Deliverable WP3 D. 6.16

Project Acronym: **PEPPOL**

Grant Agreement number: **224974**

Project Title: **Pan-European Public Procurement Online**

PEPPOL Demonstrator Client

Developer Guide

Version: 1.0

Status: In use

Editors:

Joan Farfán Armas

Carlos Dávila Pino

Revision History

Version	Date	Editor	Org.	Description
2009-04-13	2009-04-13	CDP Carlos Dávila Ravina Alfa1lab	Alfa1lab	Initial
2009-04-29	2009-04-29	Pino Alfa1lab JRR Jorge Reátegui	Alfa1lab	Corrections
2009-10-16	2009-10-16	JGB José Gonzales Biminchumo Alfa1lab	Alfa1lab	Updates
2010-05-28	2010-05-28	JGB José Gonzales Biminchumo Alfa1lab	Alfa1lab	Updates and corrections
2010-09-01	2010-09-01	JGB José Gonzales Biminchumo Alfa1lab	Alfa1lab	Updates,
2010-09-13	2010-09-13	JFA Joan Farfán Armas Alfa1lab	Alfa1lab	new features and thorough review Updates
1.0	2012-07-18	Leonardo Bertini	Consip	Consolidation of the in Use version

Contributors

Organisations

Alfa1Lab

Consip

Persons

Leonardo Bertini, Consip

Carlos Dávila Ravina, Alfa1lab

Jorge Reátegui, Alfa1lab

José Gonzales Biminchumo, Alfa1lab

Joan Farfán Armas, Alfa1lab

Tables of Contents

1. Introduction

This document describes how to download the source code of "*DEMO Client*" through SVN (Eclipse Ganymedes plugin.) and how to create a development environment on your local machine.

The "Demo Client" is an open source project oriented to developers.

2. Pre-Requisites

2.1. Eclipse

Eclipse Ganymedes with SVN, MAVEN, ANT and BPEL. You can see the features at "*Help/About Eclipse platforms*".
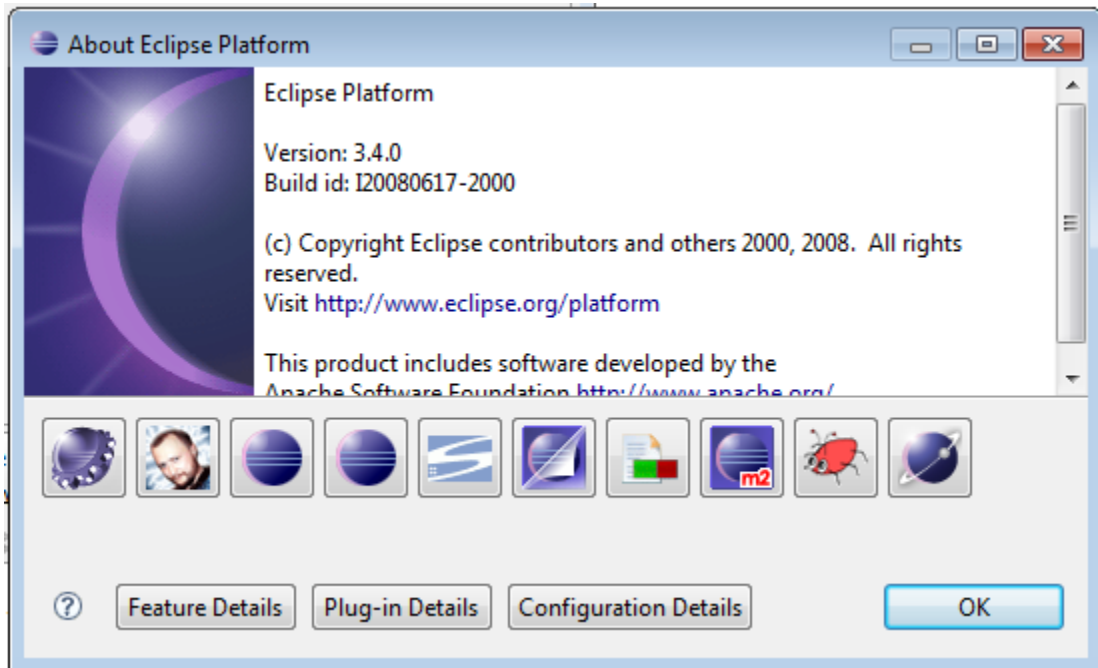

Figure 1: Some of the extra plug-ins that Eclipse Framework needs

2.2. Java version

It is necessary to install the latest JDK version for 32-bits. It is recommended to use version 20 (1.6.0_20 was used for development).

3. How to prepare your own workspace

To prepare your own workspace on Eclipse to work with the Demonstrator Client you have to install some plug-ins and download four different projects. This guide will provide you the link to download the plug-ins.

3.1. Pre-Requisites

1. Eclipse Ganymedes, you can download from http://eclipse.org/
2. Subversion (SVN) a plug-in to Eclipse, you can find a specification to download and install from: http://subeclipse.tigris.org/servlets/ProjectsProcess?pageID=p4wYuA
3. BPEL Designer a plug-in to Eclipse, install according to this guide: http://download.eclipse.org/technology/bpel/update-site/
4. To check the correct installation of BPEL and SVN, your wizard for creating new projects should look like the picture bellow:
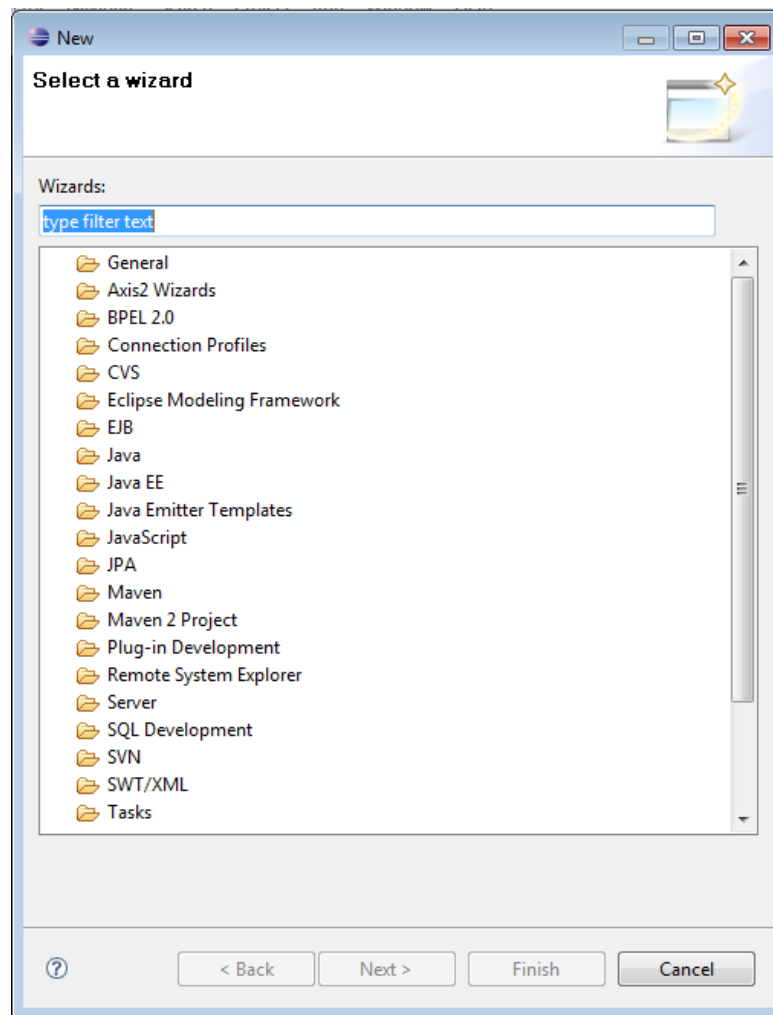

Figure 2: Kind of projects that Eclipse should allow (SVN, BPEL)

3.2. Overview

The Demonstrator Client project is split in four different Java Projects. The first one is the Demo client project (*demo_client*) which is the application itself (in charge of the visual parts), the second one is the Demo Workflow Engine (*demo_workflow_engine*) which contributes to the flow of the project and the other one is the Java Plug-in project (*java_plugin*) which supports the sending and receiving. This project deals directly with the transport libraries. To start developing with the demo client, first you have to import the project previously downloaded into to your workspace then you need to compile the Java Plug-in project and export it as a jar (plugin.jar). Then the plug-in jar needs to be placed into a folder within the Demo Client project (*demo_client/plugins/plugin.jar*). And finally the validation engine project (*validation_engine*) to validate documents in layers.
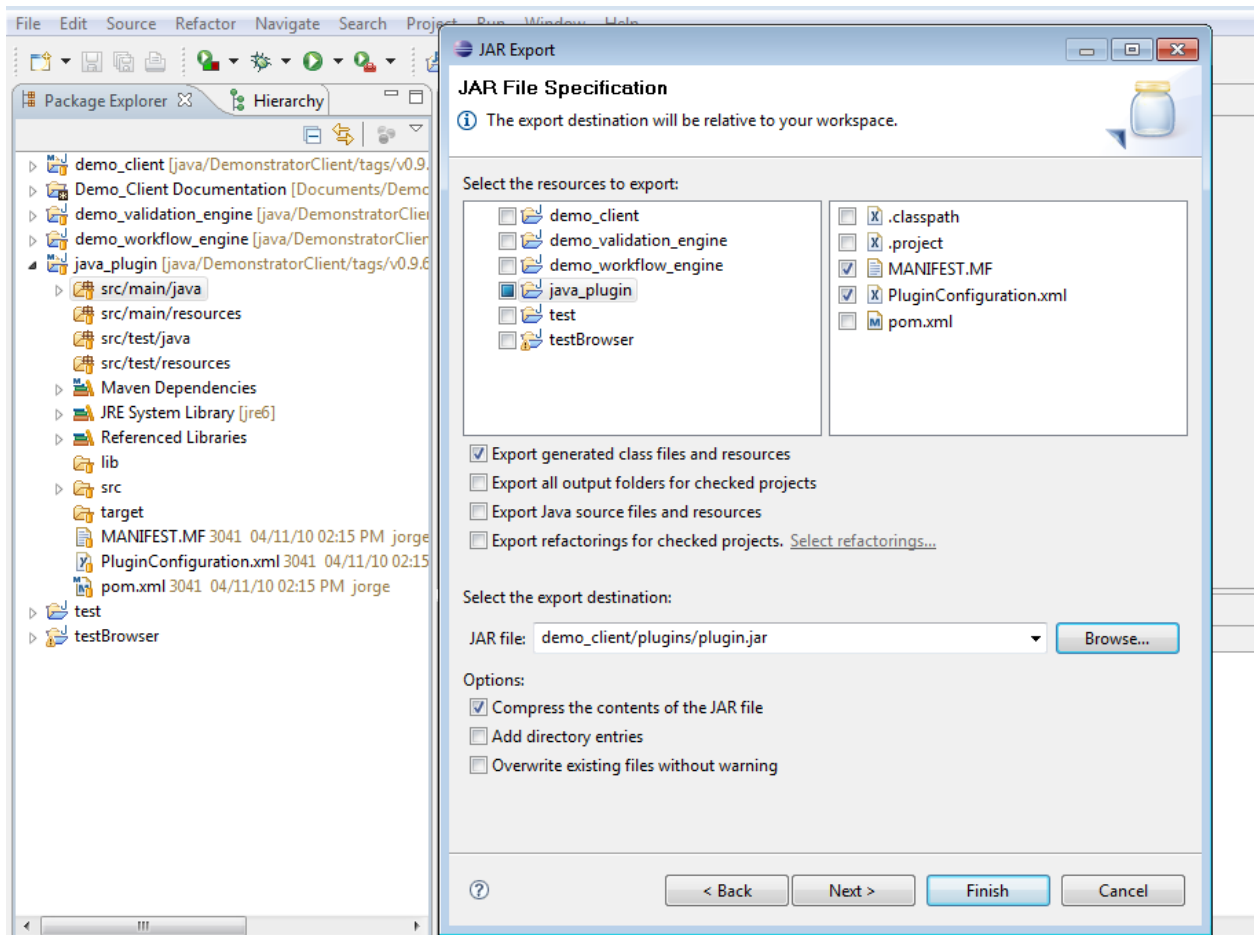
Figure 3: Internal plug-ins

The demonstrator client contains classes that enable it to support and handle all types of xml documents this is expressed in an xml configuration file.

8

DemoClienConfiguration.xml: This is a file where the configuration of the client is stored (e.g. associations between document types and document namespace + document element name.

Figure 4: DEMOClientConfiguration.xml file

3.3. MANIFEST.MF

Contains a list of jars that the application requires to work once it's converted into a binary. This file is used to build and execute the Demo Client.

Figure 5: Manifest.MF with basic plug-ins that the Client needs for a binary version

3.4. POM.XML

Contains the structure of libraries the maven project will download using nexus.


```xml
<!-- Repositories to search JAR files -->
<repositories>
    <repository>
        <id>nexus2</id>
        <name>Maven2 Release Repository</name>
        <url>http://pedri.alfa1lab.com:8081/nexus/content/groups/public</url>
    </repository>
</repositories>

<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.5</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>democlient</groupId>
        <artifactId>busdox-transport-lime-library</artifactId>
        <version>1.0.1</version>
    </dependency>
    <dependency>
        <groupId>democlient</groupId>
        <artifactId>busdox-transport-commons</artifactId>
        <version>1.0.1</version>
    </dependency>
    <dependency>
        <groupId>democlient</groupId>
        <artifactId>commons-logging</artifactId>
        <version>1.1.1</version>
    </dependency>
    <dependency>
```
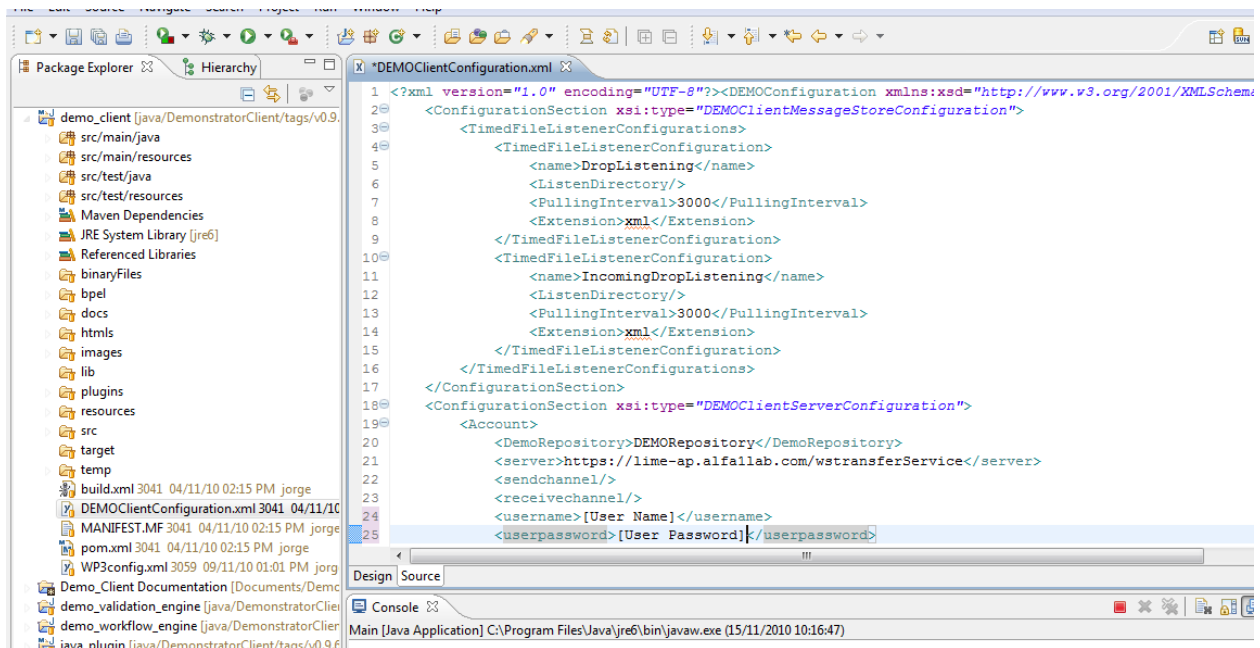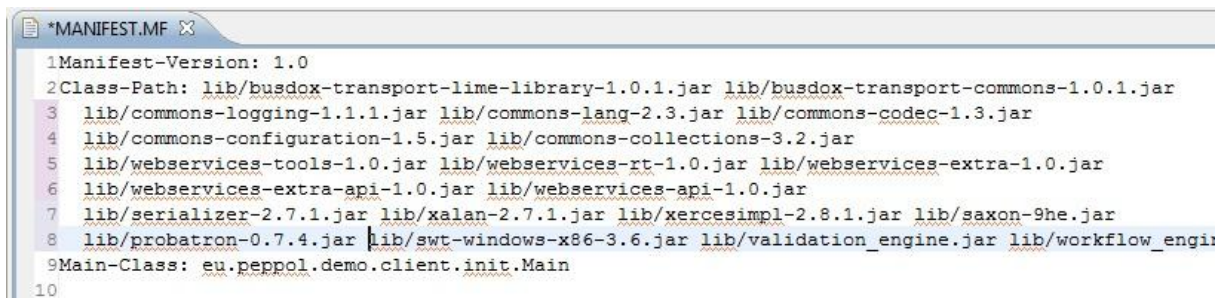
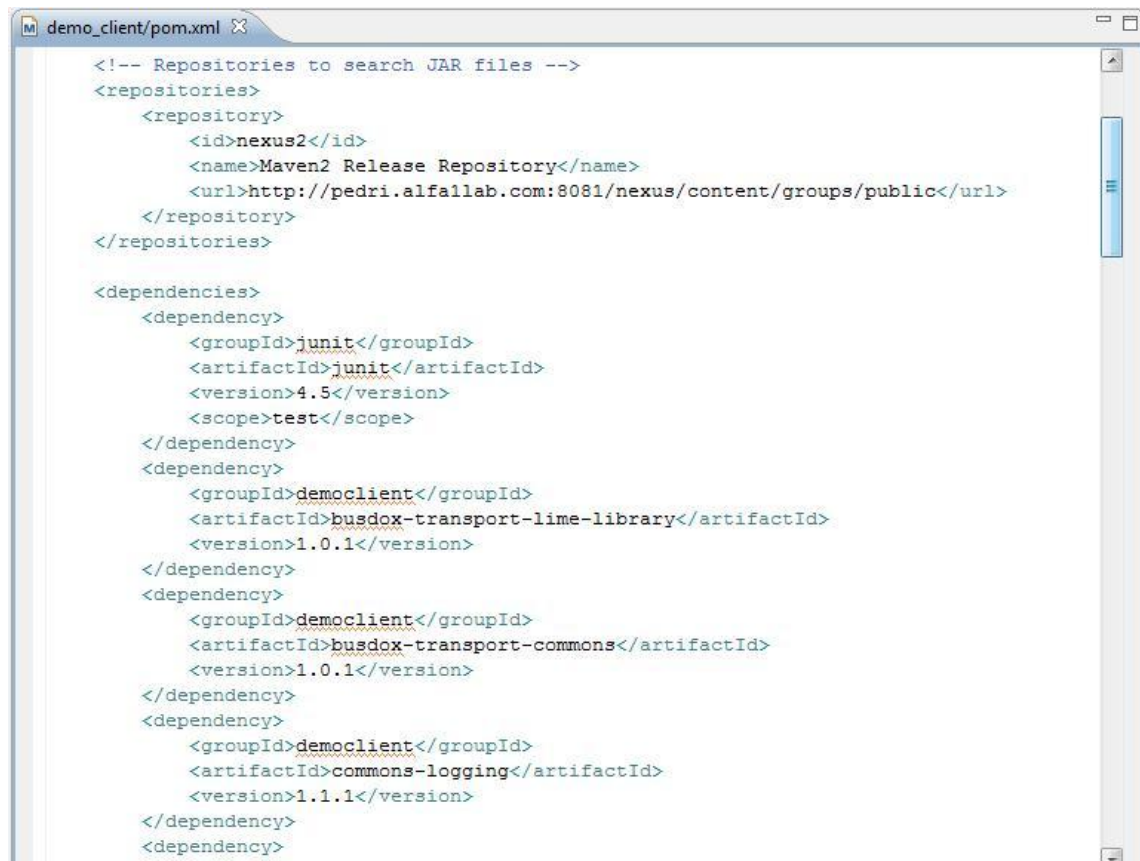*Figure 6: File that downloads basic plug-ins trough maven*

3.5. WP3Config.xml

This file allows the Demo Client to choose between two possible roles (Contract authority or Economic Operator). The Client now allows changing it trough the User Interface.


```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<config>
  <role>EcOp</role>
  <language>en</language>
</config>
```
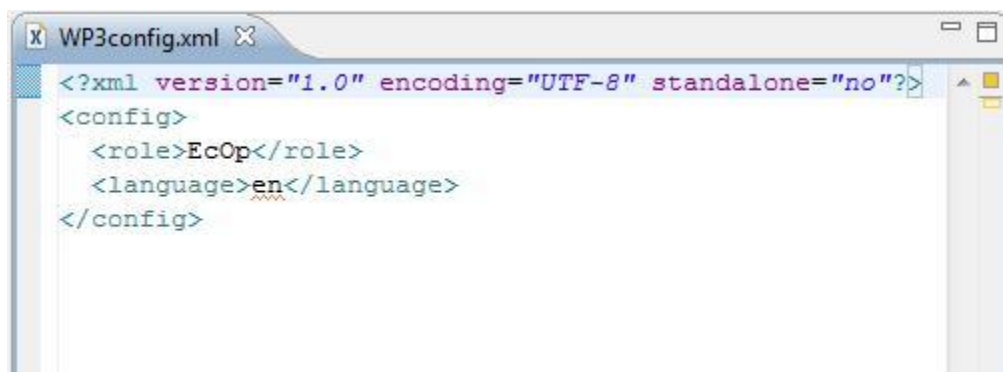
Figure 7: WP3config.xml

10

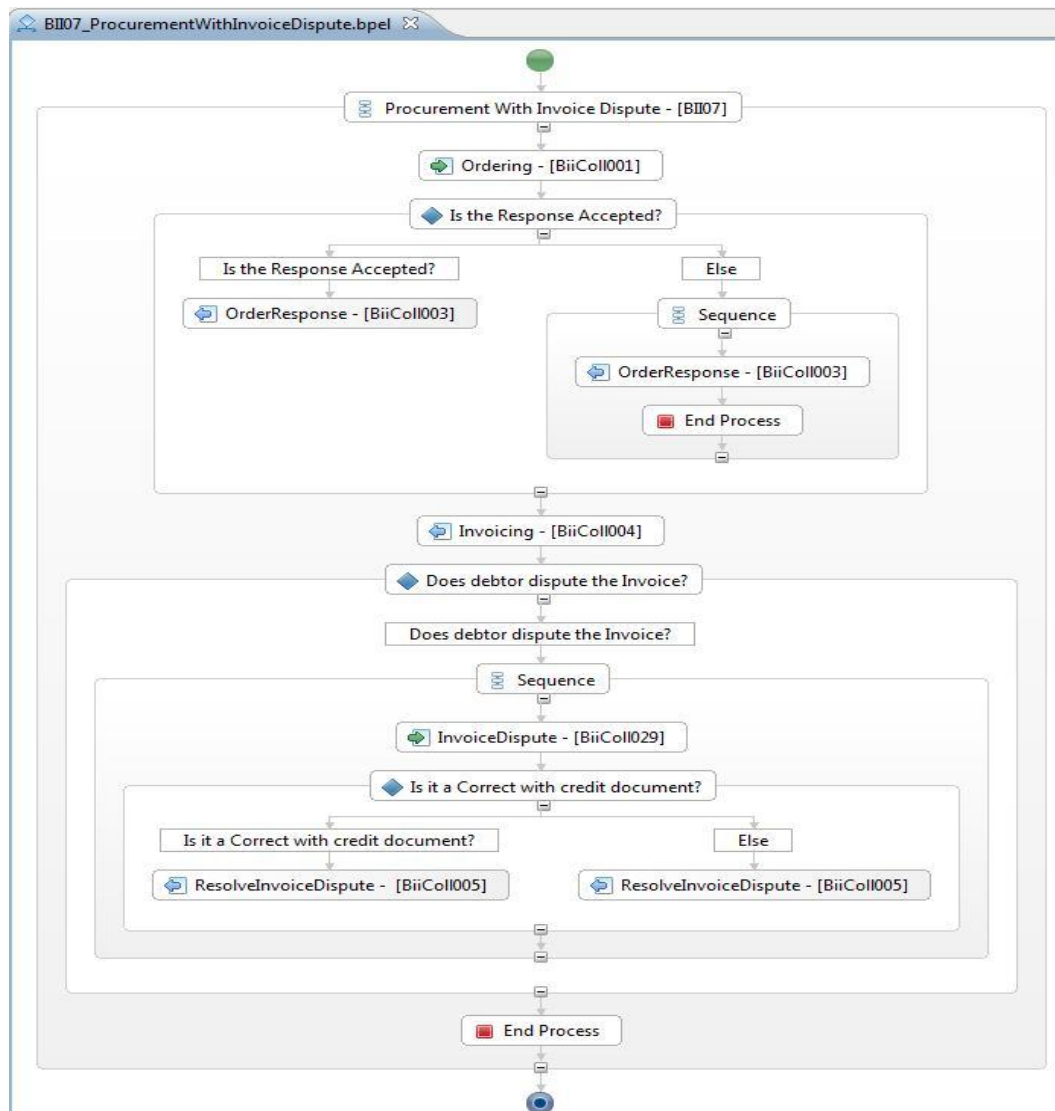## 3.6. **BII07 ProcurementWithInvoiceDispute.bpel**

Figure 8: BII07

3.7. BII19 AdvancedProcurement.bpel
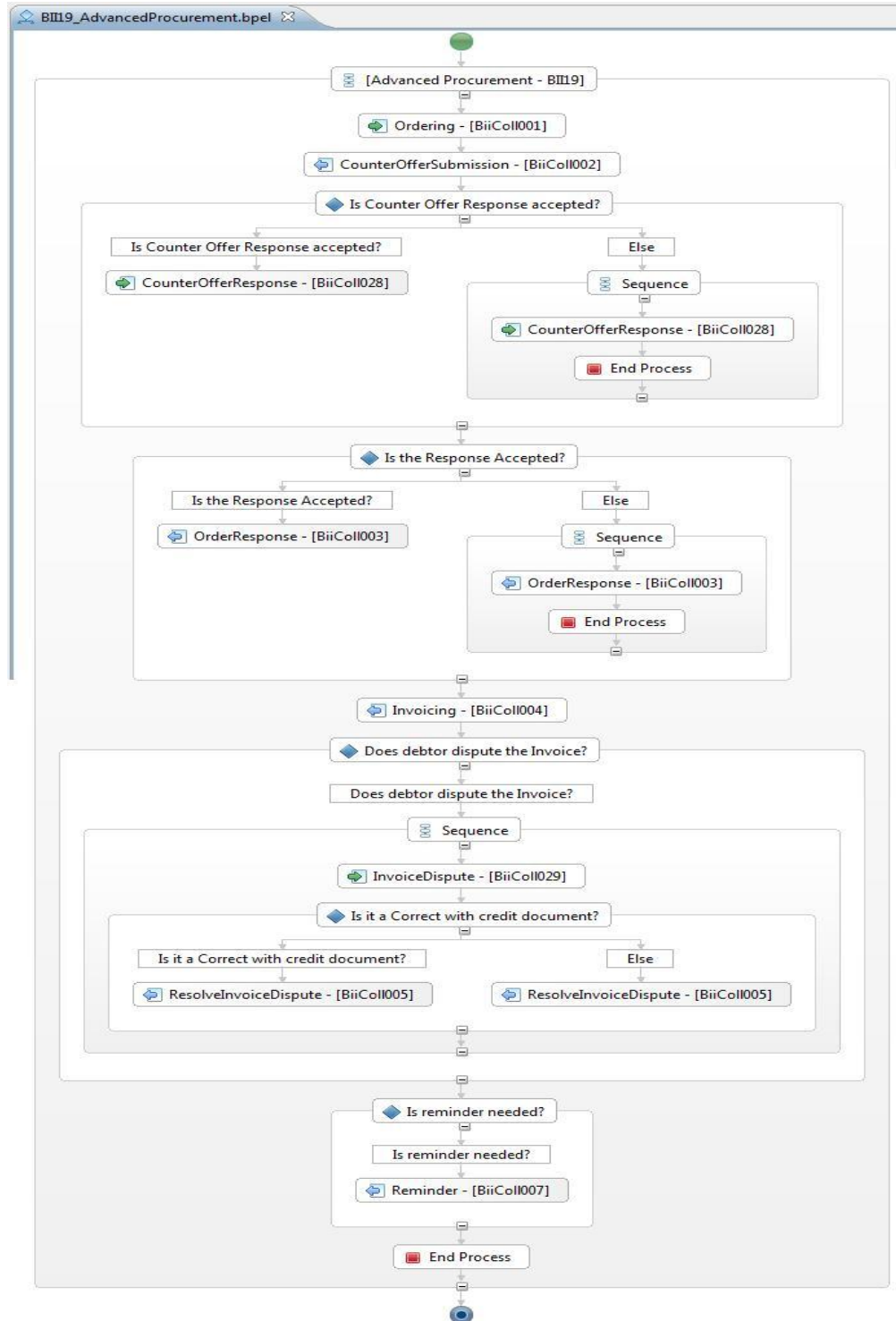
Figure 9: BII19

3.8. BII01_CatalogueOnly.bpel
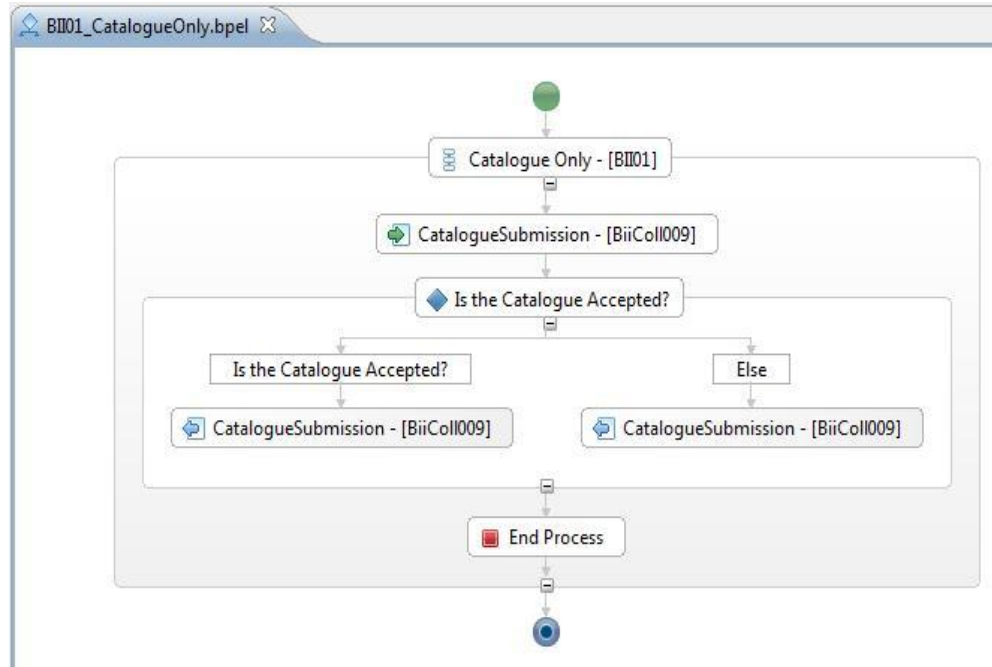
Figure 10: BII01

3.9. BII17_MultiPartyCatalogue.bpel
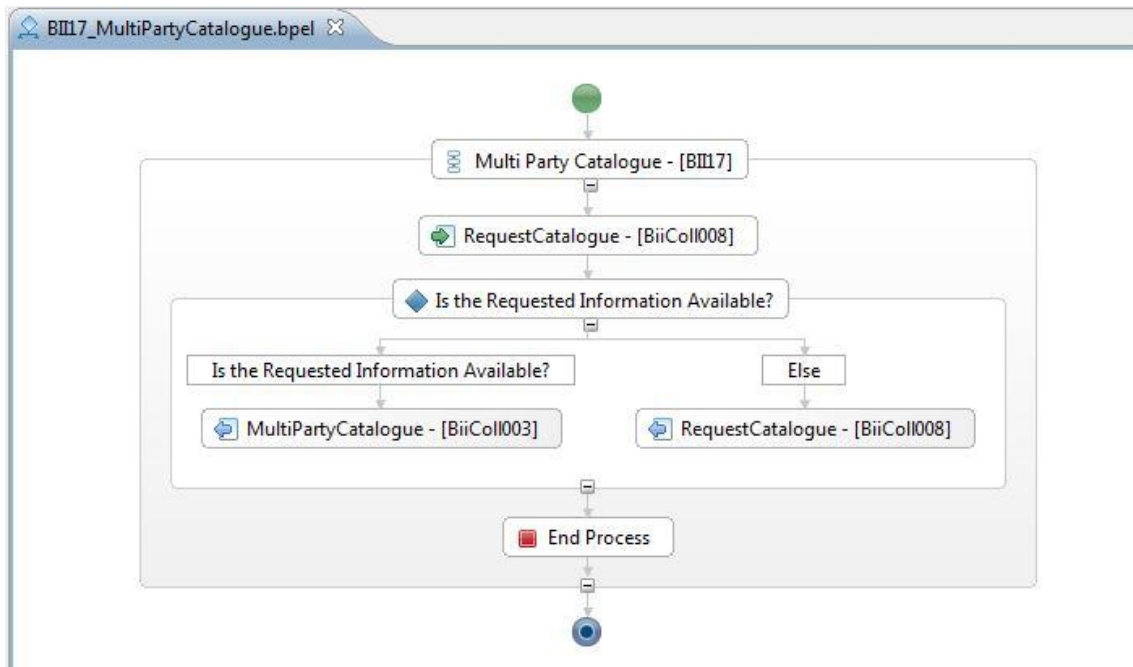
Figure 11: BII17

3.10. Folder structure

The client requires a folder structure to be in place for handling documents that are being created, sent or received. This structure is automatically created by the application the first time it is run. If it is deleted by accident the application will create it again. It is possible to create more than one "*DemoRepository*" folders in one computer but it is not possible to change a folder´s location once it is created.

Figure 12: Folder structure

4. Source code layout

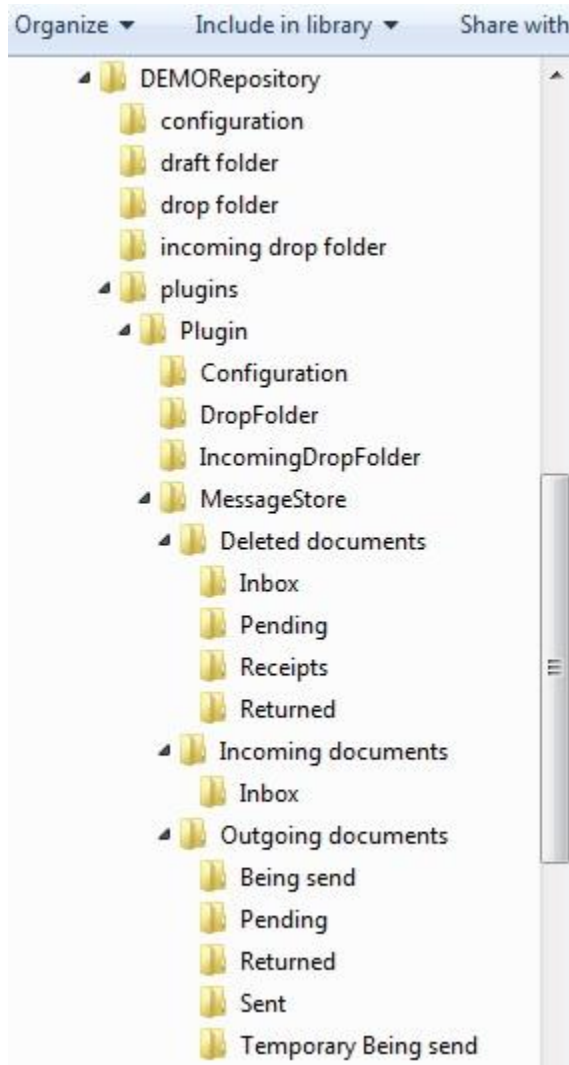| Path	Document	Description
java/DemonstratorClient/tags/v9.6.10/		PEPPOL source root
demo_client		Demonstrator client root folder
/bpel/	BII07_SimpleProcurement.bpel	BPEL document that describe the flow of the simple profile.
	BII19_AdvancedProcurement.bpel	BPEL document that describe the flow of the advanced profile.
	BII01_CatalogueOnly.bpel	Profile with catalogues.
	BII03_BasicOrderOnly.bpel	
	BII04_BasicInvoiceOnly.bpel	
	BII06_Procurement.bpel	
	PreAward.bpel	
	BII17_MultiPartyCatalogue.bpel	Profile with catalogues.
/htmls/	html.html	HTML interface for the documents
	validationResult.html	HTML for the interface of the validation
	Index.html	HTML interface for the documents creation
/htmls/css	OIOUBL.css	
	autoComplete.css	
	Calendar.css	
	displayStyle.css	
	Style.css	
/htmls/img	Images	
/htmls/js	js-disableRightClick.js	Javascript methods to disable right clicks.

	Js-autoComplete.js	Javascript methods to auto complete fields
	js-documentsCreation.js	Javascript methods to the document creation
	js-tabber.js	Javascrpit to tab.
	js-validationResult.js	Javascript validate results of the document creation.
/htmls/js/documents	types	All supported documents with their own java scripts methods
/htmls/js/calendar	calendar_db.js	
	calendar_eu.js	
/lib/		Libraries used by the Client.
/plugins/		Repository of plug-ins
/resources/		Schemas needed to validate various document types.
/resources/CSVConverter/CSV examples v1.0	BiiCoreTrdm018.csv, BiiCoreTrdm019.csv, BiiCoreTrdm054.csv, BiiCoreTrdm055.csv, BiiCoreTrdm057.csv, BiiCoreTrdm058.csv	Some templates examples to import CSV documents.
/resources/CSVConverter/export	CsvExportModel.xsl	The structure to do the export csv document.
/resources/CSVConverter/	BiiCoreTrdm018.xslt, BiiCoreTrdm019.xslt, BiiCoreTrdm054.xslt, BiiCoreTrdm055.xslt, BiiCoreTrdm057.xslt, BiiCoreTrdm058.xslt	Xslt files to build the interface.
/resources/GUIXSL/AcceptCatalogue	AcceptCatalogue-Template.xsl, AcceptCatalogue.xml, details.xsl	Group of document to implement the "document interface creation".
/resources/GUIXSL/AcceptCounterOffer	AcceptCounterOffer-Template.xsl, AcceptCounterOffer.xml, details.xsl	
/resources/GUIXSL/AcceptOrder	AcceptOrder-Template.xsl, AcceptOrder.xml, details.xsl	
/resources/GUIXSL/Catalogue	Catalogue-Template.xsl, Catalogue.xml, catalogueLine.xsl, contractorCustomer.xsl, details.xsl, sellerSupplier.xsl	

/resources/GUIXSL/CatalogueRequest	CatalogueRequest-Template.xsl, CatalogueRequest.xml, catalogueRequestLine.xsl, details.xsl, providerParty.xsl, receiverParty.xsl	
/resources/GUIXSL/CorrectWithCredit	CorrectWithCredit-Template.xsl, CorrectWithCredit.xml, details.xsl	
/resources/GUIXSL/CorrectWithDebit	CorrectWithDebit-Template.xsl, CorrectWithDebit.xml, details.xsl	
/resources/GUIXSL/DisputeInvoice	details.xsl, DisputeInvoice-Template.xsl, DisputeInvoice.xml	
/resources/GUIXSL/IssueReminder	details.xsl, IssueReminder-Template.xsl, IssueReminder.xml	
/resources/GUIXSL/MultiPartyCatalogue	catalogueLine.xsl, contractorCustomer.xsl, details.xsl, MultiPartyCatalogue-Template.xsl, MultiPartyCatalogue.xml, sellerSupplier.xsl	
/resources/GUIXSL/RejectCatalogue	details.xsl, RejectCatalogue-Template.xsl, RejectCatalogue.xml	
/resources/GUIXSL/RejectCatalogueRequest	details.xsl, RejectCatalogueRequest-Template.xsl, RejectCatalogueRequest.xml	
/resources/GUIXSL/RejectCounterOffer	details.xsl, RejectCounterOffer-Template.xsl, RejectCounterOffer.xml	
/resources/GUIXSL/RejectOrder	details.xsl, RejectOrder-Template.xsl, RejectOrder.xml	
/resources/GUIXSL/SubmitCounterOffer	details.xsl, SubmitCounterOffer-Template.xsl, SubmitCounterOffer.xml	
/resources/GUIXSL/SubmitInvoice	details.xsl, SubmitInvoice-Template.xsl, SubmitInvoice.xml,	
/resources/GUIXSL/SubmitOrder	details.xsl, SubmitOrder-Template.xsl, SubmitOrder.xml	
/resources/I18N/display/	OIOUBL_Headlines_default.xml, OIOUBL_Headlines_en.xml	Internationalization files
/resources/documents creation/	peppol_default.xsl, peppol_en.xsl	Templates
/resources/ODSConverter/functions	functions v1.0.xsl	File use to participate of the ods import files
/resources/ODSConverter/ODS examples v1.0	BiiCoreTrdm019.ods	Ods Template to use for import ods file.

/resources/ODSConverter/	BiiCoreTrdm019.xsl	
/resources/UI	MessageUndeliverable.xsl	File used to notify the user that the message was undeliverable.
/resources/UI/OIOUBL v2.01/Stylesheets	OIOUBL_AcceptCatalogue.xsl	Stylesheet for each kind of document.
	OIOUBL_AcceptCounterOffer.xsl	
	OIOUBL_AcceptOrder.xsl	
	OIOUBL_Catalogue.xsl	
	OIOUBL_CatalogueRequest.xsl	
	OIOUBL_CommonTemplates.xsl	
	OIOUBL_CorrectWithCredit.xsl	
	OIOUBL_CorrectWithDebit.xsl	
	OIOUBL_DisputeInvoice.xsl	
	OIOUBL_IssueReminder.xsl	
	OIOUBL_MultiPartyCatalogue.xsl	
	OIOUBL_RejectCatalogue.xsl	
	OIOUBL_RejectCatalogueRequest.xsl	
	OIOUBL_RejectCounterOffer.xsl	
	OIOUBL_RejectOrder.xsl	
	OIOUBL_SubmitCounterOffer.xsl	
	OIOUBL_SubmitInvoice.xsl	
	OIOUBL_SubmitOrder.xsl	
/resources/ValidationEngine/layers/BII01	BiiCoreTrdm019_Layer.xml	About validations
	BiiCoreTrdm057_Layer.xml	
	BiiCoreTrdm058_Layer.xml	
/resources/ValidationEngine/layers/BII07	BiiCoreTrdm001_Layer.xml	Layers to validate the

		BII01 profile
	BiiCoreTrdm002_Layer.xml	
	BiiCoreTrdm003_Layer.xml	
	BiiCoreTrdm010_Layer.xml	
	BiiCoreTrdm013_Layer.xml	
	BiiCoreTrdm014_Layer.xml	
	BiiCoreTrdm015_Layer.xml	
/resources/ValidationEngine/layers/BII17	BiiCoreTrdm018_Layer.xml	Layers to validate the BII17 profile
	BiiCoreTrdm054_Layer.xml	
	BiiCoreTrdm055_Layer.xml	
/resources/ValidationEngine/layers/BII19	BiiCoreTrdm001_Layer.xml	Layers to validate the BII19 profile
	BiiCoreTrdm002_Layer.xml	
	BiiCoreTrdm003_Layer.xml	
	BiiCoreTrdm004_Layer.xml	
	BiiCoreTrdm005_Layer.xml	
	BiiCoreTrdm006_Layer.xml	
	BiiCoreTrdm010_Layer.xml	
	BiiCoreTrdm013_Layer.xml	
	BiiCoreTrdm014_Layer.xml	
	BiiCoreTrdm015_Layer.xml	
	BiiCoreTrdm017_Layer.xml	
/resources/ValidationEngine/testing/xml+sch/	add_bad_namespace.xml	files that works with schematrons.
	add_bad.xml	
	add_good_namespace.xml	

	add_good.out.xml	
	add_good.xml	
	add_namespace.sch	
	add.sch	
Resources/ValidationResult/	validationResult.xml	Files to validate.
	validationResult.xsl	
Resources/temp/xmlDocuments/	OIOUBL_AcceptCatalogue.xml	Temporal templates documents
	OIOUBL_AcceptCounterOffer.xml	
	OIOUBL_AcceptOrder.xml	
	OIOUBL_Catalogue_Base.xml	
	OIOUBL_Catalogue.xml	
	OIOUBL_CatalogueRequest_Base.xml	
	OIOUBL_CatalogueRequest.xml	
	OIOUBL_CorrectWithCredit.xml	
	OIOUBL_CorrectWithDebit.xml	
	OIOUBL_DisputeInvoice.xml	
	OIOUBL_IssueReminder.xml	
	OIOUBL_MultiPartyCatalogue_Base.xml	
	OIOUBL_MultiPartyCatalogue.xml	
	OIOUBL_RejectCatalogue.xml	
	OIOUBL_RejectCatalogueRequest.xml	
	OIOUBL_RejectCounterOffer.xml	
	OIOUBL_RejectOrder.xml	
	OIOUBL_SubmitCounterOffer.xml	

	OIOUBL_SubmitInvoice.xml	
	OIOUBL_SubmitOrder.xml	
/resources/temp/	/baseDocuments/…	The structure of the documents that need to load some data based in other document.
	Communications.xml	This file will save data for the communication between clients.
	Contacts.xml	This file saves data for each contact.
	ProfileList.xml	This file contains the structure for all kind of profiles that the client works.
	temp.xml	This file contains data save for the business transaction process.
build.xml		File to build a binary from Eclipse.
DEMOClientConfiguration.xml		Save data to configure the name of the repository, endpoints, etc/
MANIFEST.MF		This file contains the sign of the author, reference to libraries and the name of the main class to allow the application run.
pom.xml		This file is created to download the libraries from a repository.
WP3config.xml		To configure what kind of authority contractor you are.

5. Lesson: Compiling the demonstrator client

To compile the demonstrator client, first you have to download the source code from SVN Repository, using eclipse.

- Demo Client
  https://svn.forge.osor.eu/svn/peppol/java/DemonstratorClient/tags/v0.9.6.0/demo_client/
- Workflow Engine
  https://svn.forge.osor.eu/svn/peppol/java/DemonstratorClient/tags/v0.9.6.0/demo_validation_engine/
- Java Plug-in
  https://svn.forge.osor.eu/svn/peppol/java/DemonstratorClient/tags/v0.9.6.0/java_plugin/
- Validation Engine
  https://svn.forge.osor.eu/svn/peppol/java/DemonstratorClient/tags/v1.0.0.14/demo_validation_engine

6. Lesson: Setting up for local testing

To test the Client's connection against an Access Point service, open the configuration and go to Test Configuration tab and press *Test Client Configuration*
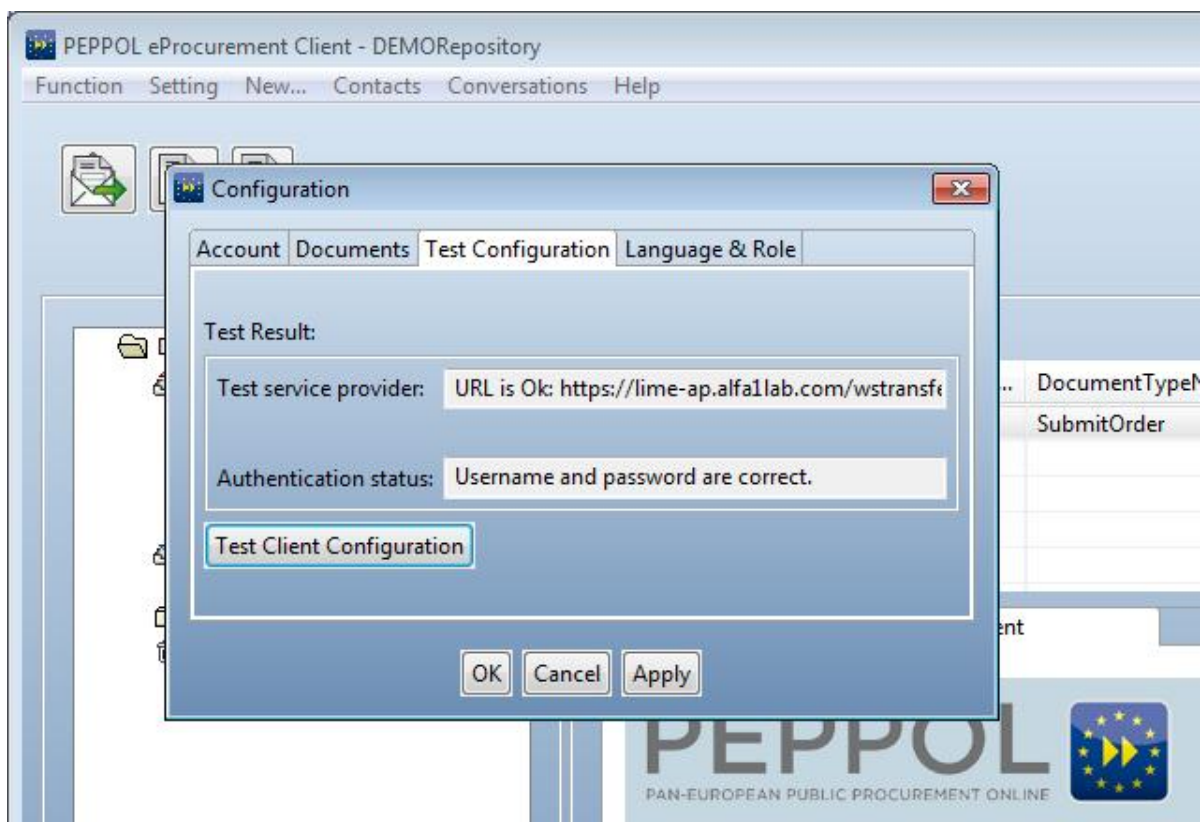
Figure 13: Testing account configuration

22

7. Lesson: About BPEL

BPEL (Business Process Execution Language) is now working with the business process of Demonstrator Client. BPEL allows creating the flow of a process with dynamic tools within the Eclipse wizard. After building a diagram flow, BPEL will create different files in your workspace. One of the files has the *.bpel* extension similar to an *.xml* file; you can parse the file using any library for parsing XML.

7.1. Creating a new BPEL

Start creating a new BPEL Project as the picture bellow:
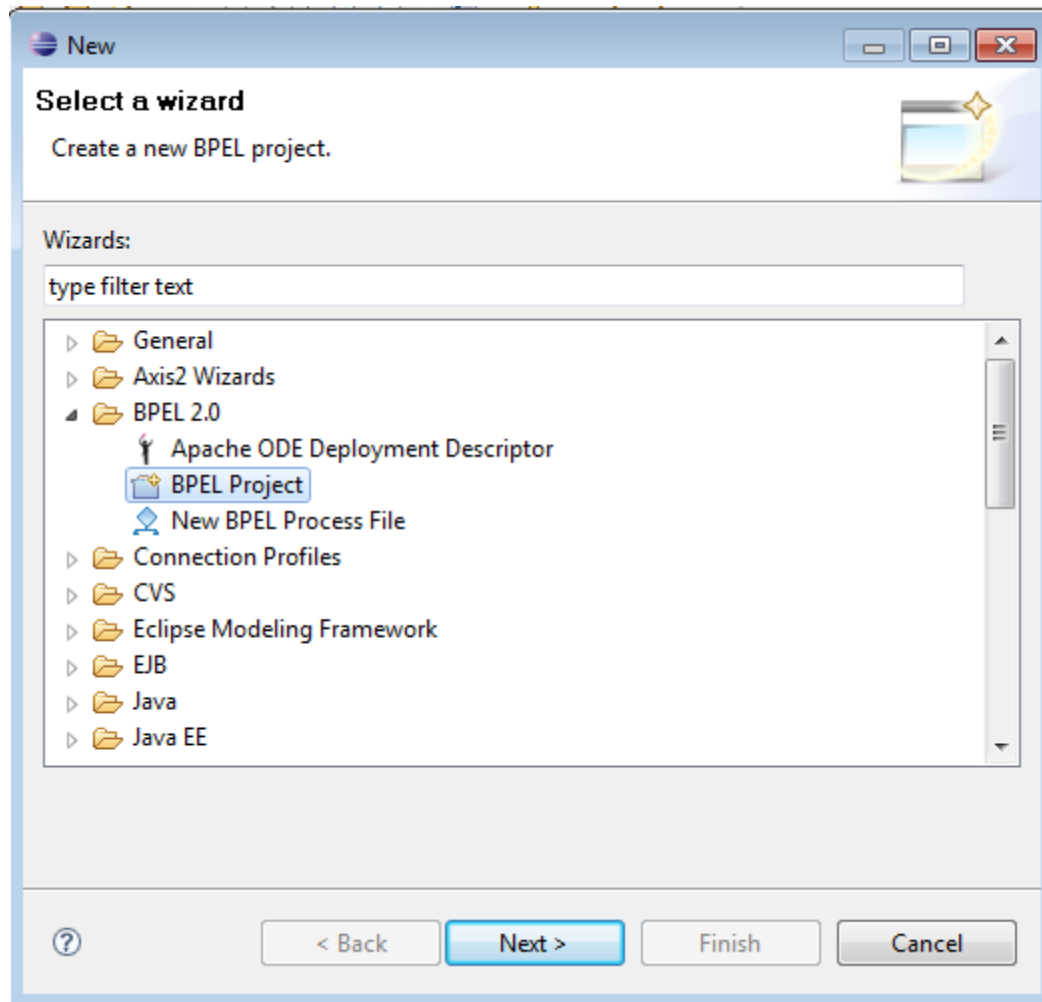
Figure 14: Choosing BPEL Project
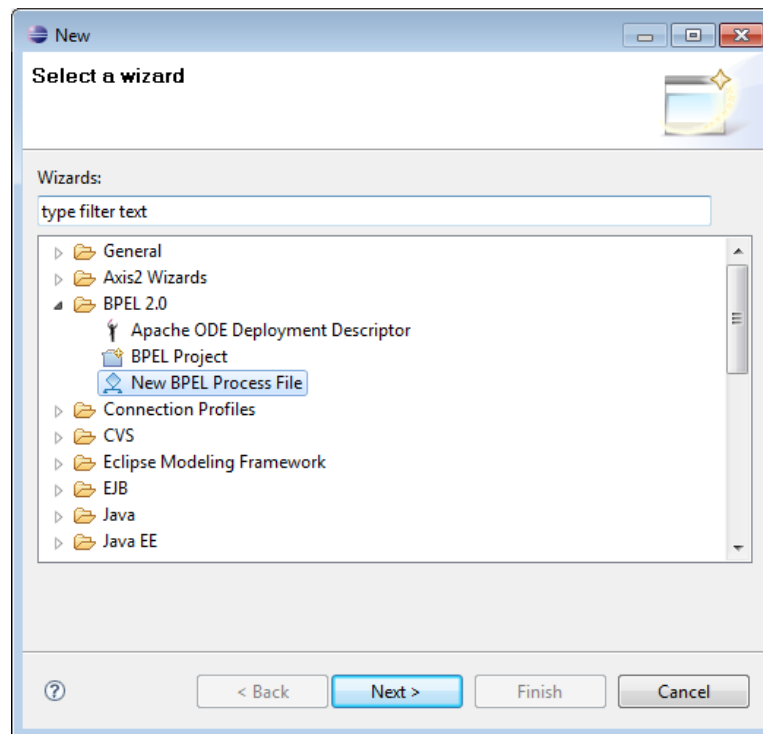
Figure 15: New BPEL Process File
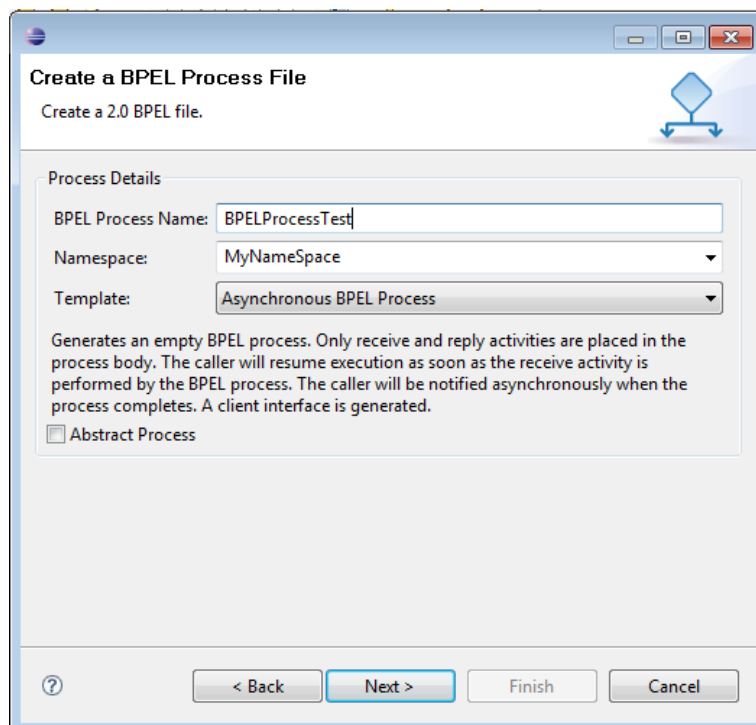
Figure 16: Fill BPEL project information

Then you have to select your BPEL Process file and finally create your own BPEL Model.
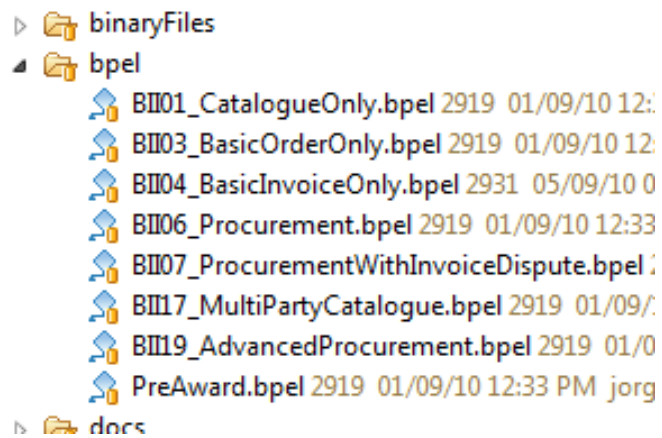
7.2. Folder structure

Figure 17: Current Profiles that Demonstrator Client supports

8. Lesson: Lime library

The documents will be sent from one client to another using the transport libraries. You can see the libraries referenced in the pom.xml file located into the Java Plug-in project.

Figure 18: Importing the lime library from maven

Demonstrator Client uses Nexus (see this link for references: http://nexus.sonatype.org/ as a repository to download libraries.

The libraries the demo client uses for the document interchange through the transport libraries are *busdox-transport-lime-library-1.0.1.jar* and *busdox-transport-common-1.0.1.jar*.

8.1.   Folder structure

/demo_client/src/eu/demo/client/transport/LightW eightProfileHandler.java	The main class that use the lime library
Maven Dependencies/busdox-transport-lime-library-1.0.1.jar	Path of the library
Maven Dependencies/busdox-transport-commons-1.0.1.jar	Path of the library
/java_plugin/pom.xml	For the libraries referenced
/demo_client/DEMOClientConfiguration.xml	To see the exact parameters information for sending and receiving documents review the last lines in this xml.
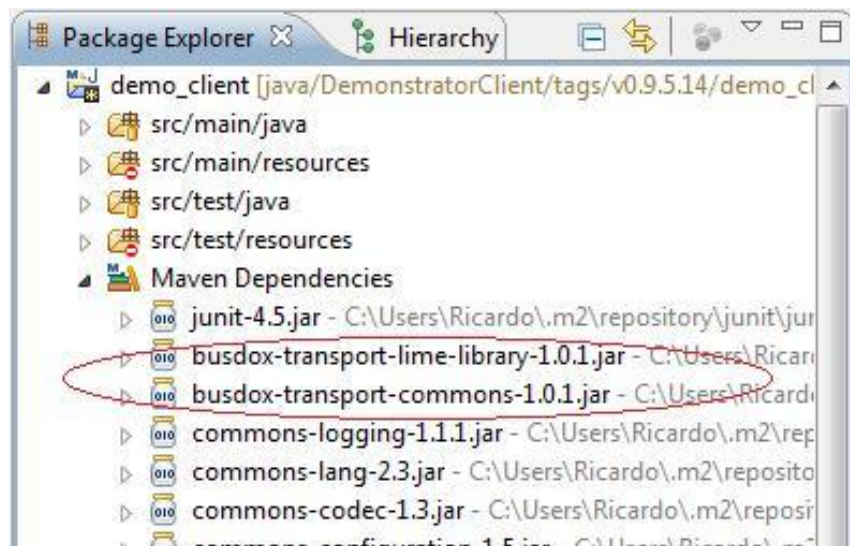
Figure 19: Imported Lime library

8.2. Set up parameters and functionality

There is a Java class into the demo client project called *LightWeightProfileHandler.java* that will be used to handle the lime library, set up parameters and use some inner methods to create the connection.

First you have to see the main parameters for the main communication.
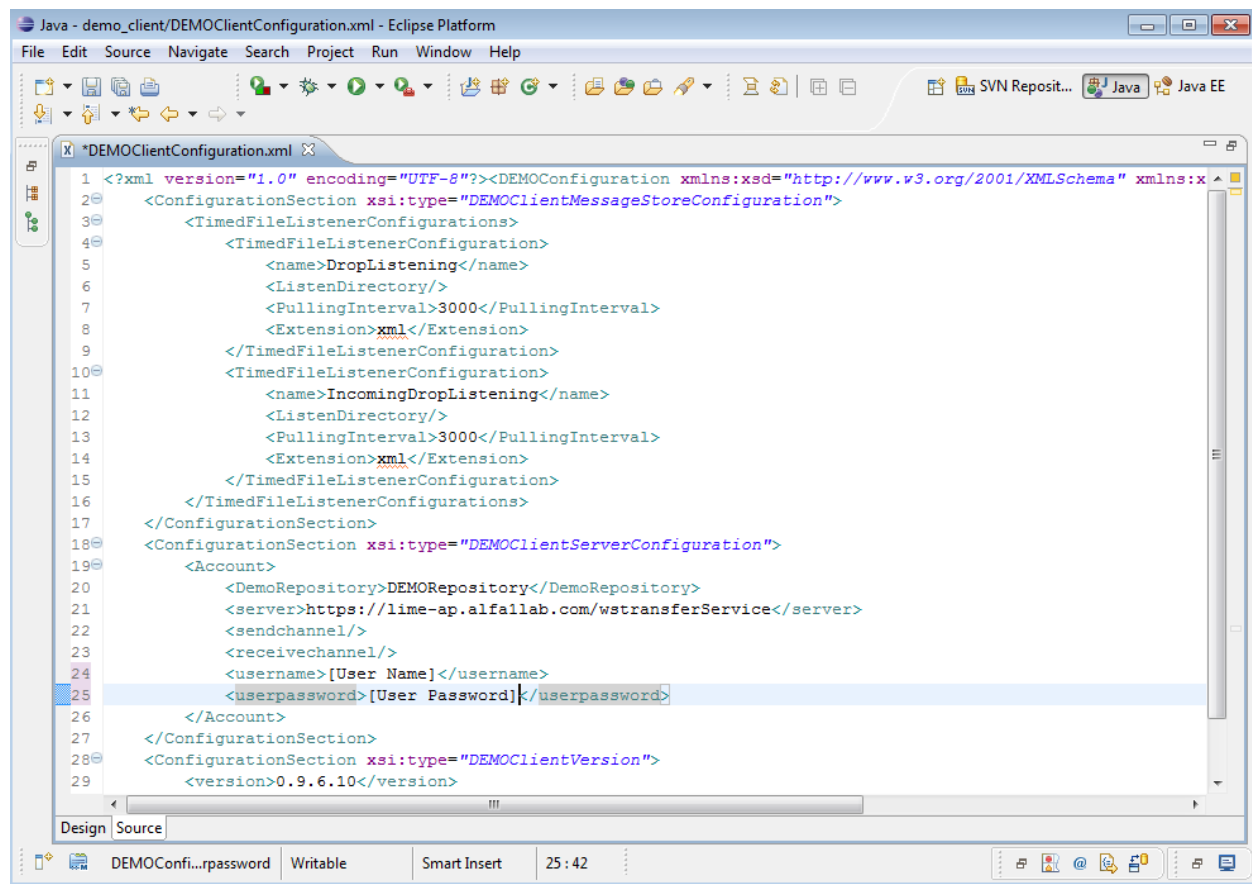
- DEMOClientConfiguration.xml

Figure 20: DEMOClientConfiguration.xml

9. How to create a "*Demo Client*" binary

To create your own binary from Eclipse it will be used ANT plug-in. First open the ANT view, drag the "build.xml" to the ant window and double click on it.
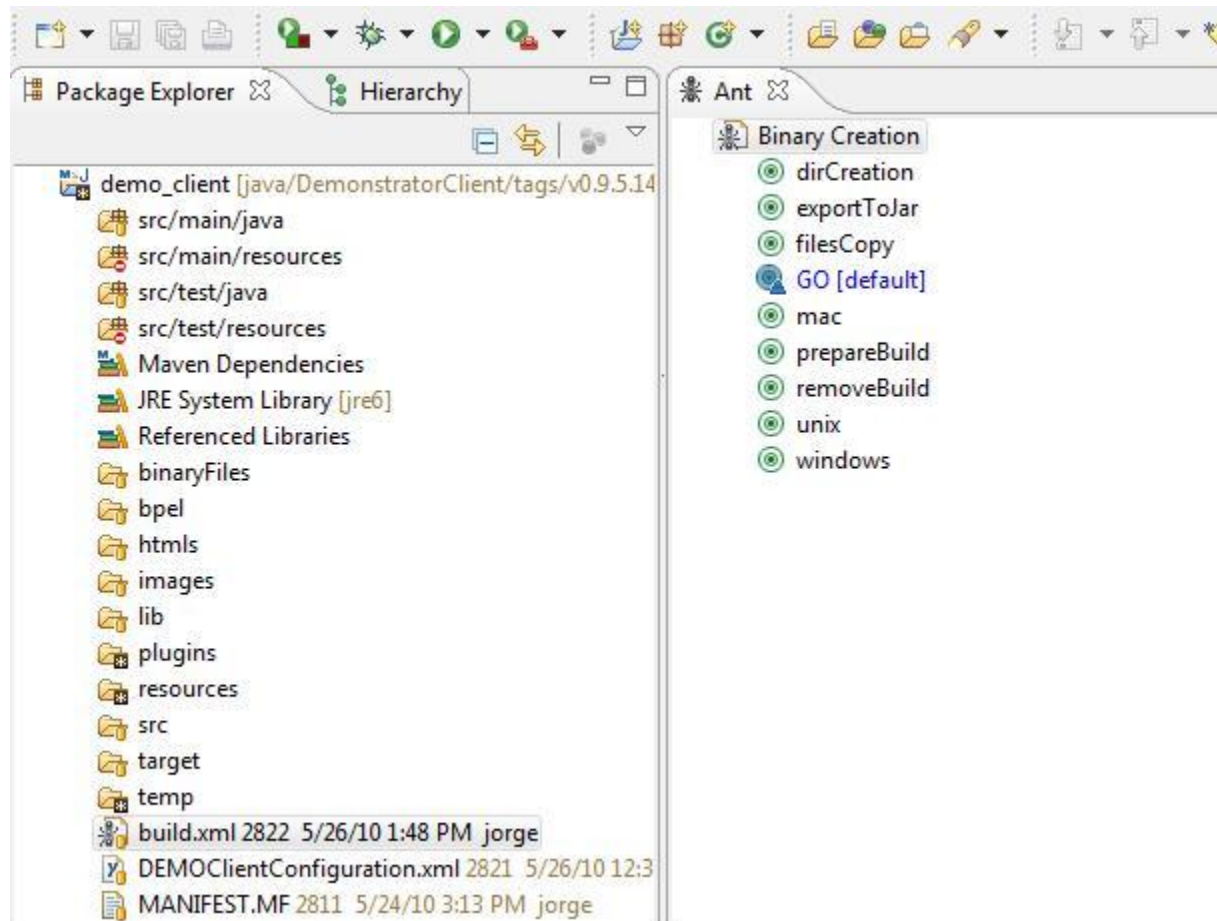
Every file will be created in your home folder.

Figure 21: Creating a binary trough Ant

10.   Validation Engine

This project is part of "*Demonstrator client*" development. The validation engine was built as a plug-in project that allows the client to perform validation by layers.
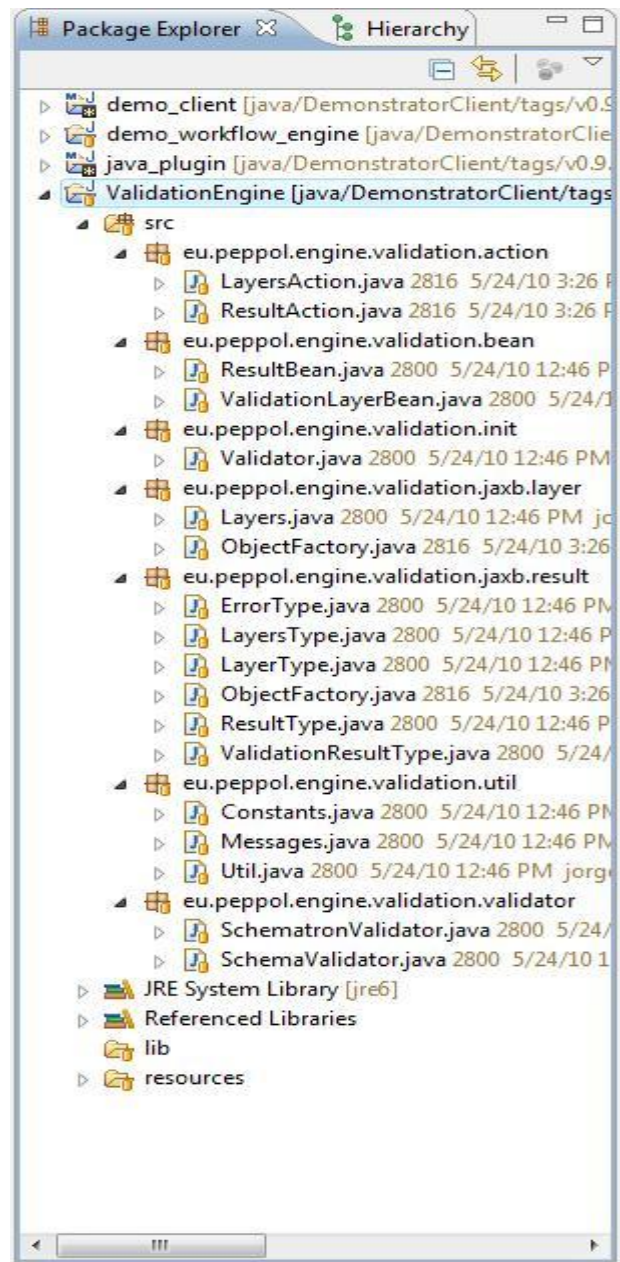

Figure 22: Source of the Validation Engine Project

When you run up the client you will see the function of the validation engine pressing the button "*Validate*" in the document creation interface.
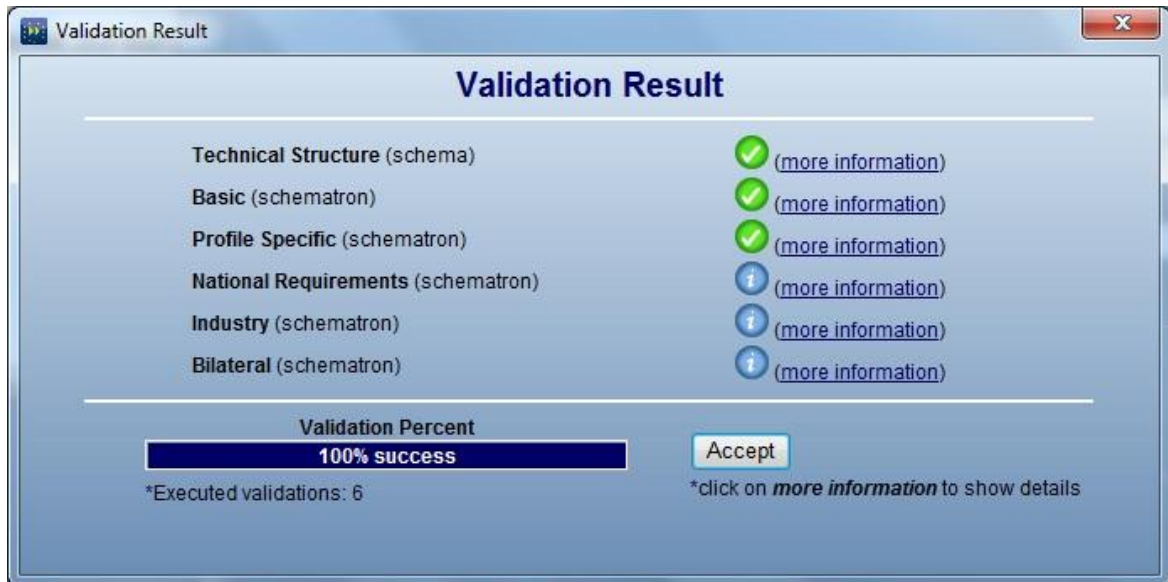

Figure 23: Example of a Validation Result