



Smart Open Services for European Patients  
Open eHealth initiative for a European large scale pilot of  
Patient Summary and Electronic Prescription

OpenNCP Continuous Integration & Conformance Test design and  
engineering details

APPENDIX F  
D 3.B.2


WORK PACKAGE	<b>WP3.B</b>
DOCUMENT NAME	OpenNCP CI & Conformance Test design and engineering details
SHORT NAME	OpenNCP CI & Conformance Test
DOCUMENT VERSION	<b>1.0.2</b>
DATE	<b>30/06/2014</b>

COVER AND CONTROL PAGE OF DOCUMENT	
Document name:	<b>OpenNCP CI &amp; Conformance Test design and engineering details</b>
Document Short name:	<b>OpenNCP CI &amp; Conformance Test</b>
Distribution level	<b>PU</b>
Status	Final
Author(s): Organization:	OpenNCP Community <b>OpenNCP Community</b>

Dissemination level: PU = Public, PP = Restricted to other programme participants, RE = Restricted to a group specified by the consortium, CO = Confidential, only for members of the consortium.


ABSTRACT
OpenNCP CI & Conformance Test design and engineering details: represents a basic reproduction from the online documentation created by the OpenNCP Community while planning, executing and controlling this epSOS Phase 2 new service implementation.

Change History					
Version	Date	Status Changes	From	Details	Review
V1.0	28/12/13	Draft	L. Mano	Reproduced from the OpenNCP Knowledge Repository	Licinio Mano Marcello Melgara
V1.0.1	22/05/2014	Draft	L. Mano	Corrections after QA	Licinio Mano Marcello Melgara
V1.0.2	30/06/2014	Final	APM	Line numbers out, version number	


	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epsOS OSS NCP Implementation	Date:	30/06/2014	

## TABLE OF CONTENTS

<b>1</b>	<b>Continuous Integration Workflow .....</b>	<b>5</b>
1.1	Macro architecture diagram .....	5
1.2	Integration Testing Component Interaction.....	5
1.3	Fluxogram of the implemented CI solution.....	6
1.3.1	Fluxogram steps .....	7
1.3.2	Used tools.....	8
1.4	Publishing SNAPSHOT versions .....	8
1.4.1	Sample POM File Part .....	9
1.5	Publishing RELEASE versions .....	10
1.6	User notifications .....	10
1.7	Unit Testing.....	10
1.8	Integration Testing .....	10
1.9	Jenkins Reporting.....	11
1.9.1	Test and Deploy .....	11
1.9.2	Static Analysis.....	12
<b>2</b>	<b>Integration of CI Environment With Gazelle Services .....</b>	<b>15</b>
2.1	Integration Scenario.....	15
2.1.1	Diagram Explanation.....	16
<b>3</b>	<b>Extended Integration scenario .....</b>	<b>18</b>
3.1	Extended Diagram Explanation .....	18
3.1.1	Store Request & Response Messages (Java Logs) to Database .....	18
3.1.2	Data-driven testing.....	19
3.1.3	Test Automation .....	19
3.1.4	soapUI vs. JUnit.....	20
<b>4</b>	<b>Messages capture mechanism.....</b>	<b>20</b>
<b>5</b>	<b>Relevant implementation material and samples .....</b>	<b>24</b>
<b>6</b>	<b>Integration points in OpenNCP .....</b>	<b>25</b>
<b>7</b>	<b>Comments.....</b>	<b>27</b>

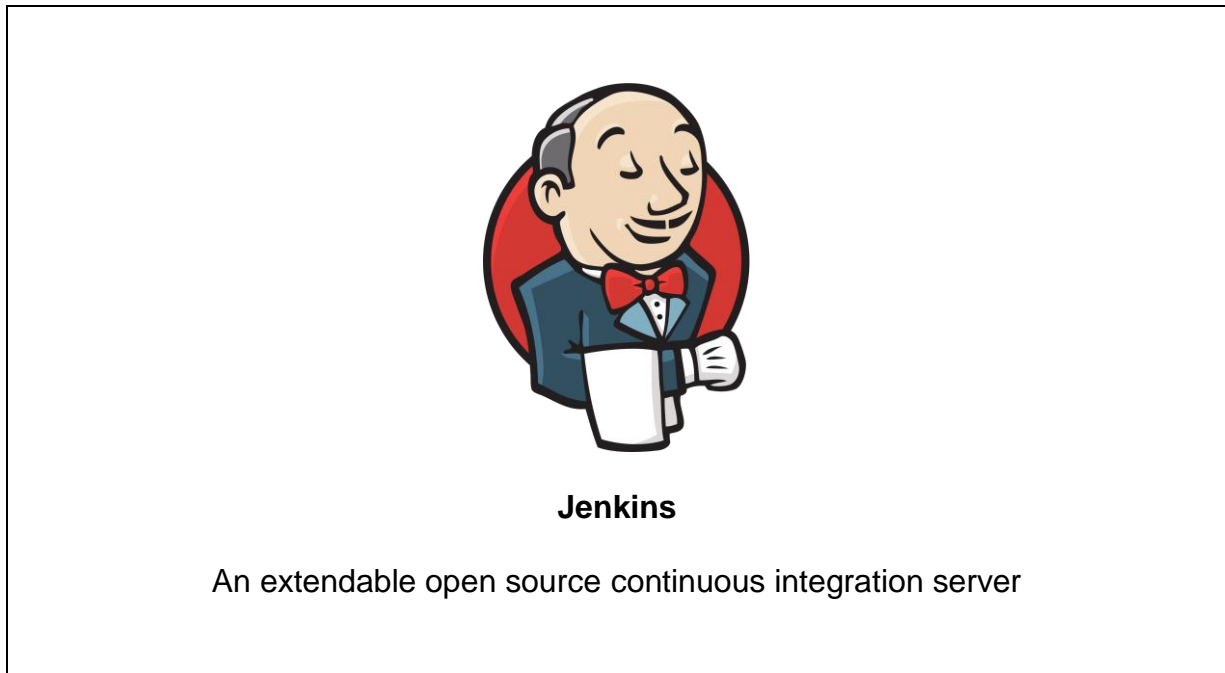
	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short OpenNCP CI & Conformance Test
		Version:	1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014

7.1 Eric Poiseau ..... 27

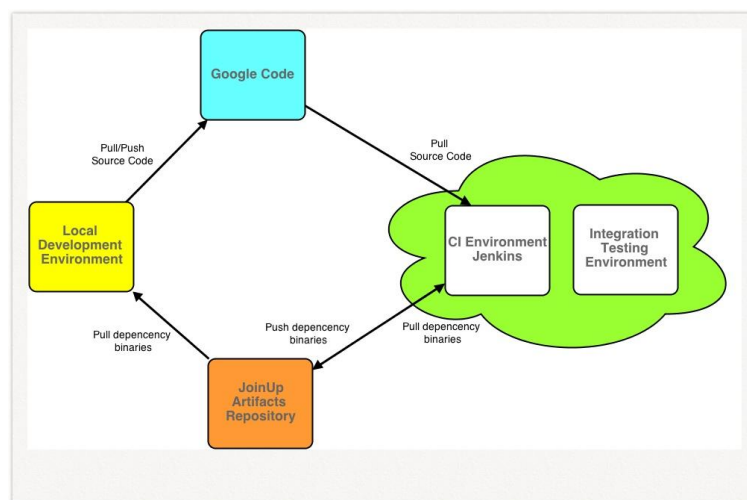
	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name: Short OpenNCP CI & Conformance Test
		Version: 1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date: 30/06/2014

# 1 Continuous Integration Workflow


The main goal of this page is to describe the Continuous Integration configured solution for the OpenNCP project.

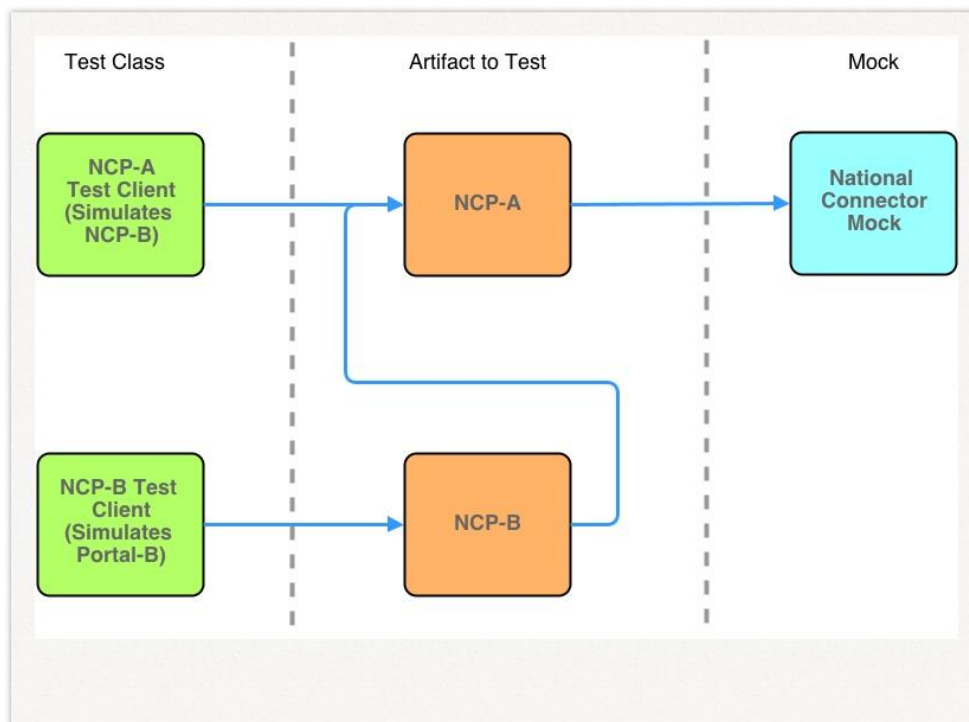


## 1.1 Macro architecture diagram




## 1.2 Integration Testing Component Interaction

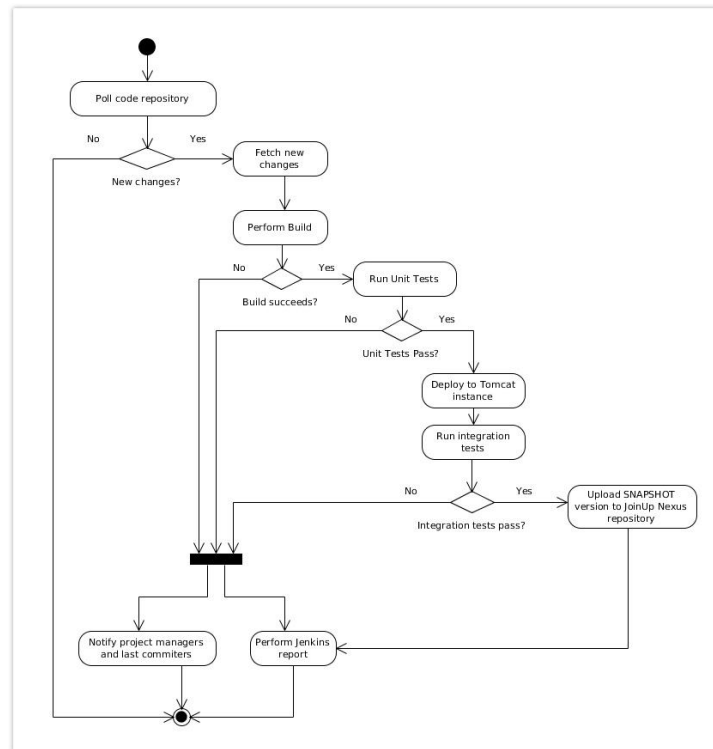
	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short OpenNCP CI & Conformance Test
		Version:	1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014



### 1.3 Fluxogram of the implemented CI solution

The following fluxogram illustrates the current process for testing and deploy artifacts (SNAPSHOT version) to JoinUp.

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short OpenNCP CI & Conformance Test
		Version:	1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014



Next, you will find a brief description of each of the steps of the CI process.

### 1.3.1 Fluxogram steps

#### 1.3.1.1 Poll code repository

This action polls the code repository for new changes and will only trigger the remaining steps if new modifications occur.


It has been set with a polling interval of **5 minutes** for small projects and **30 minutes** for large projects.

Generally the bigger projects (like Protocol Terminators) tend to produce bigger artifacts, that take longer to upload to the JoinUp repository. Also the building process takes longer. That's the reason we have set up a bigger polling time.

#### 1.3.1.2 Fetch new changes

Whenever a polling action detects new changes to the code it will trigger the next step, which will fetch the latest modifications and apply them to the existing code in the Jenkins local repository.

#### 1.3.1.3 Perform build

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014	

Only if all the unit tests are successfully passed the building process takes place, which occurs in this step.

#### 1.3.1.4 Run unit tests

Once the fetch (and merge) process is completed, the unit tests are performed, using a specific maven profile. You can find further information on the [Unit Testing](#) topic ahead on this page.

#### 1.3.1.5 Deploy to Tomcat instance

After a successful build, a specific maven plugin will perform a deploy on the Tomcat testing instance, and perform the next step, related to integration tests.

#### 1.3.1.6 Run integration tests

The integration tests step will look for testing files started with "IT\*", with the aid of advanced maven plugin configuration. This files should include all the integration testing logic.

#### 1.3.1.7 Upload SNAPSHOT version to JoinUp Nexus repository

Once all integration tests are successfully performed and passed, the resulting artifact is published to the JoinUp repository, as a **SNAPSHOT** version.

#### 1.3.1.8 Notify project managers and last committers

If any of the Unit testing, Build and Integration processes fail, a notification will be sent to related users. Further information can be found on [User Notifications](#) topic.

#### 1.3.1.9 Perform Jenkins report


If new changes are found in the first fluxogram step all the remaining steps will end-up in a Jenkins Report. Either failures or successes. More detail about the reporting process can be found in the [Reporting](#) topic.

### 1.3.2 Used tools

- Jenkins;
- Maven;
- Tomcat Instance;

## 1.4 Publishing SNAPSHOT versions



	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014	

The process of publishing SNAPSHOT versions into the JoinUp Nexus repository is completely automated by the Jenkins installation, according to the workflow presented in the previous fluxogram.


The process is automated because it may occur frequently and the publishing only **occurs when all the integration and unit tests pass.**

In order to an artifact be compliant with the JoinUp Maven Repository and to allow the uploading process it must follow a few simple rules, explained bellow by the sample POM file part and the descriptions:

#### 1.4.1 Sample POM File Part

```
<!--...-->
<groupId>eu.europa.ec.joinup.ecc</groupId>
<artifactId>epsos-project-name</artifactId>
<version>1.0-SNAPSHOT</version>
<!--...-->
```

- The GroupId should be "eu.europa.ec.joinup.ecc";
- The ArtifactId should have a prefix of "epsos-";
- The version should be SNAPSHOT;

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014	

### Please Note:

You only need to apply this configuration in the POM files.

All the namespaces and package names in the code **SHOULD** remain the same.

## 1.5 Publishing RELEASE versions

The publishing of RELEASE versions into the JoinUp Nexus repository can only be done at the Nexus web interface and **cannot be done automatically**. This way we make sure that this action is carefully taken by the responsible person and mostly because **it is not intended to happen frequently**.

## 1.6 User notifications

In the automated CI process performed by Jenkins there are a few cases that users need to be notified, mostly when a build and any test fails.

In the current setup the users notified are the **Project Managers** and the last **Committers** (possibly responsible for the faulty scenario).

This way we consider that all the interested parts are notified.

## 1.7 Unit Testing


The Unit testing process is triggered with the aid of a specific testing Maven profile. Whenever a build is performed with this profile all the tests are performed.

This way it's possible to specify when we want or not to run the tests. This is specially useful to the Jenkins configured jobs.

Further information about the tests for each component can be found at the following pages:

- Page: [\*\*XDR Client - Unit Testing\*\*](#) (Protocol Terminators)
- Page: [\*\*XCA Client - Unit Testing\*\*](#) (Protocol Terminators)
- Page: [\*\*XCPD Service - Unit Testing\*\*](#) (Protocol Terminators)
- Page: [\*\*XCPD Client - Unit Testing\*\*](#) (Protocol Terminators)
- Page: [\*\*XCA Service - Unit Testing\*\*](#) (Protocol Terminators)
- Page: [\*\*XDR Service - Unit Testing\*\*](#) (Protocol Terminators)

## 1.8 Integration Testing

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:		30/06/2014

The Integration testing process is also triggered with the aid of a specific testing Maven profile and some plugins. Whenever a build is performed, the solution is deployed to the testing server and all the integration tests are performed.

Further information about the tests for each component can be found at the following pages:

- Page: [XCPD Service - Integration Testing](#) (Protocol Terminators)
- Page: [XCPD Client - Integration Testing](#) (Protocol Terminators)
- Page: [XDR Client - Integration Testing](#) (Protocol Terminators)
- Page: [XDR Service - Integration Testing](#) (Protocol Terminators)
- Page: [XCA Client - Integration Testing](#) (Protocol Terminators)
- Page: [XCA Service - Integration Testing](#) (Protocol Terminators)

## 1.9 Jenkins Reporting


In Jenkins instance you will find several reports of the CI process, mostly in these two areas:

### 1.9.1 Test and Deploy

In the main page you will find three separators for the different categories of the Jenkins jobs, for the "Test and Deploy" the following list is presented:

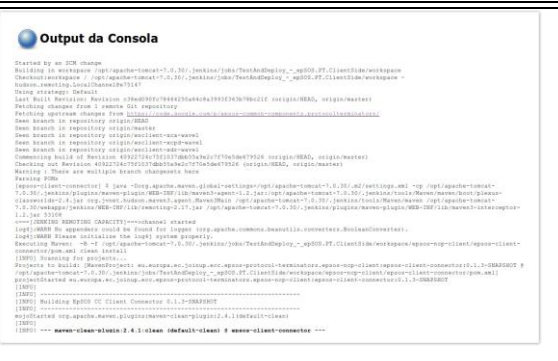
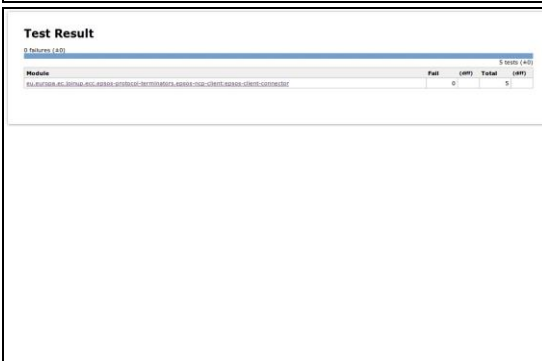
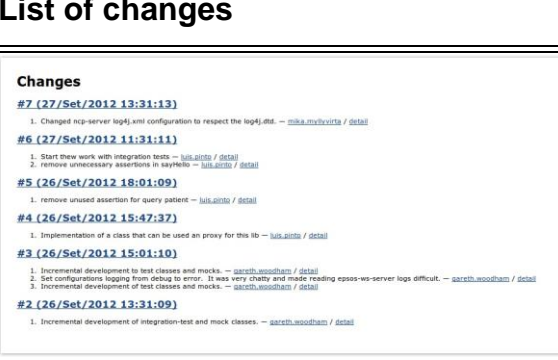
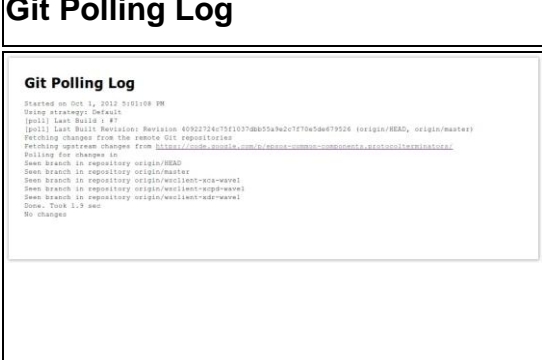
S	W	Name	Último sucesso ↑	Última Falha	Duração da última	
		TestAndDeploy - epSOS_Audit_Manager	2 hr 35 min (#11)	6 days 23 hr (#4)	37 sec	
		TestAndDeploy - epSOS_Uilities	5 hr 45 min (#24)	7 days 0 hr (#15)	30 sec	
		TestAndDeploy - epSOS_SAML_Assertion_Validator	1 day 9 hr (#13)	6 days 22 hr (#4)	36 sec	
		TestAndDeploy - epSOS_TSAM	4 days 1 hr (#6)	N/A	55 sec	
		TestAndDeploy - epSOS_PT_ServerSide	4 days 2 hr (#8)	N/A	15 min	
		TestAndDeploy - epSOS_PT_ClientSide	4 days 2 hr (#7)	N/A	18 min	
		TestAndDeploy - epSOS_PT_ClientConnector.Consumer	4 days 2 hr (#11)	5 days 3 hr (#3)	1 min 5 sec	
		TestAndDeploy - epSOS_Protocol_Terminators	5 days 0 hr (#32)	5 days 6 hr (#25)	23 min	
		TestAndDeploy - epSOS_Security_Manager	5 days 6 hr (#10)	6 days 22 hr (#5)	35 sec	
		TestAndDeploy - epSOS_Transformation_Manager	5 days 6 hr (#11)	6 days 23 hr (#3)	36 sec	
		TestAndDeploy - epSOS_Consent_Manager	5 days 6 hr (#8)	N/A	25 sec	
		TestAndDeploy - epSOS_Config_Manager	5 days 6 hr (#7)	6 days 22 hr (#3)	6 min 0 sec	
		TestAndDeploy - epSOS_Data_Model	5 days 6 hr (#12)	N/A	1 min 34 sec	

On each job you can access, the list of previous builds:

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
	WP3.B: epsOS OSS NCP Implementation	Version:		1.0.12
		Date:		30/06/2014




For each build you can view several data, such as:

<h3>Console Output</h3> 	<h3>Test Result</h3> 
<h3>List of changes</h3> 	<h3>Git Polling Log</h3> 

And many more.

## 1.9.2 Static Analysis

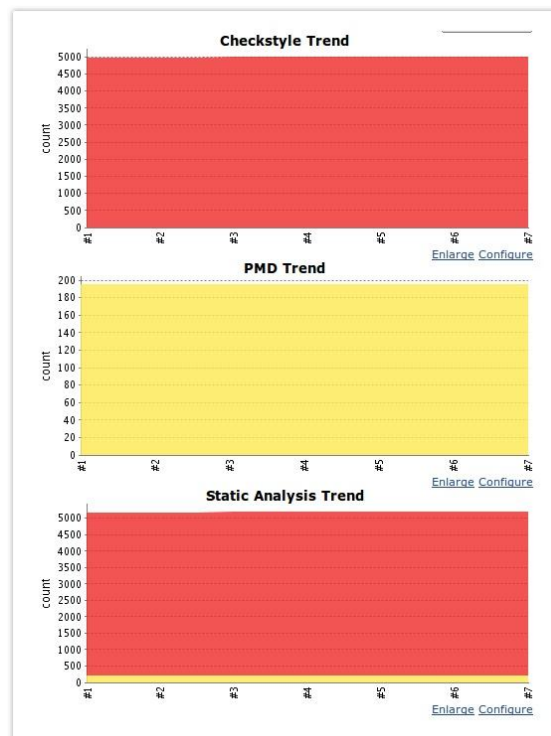
For the static analysis group you will find the following projects:

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short OpenNCP CI & Conformance Test
		Version:	1.0.12
	WP3.B: epsOS OSS NCP Implementation	Date:	30/06/2014

S	W	Name	Último sucesso	Última Falha	Duração da última
		StaticAnalysis - epsOS.Protocol.Terminators	17 hr (#7)	N/A	5 min 5 sec
		StaticAnalysis - epsOS.Data.Model	17 hr (#7)	N/A	1 min 18 sec
		StaticAnalysis - epsOS.Utilities	17 hr (#8)	N/A	2 min 11 sec

As the previous presented group you can also find the history of the previous builds, that in this case, only perform the static analysis of the code and do not compile it.


In the main page of the Static Analysis Job you can find some charts of the trend of the static analysis, shown in the following pictures:

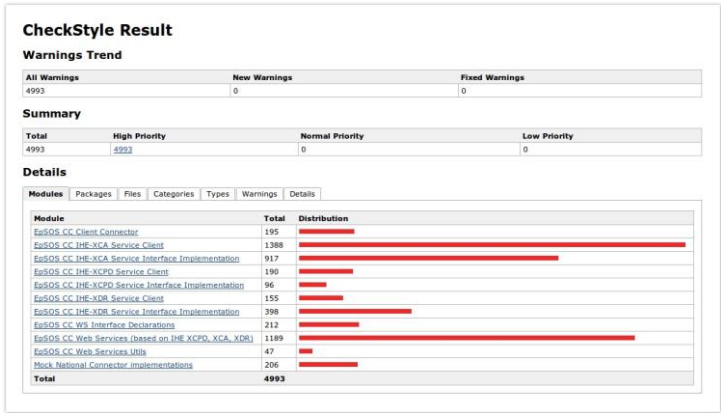


On the left menu you may also find specific trends for each plugin:




On a specific build you also have this menu and you can consult detailed information for each plugin. In the following picture you can see an example for the Checkstyle plugin:

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:		30/06/2014



For each plugin you will get a similar menu with multiple separators that will filter and display the analysis.

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short OpenNCP CI & Conformance Test
		Version:	1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014

## 2 Integration of CI Environment With Gazelle Services

The purpose of this page is to present and discuss a proposal of integration between the **CI environment** (based on Jenkins) and the **Gazelle Services**.


 <h1>Jenkins</h1> <p>An extendable open source continuous integration server</p> <p><a href="http://jenkins-ci.org">http://jenkins-ci.org</a></p>	 <h1>Gazelle</h1> <p>INTEROPERABILITY &amp; CONFORMANCE TESTING FOR E-HEALTH INFORMATION SYSTEMS</p> <p>Gazelle is an open source testbed dedicated to interoperability and conformance testing of eHealth systems</p> <p><a href="http://gazelle.ihe.net">http://gazelle.ihe.net</a></p>
--	--

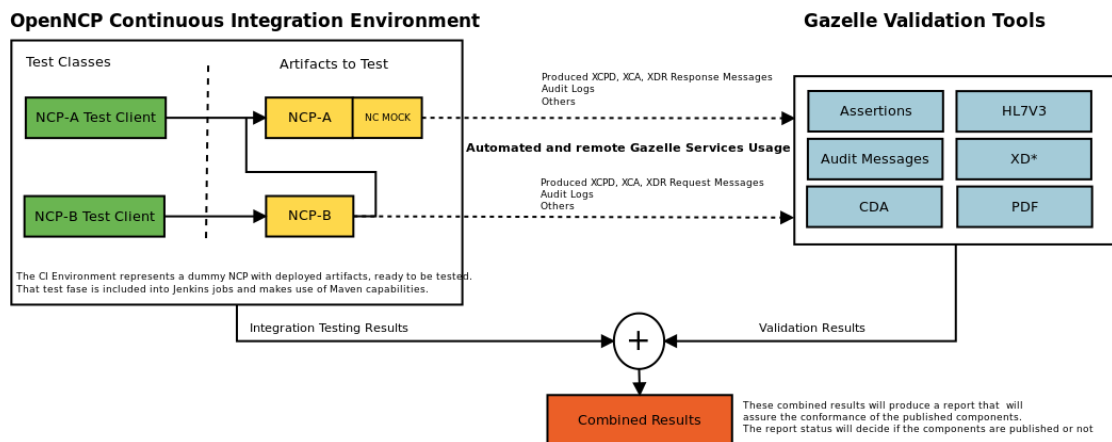
### 2.1 Integration Scenario

As a result of a F2F meeting between the Portuguese epSOS Team and Norbert Répas, representing ELGA, it was produced the first draft of an OSS road map, regarding an improved test strategy for the released components.

This page intends to present a proposed scenario to integrate the OpenNCP development Continuous Integration Environment test cycles with the services provided by IHE platform, Gazelle.

In order to better understand the conceptual model of the proposed solution, the follow diagram is presented:

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:	1.0.12	
	WP3.B: epsSOS OSS NCP Implementation	Date:	30/06/2014	




## 2.1.1 Diagram Explanation

- The current integration testing workflows are **triggered at the Jenkins job executions**, by making use of maven plugins;
- Both NCP-A and NCP-B are **tested using specific test client classes** that send hard-coded soap messages to each of the NCP sides - those soap requests **try to cover faulty situations and normal situations**, and they are based on the available data in the "mocked NI";
- In the implemented scenario we only **test the return of the correct responses and error codes** - we also check for further integration errors, that may result from code bugs and such (Exceptions, and others);
- In the proposed scenario, during the integration test phase, the **produced NCP request and response messages would also be tested**, using the Gazelle Services, **together with the assertions and audit log messages**;
- That validation information would also be **included in the final report of the Jenkins job execution** - this way the publishing of artifacts would be decided by the report overall result;

So, in order to make the described scenario possible, we would need to have the following items:

#	Requirement
1	We need a proxy that is able to <b>capture exchanged messages</b> and call the corresponding validation services;
2	In more detailed manner, we need <b>web services for conformance checking</b> of: <ul style="list-style-type: none"> <li>• XD* transactions;</li> <li>• CDA Documents;</li> <li>• SAML assertions;</li> <li>• Certificates;</li> <li>• Audit messages;</li> </ul>
3	We also need a <b>Demographic Data Generator</b> that can help simulate the test data;
4	Calls to these web services should be possible by a <b>J-Unit &amp; Integration testing program or Java</b>




	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014	

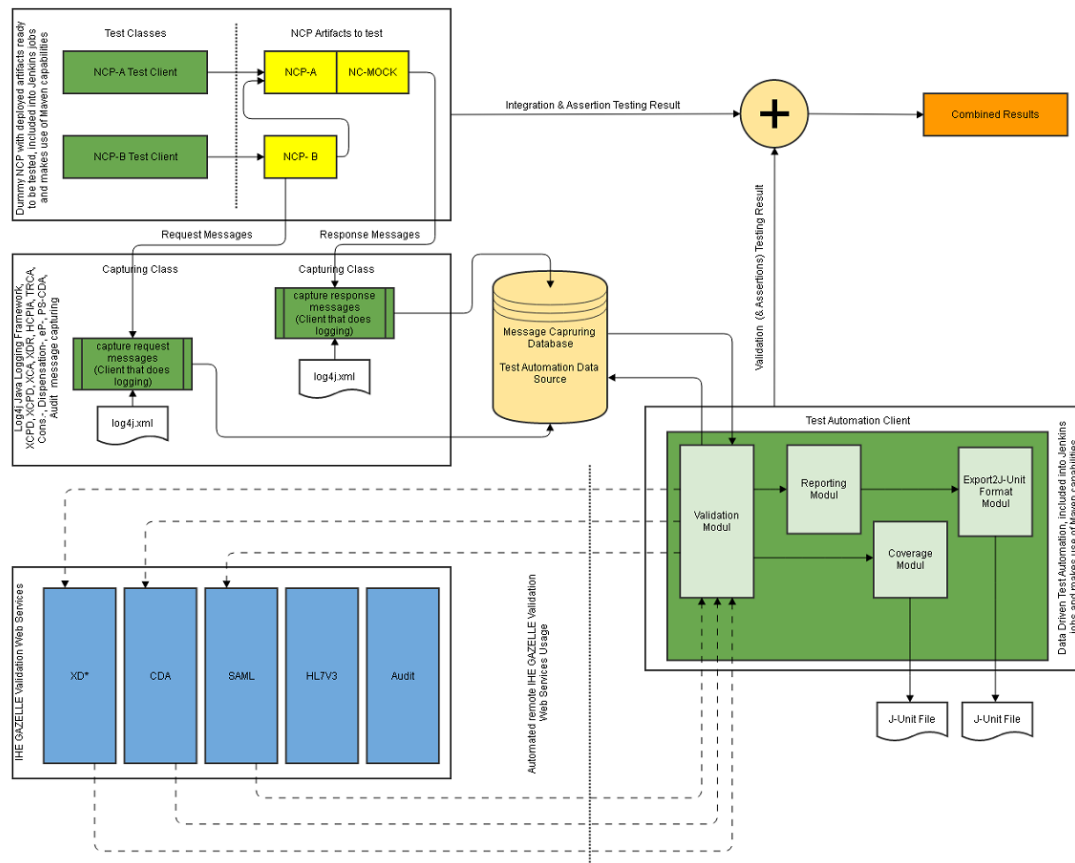
#	Requirement
	<b>Test Classes;</b>
5	Gazelle Simulators should work via <b>embedded API</b> for being managed by an external tool and not the GUI;
6	Some message validators, which we normally use through GUIs, should also <b>be available as Web Services;</b>
7	Regarding automatic capture of messages between two NCPs we need first of all the use of Gazelle Proxy if possible without the combination of the epSOS VPN. This request isn't mandatory. (In Bern - NEPC Meeting (May 2012) & in a spec. technical TConf between Eric P. and Mustafa Y., they discussed that this can be introduced in the next Projectathon (that means Istanbul). if it is really so, it should be to utilize within PPT-slots also)

Probably many of the requested elements are already available, therefore we are showing our **availability and readiness** to start the adjustment of the current workflows in order to integrate with Gazelle Services.

In order to accomplish that, we first request all the existent **documentation about the described Gazelle Services**, together with some information that may be relevant for the task.

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epsOS OSS NCP Implementation	Date:		30/06/2014

### 3 Extended Integration scenario




#### 3.1 Extended Diagram Explanation

##### 3.1.1 Store Request & Response Messages (Java Logs) to Database

There are many different ways how we can utilize it in project, for example via log4j. One of the features of log4j framework is to store Java log messages to a JDBC-enabled database. Before we get into configurations for using log4j we need:

- log4j-X.X.XX.jar - This is the main log4j library, see: <http://logging.apache.org/>
- xxxx\_jdbc.jar - This is the main Message Capturing Database JDBC driver.
- cap\_and\_log.java - A simple Java class that does capturing & logging.
- log4j.properties - In order to enable log4j we will have to create a configuration file. In this configuration we can define three appenders: Console, File and JDBC. This way, the messages will be written to all these appenders. Configuring the JDBC appender requires specifying JDBC URL and also the connection credentials.

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014	

## See also

- log4j home page <http://logging.apache.org/>
- log4jdbc home page <http://code.google.com/p/log4jdbc/>
- SLF4J manual <http://www.slf4j.org/manual.html>
- THE COMPLETE LOG4J MANUAL - Google Books

### 3.1.2 Data-driven testing

Data-driven testing is when we store test data (request, response, etc) in some external storage (database, spreadsheet, xml-files, etc) and then use that data iteratively in our tests when running them. What this boils down to is the following setup:

1. DataSource - reads test data into properties from external source
2. TestSteps (any number) - tests service functionality using the DataSource properties made available in (1)
3. DataSource Loop - loops back to (2) for each row in (1)

Depending on the way how the test cases are implemented there might potentially be a large number of different calls to the underlying Test Automation Data Source. When running test cases or test suites the connection shall preferably only be created once and then be reused whenever needed.


### 3.1.3 Test Automation

There are certainly several options to implement the test automation.

- A possible solution for the implementation of the **test automation client** is the embedding of soapUI into the Java development environment as Plugin.
- soapUI includes also a **Maven 1.x & 2.x plugin for running soapUI tests** and mocks during a maven build. Prior to using the plugin, we need to add the eviware maven 2 repository either to our project or settings.xml.

## See also

- soapUI NetBeans Plugin <http://www.soapui.org/IDE-Plugins/netbean.html>
- SoapUI eclipse-plugin <http://www.soapui.org/IDE-Plugins/eclipse-plugin.html>
- Test Automation via Maven <http://www.soapui.org/Test-Automation/maven-2x.html>
- Getting Started with Reporting <http://www.soapui.org/Reporting/getting-started-with-reporting.html>

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epsOS OSS NCP Implementation	Date:		

- Web Services Coverage & Coverage Visualization  
<http://www.soapui.org/Coverage/introduction.html>

### 3.1.4 soapUI vs. JUnit

Open source soapUI tool has a number of benefits over developing test cases in Java using JUnit framework. Some of these benefits include:

- Use of UI for creating/configuring tests. This makes it possible for non-programmers to develop tests, which means that testing of Web services can be performed by a QA/system test group.
- Auto-generation of test messages based on WSDL.
- Ability to externalize test data using property files.
- Built-in assertions, such as checking for SOAP faults and validity of the schema.
- Built-in stress testing capabilities.
- Management of endpoint URLs.

Overall, soapUI allows developers and testers to create tests quicker and also makes tests easier to manage and update. However, one needs to understand shortcomings of this approach. soapUI uses its own Web services client. The way it creates test messages is different from how regular (JAX-WS-based and other) Web services clients work. soapUI completely skips the step of generating Java classes from WSDL. It does not have to deal with XML marshalling/unmarshalling from/to Java objects. As a result, you are not invoking your service the way it will be invoked by real Web service consumers.


## 4 Messages capture mechanism

The following message capture and store mechanism has already been presented. An can be used to provide reports on the transacted messages or for evaluation use.

In order to easily capture the needed transaction messages from epsos-client-connector, epsos-ws-server, openncp portal you can easily configure a database appender in these components. We describe how to in the following steps:

1. Create a table (i.e. OpenNCP\_LOGS) in one database (i.e. named LOGS) using the following script:

```
CREATE TABLE IF NOT EXISTS `OpenNCP_LOGS` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `component` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `logger` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `priority` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `message` longtext COLLATE utf8_unicode_ci NOT NULL,
```

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epsOS OSS NCP Implementation	Date:		30/06/2014

```

`stacktrace` text COLLATE utf8_unicode_ci,
`creationTime` datetime NOT NULL,
`ip` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
PRIMARY KEY (`id`),
KEY `priority` (`priority`),
KEY `creationTime` (`creationTime`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO INCREMENT=1 ;

```

2. **Configure epsos-client-connector log4j.properties (located under webapps/epsos-client-connector/WEB-INF/classes/log4j.properties). You can capture all XCPC, XCA, XDR messages**

```

# Set root category priority to INFO and its only appender to CONSOLE.
log4j.rootLogger=INFO, Stdout
# Roll-over the log once per day
log4j.appender.Stdout=org.apache.log4j.ConsoleAppender
log4j.appender.Stdout.layout=org.apache.log4j.PatternLayout
# Print the date in ISO 8601 format
log4j.appender.Stdout.layout.conversionPattern=%d [%t] %-5p %c - %m%n
# Define the DB appender
log4j.appender.DB.driver=com.mysql.jdbc.Driver
log4j.appender.DB.eu.epsos.log4j.appender.DatabaseAppender
log4j.appender.DB.URL=jdbc:mysql://localhost:3306/LOGS
log4j.appender.DB.user=<username>
log4j.appender.DB.password=<password>
log4j.appender.DB.sql=INSERT INTO OpenNCP_LOGS (component, creationTime,
logger, \
priority, message, stacktrace, ip) VALUES ('NCP-B', \
?{d{yyyy-MM-dd HH:mm:ss}}, ?{c}, ?{p}, ?{m}, ?{stacktrace}, ?{ip})
log4j.appender.DB.layout=org.apache.log4j.PatternLayout
log4j.appender.DB.layout.ConversionPattern=%-5p [%d{yyyy-MM-dd HH:mm:ss}]
%C{1}: %m (%F:%L, %t)%n

# Specific log Levels
log4j.logger.eu.epsos.pt.cc=DEBUG,DB
log4j.logger.tr.com.srdc.epsos=DEBUG,DB
log4j.logger.epsos.ccd.gnomon.configmanager=DEBUG
log4j.logger.epsos.ccd.gnomon.auditmanager=DEBUG
log4j.logger.epsos.ccd.posam=DEBUG
log4j.logger.org.hibernate=ERROR
log4j.logger.org.apache.axis2=ERROR
log4j.logger.com.mchange=ERROR
log4j.logger.com.spirit.epsos.cc.adc=ERROR
log4j.logger.eu.epsos.pt.eadc=ERROR
log4j.logger.epsos.ccd.posam=ERROR

```


3. **Configure epsos-ws-server log4j.xml (located under webapps/epsos-ws-server/WEB-INF/classes/log4j.xml). You can capture all XCPC, XCA, XDR messages**

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">

<log4j:configuration xmlns:log4j=" debug="false">
  <appender name="LOGFILE"
class="org.apache.log4j.DailyRollingFileAppender">
    <param name="File" value="${catalina.base}/logs/epsos-srdc.log" />
    <param name="DatePattern" value="'. 'yyyy-MM-dd"/>
    <param name="Append" value="true"/>
    <param name="Threshold" value="DEBUG"/>
    <param name="Encoding" value="UTF-8"/>
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%-5p %d %c %M.%L %x -
%m\n"/>
    </layout>

```

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epsOS OSS NCP Implementation	Date:		30/06/2014


</appender>

```
<appender name="DB" class="eu.epsos.log4j.appender.DatabaseAppender">
  <param name="URL" value="jdbc:mysql://localhost/LOGS" />
  <param name="driver" value="com.mysql.jdbc.Driver" />
  <param name="user" value="<username>" />
  <param name="password" value="<password>" />
  <param name="sql"
    value="INSERT INTO OpenNCP_LOGS (component, creationTime,
logger, \
priority, message, stacktrace, ip) VALUES ('NCP-A', \
?{d{yyyy-MM-dd HH:mm:ss}}, ?{c}, ?{p}, ?{m}, ?{stacktrace}, ?{ip})" />
</appender>
```

```
<appender name="console" class="org.apache.log4j.ConsoleAppender">
  <param name="Target" value="System.out"/>
  <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="%-4r %-5p %c %M.%L %x -
%m\n"/>
  </layout>
</appender>
```

```
<logger name="org.apache.axis2">
  <level value="error"/>
</logger>
<logger name="org.apache.axiom">
  <level value="error"/>
</logger>
<logger name="org.apache.xml">
  <level value="error"/>
</logger>
<logger name="org.opensaml">
  <level value="error"/>
</logger>
<logger name="org.springframework">
  <level value="error"/>
</logger>
<logger name="org.hibernate">
  <level value="error"/>
</logger>
<logger name="net.sf.ehcache">
  <level value="error"/>
</logger>
```

```
<logger name="epsos.ccd.posam.tm">
  <level value="info"/>
</logger>
<logger name="epsos.ccd.posam.tsam">
  <level value="info"/>
</logger>
<logger name="de.hunsicker.jalopy.io">
  <level value="fatal"/>
</logger>
<logger name="httpClient.wire.header">
  <level value="fatal"/>
</logger>
<logger name="org.apache.commons.httpClient">
  <level value="fatal"/>
</logger>
<logger name="org.apache.commons.configuration">
  <level value="error"/>
</logger>
<logger name="com.mchange">
  <level value="warn"/>
</logger>
<logger name="_2009.xcpd.iti.ihe">
  <level value="debug"/>
```


	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epsOS OSS NCP Implementation	Date:		

```

    <appender-ref ref="DB"/>
  </logger>
<logger name=" 2007.xds.b.it.ihe">
  <level value="debug"/>
  <appender-ref ref="DB"/>
</logger>

<logger name="eu.epsos.protocolterminators.ws.server.xca.impl">
  <level value="debug"/>
  <appender-ref ref="DB"/>
</logger>
<root>
  <priority value="debug" />
  <appender-ref ref="LOGFILE"/>
</root>
</log4j:configuration>

```

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epsOS OSS NCP Implementation	Date:		30/06/2014

#### 4. Configure openncp portal (located under webapss/epsosportal/WEB-INF/classes/log4j.xml). You can capture:

1. HCPA
2. TRCA
3. Consent Documents (CDA Format)
4. Dispensation Documents (CDA Format)
5. ePrescription Documents (CDA Format)
6. Patient Summary Documents (CDA Format)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration xmlns:log4j="debug="false">
  <appender name="LOGFILE" class="org.apache.log4j.DailyRollingFileAppender">
    <param name="File" value="\${catalina.base}/logs/portal.log" />
    <param name="DatePattern" value="'. 'yyyy-MM-dd"/>
    <param name="Append" value="true"/>
    <param name="Threshold" value="INFO"/>
    <param name="Encoding" value="UTF-8"/>
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%-5p %d %c %M.%L %x - %m\n"/>
    </layout>
  </appender>


  <appender name="DB" class="eu.epsos.log4j.appender.DatabaseAppender">
    <param name="URL" value="jdbc:mysql://localhost/LOGS" />
    <param name="driver" value="com.mysql.jdbc.Driver" />
    <param name="user" value="<username>" />
    <param name="password" value="<password>" />
    <param name="sql"
      value="INSERT INTO OpenNCP_LOGS (component, creationTime, logger, \
priority, message, stacktrace, ip) VALUES ('Portal', \
?{d{yyyy-MM-dd HH:mm:ss}}, ?{c}, ?{p}, ?{m}, ?{stacktrace}, ?{ip})" />
  </appender>
  <appender name="console" class="org.apache.log4j.ConsoleAppender">
    <param name="Target" value="System.out"/>
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%-4r %-5p %c %M.%L %x - %m\n"/>
    </layout>
  </appender>
  <logger name="com.gnomon.epsos.servlet">
    <level value="INFO"/>
    <appender-ref ref="DB"/>
  </logger>
  <logger name="com.gnomon.epsos.service">
    <level value="INFO"/>
    <appender-ref ref="DB"/>
  </logger>
</root>
  <priority value ="debug" />
  <appender-ref ref="LOGFILE"/>
</root>
</log4j:configuration>
```

You should use also different tables for each component to keep separate the log messages. Also you can easily adapt this to any other epsOS component needed.

## 5 Relevant implementation material and samples

File	Modified
------	----------



	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epsOS OSS NCP Implementation	Date:		

File	Modified
ZIP Archive <a href="#">LOG2XML_example.zip</a>	Oct 17, 2013 by <a href="#">Marcelo Fonseca</a>
ZIP Archive <a href="#">LOG2XML_source.zip</a>	Oct 17, 2013 by <a href="#">Marcelo Fonseca</a>

## 6 Integration points in OpenNCP

-----  
XD\* transactions  
-----

Client:

XCPD request  
tr.com.srdc.epsos.ws.xcpd.client.RespondingGateway\_ServiceStub 219

XCPD response  
tr.com.srdc.epsos.ws.xcpd.client.RespondingGateway\_ServiceStub 236

XCA request  
tr.com.srdc.epsos.ws.xca.client.RespondingGateway\_ServiceStub 247

XCA response  
tr.com.srdc.epsos.ws.xca.client.RespondingGateway\_ServiceStub 266

XDR request  
tr.com.srdc.epsos.ws.xdr.client.RespondingGateway\_ServiceStub 229

XDR response  
tr.com.srdc.epsos.ws.xdr.client.DocumentRecipient\_ServiceStub 268

Server:

Incoming XCPD request  
\_2009.xcpd.iti.ihe.XCPD\_ServiceMessageReceiverInOut 104

XCPD response  
\_2009.xcpd.iti.ihe.XCPD\_ServiceMessageReceiverInOut 152

Incoming XCA request  
\_2007.xds\_b.iti.ihe.XCA\_ServiceMessageReceiverInOut 118

XCA response  
\_2007.xds\_b.iti.ihe.XCA\_ServiceMessageReceiverInOut 138

Incoming XDR request  
\_2007.xds\_b.iti.ihe.XDR\_ServiceMessageReceiverInOut 116


XDR response  
\_2007.xds\_b.iti.ihe.XDR\_ServiceMessageReceiverInOut 138

-----  
CDA Documents  
-----

Client:

Pivot XCA Retrieve  
eu.epsos.pt.ws.client.xca.XcaInitGateway 175

Friendly XCA Retrieve  
eu.epsos.pt.ws.client.xca.XcaInitGateway 176

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epsSOS OSS NCP Implementation	Date:		30/06/2014

Friendly XDR Submit  
tr.com.srdc.epsos.ws.xdr.client.XDSbRepositoryServiceInvoker 129

Friendly XDR Submit  
tr.com.srdc.epsos.ws.xdr.client.XDSbRepositoryServiceInvoker 143

Server:

Friendly XCA Retrieve  
tr.com.srdc.epsos.ws.server.xca.impl.XCAServiceImpl 880

Pivot XCA Retrieve  
tr.com.srdc.epsos.ws.server.xca.impl.XCAServiceImpl 929

Pivot XDR Submit  
tr.com.srdc.epsos.ws.server.xdr.XDRServiceImpl 410

Friendly XDR Submit  
tr.com.srdc.epsos.ws.server.xdr.XDRServiceImpl 413


-----  
SAML assertions  
-----

Treatment Relationship Care Assertion (TRC)  
epsos.ccd.netSMART.securitymanager.SamlTRCIssuer 241

Health Care Professional Identity Assertion (HCP)  
(Created by the Web Portal) ?

-----  
Audit messages  
-----

epsos.ccd.gnomon.auditmanager.AuditTrailUtils 139

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:	1.0.12	
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014	

## 7 Comments

### 7.1 Eric Poiseau


#### Requirement 1 :

Currently the Gazelle Proxy (<http://gazelle.ihe.net/proxy>) can be used to capture the exchanged messages. It allows to call Validation services in order to validate the captured messages. As of today it can not capture messages in the context of secure transactions.

#### Requirement 2 :

Gazelle Test Bed offers validation services for the following type of messages :

- XD\* Transactions :
  - GUI : <http://gazelle.ihe.net/EVSCClient/xds/validator.seam?extension=epSOS>
  - WSDL : <http://131.254.209.20:8080/XDStarClient-XDStarClient-ejb/XDSMetadataValidatorWS?wsdl>
- CDA Documents :
  - GUI :
  - WSDL (Model Based) : <http://131.254.209.20:8080/CDAGenerator-CDAGenerator-ejb/CDAValidatorWS?wsdl>
  - WSDL (Schematron Based) : <http://131.254.209.12:8080/SchematronValidator-SchematronValidator-ejb/GazelleObjectValidatorWS?wsdl>
- SAML Assertions :
  - GUI : <http://gazelle.ihe.net/EVSCClient/saml/validator.seam?extension=epSOS>
  - WSDL (Schematron Based) : <http://131.254.209.12:8080/SchematronValidator-SchematronValidator-ejb/GazelleObjectValidatorWS?wsdl>
- Certificates :
  - GUI : <http://gazelle.ihe.net/EVSCClient/tls/validator.seam>
  - WSDL : <http://131.254.209.14:8080/gazelle-pki-gazelle-atna-ejb/CertificateValidator?wsdl>
- Audit Messages :
  - GUI : <http://gazelle.ihe.net/EVSCClient/atna/validator.seam?extension=epSOS>
  - WSDL (Schematron Based) : <http://131.254.209.12:8080/SchematronValidator-SchematronValidator-ejb/GazelleObjectValidatorWS?wsdl>
  - WSDL (Model Based) : <http://131.254.209.20:8080/XDStarClient-XDStarClient-ejb/AuditMessageValidatorWS?wsdl>

	Appendix F: OpenNCP Continuous Integration & Conformance Test design and engineering details	Document name:	Short	OpenNCP CI & Conformance Test
		Version:		1.0.12
	WP3.B: epSOS OSS NCP Implementation	Date:	30/06/2014	

### Requirement 3 :

The demographic data generator exists in the Gazelle Test Bed. We call it DDS : Demographic Data Server

- GUI : <http://gazelle.ihe.net/DDS/home.seam>
- WSDL : <http://gazelle.ihe.net/DDSWS/DemographicDataServerBean?wsdl>
- Documentation : <http://gazelle.ihe.net/content/dds-user-manual>

### Requirement 4 :

It is possible to call this web services from J-Unit or form Java Test Classes. The Proxy, the simulators and Test Management make extensive use of calls to this web services using java classes

### Requirement 5:

We need to interact in order to identify the specific needs in that API ! We already have an API that Gazelle Test Management Uses to interact with the simulator. We are not using it in production now, but it can be used to automate the interactions with SUT.

### Requirement 6 :

See requirement 2. IMHO this is the same requirement.

### Requirement 7 :

As specified in the comments to Requirement 1, the proxy is not yet able to handle TLS transactions.