Part 1, Core Features OASIS Standard, 1 October 2007

Restricted to e-CODEX consortium

element eb:Messaging/eb:UserMessage/eb:PartyInfo/eb:To/eb:Roleof each message in this MEP must have this value, depending on the direction of message transfer.

- PMode[1].Protocol.Address:
  The value of this parameter represents the address (endpoint URL) of the Receiver MSH (or Receiver Party) to which Messages under this P-Mode leg are to be sent. Note that a URL generally determines the transport protocol (for example, if the endpoint is an email address, then the transport protocol must be SMTP; if the address scheme is "http", then the transport protocol must be HTTP).

- PMode[1].Protocol.SOAPVersion:
  This parameter indicates the SOAP version to be used (1.1 or 1.2). In some implementations, this parameter may be constrained by the implementation, and not set by users.

- PMode[1].BusinessInfo.Service:
  Name of the service to which the User message is intended to be delivered. Its content should map to the element eb:Messaging/eb:UserMessage/eb:CollaborationInfo/eb:Service.

- PMode[1].BusinessInfo.Action:
  Name of the action the User message is intended to invoke. Its content should map to the element eb:Messaging/eb:UserMessage/eb:CollaborationInfo/eb:Action

- PMode[1].BusinessInfo.MPC:
  The value of this parameter is the identifier of the MPC (Message Partition Channel) to which the message is assigned. It maps to the attribute eb:Messaging/eb:UserMessage/@mpc.

The structure of a P-Mode can be defined in different ways.

1) First Scenario

In this approach, all the nodes that a P-Mode node needs for working are defined inside it.
This configuration does not allow reusing these internal nodes.

Here is an example of this kind of configuration:

```
<PModes>
    <PMode name="PMode01">
      <Producer name="producer01">
                    ...
      </Producer>
      <UserService name="userService01">
                    ...
      </UserService>
      <Binding name="binding01">
                    ...
      </Binding>
   </PMode>
   <PMode name="PMode02">
```

Restricted to e-CODEX consortium

```
        <Producer name="producer02">
                        ...
        </Producer>
        <UserService name="userService02">
                        ...
        </UserService>
        <Binding name="binding02">
                        ...
        </Binding>
    </PMode>

</PModes>
```

2) Second Scenario

In this other approach is to define, separately, the <Producer>, <UserService> and <Binding> nodes, so it is possible to re-using them in different <PMode> configurations.

In this approach, the nodes that a P-Mode node needs for working are defined separately, so they can be re-used by different P-Mode nodes.

The first scenario will require larger configuration files than the second because you can define several P-Modes with the same UserService and different Producers, or you can define different Bindings with the same P-Mode and different Producers.

e-CODEX will use this approach.

Here is an example of this kind of configuration:

```
<PModes>
  <Producer name="producer01">
            ...
  </Producer>
  <UserService name="userService01">
            ...
  </UserService>
  <Binding name="binding01">
      <MEP ....>
          <Leg userService=" userService01" producer=" producer01" ...>
              ....
          </Leg>
      </MEP>
  </Binding>

  <PMode name="PMode01" binding="binding01" />
</PModes>
```

Following it is described a complete example of a P-Mode syntax definition:

```
<PModes>
            <Producer name="GW-AT-EPO">
                        <PartyId>GW-EE</PartyId>
```

```
                <Role>GW-EPO</Role>
        </Producer>
        <UserService name="EPO_AT">
                <ToPartyInfo>
                    <PartyId>AT</PartyId>
                    <Role>GW</Role>
                </ToPartyInfo>
                <CollaborationInfo>
                <Service>EPO</Service>
                <Action>EPO</Action>
                </CollaborationInfo>
                <PayloadInfo>
                <Message label="shipMsg" maxSize="3000">
                        <SoapBody />
                </Message>
                </PayloadInfo>
        </UserService>

        <Binding name="SC_AT">
                <MEP name="One-Way/Push">
                        <Leg number="1" mpc="epo" userService="EPO_AT" security="sign-encrypt-body-header" producer="GW-
AT-EPO">
                        <Endpoint address="http://localhost:8280/holodeck/services/msh" soapVersion="1.1" />
                        </Leg>
                </MEP>
        </Binding>

        <PMode name="EPO_AT" binding="SC_AT" />
</PModes>
```

2.2.7. DB Schemas

ER Diagram of Databases and some description (automatically generated)

Restricted to e-CODEX

Table Name	ACKS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	CALLBACKACK_ID	P		Y	VARCHAR (255 CHAR)	LT		
2	ACKTO				VARCHAR (255 CHAR)	LT		
3	GROUPID				VARCHAR (255 CHAR)	LT		
4	SENT				NUMERIC (1)	LT		
5	SEQ_NUMBER				VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
ACKS_PK	PK				CALLBACKACK_ID	ASC

Restricted to eSODEX consortium

Table Name	ATTACHMENTS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	ID	P		Y	VARCHAR (255 CHAR)	LT	Unique identifier of the record	
2	CONTENTID				VARCHAR (255 CHAR)	LT	Name of the Payload	
3	CONTENTTYPE				VARCHAR (255 CHAR)	LT	MIME type of the transmitted file	
4	FILEPATH				VARCHAR (255 CHAR)	LT	Complete path where the attachment will be temporary stored until the transmission or downloading of the message	

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
ATTACHMENTS_PK	PK				ID	ASC

Restricted to eSODEX consortium

Table Name	LOGGEREVENT

This table has been implemented to supersede the conventional use of text log files that difficults the auditing tasks.

The object of this table is the recording of the actions executed. It is used mostly for debugging purposes because it records technical information not business information.

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	ID	P		Y	VARCHAR (255 CHAR)	LT	Unique identifier of the record	
2	LOGDATE				Timestamp (6)	LT	Date and Time of the event	
3	LOG_CLASSNAME				VARCHAR (255 CHAR)	LT	Class in the code that has triggered the recording of the action	
4	LOG_LINENUMBER				VARCHAR (255 CHAR)	LT	Line number of the class that contains the instruction that has triggered the recording of the action	
5	LOG_METHODNAME				VARCHAR (255 CHAR)	LT	Method inside the class that has triggered the recording of the action	
6	LOGGER				VARCHAR (255 CHAR)	LT	Class in the code that has triggered the recording of the action	
7	MSG				VARCHAR (2000 CHAR)	LT	Description of the action that is occurring when the logger has been launched	
8	PRIORITY				VARCHAR (255 CHAR)	LT	Assigned level to the message. Enables the filtering of the messages	

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
LOGGEREVENT_PK	PK				ID	ASC

Restricted to eCODEX consortium

Table Name	LOGGERMESSAGE

This table has been implemented to supersede the conventional use of text log files that difficult the auditing tasks.

The object of this table is the recording of the actions executed on the messages.

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	ID	P		Y	VARCHAR (255 CHAR)	LT	Unique identifier of the record	
2	ACTION				VARCHAR (255 CHAR)	LT	Identifies the action made, e.g. 'push', 'sendMessage', 'downloadMessage',…	
3	CONVERSATIONID				VARCHAR (255 CHAR)	LT	Identifier assigned by the user to correlate the message dialogue	
4	FROMROLE				VARCHAR (255 CHAR)	LT	Role of the sending gateway	
5	MESSAGEID				VARCHAR (255 CHAR)	LT	Identifier assigned by Holodeck to message	
6	PMODE				VARCHAR (255 CHAR)	LT	P-Mode used by the sending gateway to transmit the message	
7	RECIPIENT				VARCHAR (255 CHAR)	LT	Destination country	
8	SENDER				VARCHAR (255 CHAR)	LT	Sender country	
9	SERVICE				VARCHAR (255 CHAR)	LT	Back end service invoked, e.g. 'DownloadMessageService', 'SendMessageService',…	
10	STATUS				VARCHAR (255 CHAR)	LT	Result of the action, e.g. 'SENT_OK', 'MESSAGE_DOWNLOADED',…	
11	TIMESTAMP				Timestamp (6)	LT	Date and Time of the event	
12	TOROLE				VARCHAR (255 CHAR)	LT	Role of the receiving gateway	

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order

Restricted to e-CODEX consortium

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
LOGGERMESSAGE_PK	PK				ID	ASC

Restricted to e-CODEX consortium

Table Name	MESSAGE

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	IDMESSAGE	P		Y	NUMERIC (10)	LT	Unique identifier of the message for this table	
2	DELETED				NUMERIC (1)	LT	Indicator that informs if the message has been deleted or not (0->no, 1->yes)	
3	DIRECTORY				VARCHAR (1024 CHAR)	LT	Folder that contains temporary the message until the message transmission has been completed or it has been downloaded	
4	DOWNLOADED				NUMERIC (1)	LT	Indicator that informs if the message has been downloaded or not (0->no, 1->yes)	
5	MESSAGEDATE				Timestamp (6)	LT	Date and Time of the event	
6	MESSAGEUID				VARCHAR (128 CHAR)	LT	Identifier assigned by Holodeck to message	
7	PMODE				VARCHAR (128 CHAR)	LT	P-Mode used by the sending gateway to transmit the message	

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
MESSAGE_PK	PK				IDMESSAGE	ASC

Foreign Keys (referred from)

Name	Referred From	Mandatory	Transferable	In Arc	Column Name
FK3454796EA6FCC7E9	PAYLOAD		Y		IDMESSAGE

Restricted to e-CODEX consortium

Table Name	MSG_CALLBACK

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	ID	P		Y	VARCHAR (255 CHAR)	LT		
2	CALLBACK_CLASS				VARCHAR (255 CHAR)	LT		
3	LEGNUMBER				NUMERIC (10)	LT		
4	MESSAGEID				VARCHAR (255 CHAR)	LT		
5	PMODE				VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
MSG_CALLBACK_PK	PK				ID	ASC

Restricted to eSODEX consortium

Table Name	ORDERED_MSG

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	OMESSAGE_ID	P		Y	VARCHAR (255 CHAR)	LT		
2	CONTENT_TYPE				VARCHAR (255 CHAR)	LT		
3	MIME_FILE				VARCHAR (255 CHAR)	LT		
4	MSG_CONTEXT				BLOB (4000)	LT		
5	DELIVERED				NUMERIC (1)	LT		
6	EXPIRYTIME				Timestamp (6)	LT		
7	FAULTED				NUMERIC (1)	LT		
8	GROUPID				VARCHAR (255 CHAR)	LT		
9	SEQ_NUMBER				NUMERIC (10)	LT		
10	SERVICE_URL				VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
ORDERED_MSG_PK	PK				OMESSAGE_ID	ASC

Foreign Keys (referred from)

Name	Referred From	Mandatory	Transferable	In Arc	Column Name
FKFD00A3C0112E73C	ORDERED_MSG_ATTACHMENTS	Y	Y		OMESSAGE_ID

Restricted to e-CODEX consortium

Table Name	ORDERED_MSG_ATTACHMENTS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	ORDERED_MSG_OMESSAGE_ID		F	Y	VARCHAR (255 CHAR)	LT		
2	ATTACHMENTS_ID			Y	VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
ORDERED_MSG_ATTACHMENTS_ATTACHMENTS_ID_UN	UK				ATTACHMENTS_ID	ASC

Foreign Keys (referring to)

Name	Refering To	Mandatory	Transferable	In Arc	Column Name
FKFD00A3C0112E73C	ORDERED_MSG	Y	Y		OMESSAGE_ID

Restricted to e-CODEX consortium

Table Name	PAYLOAD

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	IDPAYLOAD	P		Y	NUMERIC (10)	LT		
2	FILENAME				VARCHAR (256 CHAR)	LT		
3	IDMESSAGE		F		NUMERIC (10)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
PAYLOAD_PK	PK				IDPAYLOAD	ASC

Foreign Keys (referring to)

Name	Refering To	Mandatory	Transferable	In Arc	Column Name
FK3454796EA6FCC7E9	MESSAGE		Y		IDMESSAGE

Restricted to e-CODEX consortium

Table Name	RECEIPTS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	RECEIPT_ID	P		Y	VARCHAR (255 CHAR)	LT		
2	MESSAGEID				VARCHAR (255 CHAR)	LT		
3	NON_REPUDIATION_INFO				CLOB (4000)	LT		
4	REFTOMESSAEID				VARCHAR (255 CHAR)	LT		
5	SENT				NUMERIC (1)	LT		
6	TIMESTAMP				Timestamp (6)	LT		
7	TO_URL				VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
RECEIPTS_PK	PK				RECEIPT_ID	ASC

Restricted use e-CODEX consortium

Table Name	RECEIPT_TRACKING

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	ID	P		Y	VARCHAR (255 CHAR)	LT		
2	MESSAGEID				VARCHAR (255 CHAR)	LT		
3	PMODE				VARCHAR (255 CHAR)	LT		
4	RECEIPT_RECEIVED				NUMERIC (1)	LT		
5	RECEIPT				CLOB (4000)	LT		
6	TOURL				VARCHAR (255 CHAR)	LT		
7	REQUEST_ID		F		VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
RECEIPT_TRACKING_PK	PK				ID	ASC
RECEIPT_TRACKING_MESSAGEID_UN	UK				MESSAGEID	ASC

Foreign Keys (referring to)

Name	Refering To	Mandatory	Transferable	In Arc	Column Name
FK78EC851E85558B8E	USERMSG_PUSH		Y		ID

Restricted use e-CODEX consortium

Table Name	RECEIVED_RANGES

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	RECEIVEDRANGE_ID	P		Y	VARCHAR (255 CHAR)	LT		
2	GROUPID				VARCHAR (255 CHAR)	LT		
3	MAX_SEQ				NUMERIC (10)	LT		
4	MIN_SEQ				NUMERIC (10)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
RECEIVED_RANGES_PK	PK				RECEIVEDRANGE_ID	ASC

Restricted to e-LODEX consortium

Table Name	RECEIVED_USERMSG

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	ID	P		Y	VARCHAR (255 CHAR)	LT		
2	CONTENT_TYPE				VARCHAR (255 CHAR)	LT		
3	MIME_FILE				VARCHAR (255 CHAR)	LT		
4	MSG_CONTEXT				BLOB (4000)	LT		
5	EBMS3_ACTION				VARCHAR (255 CHAR)	LT		
6	FROM_PARTY				VARCHAR (255 CHAR)	LT		
7	MESSAGEID				VARCHAR (255 CHAR)	LT		
8	MPC				VARCHAR (255 CHAR)	LT		
9	MSG_INFOSET				BLOB (4000)	LT		
10	REFTOMESSAGEID				VARCHAR (255 CHAR)	LT		
11	EBMS3_SERVICE				VARCHAR (255 CHAR)	LT		
12	TO_PARTY				VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
RECEIVED_USERMSG_PK	PK				ID	ASC

restricted to e-CODEX consortium

Foreign Keys (referred from)

Name	Referred From	Mandatory	Transferable	In Arc	Column Name
FK84D1352918A1EF0F	RECEIVED_USERMSG_ATTACHMENTS	Y	Y		ID

Restricted to e-CODEX consortium

Table Name	RECEIVED_USERMSG_ATTACHMENTS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	RECEIVED_USERMSG_ID		F	Y	VARCHAR (255 CHAR)	LT		
2	ATTACHMENTS_ID			Y	VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
RECEIVED_USERMSG_ATTACHMENTS_ATTACHMENTS_ID_UN	UK				ATTACHMENTS_ID	ASC

Foreign Keys (referring to)

Name	Refering To	Mandatory	Transferable	In Arc	Column Name
FK84D1352918A1EF0F	RECEIVED_USERMSG	Y	Y		ID

Restricted to e-CODEX consortium

Table Name	RECEIVER_GROUPS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	GROUP_ID	P		Y	VARCHAR (255 CHAR)	LT		
2	CLOSED				NUMERIC (1)	LT		
3	COMPLETE				NUMERIC (1)	LT		
4	DELIVERED_COUNT				NUMERIC (10)	LT		
5	GROUP_EXPRITYTIME_UTC				VARCHAR (255 CHAR)	LT		
6	GROUPID				VARCHAR (255 CHAR)	LT		
7	MAX_IDLE_DURATION				VARCHAR (255 CHAR)	LT		
8	HIGHEST_SEQ_RECEIVED				NUMERIC (10)	LT		
9	LAST_DELIVERED_SEQ				NUMERIC (10)	LT		
10	LAST_MSG_TIMESTAMP				Timestamp (6)	LT		
11	MAX_MSG_EXPIRYTIME				Timestamp (6)	LT		
12	ORDERED				NUMERIC (1)	LT		
13	REMOVED				NUMERIC (1)	LT		

Restricted to e-CODEX consortium

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
RECEIVER_GROUPS_PK	PK				GROUP_ID	ASC

Restricted to e-CODEX consortium

Table Name	RELIABILITY

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	RELIABILITY_ID	P		Y	VARCHAR (255 CHAR)	LT		
2	ACKREPLY				VARCHAR (255 CHAR)	LT		
3	ACKREPLYELEMENT				Raw (255)	LT		
4	ACKTO				VARCHAR (255 CHAR)	LT		
5	ATLEASTONCE				NUMERIC (1)	LT		
6	ATMOSTONCE				NUMERIC (1)	LT		
7	EXPONENTIAL_BACKOFF				NUMERIC (1)	LT		
8	INORDER				NUMERIC (1)	LT		
9	MAXIMUM_RETRANSMISSION_COUNT				NUMERIC (10)	LT		
10	NAME				VARCHAR (255 CHAR)	LT		
11	RETRANSMISSION_INTERVAL				NUMERIC (10)	LT		
12	RETRANSMIT_CALLBACK				VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
RELIABILITY_PK	PK				RELIABILITY_ID	ASC

Restricted to e-CODEX consortium

Foreign Keys (referred from)

Name	Referred From	Mandatory	Transferable	In Arc	Column Name
FKA6DEE31E894F77F3	SENDER_GROUPS		Y		RELIABILITY_ID

Restricted to e-CODEX consortium

Table Name	RETRANSMIT_MSG

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	GMESSAGE_ID	P		Y	VARCHAR (255 CHAR)	LT		
2	CONTENT_TYPE				VARCHAR (255 CHAR)	LT		
3	MIME_FILE				VARCHAR (255 CHAR)	LT		
4	MSG_CONTEXT				BLOB (4000)	LT		
5	ACKNOWLEDGED				NUMERIC (1)	LT		
6	DELIVERY_FAILED				NUMERIC (1)	LT		
7	EXPIRYTIME				Timestamp (6)	LT		
8	EXP_BACKOFF				NUMERIC (1)	LT		
9	FAULTED				NUMERIC (1)	LT		
10	GROUPID				VARCHAR (255 CHAR)	LT		
11	MAX_RETRANS_COUNT				NUMERIC (10)	LT		
12	SEND_TO				VARCHAR (255 CHAR)	LT		
13	REPLY_PATTERN				VARCHAR (255 CHAR)	LT		
14	RESEND_COUNT				NUMERIC (10)	LT		
15	RETRANS_INTERVAL				NUMERIC (10)	LT		

Restricted to e-CODEX consortium

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
16	RETRANSMIT_CALLBACK				VARCHAR (255 CHAR)	LT		
17	SEQ_NUMBER				NUMERIC (10)	LT		
18	TIME_TO_SEND				NUMERIC (19)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
RETRANSMIT_MSG_PK	PK				GMESSAGE_ID	ASC

Foreign Keys (referred from)

Name	Referred From	Mandatory	Transferable	In Arc	Column Name
FK63ABC176768643B2	RETRANSMIT_MSG_ATTACHMENTS	Y	Y		GMESSAGE_ID

Restricted to e-CODEX consortium

Table Name	RETRANSMIT_MSG_ATTACHMENTS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	RETRANSMIT_MSG_GMESSAGE_ID		F	Y	VARCHAR (255 CHAR)	LT		
2	ATTACHMENTS_ID			Y	VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
RETRANSMIT_MSG_ATTACHMENTS_ATTACHMENTS_ID_UN	UK				ATTACHMENTS_ID	ASC

Foreign Keys (referring to)

Name	Refering To	Mandatory	Transferable	In Arc	Column Name
FK63ABC176768643B2	RETRANSMIT_MSG	Y	Y		GMESSAGE_ID

Restricted to e-CODEX consortium

Table Name	SENDER_GROUPS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	GROUP_ID	P		Y	VARCHAR (255 CHAR)	LT		
2	ACKNOWLEDGED_COUNT				NUMERIC (10)	LT		
3	CAPACITY				NUMERIC (10)	LT		
4	CLOSED				NUMERIC (1)	LT		
5	CURRENT_SEQ				NUMERIC (10)	LT		
6	DATE_STARTED				Timestamp (6)	LT		
7	FAILED_COUNT				NUMERIC (10)	LT		
8	GROUPID				VARCHAR (255 CHAR)	LT		
9	GROUP_LIFETIME				VARCHAR (255 CHAR)	LT		
10	MAX_IDLE_DURATION				VARCHAR (255 CHAR)	LT		
11	LAST_MSG_TIMESTAMP				Timestamp (6)	LT		
12	MAX_MSG_EXPIRYTIME				Timestamp (6)	LT		
13	MESG_LIFETIME				VARCHAR (255 CHAR)	LT		
14	NAME				VARCHAR (255 CHAR)	LT		
15	QUALITY_RELIABILITY_ID		F		VARCHAR (255 CHAR)	LT		

Restricted to e-CODEX consortium

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
SENDER_GROUPS_PK	PK				GROUP_ID	ASC

Foreign Keys (referring to)

Name	Refering To	Mandatory	Transferable	In Arc	Column Name
FKA6DEE31E894F77F3	RELIABILITY		Y		RELIABILITY_ID

Restricted to e-CODEX consortium

Table Name	SEQUENCE_TABLE

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	SEQ_NAME				VARCHAR (255 CHAR)	LT		
2	SEQ_COUNT				NUMERIC (10)	LT		

Restricted to e-CODEX consortium

Table Name	SYNC_RESPONSES

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	ID	P		Y	VARCHAR (255 CHAR)	LT		
2	CONTENT_TYPE				VARCHAR (255 CHAR)	LT		
3	MIME_FILE				VARCHAR (255 CHAR)	LT		
4	MSG_CONTEXT				BLOB (4000)	LT		
5	MEP				VARCHAR (255 CHAR)	LT		
6	MPC				VARCHAR (255 CHAR)	LT		
7	MSG_INFOSET				BLOB (4000)	LT		
8	PMODE				VARCHAR (255 CHAR)	LT		
9	SENT				NUMERIC (1)	LT		
10	TIME_IN_MILLIS				NUMERIC (19)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
SYNC_RESPONSES_PK	PK				ID	ASC

Foreign Keys (referred from)

Name	Referred From	Mandatory	Transferable	In Arc	Column Name
FK6084EC7F10131110	SYNC_RESPONSES_ATTACHMENTS	Y	Y		ID

restricted to e-CODEX consortium

Table Name	SYNC_RESPONSES_ATTACHMENTS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	SYNC_RESPONSES_ID		F	Y	VARCHAR (255 CHAR)	LT		
2	ATTACHMENTS_ID			Y	VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
SYNC_RESPONSES_ATTACHMENTS_ATTACHMENTS_ID_UN	UK				ATTACHMENTS_ID	ASC

Foreign Keys (referring to)

Name	Refering To	Mandatory	Transferable	In Arc	Column Name
FK6084EC7F10131110	SYNC_RESPONSES	Y	Y		ID

Restricted to e-CODEX consortium

Table Name	USERMSG_PULL

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	ID	P		Y	VARCHAR (255 CHAR)	LT		
2	CONTENT_TYPE				VARCHAR (255 CHAR)	LT		
3	MIME_FILE				VARCHAR (255 CHAR)	LT		
4	MSG_CONTEXT				BLOB (4000)	LT		
5	MPC				VARCHAR (255 CHAR)	LT		
6	MSG_INFOSET				BLOB (4000)	LT		
7	PMODE				VARCHAR (255 CHAR)	LT		
8	PULLED				NUMERIC (1)	LT		
9	TIME_IN_MILLIS				NUMERIC (19)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
USERMSG_PULL_PK	PK				ID	ASC

Foreign Keys (referred from)

Name	Referred From	Mandatory	Transferable	In Arc	Column Name
FK4E9D823FDAD13FDA	USERMSG_PULL_ATTACHMENTS	Y	Y		ID

Restricted to e-CODEX consortium

Table Name	USERMSG_PULL_ATTACHMENTS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	USERMSG_PULL_ID		F	Y	VARCHAR (255 CHAR)	LT		
2	ATTACHMENTS_ID			Y	VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
USERMSG_PULL_ATTACHMENTS_ATTACHMENTS_ID_UN	UK				ATTACHMENTS_ID	ASC

Foreign Keys (referring to)

Name	Refering To	Mandatory	Transferable	In Arc	Column Name
FK4E9D823FDAD13FDA	USERMSG_PULL	Y	Y		ID

Restricted to e-CODEX consortium

Table Name	USERMSG_PUSH

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	ID	P		Y	VARCHAR (255 CHAR)	LT		
2	CONTENT_TYPE				VARCHAR (255 CHAR)	LT		
3	MIME_FILE				VARCHAR (255 CHAR)	LT		
4	MSG_CONTEXT				BLOB (4000)	LT		
5	CALLBACK_CLASS				VARCHAR (255 CHAR)	LT		
6	LEGNUMBER				NUMERIC (10)	LT		
7	MEP				VARCHAR (255 CHAR)	LT		
8	MSG_INFOSET				BLOB (4000)	LT		
9	PMODE				VARCHAR (255 CHAR)	LT		
10	PUSHED				NUMERIC (1)	LT		
11	TIME_IN_MILLIS				NUMERIC (19)	LT		
12	TOURL				VARCHAR (255 CHAR)	LT		

Restricted to e-CODEX consortium

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
USERMSG_PUSH_PK	PK				ID	ASC

Foreign Keys (referred from)

Name	Referred From	Mandatory	Transferable	In Arc	Column Name
FK78EC851E85558B8E	RECEIPT_TRACKING		Y		ID
FKC1A1EB94DB3213BA	USERMSG_PUSH_ATTACHMENTS	Y	Y		ID

Restricted to e-CODEX consortium

Table Name	USERMSG_PUSH_ATTACHMENTS

Columns

No	Column Name	PK	FK	M	Data Type	DT kind	Description	Formula (Default Value)
1	USERMSG_PUSH_ID		F	Y	VARCHAR (255 CHAR)	LT		
2	ATTACHMENTS_ID			Y	VARCHAR (255 CHAR)	LT		

Indexes

Index Name	State	Functional	Spatial	Expression	Column Name	Sort Order
USERMSG_PUSH_ATTACHMENTS_ATTACHMENTS_ID_UN	UK				ATTACHMENTS_ID	ASC

Foreign Keys (referring to)

Name	Refering To	Mandatory	Transferable	In Arc	Column Name
FKC1A1EB94DB3213BA	USERMSG_PUSH	Y	Y		ID

Restricted to e-CODEX consortium

Figure 14 DB Schema Gateway

Restricted

1 Error Codes

Restricted to e-CODEX consortium

3. SW Module Generic Connector Framework

The second main SW delivery of WP5 is the generic connector framework, which tries to provide as much as possible the common functionality which is needed by every piloting MS. The MS specific parts can be provided by some dedicated implementation classes via well defined interfaces.

3.1. Workflow

The connector framework implements basically two workflows one for sending messages from the national backend system to the Gateway and the other partner Gateway in Europe (outgoing workflow) and one for receiving messages from the Gateway and forwarding them to the national backend system (incoming workflow).

Restricted to e-CODEX consortium

Figure 15 Incoming Workflow

The incoming workflow consists of the following steps:

1) The connector queries the pending messages from the Gateway and downloads them.
2) Then the Trust Ok Token is validated and
3) A REMMDAcceptanceRejection Evidence is created and sent back to the Gateway.
4) Afterwards the content is transformed (if necessary) to the national format and the message is send to the national backend system.
5) Two timers are started, one for the delivery confirmation (REMMDDeliveryNonDelivery) from the national backend system and one for the real retrieval of the message by the end recipient (REMMDRetrievalNonRetrievalbyRecipient). The second event is optional and can be switched off depended if the national subsystem provides such information.
6) Then the connector tries to receive a delivery confirmation from the national backend system and depending whether this confirmation is received or the timer is finished a
7) REMMDDeliveryNonDelivery Evidence is generated and sent back to the Gateway.
8) The connector tries to receive a recipient confirmation form the national backend system and depending whether this confirmation is received or the timer is finished a
9) REMMDRetrievalNonRetrievalbyRecipient Evidence is generated and it is sent back to the Gateway.

Restricted to e-CODEX consortium

Figure 16 Outgoing Workflow

The outgoing workflow consists of the following steps:

1) The connector queries the pending messages from the national backend system and downloads them.
2) Afterwards the content is transformed (if necessary) to the e-CODEX format and
3) The Trust Ok Token is created as well as the ASiCS container.
4) A REMMDSubmissionAcceptanceRejection Evidence is created. Only in case of errors a rejection evidence is sent back to the national subsystem immediately.
5) Then a timer is started. The system waits for the acceptance of the message by the gateway, in this case will be created and evidence confirmation (RelayREMMD Evidence).
6) The ebMS message is created including the message content, the ASiCS container and the REMMDSubmissionAcceptanceRejection Evidence. The EbMS message is forwarded to Gateway via the backend web service.
7) Afterwards the different possible evidences are retrieved by the Gateway and forwarded to the national backend system. First a RelayREMMD Evidence will be retrieved and
8) Second a REMMDDeliveryNonDelivery Evidence will be retrieved and finally a
9) REMMDRetrievalNonRetrievalbyRecipient Evidence.

3.1.1. Configuration

The following configuration parameters can be set:

gateway.endpoint.address=127.0.0.1:8080

gateway.name=AT

gateway.role=GW

Defines the endpoint address of the gateway backend webservice and the corresponding MS (Iso Code) and role.

connector.national.backend.client.implementation.class.name=eu.ecodex.connector.nbc.ECodexConnectorNationalBackendClientImpl

As every nation has its own backend client, an implementation of the ECodexConnectorNationalBackendClient interface has to be implemented which handles the connection to its own backend. Here the full qualified name of the implementation class which implements ECodexConnectorNationalBackendClient must be given.

connector.use.content.mapper=true

connector.content.mapper.implementation.class.name=eu.ecodex.connector.mapping.ECodexConnectorContentMapperImpl

Defines if a content mapper module should be used. If there is a certain national format the e-CODEX message should be transformed to this content and the mapper handles the mapping. The main class ECodexConnectorContentMapperImpl has to be extended and the full qualified name of the implementation class which implements ECodexConnectorContentMapper must be given.

connector.use.security.toolkit=true

Defines if the security toolkit from WP4 should be used. If messages should be added with a trustOkToken and be sent in an encrypted container, this module has to be activated.

Restricted to e-CODEX consortium

connector.use.evidences.toolkit=true

Defines if evidences toolkit should be used and ebMS standard messages should be confirmed or declined by evidences messages sent back to the message sender. Those evidences contain the state of the message sent and are therefore good to hold reliability.

connector.check.messages.period.ms=30000

Defines how often the gateway and the national backend system should be checked for messages. All messages in both directions should be handled entirely before next period starts. So no conflicts are produced. The Value is defined in milliseconds.

connector.database.dialect=

connector.database.driverClassName=

connector.database.url=

connector.database.username=

connector.database.password=

These properties define the connection to the database where message states should be logged.

The dialect and driverClassName values depend on the dbms that is in use.

java.keystore.path=

java.keystore.password=

key.alias=

key.password=

These parameters define the needed java keystore holding the certificates for encryption and authentication with the partner Gateway.

Restricted to e-CODEX consortium

3.2. Component Structure

Figure 17 e-CODEX Connector Structure

The connector consists of a controller, which implements and controls the whole workflow between the components. For the needed e-CODEX functionality, which is common for all the piloting MS, such as the generation of TrustOkToken und generation of evidences, there exist specific components, which have been made optional to be able to support future use in other areas than e-CODEX. For the connection to the Gateway there exist a full implementation of the Backend Web Service. The connection to the national Backend System must be provided by the MS and be configured as a plugin. For internal use, e.g. storing message Ids, a small DB is needed. The following chapters describe the components in more detail. The source of the components is available at the BSCW server as well. Within the source you will find additional documentation describing the different methods and parameters in more detail.

3.2.1. ECodexConnectorCommon

This module implements the basic classes and structures used by all the other components such as the exception handling, DB objects and a common unique definition of a message object. The DB is used to hold the internal state of the messages and the message object is used to wrap different national formats and the e-CODEX format to one generic JAVA Object. This general objects are used internally by the Framework as well as by the needed national extensions. It is a very common practice to extract such basic structures within a separate package. For a detailed description please refer to the inline documentation of the source code found as part of the deliverable on the BSCW server.

3.2.2. ECodexConnectorNationalBackendClient

This module has to be implemented by the MS. It covers the message forwarding to and from the national legal communication system. Every national legal communication system uses different

technologies therefore this systems have to be mapped to a generic interface which can be used by the connector framework.

The interface is defined as follows:

```
/**
    * This method delivers a message received by the gateway. The message
    * content is already transformed into a national format, if there is a
    * content mapper configured and implemented.
    *
    * @param message
    *         A {@link Message} object with all data concerning the message.
    * @throws ECodexConnectorNationalBackendClientException
    * @throws ImplementationMissingException
    */
   public void deliverMessage(Message message) throws ECodexConnectorNationalBackendClientException,
         ImplementationMissingException;

   /**
    * If there is a new evidence generated for a message and sent to the
    * connector, this new evidence must be sent to the national system.
    *
    * @return A {@link MessageConfirmation} object containing the messageId,
    *       the evidence was generated for, the {@link ConfirmationStateEnum}
    *       and the evidence itself.
    * @throws ECodexConnectorNationalBackendClientException
    * @throws ImplementationMissingException
    */
   public void deliverLastEvidenceForMessage(MessageConfirmation confirmation)
         throws ECodexConnectorNationalBackendClientException, ImplementationMissingException;

   /**
    * Requests all messages from the national system that are not yet handled
    * by this connector instance, therefore not sent over the gateway.
    *
    * @return an Array of messageId's that are queued in the national backend
    *       system and have to be handled by the connector.
    * @throws ECodexConnectorNationalBackendClientException
    * @throws ImplementationMissingException
    */
   public String[] requestMessagesUnsent() throws ECodexConnectorNationalBackendClientException,
         ImplementationMissingException;

   /**
    * Requests a certain message that has to be handled by the connector and
    * sent over the gateway.
    *
    * @param message
    *         A {@link Message} object with all data concerning the message.
    *         This object contains {@link MessageDetails} which holds the
    *         messageId of the message that is requested.
    * @throws ECodexConnectorNationalBackendClientException
    * @throws ImplementationMissingException
    */
   public void requestMessage(Message message) throws ECodexConnectorNationalBackendClientException,
```

Restricted to eCODEX consortium

```
        ImplementationMissingException;

    /**
     * Requests all new confirmations for messages delivered to the national
     * backend system before. If the national system marks a message as
     * delivered, or retrieved a {@link MessageConfirmation} should be created
     * and queued.
     *
     * @return an Array of {@link MessageConfirmation} Objects which contain
     *       informations on what message is in which confirmation state.
     * @throws ECodexConnectorNationalBackendClientException
     * @throws ImplementationMissingException
     */
    public MessageConfirmation[] requestConfirmations() throws
ECodexConnectorNationalBackendClientException,
        ImplementationMissingException;
```

These interface methods have to be implemented by the MS. For a detailed description please refer to the inline documentation of the source code found as part of the deliverable on the BSCW server.

3.2.3. ECodexConnectorEvidencesToolkit

This module is responsible for the generation of the evidences which needs to be sent by the connector according to the workflow described in the chapter 3.1. This is a library reused from a former LSP project SPOCS and it is based on the ETSI REM Standard. The usage of this module is optional and it can be switched off via the configuration file. In the sense of e-CODEX this module is mandatory because with the evidence the proof for a legal valid delivery is done. But for future use as a convergence solution this module might be optional. For a detailed description please refer to the inline documentation of the source code found as part of the deliverable on the BSCW server.

3.2.4. ECodexConnectorSecurityToolkit

This module is responsible for the generation of the Trust Ok token and the ASiCs container packaging. Basically it contains and uses the deliverables provided by WP4. The Trust Ok Token will be generated in a XML and PDF form and defines the result of the signature verification or the user information verification in the context of an advanced electronic system. The ASiCs container is used to store the payload in a signed and secure way as a ZIP Container. The usage of this module is optional and it can be switched off via the configuration file. In the sense of e-CODEX this module is mandatory, because the Trust Ok Token is needed as a basic instrument for the circle of trust. But for future use as a convergence solution this module might be optional. For a detailed description please refer to the inline documentation of the source code found as part of the deliverable on the BSCW server.

3.2.5. ECodexConnectorContentMapper

This module is responsible for the mapping between the national and e-CODEX XML schemas. The usage of this module is optional and it can be switched off via the configuration file, in the case the piloting MS does not have an existing national schema yet and will use the e-CODEX XML schema.

Restricted to e-CODEX consortium

If a MS has a national XML Schema than this module must be implemented by the MS.

The interface is defined as follows:

```
/**
 *
 * Method to map international eCodex XML to national format. Must be
 * overridden when ContentMapper is used by configuration.
 *
 * @param internationalContent
 *        - eCodex XML.
 * @return nationalContent as byte array.
 * @throws ECodexConnectorContentMapperException
 * @throws ImplementationMissingException
 */
public byte[] mapInternationalToNational(byte[] internationalContent) throws
ECodexConnectorContentMapperException,
        ImplementationMissingException;

/**
 * Method to map national XML to international eCodex format. Must be
 * overridden when ContentMapper is used by configuration.
 *
 * @param nationalContent
 * @return eCodex XML as byte array.
 * @throws ECodexConnectorContentMapperException
 * @throws ImplementationMissingException
 */
public byte[] mapNationalToInternational(byte[] nationalContent) throws
ECodexConnectorContentMapperException,
        ImplementationMissingException;
```

These two methods have to be implemented by the MS according to their national schemas. It is fully independent and up to the MS which type of technique will be used for the content (XML) mapping. If the MS implemented additional validation rules as well is also up to the MS. The interface itself is easy to understand and self explaining by just exchanging bytestreams. The e-CODEX content format, which needs to be mapped to national schemas has been defined by WP6. Therefore the deliverables of WP6 are a necessary input.

3.2.6. ECodexConnectorController

This is the main class of the connector framework and it implements the incoming and outgoing workflow as described in chapter 3.1 based on the configuration parameters. This is the main central component and it calls all other module described in this chapter according to the workflow. For a detailed description please refer to the inline documentation of the source code found as part of the deliverable on the BSCW server.

3.2.7. ECodexConnectorGatewayWebserviceClient

This module implements the Backend Web Service provided by the e-CODEX Gateway. e-CODEX has defined such a Web Service for an easy access to the Gateway. Due to the fact that this interface is the same for all MS it could be fully implemented and provided to the MS within the Connector Framework. The internal call of this module is done by the Controller Module and therefore transparent. For a detailed description please refer to the inline documentation of the source code found as part of the deliverable on the BSCW server.

Restricted to e-CODEX consortium