




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

Work Package 3.9

D3.9.1: Appendix A FET Solution


WORK PACKAGE	WP3.9, JWG 3.8/3.9
DOCUMENT NAME	D3.9.1: Appendix A FET Solution
SHORT NAME	D3.9.1 Appendix A
DOCUMENT VERSION	1.0
DATE	01/10/2010

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

COVER AND CONTROL PAGE OF DOCUMENT	
Document name:	D3.9.1: Appendix A FET Solution
Document Short name:	D3.9.1 Appendix A
Distribution level	PU
Status	Final
Author(s): Organization:	B.Horvat, S.Evdokimov, R.Melgaard, P.Loubjerg, S.Bittins, J.Caumanns, A.Estelrich, P.Ruestchmann, A.Périé, G.De Bejarry, A.Hansen, M.Kovarova, G:Heider, P.Gross, T.Pass, G.Cangioli, S.Lotti, M.Melgara, E. Albertini, G.Orsi JWG-Technical Core Team and WP3.9

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
<p>“D3.9.1: Appendix A FET Solution” provides the information concerning the software component development activities performed by the Proof of Concept implementation streamline of WP3.9, in co-operation with WP3.8 in the Joint Working Group WP3.8 / 3.9.</p> <p>Strategic activities like implementation strategy definition and implementation proposal selections were performed in co-operation with TPM task force and PMT.</p> <p>This document is heavily based on the High Level Design document, epSOS Internal Deliverable: “Architecture of the National Contact Point in a Box (NCP-in-a-Transparent-Box)”, developed by the Joint Activity Group (JWC) 3.8 / 3.9.</p> <p>The document is addressed to the MS and to the industries who wants to develop the NCP, the Front-end Portal. It represent a bridge among the Specifications of WP3.3 – 3.7, the Guidelines of WP3.8, the testing procedures defined in WP3.9 and applied in WP3.10, and the MS implementation of PD4.</p> <p>After a summary of WP3.9 organisational and methodological approaches, the NCP implementation guidelines, provided by PEB/PSB are introduced.</p> <p>Basic requirements for NCP implementation are analysed. High Level Design of the NCP-in-a-Box concept, both considered as gateway toward Country A and toward Country B, is provided. It contains general description of the NCP-in-a-Box architecture as well as details about individual software components and Country B Front-end.</p>

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

Software Component specifications are provided in the document, while Detailed Technical Specifications of some of them are reported as Appendixes.

A specific chapter is devoted to Semantic Group activity description, including the definition of Central Services for Terminology Management and processes to develop and maintain Master Value Sets and Translation/Transcoding Catalogues.

Implementation strategies and scenarios are analysed, describing the procurement procedures defined and actuated by epSOS.

The Proof of Concept NCP implementation, assigned Fraunhofer ISST / Elga and IT consortium are finally described.

Change History					
Version	Date	Status Changes	From	Details	Review
V0.47	02/08/2010	Draft	Digital Health	Initial Structure based on full D3.9.1 v0.46 document, split appendix A into separate document	JWG-TC
V0.48	06/08/2010	Draft	Digital Health	Minor Editorial changes & links to ProjectPlace	JWG-TC
V0.49	09/08/2010	Draft	Lombardy	Re-formatted by Marcello	JWC-TC
V0.50	24/09/2010	Draft	Fraunhofer ISST	Revised the full document to align to current release status	
V0.51	24/09/2010	Draft	Lombardy	Revised for the final document release	
V0.90	24/09/2010	Draft	Lombardy	Version for PEB approval	
V1.0	01/10/2010	Final	Lombardy	Version approved by PEB	




	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

TABLE OF CONTENTS


Preface	7
1 Architecture Overview	7
1.1 NCP-A Architectural Overview	8
1.2 NCP-B Architectural Overview	9
1.3 National Connectivity Options Overview	10
1.4 General NCP Workflow, Interfaces, and Interactions	11
1.5 Exemplary Demonstrator / Deployment Scenario	11
2 Common Components General Requirements.....	16
3 Common Components Documentation	17
3.1 The Common Components Repository - SVN	17
3.1.1 Accessing the SVN	17
3.1.2 Requesting a SVN Account.....	17
3.1.3 Tools for SVN Access.....	17
3.2 Backbone Components – Documentation Access.....	18
3.2.1 Documentation Access	18
3.2.2 National Connector/Interface (SpiritEHRWSCClient)	18
3.2.3 NCP-A-centred documentation	18
3.2.4 NCP-B-centred documentation	19
3.3 Testing Facilitators and Provisions.....	19
4 Security Manager	20
4.1 Components Overview	20
4.2 NCP Signature Manager.....	20
4.3 NCP Certificate Validator	20
4.4 TRC Assertion Issuer.....	21
4.5 TRC STS.....	21
5 Configuration Manager	22
5.1 NCP Configuration Manager.....	22
5.2 NCP Routing Manager	23
5.3 NCP TSL Synchronizer.....	23

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

5.4	TSL Editor	24
5.5	Configuration Attributes of the Configuration Manager.....	24
6	Audit Manager	25
6.1	Audit Writer	25
6.2	Syslog Client	26
6.3	Audit Repository.....	26
6.4	Syslog-Interface	27
6.5	Information Security Safeguards	27
6.6	Export Operation	28
6.7	Export for Reporting	29
6.8	Monitoring Manager	29
6.9	Partner Contributions / Responsibilities	30
7	Policy Manager.....	31
7.1	Components Overview	31
7.2	Attribute Service.....	31
7.3	Role Configuration.....	32
7.4	PoC Registration Data.....	34
7.5	Authz Query Creator.....	35
7.6	Policy Decision Point (PDP)	35
8	Transformation Manager.....	37
8.1	Operations.....	37
8.1.1	Component behaviour (called operations).....	39
8.2	Functional requirements	39
8.2.1	Document transformation.....	39
8.3	Non-functional requirements.....	40
8.3.1	Specific requirements for TM	40
8.4	epSOS Transcoder.....	41
9	TSAM.....	42
9.1	Components Overview	42
9.1.1	<i>Database Schema</i>	42
9.1.2	<i>ValueSet Retriever</i>	42
9.1.3	<i>TSAM Synchronizer</i>	43

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

9.1.4	<i>Taxonomy Mapping Service</i>	43
9.2	Component responsibilities.....	44
9.3	Operations.....	44
9.4	Functional requirements for TSAM	46
9.4.1	Transcoding/Translation functionality	46
9.4.2	Local Terminology Repository	46
9.5	Non-functional requirements for TSAM.....	46
9.5.1	Specific requirements for TSAM.....	46
10	National Interface/Connector.....	48
10.1	National infrastructure to NCP-B	48
10.1.1	An IHE compliant interface / infrastructure to NCP-B	48
10.1.2	A proprietary infrastructure to NCP-B.....	49
10.2	NCP-A to a national infrastructure	49
10.2.1	NCP-A to a IHE compliant interface / infrastructure.....	50
10.2.2	NCP-A to a NON - IHE compliant infrastructure	50
11	Front-end Portal	52
11.1	View of Patient Summary	52
11.2	View of ePrescription.....	53
11.3	Consent in Country A for several Country B.....	53
11.4	Web services.....	53
12	Change Proposal and HLDD-Delta in CCD	55
12.1	Change Proposals Listing.....	55
12.2	Major Change Proposal Compilation	58
12.2.1	Backbone Components.....	58
12.2.2	Policy Manager	59
12.2.3	Audit Manager	59
12.2.4	Security Manager.....	59
12.2.5	<i>Transformation Manager and TSAM</i>	59

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

Preface

This appendix describes the general framework of the NCP-in-a-Box reference implementation, including:

- the general architecture of the NCP-in-a-Box
- generalized work- and information flows
- an exemplary deployment scenario
- a scoping and brief overview of each common component

A detailed description of the common components is provided at ProjectPlace in the virtual folder: “**epSOS Common Component Development**”

ProjectPlace link: <https://service.projectplace.com/pp/pp.cgi/0/499293774>

Additional documentation, including an extensive collection of developer-centred information and concrete software components, is available at:


- the FhGISST SVN (see 3.1, this document)
- the TIANI Wiki (see 3.2, this document)
- the individual change proposals (see 12, this document)

The information provided in this annex and the referred information sources however is not sufficient in order to fully comprehend the epSOS reference implementation. Extensive knowledge of any of the epSOS PD3 specification is of crucial importance.

1 Architecture Overview

The following sections illustrate the general architecture of the NCP-in-a-Box reference implementation. Open Source common components are presented as yellow entities, while the closed-source backbone components feature a purple shading. The concrete services and their distinctive functionalities are organisationally grouped into more general entities: the common components. Each of the NCP's components provides a set of functionalities and is designed to be as reusable as possible by default. In order to facilitate complex deployment scenarios as potentially required by some Member States, the components also follow the separation-of-concern and –duty principle by a strict encapsulation of their assigned tasks. As a result, Member States may take liberty of replacing individual components with their own existing services for a seamless integration of epSOS into the national infrastructure.

The following subsections illustrate the current architecture of the NCP-in-a-Box.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

NCP-A Architectural Overview

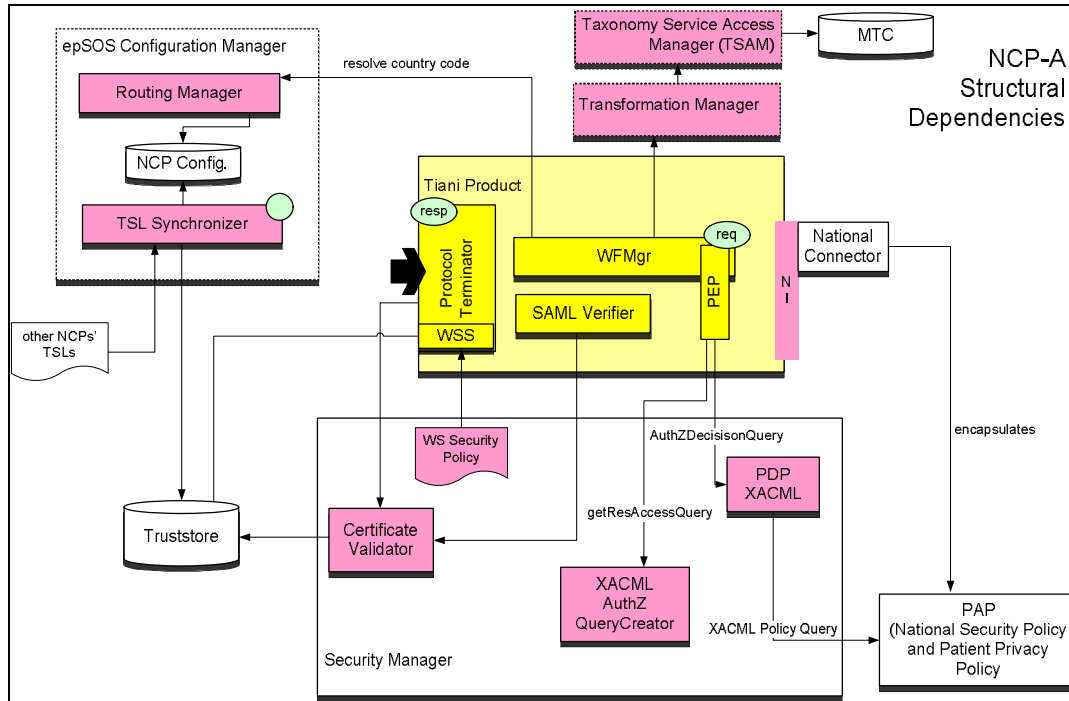
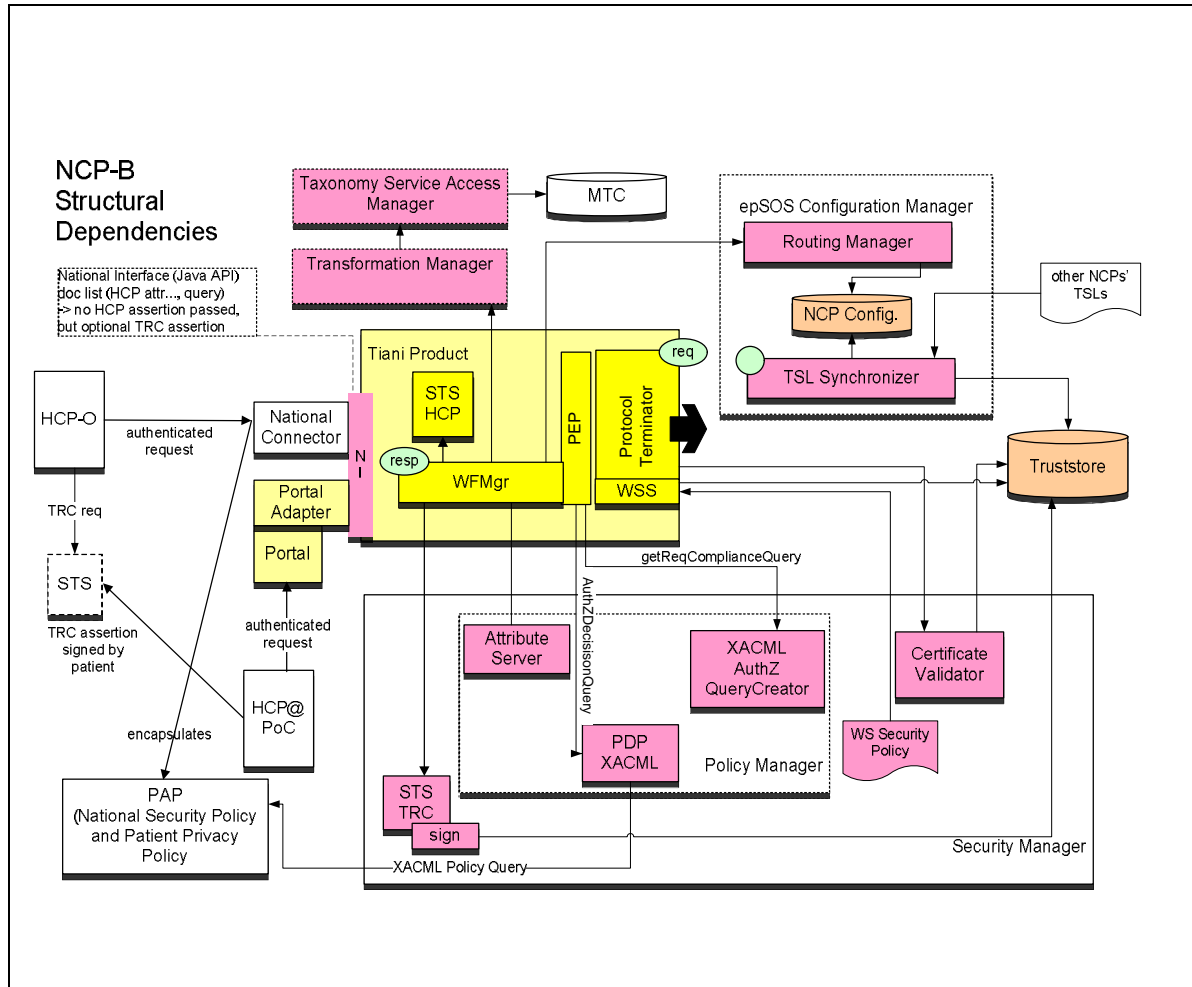



Figure 1 NCP-A

NCP-B Architectural Overview



	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

National Connectivity Options Overview

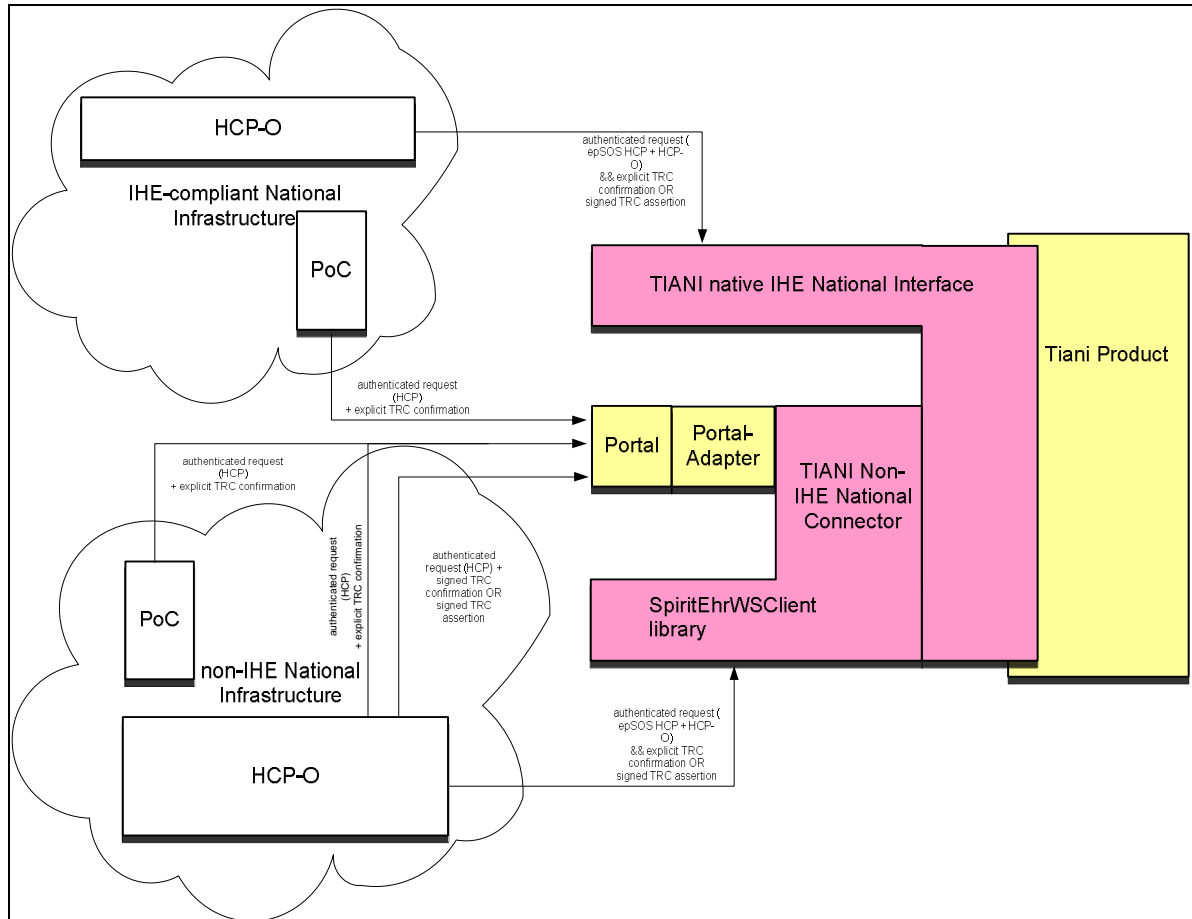

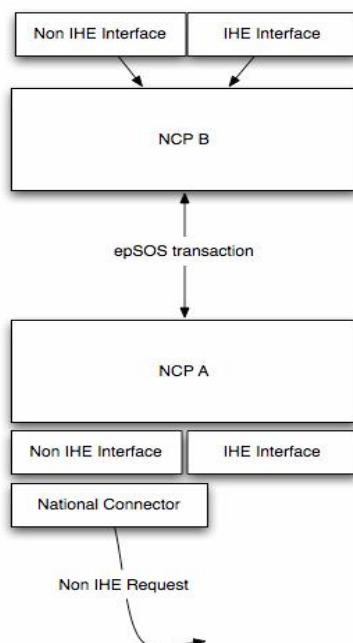


Figure 3 TIANI/epSOS Connector

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

General NCP Workflow, Interfaces, and Interactions



Exemplary Demonstrator / Deployment Scenario

During the first official presentation of the epSOS reference implementation, a demonstrator was compiled and presented at the CCD workshop-I in Vienna. Additionally to the demonstration of the epSOS services and use-cases, this demonstrator also featured an exemplary deployment option of the NCP-in-a-Box.


The demonstrator was built on the following set of baseline assumptions:

- Country-A is Greece (using a simulated shadow national infrastructure)
- Country-B is Austria (using a pre-configured epSOS- portal-B)
- demonstration both, a Non-IHE compliant national infrastructure and an IHE compliant national infrastructure
- operating all epSOS-specific components on a dedicated CISCO system
- utilising existing Austrian building-blocks as backbone components

Each demonstrator node features and/or provides:

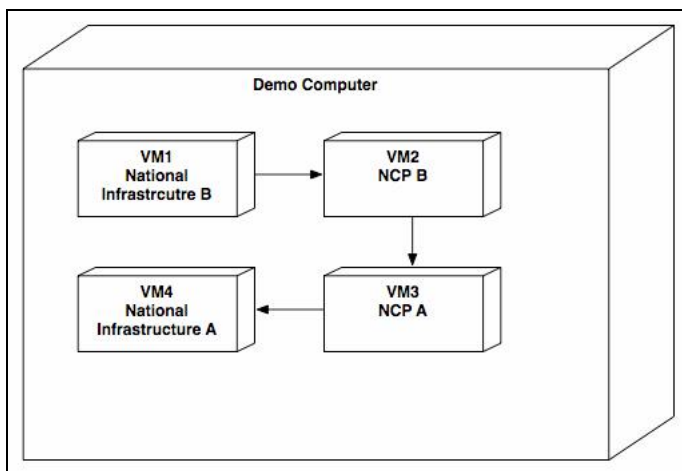
- IHE ATNA Secure Node capabilities, with no file-system access
- a complete separation of all virtual machines
- all virtual machines communicate by web-services with each other
- exclusive communication by either the epSOS NCP2NCP interface or the National Interface connectivity

The demonstrator incorporated two distinctive use-cases:

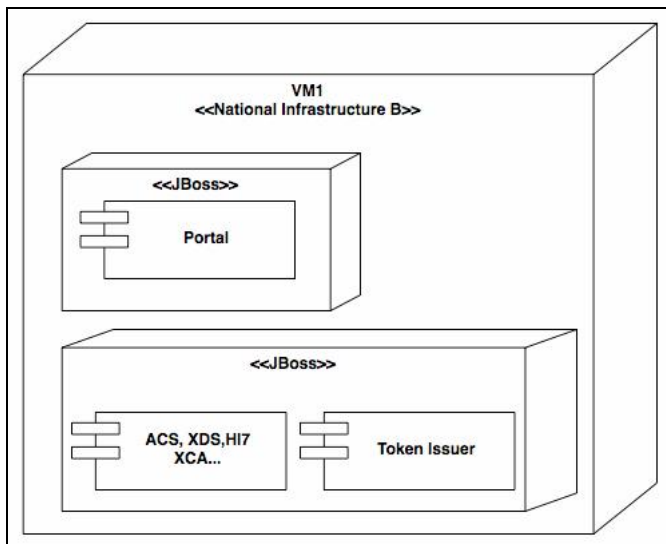
	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

1. “A healthcare provider located in Austria needs to perform a set of epSOS transactions to retrieve the documents for a patient originating from Greece seeking treatment”
2. “The HCP performs a set of local queries to the HIS/CIS and then performs a generic epSOS query (identifying a patient and retrieving the documents)”


The epSOS demonstrator setup incorporated four individual systems deployed in one physical machine as virtual machines.

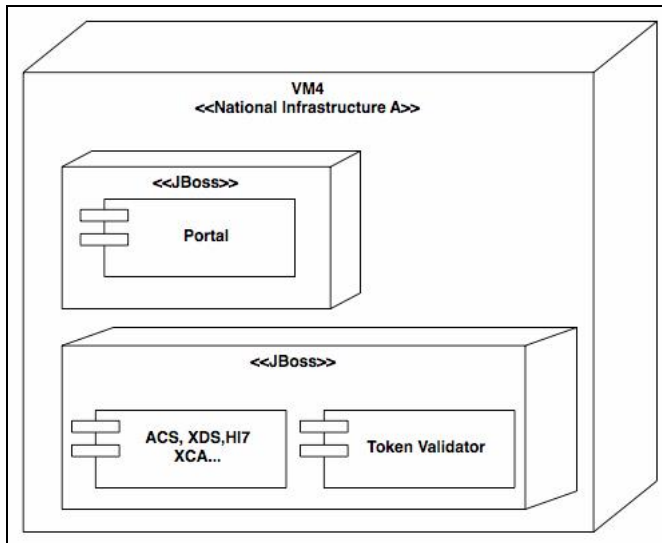


The simulated national infrastructure of Country-B was deployed as follows:

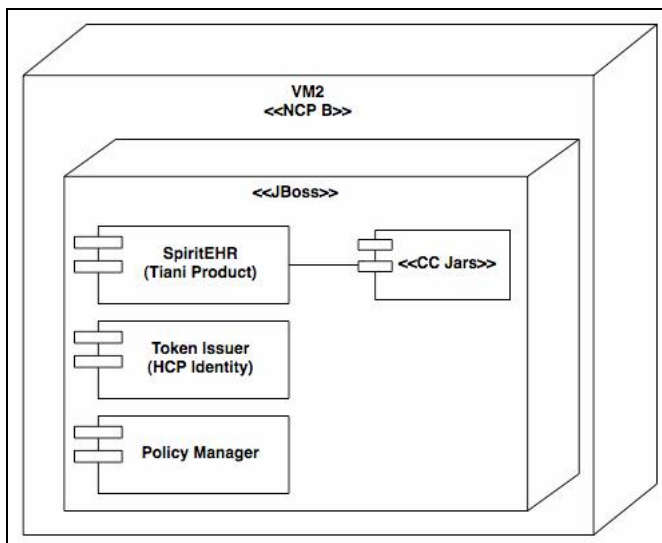


The national infrastructure of country-A was deployed likewise:


	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

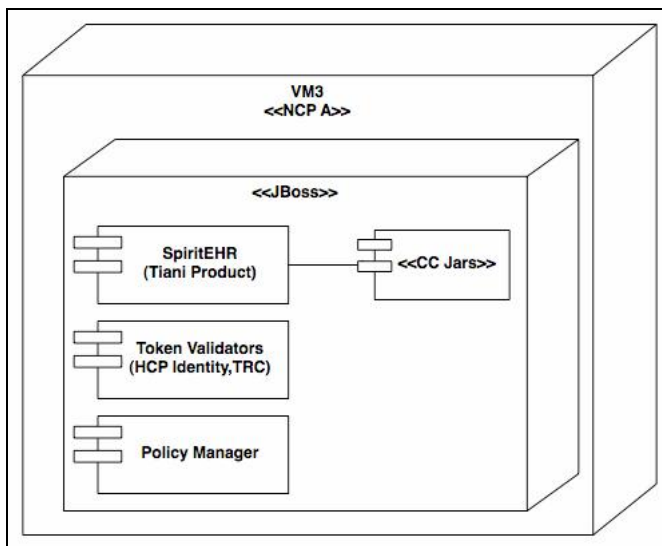


The NCP-B, including a detached Identity Provider for authentication of the HCP's and the policy manager, was assigned and deployed:

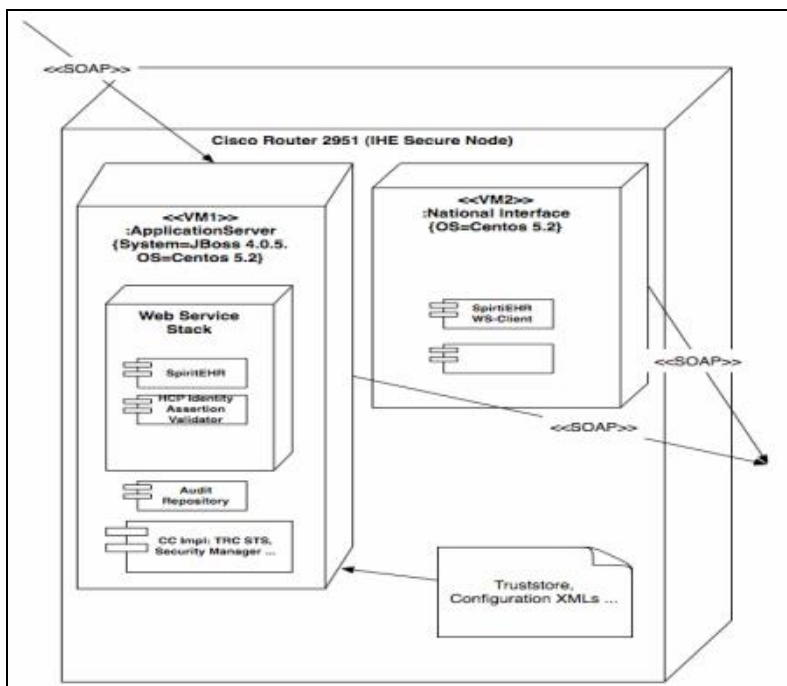


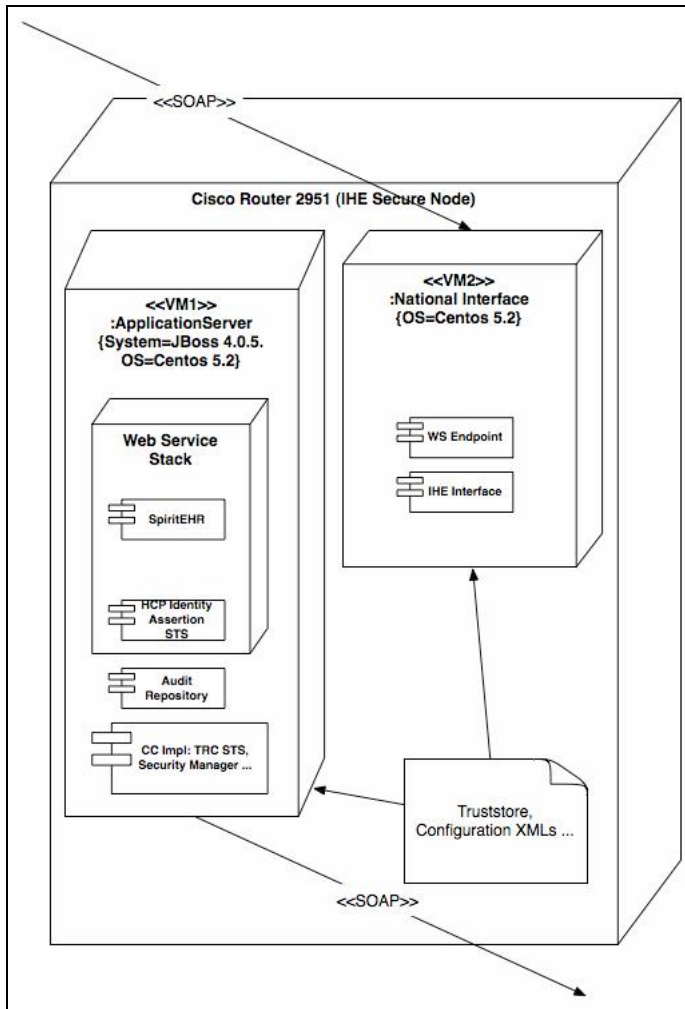
Likewise was the NCP-A of the demonstrator:


	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010



NCP-A and NCP-B more detailed (with valid information-/control-flow indicated by the arrows):





	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

2 Common Components General Requirements

All common components are provided as open source and free of licensing fees. Libraries and frameworks used for component development SHOULD be open source but MUST NOT be published under a copy-left license. Before a contract is signed with a member of the epSOS IT for common component development the vendor MUST provide FHGISST/ELGA with a list of all frameworks and libraries that will be used for component development and that are required for compiling and operating the component.


- The following requirements MUST be met by all developed common components:
- Programming language MUST be Java, exceptions can be made for GUI tools and batch tools.
- All components MUST be thread-safe.

Test data MUST be assembled for each interface. Components MUST be tested by vendors. FHGISST/ELGA will test each component before approval.

A full documentation MUST be provided in English language. The documentation MUST provide clear guidance for installing and integrating the component with a typical environment.

Javadoc MUST be used for external and internal interface description.

Bugs (faults in the interfaces, wrong output to correct input and unwanted runtime behavior) MUST be fixed by the vendor without any additional charge until the end of the first pilot phase.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

3 Common Components Documentation

The Common Components Repository - SVN

This document describes how to access the epSOS CCD project versioning control system. The versioning control system is a Subversion (SVN). In order to facilitate the distribution of the CC, FhGISST has set up a component repository in which all Open-Source common components are stored and made accessible for any epSOS party.

The FhGISST SVN contains the software components itself, the developer documentation (javadoc), and the environmental requirements for necessary for operating the respective components, such as configuration provisions, library and framework dependencies, and deployment options.

Accessing the SVN

1. Request a epSOS CCD SVN account and wait for account confirmation mail
2. URL SVN: <http://svnberlin.isst.fraunhofer.de/subversion/epsosCCD>
3. Use a web browser to read-only access or a SVN client (see below)

Requesting a SVN Account

To access the SVN, authentication and authorization is needed. No anonymous account is provided for reasons of data security. To request a SVN account, please send a request mail to:


hannes.restel@isst.fraunhofer.de

The real name and the company name must be given. FhGISST will create an account with username (following the pattern "<first letter of first name><family name>") and a password. Both will be sent to you by encrypted email.

Tools for SVN Access

In this section, a selection of tools is presented for accessing the SVN.

- Web browser [all OS]: Use a web browser to gain read-only access to SVN - Subversion command shell client [all OS]: Command shell client (<http://subversion.apache.org/packages.html>)
- Eclipse Tigris Subclipse [all OS]: SVN plugin for Eclipse (http://subclipse.tigris.org/update_1.6.x)
- Tortoise SVN [MS Windows only]: Client that integrates into Windows Explorer (<http://tortoisesvn.net/downloads>)
- Versions [Mac OS X only]: SVN GUI client (<http://versionsapp.com/>)

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

Backbone Components – Documentation Access

The CCD team reference implementation of the NCP-in-a-Box builds on a set of industry components for infrastructure services. Those components are provided as closed-source, which means that no source code is provided but the interface and transaction specifications.

The respective national infrastructure connectivity may be achieved by either integrating a small client that manages the communication with epSOS or by deploying the NCP-in-a-Box as a “pass-through” service. In the first option, the NCP-in-a-Box is performing all epSOS-specific tasks, while the latter option relays the original epSOS communication to the national infrastructure that subsequently deals with all further processing exclusively.

Documentation Access

All backbone components are documented in a Wiki provided by the CCD team partner Tiani. The Wiki page is access restricted, using “epsos” as username and “epsos” as password. The backbone components documentation overview page is located at:

<http://wiki.tiani-spirit.com/>

National Connector/Interface (SpiritEHRWSCClient)

Member States that choose the integration option are invited to navigate to the NCP-in-a-Box overview page, featuring:

- the full documentation of the National Connector/Interface using the client provided by Tiani
- a comprehensive set of code examples and code snippets to demonstrate major tasks and functionality
- a step-by-step How-To on how to connect your national infrastructure to the epSOS NCP-in-a-Box reference implementation
- development hints and best practices that have been identified by other epSOS partners connecting their national infrastructure


The SpiritEHRWSCClient How-To and overview page is located at:

<http://wiki.tiani-spirit.com/display/javadocspace/How+to+epSOS+with+SpiritEhrWS>

NCP-A-centred documentation

In order to provide the NCP-A functionality and required connectivity, a dedicated page has been set:

<http://wiki.tiani-spirit.com/display/javadocspace/Using+SpiritEhrWsServer>

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

This documentation page features:

- all epSOS-specific documentation and information required to connect your national infrastructure to the NCP-A interfaces of the NCP-in-a-Box
- a comprehensive set of code examples and snippets to demonstrate and test the connection to the NCP-A side (Example Implementation of the Interface)
- a test class (ServerTest) within the module SpiritEHRWSServerTestClient that enables Member States to test the local NCP-A implementation

NCP-B-centred documentation

In order to provide the NCP-B functionality and required connectivity, a dedicated page has been set:

<http://wiki.tiani-spirit.com/display/javadocspace/Using+SpiritEhrWSClient+%28NationalInterface%29>

This documentation page features:


- all epSOS-specific documentation and information required to connect your national infrastructure to the NCP-B interfaces of the NCP-in-a-Box
- a class that covers the full capabilities of the SpiritEHRWSClient including nonepSOS-specific aspects for a potential pass-through scenario
- a comprehensive set of code examples and snippets to demonstrate and test the connection to the NCP-B side (NationalInterfaceTest)
- a hands-on demonstrator implementation (as shown in the workshop in Vienna) that demonstrates the connection to the NCP-B-side and the fulfilment of a selection of epSOS tasks and services (SWSClientDemo)

Testing Facilitators and Provisions

Each connectivity option for epSOS (NCP-A, NCP-B) is fully documented with live code examples, exemplary code snippets, and demonstrators. Additionally to those provisions, certain test classes are also provided to enable Member States to test their respective implementation.

Since the NCP-A-side implementation is of particular significance and critical towards security and safety of the processed data, a dedicated test class is provided to locally test the national implementation of the NCP-A-side connection.

The class “ServerTest” within the module “SpiritEHRWSServerTestClient” enables Member States to perform critical testing at their own discretion and scheduling without being required to have a full connection to the epSOS network already set up. The test class features all epSOS-specific work flows and functionalities.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

4 Security Manager

Components Overview

The epSOS Security Manager consists of 4 parts:

- **NCP Signature Manager:** operations for applying and verifying digital signatures.
- **NCP Certificate Validator:** operation for verifying the validity of a certificate or a key
- **TRC Assertion Issuer:** operation for issuing TRC assertions acc. D3.4.2.
- **TRC STS:** WS Trust RST/RSTR wrapper for the TRC assertion issuer.

NCP Signature Manager

The NCP Signature Manager is a JAVA library for applying and verifying detached digital signatures on XML documents and for applying and verifying enveloped signatures on SAML assertions. At least the following methods **MUST** be provided:

- XMLDSig signData(xmlData, keyID)
- Boolean verifyDetachedSignature(xmlData, xmlDSig, keyTypeID)
- SamlAssertion signAssertion(samlAssertion, keyID)
- Boolean verifySamlSignature(samlAssertion)


The NCP Signature Manager **MUST** support all normalization, digesting, transformation and signing algorithms that are referenced in the epSOS D3.3.2/3, D3.4.2 and D3.7.2 documents. Which algorithms are to be used for signing **MUST** be read from the NCP configuration.

The verification of a signature includes the following assessments:

- recalculation of the hash
- validation of the validity of the signature key (certificate is neither outdated nor rejected); see NCP certificate validator below.
- Validation that the certificate of the signer belongs to the requested class of certificates (e.g. NCP)
- Verification of the certificate attributes acc. to the certificate profiles as defined in D3.4.2

NCP Certificate Validator

The NCP Certificate Validator verifies the validity of a certificate by a given KEYINFO structure:

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

- Boolean verifyCertificate(KEYINFO as DOM element)
- The certificate validation consists of checking if the certificate is trusted and if it is not revoked:
- Trust checks are performed against the TrustStore (see Configuration manager). All certificates registered with the local NCP trust store are assumed as trusted.
- For revocation check both CRL retrieval and OCSP MUST be supported. The respective addresses are provided with all epSOS certificates.

TRC Assertion Issuer

The TRC Assertion issuer is a JAVA class (subcomponent if the security manager) that issues Treatment Relationship Assertions as specified in D3.4.2. It makes use of the signature manager for signing the assertions. An audit trail entry is written after the successful issuance of a TRC assertion.

The TRC Assertion Issuer MUST at least provide the following methods:

- SAMLAssertion issueTrcToken(hcplIdentityAssertion, patientID, purposeOfUse[], optional attrValuePair[])
 - Boolean verifyTrcToken(trcAssertion, hcplIdentityAssertion, patientID)

The verification of a TRC assertion includes:

- Verification of the signature (using the repective functionality of the signature manager)
- Verification of conformance to the D3.4.2 TRC assertion spec
- Verification of the validity of the assertion (e.g. time valid)


For the processing of SAML assertions an established framework such as SUN WSIT or OpenSAML MUST be used.

TRC STS

The TRC STS encapsulates the TRC Assertion Issuer as a WS Trust security token service by providing a WS Trust RST/RSTR interface to the issueTrcToken method. The HCP Identity Assertion is provided within the security header of the RST message. Patient ID and "purpose of use" are given as arguments using the WS Trust 1.3 extension mechanism.

The objective of this additional interface is to allow alternative NCP deployments that connect to the TRC STS as an external service (e.g. using WS SecurityPolicy at the national connector interface).

An established WS* framework MUST be used (e.g. SUN WSIT).

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

5 Configuration Manager

The epSOS Configuration Manager consists of 4 parts:

- **NCP Configuration Manager:** library for encapsulating access to the epSOS NCP configuration (path names, URLs, etc.).
- **Routing Manager:** library for runtime access of routing information.
- **NCP TSL Synchronizer:** batch tool for making local copies of all NCPs TSLs. Certificate data will be stored in a format that can easily be copied into gateway and web server configuration files. Routing information will be written to the NCP configuration.
- **TSL Editor:** GUI tool for editing the NCPs own TSL. Required functionalities include certificate import, editing of all elements, signature application and upload.

NCP Configuration Manager


The NCP Configuration Manager is a JAVA class for accessing the epSOS NCP configuration.

The epSOS NCP configuration is a human readable and editable file that contains key=value pairs. Its syntax MUST follow the constraints given for JAVA properties. Values MAY include string values within %-signs that MUST be replaced by the respective values of the NCP configuration or the operating system configuration (in exactly this order: if the key is not part of the NCP configuration it MUST be looked for at the operating system configuration) E.g. for a value of "%epsos%/drop/NSL/" the NCP Configuration Manager will replace %epsos% by the respective value that is assigned to this attribute with the OS configuration.

The epSOS NCP Configuration Manager SHOULD use the existing JAVA Properties or Preferences class libraries. At least the following methods MUST be provided:

- String getProperty(String key) for reading a property value from a given key. If the key is not defined an empty string value is returned.
- Object setProperty (String key, String value) for assigning a value to a key. If the key already exists, the old value is replaced by the new value. The new value is visible immediatly and written to the property file immediately.

The constructor MUST NOT require any arguments. It MUST initialize the NCP configuration from a pathname that is provided by the system configuration (e.g. epsos.config=/usr/epsos/ncp.config). If no configuration file exists at the given location, an error is thrown. The configuration file MUST be read only once when the first instance of the NcpProperties class is created.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

NCP Routing Manager

The NCP Routing Manager is a JAVA class for accessing the NCP WSEs. All WSEs MUST be kept with the epSOS NCP configuration. The keys to be used MUST be of the syntax "countrycode.servicename.URL" (e.g. "AT.OrderService.URL" for the Austria epSOS Order Service WSE). The routing manager MUST use the NCP Configuration Manager for accessing routing information.

The routing manager MUST at least provide the following methods:


- URL getServiceWSE(ISOCountryCode countryCode, CV serviceName) for obtaining the full URL of a given service provider of a given country. If the requested service is not defined or unknown, a NULL value is returned.
- URL setServiceWSE(ISOCountryCode countryCode, CV serviceName, URL wseURL) for writing the WSE of a service to the NCP configuration. The result is the new service WSE if the operation succeeded or NULL on failure.
- ISOCountryCode getCountryFromCertificate(signatureCertificateID)

The constructor MUST NOT require any arguments.

NCP TSL Synchronizer

The NCP TSL Synchronizer is a batch tool for retrieving required information about other countries' NCPs. The tool performs the following actions:

- read the country codes of the epSOS countries from the NCP configuration
- for each country:
- read the country TSL from a known location (e.g. via HTTPS)
- Verify the authenticity and integrity of the TSL (by using the epSOS Signature Manager)
- Extract the service WSEs from the TSL and write them to the NCP configuration
- Extract the NCP SSL certificates from the TSL, verify their validity (via OCSP or CRL) and write them to the file system in a format that can be used for the configuration of the web server (DER format)
- Extract the NCP IPsec certificates from the TSL, verify their validity (via OCSP or CRL) and write them to the file system in a format that can be used for the configuration of the router (DER format)
- Extract the NCP Signature certificates from the TSL, verify their validity (via OCSP or CRL) and write them to a trust store (DER format). In addition each certificate is registered in the NCP-configuration as a key-value pair where the key is derived from the certificate number and the value contains the country code.
- Write an audit trail entry for the successful processing of the TSL

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

The format of epSOS TSLs is specified in D3.4.2. The format of the audit trail entry will be added to the next draft of D3.4.2 (end of May).

TSL Editor


The TSL editor is a graphical user interface tool for editing the NCPs own TSL. An administrator **MUST** be able to edit all fields of a TSL as specified in D3.4.2. In addition the following features **MUST** be provided:

- by default the previous version of the TSL is loaded for editing
- version numbers are managed by the tool
- fields with fixed values are set by the tool
- certificates can be imported in a standard format from a local trust store or from the file system
- the TSL is managed internally as a DOM on the ETSI schema
- for signing the TSL the administrator may choose between an existing certificate and a certificate on a smart card. In any case the ID of the certificate **MUST** be registered with the NCP configuration
- the TSL can be uploaded to the defined location (plus a local copy)
- the tool ensures that only properly signed TSLs can be uploaded.

Configuration Attributes of the Configuration Manager

The following table lists the keys and respective values of the epSOS configuration that are used by the Configuration Manager. The provider of the Configuration Manager **MAY** add further keys for internal use by the Configuration Manager.

Key	Value
Local.CentralServices	Path where local copies of configuration data (NSL, MTC) is stored. the default value is "%epsos%/centralservices"
Local.NslDropZone	Path where the NCP's own NSL is maintained The default value is "%Local.CentralServices%/drop/NSL"
Local.NslRunningZone	Path where the current version of the NCP's NSL is provided for access by other NCPs. The default value is "%Local.CentralServices%/running/NSL".
CoT.CountryCodes	Semicolon-separated list of all countries that are assigned to the same circle of trust as the NCP. Example: "dk;se;it;at"
at.Nsl.URL	URL of the Austrian, German, Danish, etc. NSL. The default is " https://countrycode.epsos.eu/centralservices/running/nsl/nsl.tsl "
de.Nsl.URL	
dk.Nsl.URL	
...	
at.OrderService.WSE	Webservice endpoints of the national epSOS services. The WSE are taken from the countries' NSLs by the TSL Synchronizer and written into the configuration. Therefore these entries MUST NOT be managed manually.
de.OrderService.WSE	
...	
at.PatientService.WSE	
de.PatientService.WSE	
...	
at.DispensationService.WSE	
...	
at.ConsentService.WSE	

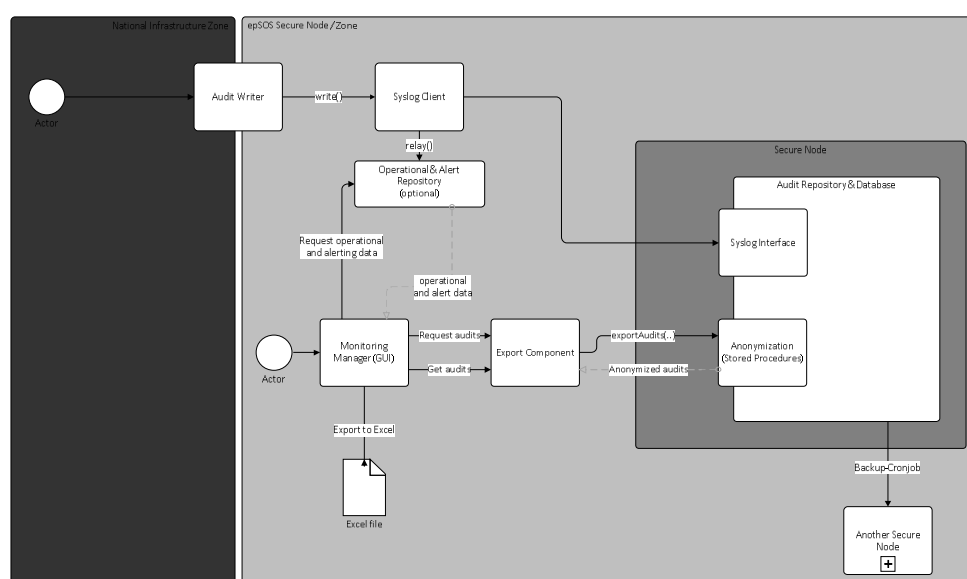
	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

6 Audit Manager

The epSOS Audit Manager consists of 5 parts:

- Audit Writer:** library for populating audit trail entries
- Syslog Client:** library to send the audit trail data to repository
- Audit Repository:** multithreaded server rfc5424,rfc5425 compliant
- Anonymization:** library for anonymizing the audit trail
- Monitoring Manager:** gui for analyzing the audit trail data


epSOS CCD Audit Manager



Audit Writer

The Audit Writer is a Java Class (e.g. EpsosAuditTrailEntry) that MUST at least provide the following methods for the population of an audit trail entry:

- Constructor()
- setEvent(String eventId)
- setPointOfCare(String hcpoID, String hcpoType)
- setHumanRequestor(String userID, string userName, string userRole, opt String assertionID)
- setConsumerNCP(String oidNCP)
- setProviderNCP(String oidNCP)
- setPatient(String patientID, opt String sourcePatientID)
- setMappingService(String oidService, String serviceType)
- setError(SOAPErrorType err, int outcomeIndicator)
- setRequestHeader(String uuidMsg, String securityHeader)

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

- setResponseHeader(String uuidMsg, String securityHeader)
- addParticipantObject(String typeCode, String dataLifeCycle, String idTypeCode, String id, AttributeValuePair[] details)
- write()

Syslog Client

The Syslog client is triggered by the Audit Writer's write() operation. It performs the following actions:


- assemble a syslog payload (octet stream) from the audit data
- apply CMS Signature to the payload (using the NCP signature certificate) by applying a XML-DSig detached signature (functionality is provided by the Signature Manager of the Security Manager)
- compile and assemble a syslog-compliant message (RFC 5424) with a RFC3881-conformant payload alignment
- manages a TCP over TLS secure channel for communicating with the selected audit repository
- provides an configurable interceptor for relaying operational (performance) and alerting information to the monitoring facilities
- decide upon configuration data (facility name) to dispatch the respective syslog message to either audit repository & operational log, audit repository only, or operational log only
- send the syslog message to the audit repository (all information that is needed for this is given in the NCP configuration)
- temporarily caches message objects when the audit repository is currently unable to process further messages (when a communication and/or TLS error is received)

Audit Repository

The Audit (Record) Repository is a multithreaded server that provides a secure and persistent on-line storage facility for auditable information chunks (audit records).

The audit repository included in the reference "NCP-in-a-Box" package is exclusively designed to store and to provide epSOS-compliant auditable information chunks on request. In order to facilitate the management of auditable information chunks, the reference audit repository must be fully aligned to the current standards concerning auditable information processing such as IHE ATNA and syslog.

As underlying technology, the audit repository must exploit existing database technology in order to benefit from its secure storage capabilities, efficient data processing, information encapsulation and the full transactional safety, and advanced encryption opportunities.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

However, the audit repository may not be considered to be able to replace a Member-State-specific long-term archiving facility.

Syslog-Interface

This interface is capable to parse and store rfc5424-style syslog messages whose payload is UTF-8 encoded. Messages must be sent on a specified port (e.g. 6514) using the grammar defined in RFC 5425 via TCP over TLS. The audit XML must be set as MSG part of the syslog message and it must start with the BOM.

The Syslog interface of the Audit Repository performs the following actions:

- ensures an active TCP over TLS channel towards its communication partner (Syslog client sub-component)
- verifies the validity of the enveloped XML signature using the Signature Manager sub-component of the Security Manager
- accepts RFC5424-compliant syslog messages with a RFC3881-compliant payload alignment
- verifies the compliance of the received syslog message
- persistently stores the syslog message in the repository
- disconnects the TLS channel when invalid or incompliant data is received from the syslog client and stores the received syslog message with an additional error code flagging the incompliant message
- disconnects the TLS secure channel when the message signature is invalid and stores the received syslog message with an additional error code flagging the invalid signature

Information Security Safeguards


Audit Records are stored in a database running in an ATNA Secure Node. A cron job running on the secure node is responsible to dump all the tables belonging to the audit repository to another secure node (the communication must be TLS encrypted) for backup/restore and long-term archiving purposes.

The database of the audit record repository must permit access only to the local host (no remote access). All the applications working with audit data must be deployed locally (according with the ATNA secure node definition).

It is implied that all servers executing the audit repository are fully subject to the epSOS Security Policy, the provisions of the FWA, and compliance checks to the baseline security policy of each Member State.

Furthermore, the local database containing the audit information must be secured appropriately by preferably technical means:

All audit information that is to be persistently stored in the audit repository must be fully safeguarded to ensure its integrity. Additionally to the integrity safeguards inherited by the digital signatures of each auditable information chunk, the

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

underlying database must feature further means such as fully thread- and transaction-safe operations.

All information stored in the audit repository must be fully safe-guarded concerning its full confidentiality. The audit information must be either stored individually encrypted or alternatively in an encrypted segment of the database system. All access to the information contained in the audit repository must be protected from unauthorized access by safeguarding the:

- traditional database interaction patterns (database-system user access control),
- communication and transaction safeguards, such as an encrypted data input data stream (preferably TCP over TLS),

as well as supporting safeguards with regards to the system access control on the infrastructural level (such as operating systems access control, storage media control, and environmental security).

The audit repository must ensure at least a full virtual availability with a full insurance of a complete forward data conservation. Even if the underlying database system is temporarily unavailable, the auditable information chunks that are to be stored in the audit repository must be safely and securely cached in the supplying components until the audit repository is able to process this information in-transfer.

With regards to the audit repository, each Member State that operates the audit repository as provided by the NCP-in-a-Box package must implement an adequate data backup and conservation strategy. The audit repository is providing the means to unload the complete auditable information chunks to a file that may be archived in the concrete Member State-specific way. Additionally, the audit repository server should be subjected to an on-line backup process in compliance to the FWA and the individual risk-management process of the Member State.

Export Operation


Due to the sensitive characteristics with regards to the information held in the audit repository, this operation is to be considered a part of the repository and protected by a secure node and a constrained one-way data movement.

This component is responsible for exporting the audit trail data, in order to be transferred into an archive. This functionality must run within the database.

The component will perform a TLS and audited web service call to a service running on the audit record repository secure node. The client invokes the

- exportAudits(Date from, Date to)
- exportAudits(UUID audit_id)

Before sending the audits to the requester, the service sends an audit trail containing the Cn of the certificate of the requestor.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

Export for Reporting

In order to facilitate the evaluation and reporting requirements of epSOS, this component is responsible of performing two integral tasks:

- to securely anonymise the audit data for further processing by the reporting tools, and
- to reliably communicate the results to the Monitoring Manager

The anonymising export functionality must be performed within the secure database in order to avoid any disclosure of personal information and to lower the protection demands of the subsequent components. The implementation of this functionality must feature safeguards against unintended modification and the anonymising business logic and data access must be fully encapsulated and separated from other users access.

The anonymisation functionality will be provided by exploiting the capabilities of stored procedures within the audit repository:

- stored procedures are executed within the database and only the result of the procedure is made available to the end user.
- stored procedures encapsulate the data and business logic. End users of a stored procedure do not need to have access rights for the data base but only for the stored procedure call (delegation of rights).
- stored procedures may not dynamically be altered after their definition (defined once – executed often principle).
- the result of the stored procedure is a fully anonymised data set that can be taken over and communicated by the exporting tool. The protection demands of the exporting tool and the Monitoring Manager are comparably low
- stored procedures abstract from the database so that the secure node principle is fully provided.

The result of the stored procedure is provided on the local node to the second part of the exporting component. The exporting client invokes


`exportAudits(Date from, Date to)`

procedure in the database is and is provided with the result. This result data set is made available to the end user by a separate web service interface. The Monitoring Manager may contact the web service of the Exporting Component in order to receive the data set for further evaluation.

At no point a direct connection to the audit repository is performed, nor is data that contains personally identifiable information processed at this sub-component.

Monitoring Manager

This component consists of a GUI tool for analysing and searching exported anonmymised audit data. In order to perform this task it will take over an

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

anonymized reporting export set from the audit repository and provide the possibility to search this reporting export for predefined criteria. Additionally, this component may convert and save the reporting export as an Excel file that may serve as an input for external reporting generators.

Main functionalities of this subcomponent will be:


- Retrieval of exported anonymized audit trail data
- Search audit trail data, based on several criteria
- Convert and store the audit trail data as Excel file
- receives the operational and alerting information from the (optional) monitoring facility over a separate communication channel
- displays the received operational and alerting information

Apart from the rather rudimental reporting capabilities of the Monitoring Manager the exported Excel file may be used as an input for more sophisticated reporting tools (such as iReport or Jasper Reports).

Additional functionality may be to analyse the reporting export using security alerting tools. The Monitoring Manager may be invoked by an external trigger, such as a scheduled cron-job (potentially using existing solutions such as the Quartz-Scheduler), relay the exported audit trail to an external security analyser, and distribute the results to a pre-defined set of recipients.

Partner Contributions / Responsibilities

- Test environment for audit trail repository, in order to do audit read/write testing
- Sample configuration files for tsl, example of URLs
- We have to verify signatures of configuration data in the configuration manager.
So we need the stubs (for beginning) of a web service method that verifies a signature
- In NCP TSL Synchronizer we have to verify the authenticity and integrity of the TSL (by using the epSOS Signature Manager). So we need the web service stubs for this services
- the signature validation capabilities from the Security Managers are required to operate this component.
- Private (national) and epSOS certificates are required in order to establish TLS channels.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

7 Policy Manager

This formerly common component is now integrated into the Tiani backbone components (in reference to change proposal CCD-PM). This decision was taken in favour of a new courtesy common component – the consent manager – that actually provides real epSOS-specific functionality in contrast to the policy manager that features a standard-compliant interface anyway. Naturally, all services of the policy manager are delivered in the agreed extent.

Components Overview

The epSOS policy manager is part of the epSOS security manager. It includes all functionality that is related to roles, permissions, policies and their proper assignment and enforcement.

The epSOS Policy Manager consists of 3 parts:

- **Attribute Service:** library for assembling the epSOS-coded HCP identity attributes.
- **AuthzQuery Creator:** library for assembling an authorization decision query from a given set of attributes and resources.
- **PDP:** XACML 2.0 compliant PDP that communicates with the PAP (national infrastructure) and accepts SAML XACMLAuthzDecisonQuery requests.


Attribute Service

The Attribute Service is deployed at NCP-B only. It is responsible for assembling an SAML attribute statement acc. to D3.4.2 from a set of HCP identity attributes. It provides the following interface:

SAMLAttrStmtDOM getAttributeStatement(hcp-name, hcp-national-role, hcp-on-behalf-of, hcp-speciality, poc-id, purpose-of-use)

The input arguments are processed as follows:

- hcp-id: The HCP identifier is accepted as-is and wrapped as a XSPA subject attribute
- hcp-national-role, hcp-on.behalf-of: The structural role of the HCP (and optionally the role the HCP is acting in behalf of) are provided in country-B encoding. The Attribute Service maps the national roles on the epSOS roles and permissions (see section 2.1) and wraps the result as a set of SAML attributes acc. D3.4.2.
- hcp-speciality: The HCP speciality is accepted as-is and wrapped as a HITSP clinical speciality attribute


	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

- poc-id: It is assumed that each country-B PoC is registered with NCP-B. Based on the provided identifier of the PoC the Attribute Service can extract all other required attributes from the PoC registration data (see section 2.2).
- purpose-of-use: The PoU is accepted as-is and wrapped as a PoU attribute acc to D3.4.2

Role Configuration

Each NCP-B holds a configuration that describes the mapping of roles and the assignment of permissions to roles. The following XML excerpt is outlining the current configuration (presented as XML schema definition):

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:epsos="urn:epsos:access-control:role-mapping" xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:epsos:access-control:role-mapping" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="0.1">
<!--
  This is the basic epSOS' Role type, as defined in 3.4.2.
  A Role is defined with a role name (as defined in the local
  language of the member state, e.g. ``Dentista'', ``Arzt'')
  a translation in the epSOS role name and a set of HI7
  permissions
-->
<xs:element name="role-mapping" type="epsos:RoleMappingType"/>
<xs:complexType name="RoleMappingType">
<xs:annotation>
<xs:documentation>
  The RoleMapping type defines the root element for the
  epSOS' role mapping. It is an unbounded sequence of
  RoleTypes and a free section for the MS' extensions.
</xs:documentation>
</xs:annotation>
<xs:sequence minOccurs="0" maxOccurs="unbounded">
<xs:element name="role" type="epsos:RoleType"/>
</xs:sequence>
<xs:anyAttribute/>
</xs:complexType>
<xs:complexType name="RoleType">
<xs:annotation>
<xs:documentation>
  The RoleType is the basic epsos role mapping. It contains
  a mandatory epsosRoleElementType (the mapping) and a sequence
  of permissions, as defined in 3.4.2.
</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:sequence>
<xs:element name="epsosRole" type="epsos:epsosRoleElementType"/>
</xs:sequence>
<xs:sequence maxOccurs="unbounded">
<xs:element name="permission" type="epsos:PermissionType"/>
</xs:sequence>
</xs:sequence>
<xs:attribute name="role-name" type="xs:string" use="required"/>
<xs:attribute name="on-behalf-of" type="xs:string" use="optional"/>
<xs:anyAttribute/>
</xs:complexType>
<xs:complexType name="epsosRoleElementType">
<xs:annotation>
<xs:documentation>
```


	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

The epsosRoleElementType contains the attribute that maps the local name with the epsos defined name

```

</xs:documentation>
</xs:annotation>
<xs:attribute name="role-name" type="epsos:epsosRoleType" use="required"/>
</xs:complexType>
<xs:simpleType name="epsosRoleType">
<xs:annotation>
<xs:documentation>
    The epSOS roles are defined by 3.4.2. For this pilot these
    are only 2 possible values, MedicalDoctor and Nurse
</xs:documentation>
</xs:annotation>
<xs:restriction base="xs:string">
<xs:enumeration value="medical doctor"/>
<xs:enumeration value="pharmacist"/>
<xs:enumeration value="nurse"/>
<!-- GP and Specialist are "medical doctor" -->
<!-- midwife is a nurse too -->
</xs:restriction>
<!-- The administrator can't exist in epSOS, since no administrative tasks
    can be performed outside the NCP's trust zone -->
</xs:simpleType>
<xs:complexType name="PermissionType">
<xs:attribute name="id" type="epsos:HL7PermissionType" use="required"/>
</xs:complexType>
<xs:simpleType name="HL7PermissionType">
<xs:annotation>
<xs:documentation>
    These are the HL7 Permissions defined by 3.4.2.
    Only these are allowed, and they will be used
    for creating an access control request
</xs:documentation>
</xs:annotation>
<xs:restriction base="xs:string">
<xs:enumeration value="POE-006"/>
<xs:enumeration value="PRD-003"/>
<xs:enumeration value="PRD-004"/>
<xs:enumeration value="PRD-005"/>
<xs:enumeration value="PRD-006"/>
<xs:enumeration value="PRD-010"/>
<xs:enumeration value="PRD-016"/>
<xs:enumeration value="PPD-032"/>
<xs:enumeration value="PPD-033"/>
<xs:enumeration value="PPD-046"/>
<!-- Change/Discontinue/Refill Outpatient Prescription Order -->
<!-- Review Medical History -->
<!-- Review Existing Orders -->
<!-- Review Vital Signs/Patient Measurements -->
<!-- Patient Identification and Lookup -->
<!-- Review Patient Medications -->
<!-- Review Problem List -->
<!-- New Consents and Authorizations -->
<!-- Edit/Addend/Sign Consents and Authorizations -->
<!-- Record Medication Administration Record -->
</xs:restriction>
</xs:simpleType>
</xs:schema>


```

A typical instance of that schema may be:

```

<?xml version="1.0" encoding="UTF-8"?>
- <epsos:role-mapping xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="urn:epsos:access-control:role-mapping role-mapping.xsd" xmlns:epsos="urn:epsos:access-control:role-mapping">

```

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

```

- <epsos:role role-name="Arzt">
  <epsos:epsosRole role-name="medical doctor"/>
  <epsos:permission id="PRD-006"/>
  <epsos:permission id="PRD-003"/>
  <epsos:permission id="PRD-004"/>
  <epsos:permission id="PRD-005"/>
  <epsos:permission id="PRD-010"/>
  <epsos:permission id="PRD-016"/>
  <epsos:permission id="PPD-032"/>
  <epsos:permission id="PPD-033"/>
</epsos:role>
- <epsos:role role-name="Stationsschwester" on-behalf-of="Arzt">
  <epsos:epsosRole role-name="nurse"/>
  <epsos:permission id="PRD-006"/>
  <epsos:permission id="PPD-032"/>
  <epsos:permission id="PPD-033"/>
</epsos:role>
- <epsos:role role-name="Apotheker">
  <epsos:epsosRole role-name="pharmacist"/>
  <epsos:permission id="PRD-006"/>
  <epsos:permission id="PRD-004"/>
  <epsos:permission id="PRD-010"/>
  <epsos:permission id="PPD-046"/>
</epsos:role>
</epsos:role-mapping>

```

The vendor who is responsible for implementing the policy manager must specify the schema of the role configuration and implement the attribute service that makes use of this configuration. The schema may use a different structure than the example above. The only functional requirements are:

- it MUST allow for mapping a national role onto an epSOS role
- It MUST consider mapping pairs of roles for scenarios where one role is acting on behalf of another role
- It MUST allow to assign permissions to a national role (pair)
- The schema MUST allow for member state specific extensions

PoC Registration Data

It is assumed that each point of care in country B that takes part in an epSOS pilot is registered with the respective NCP-B. The registration data MUST at least contain all information that is needed to fill the PoC/HCPO related attributes of the HCP Identity Assertion.


The following XML excerpts shows how such a configuration might look like.

```

<PoC id="...." x509cn="...">
  <HCPO="...."/>
  <HealthcareFacilityType="...."/>
</PoC>
<PoC id="...." x509cn="...">
  ....

```

The vendor who is responsible for implementing the policy manager must specify the schema of the PoC registry and implement the attribute service that makes use of this registry. The schema may use a different structure than the example above. The only functional requirements are:

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

- it MUST allow for retrieving all PoC/HCP related HCP Identity Assertion attributes from a given PoC identifier. This identifier may either be a national PoC ID or the CN of a TLS X.509 certificate
- The schema MUST allow for member state specific extensions

Authz Query Creator

The Authz Query Creator is a JAVA library for assembling an XACML authorization decision query from a given set of attributes and resources.

The Authz Query Creator MUST at least provide the following methods:

- XACMLAuthzQueryRequest getResourceAccessQuery(hcplIdentityAssertion, trcAssertion, countryCode, requestedOperation, opt objectID)
- XACMLAuthzQuery getRequestComplianceQuery(hcplIdentityAssertion, requestedOperation)
- Subject: HCP Identity Assertion Attributes, authentication method (from Identity Assertion)
- Action: requested operation (derived from SOAP message and metadata)
- Resource: patient ID (from TRC assertion), opt objectID (from message body)
- Environment: NCP-B country (from message signature), purpose of use (from identity assertion; overwritten by TRC assertion of PoU is "EMERGENCY")

The getResourceAccessQuery is used at NCP-A. It considers all attributes provided with the assertions, the requested operation and the patient whose data is to be accessed. The objective is to assemble a query that can be used to assess if a requested access to a given patients data is allowed.


The getRequestComplianceQuery is used at NCP-B. It only considers the role attributes, the permission attributes and the requested operation. The objective is to assemble a query that can be used to verify if the HCP is allowed to perform a given epSOS request.

The requestedOperation MUST be encoded as an eventID as defined for the epSOS audit trail. An additional object ID MUST be provided if the operation is on a specific object rather than on the patient. The patient identifier is encoded within the TRC Assertion.

The provided queries MUST comply with the authorization decision query request as specified in the XACML SAML 2.0 profile.

Policy Decision Point (PDP)


The PDP is deployed at both NCPs. It takes as input an AuthzDecisionQuery that was assembled by the Authz Query Creator. It MUST assess the query with the following policies:

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

- XACML encoded national security policy (NCP-A: retrieved by country code; NCP-B: policy ID is part of the NCP configuration)
- XACML encoded patient privacy policy (NCP-A only; based on the patient's consent and additional constraints stated by the patient; retrieved by patient ID)

It is task of the PDP to obtain the required policies from a PAP that is implemented within the national infrastructure. The implementation **MUST** make use of the respective mechanisms as defined in the XACML SAML 2.0 profile (XACMLPolicyQuery).

The patient privacy policy is derived from the patient's consent. If a patient has not given consent, a default deny-policy is provided by the PAP.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

8 Transformation Manager

The epSOS Transformation Manager is part of the epSOS semantic service. It includes all functionality that is related to rendering of the epSOS documents:


Component responsibilities:

- Translating and/or transcoding (if necessary) the original data compliant to epSOS CDA syntax from the national language and possibly from the national code system(s) in the document creator country (in most cases Country A) to the epSOS Reference Terminology. From a functional point of view the translation and transcoding are the same operation for the Transformation Manager from the point of view of the document creator country (in most cases Country A).

Operations

ToEpSOSPivot(EpSOSOriginalData) :EpsosCDA


Textual description of operation	
<i>Transformation of national data to epSOS pivot format.</i>	
Input parameters	<p>EpSOSOriginalData</p> <p><i>Medical document in its original data format as provided from the NationalConnector to this component.</i></p> <p><i>The provided document is compliant with the epSOS pivot CDA (see D 3.5.2 Appendix C) unless the adoption of the element binding with the epSOS reference Value Sets.</i></p> <p><i>[Mandatory]</i></p>
Output parameters	<p>EpSOS CDA structure</p> <p><i>Response structure including the epSOS pivot CDA and the response status structure.</i></p> <p><i>The response status structure provides information about the operation results, including possible errors and warning.</i></p>
Component behaviour	<p><i>After having received a toEpSOSPivot() request, this component takes the EpSOSOriginalData (already compliant to epSOS CDA syntax) and using the TSAM capabilities, accomplishes the eventual transcoding of the terms present in the epSOS value sets, while also keeping the original codes and display name. An epSOS pivot document with epSOS coded concepts is therefore produced. The epSOS pivot document shall to have a link to the EpSOSOriginalData.</i></p>

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

	<i>Exceptions: in case of processing error or warning, the responseStatusStructure will be used to convey this information to the calling component with an appropriated error and warning code. A detailed list of the managed exceptions will be provided in the Detail Design Specification document. Each exception condition occurred will be logged (standard and audit), reporting both the exception code and its English description.</i>
Notes	

Translate (EpsosCDA; TargetLanguageCode) : TranslatedEpSOSCDA

Textual description of operation	
Translation from epSOS pivot data to consumer country language.	
Input parameters	<p>EpSOSosCDA Document in epSOS pivot format (with epSOS codes)</p> <p>TargetLanguageCode. Identifier (code) of the target language.</p>
Output parameters	TranslatedEpSOSCDA epSOS pivot CDA with translated epSOS codes into the consumer country language.
Component behaviour	<p>After having received a translate() request, this component starts to process the received EpSOSCDA in order extract the epSOS coded concepts.</p> <p>Subsequently, for each coded concept found, it makes use of the TSAM capabilities to obtain the representation of that concept in the target TargetLanguageCode identifier. This information is therefore used by this component to update the displayName attribute of that coded entry.</p> <p>After the completion of this translation phase, an epSOS pivot document with “translated” concepts is obtained.</p> <p>This document is therefore returned to the requesting party. No changes are applied to the document identifiers.</p> <p><i>Exceptions: in case of processing error or warning, the responseStatusStructure will be used to convey this information to the calling component with an appropriated error and warning code. A detailed list of the managed exceptions will be provided in the Detail Design Specification document. Each exception condition occurred will be logged (standard and audit), reporting both the exception code and its English description.</i></p>
Notes	

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

Component behaviour (called operations)

getEpSOSConceptByCode()

This component issues a `getEpSOSConceptByCode()` request in order to know the best matching epSOS Concept, according to the information provided.

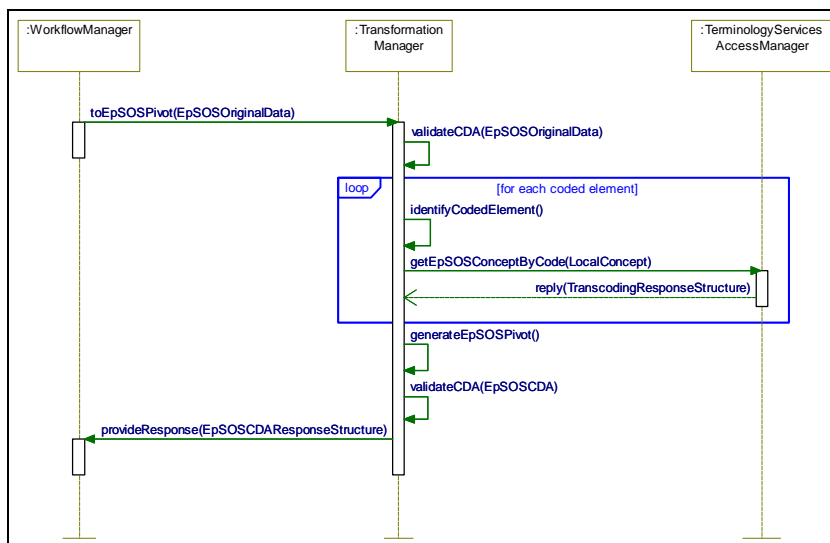
Exceptions: in case of returned errors this component shall appropriately use NullFlavors for valorising the “main” coded concept; nevertheless the original code shall be provided through the <translation> element.

getDesignationByEpSOSConcept()

This component issues `getDesignationByEpSOSConcept()` request in order to know the target language epSOS Designation, according to the information provided.

Exceptions: in case of returned errors no actions are required for displayName translation.

This component provides capabilities for coded concepts translation and transcoding if necessary.




Functional requirements

This section summarizes requirements on Transformation Manager.

Document transformation

1. For a provided document containing original data in form of epSOS compliant structured CDA level 3, create epSOS pivot document as transcoding of all coded elements included.
2. For a provided document containing original data in form of epSOS compliant CDA level 1 embedding PDF, create epSOS pivot document as transcoding of all coded elements included.


	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

3. For provided epSOS pivot document in structured form (both CDA level 1 and level 3), extend the document with translation of all coded elements.
4. Country A has to provide original data in epSOS CDA compliant form, i.e. document has exactly the form of epSOS pivot document, but coded elements use local codes.
5. Country A has to provide original data in human readable form of PDF file, which is included into CDA document. Content of PDF file should correspond to documents defined by epSOS, it is in language and coding of country A and will not be modified or extended during transformation process.
6. The relationship (linkage) between the two provided documents (structured and containing PDF) has to be created by national infrastructure/national connector of country A.
7. When creating epSOS pivot document, in coded elements information about code, code system and its version must not to be repeated in translation element, if it is the same as in original data or other translation element.
8. Country B may decide, that some coded elements will not be translated into target language but kept in English.
9. All exceptions occurred during transcoding or translation has to be reported to Workflow Manager using response structure.

Non-functional requirements

Specific requirements for TM

1. Provide public interface for Workflow Manager with methods for:
 - transformation of original data in structured form to epSOS CDA pivot document,
 - translation of epSOS pivot document into target language.
2. Result of transformation of original data in structured form to epSOS pivot document is compliant with HL7 CDA level 3 specification.
3. Result of transformation of original data in unstructured form to epSOS pivot document is compliant with HL7 CDA level 1 specification.
4. Input and output data of TM public interface has to be harmonized with requirements on Workflow Manager.
5. For all operations response structure containing result of the operation has to be provided.
6. Each exception condition occurred will be logged, reporting both the exception code and its English description.
7. All exceptions occurred during any operation has to be reported to Workflow Manager using response structure.
8. MVC contains Value set for error codes, which has to be used when reporting error or warning.
9. Audit trail of all transformation operations has to be written.
10. For writing an audit trail Audit Manager will be used.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010


11. For configuration parameters Configuration Manager will be used.

epSOS Transcoder

The epSOS Transcoder component implements the transformation of CDA documents as sketched in the [HLDD]:

- *ToEpSOSPivot(EpSOSOriginalData; ReplacedDocId) :EpsosCDA*
- *Translate (EpsosCDA; TargetLanguageCode) : TranslatedEpSOS CDA*

An audit trail entry MUST be written.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

9 TSAM

Components Overview

The epSOS Terminology Service Access Manager (TSAM) is part of the epSOS semantic service. It includes all functionality that is related to mapping national codes on epSOS MVS codes and vice versa. While the mapping tables are managed centrally in the CareCom tool, the NCPs always only work with local copies of these mappings.

The epSOS Terminology Service Access Manager (TSAM) is part of the epSOS semantic service. It includes all functionality that is related to mapping national codes on epSOS MVS codes and vice versa. While the mapping tables are managed centrally in the CareCom tool, the NCPs always only work with local copies of these mappings.

The epSOS Terminology Service Access Manager consists of 4 parts:

- **Database Schema:** Schema for the local term mapping table.
- **ValueSet Retriever:** JAVA library for synchronizing a single value set by copying the centrally managed table into the local database
- **TSAM Synchronizer:** Batch job for synchronizing all value sets' local copies at once.
- **Taxonomy Mapping Service:** JAVA library for switching a single term

Database Schema

Acc. to JWG HLDD a database is used for storing the local copies of the epSOS MVC value sets. While the database is provided by the NCP core, a dedicated set of tables must be set up to manage the epSOS value sets.

The result of this work item is a schema for the local database table(s) as a SQL DDL.


ValueSet Retriever

The ValueSet Retriever is a library that allows for extracting a national mapping of a whole value set from the central repository (e.g. using CTS(2)). The ValueSet Retriever makes use of DAO/JDBC to write these mapping into the local NCP database table.

The ValueSet Retriever must at least implement the following operation:

syncValueSet(valuesetID, countrycode)

This operation copies the epSOS coding and the national coding (of the given country) of the given value set into the national mapping table (database). It MUST

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

be thread-safe in a way that all concurrent read requests are blocked until the value set is copied in total. In advance to copying the national coding to the local NCP database, the ValueSet Retriever MUST verify the integrity and authenticity of the MTC.

All information that is needed for connecting to the central repository, for accessing the local database and for verifying the MTC signature must be read from the epSOS NCP configuration (see Configuration Manager Common Component).

Additional functionality includes:

- writing of an audit trail entry on successful copying of a value set translation table (see D3.4.2 for details)
- verification of the authenticity of the central service (mutual authentication based on SSL and a registered service certificate) and the value set mapping table

TSAM Synchronizer

The TSAM Synchronizer is a batch tool that copies all mapping for a country into the local database. In a first step it obtains the IDs of all value sets from the central repository and then copies them by calling the ValueSet Retriever for each of these value sets.

Taxonomy Mapping Service


The Taxonomy Mapping Service is a library that builds upon DAO/JDBC and that allows for simple SQL queries for mappings (ValueSet Identifier together with either epSOS or national Code as a search key). By this it implements the following operations as specified in the HLDD:

- getEpSOSConceptByCode(LocalConcept) : TranscodingResponseStructure*
- getDesignationByEpSOSConcept(EpSOSRefConcept; TargetLanguageCode) : TranslatingResponseStructure*

The specification of the strucs (classes) used in the Taxonomy Mapping Service interface is part of the Taxonomy Mapping Service implementation.

Additional functionalities are:

- if a mapping cannot be found, the respective value set is synchronized using the library developed in 3). It must be possible to switch off this feature by setting a respective flag in the configuration (using the epSOS Config Manager common component).
- The library must be thread-safe and therefore all access to a value set must be blocked during synchronization.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010


Component responsibilities

- Translating a given concept designation into the requested target language using the information present in the Terminology Repository.
- Transcoding a given “local” coded concept into the appropriate epSOS coded concept using the information present in the Terminology Repository.

Operations

getEpSOSConceptByCode(LocalConcept):TranscodingResponseStructure


Textual description of operation	
<i>Transcoding a given “local” coded concept into the appropriate epSOS coded concept using the information present in the Terminology Repository.</i>	
Input parameters	<p>LocalConcept;</p> <p>structure used to convey the concept derived from the epSOS Original Data.</p> <p>It shall include at least the concept code and the concept code system.</p> <p>Code System Version, Country Code and value set OID - if available - should be provided.</p>
Output parameters	<p>TranscodingResponseStructure</p> <p>Response structure including:</p> <ol style="list-style-type: none"> 1.the epSOS Reference Concept: this means the Concept Code, the English designation, the concept code system (OID), Code System Version, Value Set OID; Value Set Version, 2.The responseStatusStructure, providing information about operation result, including possible errors and warning.
Component behaviour	<p>When this component receives a getEpSOSConceptByCode() request, it uses all the data extracted from the LocalConcept structure in order to search within the Terminology Repository for the best matching epSOS Concept, according to the local information provided (e.g., if no code system version is indicated, the latest version will be provided). All information retrieved is finally returned to the requesting component.</p> <p>Exceptions: if there is no transcoding or a processing error occurs, the responseStatusStructure will be used to convey this information to the calling component with an appropriated error and warning code. A detailed list of the managed exceptions will be provided in the Detail Design Specification document.</p> <p>Each exception condition occurred will be logged (standard and audit), reporting both the exception code and its English</p>

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

	<i>description.</i>
Notes	<i>Makes use of Terminology Repository</i>

getDesignationByEpSOSConcept(EpSOSRefConcept;TargetLanguageCode) :
TranslatingResponseStructure

Textual description of operation	
<i>Translating a given concept designation into the requested target language using the information present in the Terminology Repository</i>	
Input parameters	<p>EpSOSRefConcept.</p> <p><i>Structure used to convey the concept derived from the epSOS pivot CDA.</i></p> <p><i>It shall include at least the concept code and the concept code system.</i></p> <p><i>Code System Version, Country Code and value set OID - if available - should be provided.</i></p> <p><i>TargetLanguageCode identifier (code) of the target language.</i></p>
Output parameters	<p>translatingResponseStructure</p> <p><i>Response structure including:</i></p> <ol style="list-style-type: none"> <i>1.the target language concept designation;</i> <i>2.the responseStatusStructure providing information about operation result, including possible errors and warning.</i>
Component behaviour	<p><i>When this component receives a getDesignationByEpSOSConcept() request, it uses all the data extracted from the EpSOSRefConcept structure in order to search within the Terminology Repository for the target language epSOS Designation, according to the local information provided (e.g., if no code system version is indicated, the latest version will be provided). All information retrieved are finally returned to the requesting component.</i></p> <p><i>Exceptions: if there is no translation or a processing error occurs, the responseStatusStructure will be used to convey this information to the calling component with an appropriated error and warning code. A detailed list of the managed exceptions will be provided in the Detail Design Specification document.</i></p> <p><i>Each exception condition occurred will be logged (standard and audit), reporting both the exception code and its English description.</i></p>
Notes	<i>Makes use of Terminology Repository</i>

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

Functional requirements for TSAM

Transcoding/Translation functionality

1. For a provided local concept find transcoding to an epSOS concept.
2. For a provided epSOS concept in local language find translation to the epSOS reference language.
3. For a particular coded element, local code system in Country A may be either different from epSOS code system necessary to create epSOS pivot document, or it can be the same code system.
4. Country A may use various languages in which original data are provided.
5. Original data based on older version of coded system may be provided by Country A.
6. Country B may use various languages in which translated data are requested.
7. Country B may decide, that some coded elements will not be translated into target language but kept in English.
8. During translation/transcoding if no version of code system or value set is provided, try to use current (last known) version.
9. All exceptions occurred during transcoding or translation has to be reported to Transformation Manager using response structure.


Local Terminology Repository

1. Structure of Local Terminology Repository will be provided as DDL describing database schema.
2. Structure and content of LTR is based on MVC and MTC of a particular country.
3. LTR contains epSOS reference terminology in local languages of a member state and in English (as reference language).
4. LTR is synchronized with eCRTS when new version of epSOS reference terminology is released.
5. If transcoding / translation fails because searched concept was not found in LTR, corresponding value set has to be synchronized.

Non-functional requirements for TSAM


Specific requirements for TSAM

1. Provide public interface for Transformation Manager with methods for

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

- transcoding of concept and
 - translating of concept.
- 2.Input and output data of TSAM public interface has to be harmonized with requirements on TransformationManager.
 - 3.For all operations response structure containing result of the operation has to be provided.
 - 4.Each exception condition occurred will be logged, reporting both the exception code and its English description.
 - 5.All exceptions occurred during transcoding or translation has to be reported to Transformation Manager using response structure.
 - 6.MVC contains Value set for error codes, which has to be used when reporting error or warning.
 - 7.On-demand synchronization has to provide possibility to switch it off.
 - 8.All access to LTR must be blocked during synchronization.
 - 9.For writing an audit trail component Audit Manager will be used.

For configuration parameters Configuration Manager will be used.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

10 National Interface/Connector

The *Spirit EHR* for NCP is an IHE compliant application that enables secure and audited NCP to NCP communications and an interface for national infrastructures that any member state can use for providing, querying, and retrieving Patient Summaries, ePrescriptions, and eDispensation documents, which will fulfil the epSOS use cases. The *Spirit EHR*, together with the other Common Components, constitutes the so called “NCP-in-a-Transparent-Box”.

The use of JAVA library calls, as described in the High Level Design Document, were replaced with WS calls, due to the fact, that the High Level Design Document was written under the assumption that the Common Components will be developed in common in an open source way. Since *Spirit EHR* was chosen as the solution to be the core part of the NCP, the WS calls are used in order to perform an easy implementation, a highly scalable solution, and a minimum of dependencies are guaranteed. Furthermore, *Spirit EHR* is a “closed code” product and therefore this is the proposed compromise to achieve all the functional specifications and meet the functional and non-functional requirements of the High level Design Document.

National infrastructure to NCP-B

Member states connect to NCP-B via a National Connector (NC) that invokes the National Interface using data coming from the existing national infrastructure. *Spirit EHR* gives two possibilities for NC to NI invocation. A first one, based on IHE transactions (depicted in Figure 4) and a second one (Figure 5) for member states that do not implement an IHE compliant messaging interface.

An IHE compliant interface / infrastructure to NCP-B

The national connector of country B can connect to the *Spirit EHR* in two different ways (NCP-B):

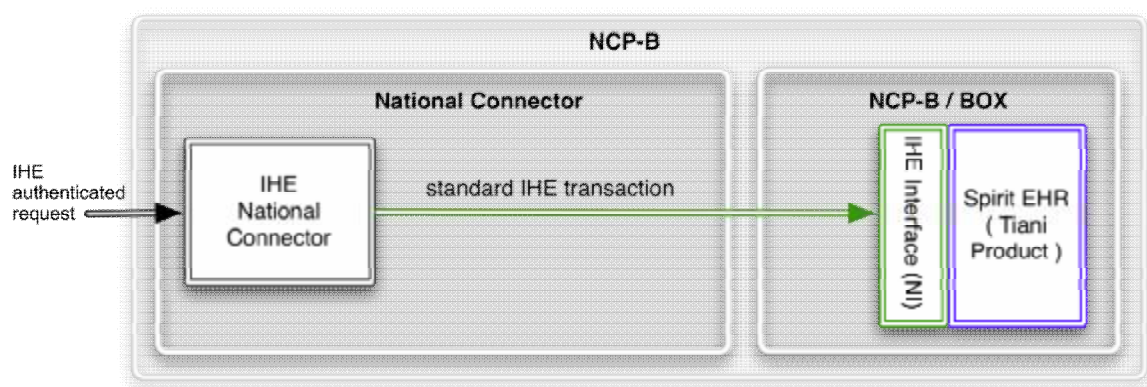



Figure 4 IHE-Compliant National Connector

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

The first one is to use standard IHE transactions such as PDQ, XCA query, XCA retrieve and XDS submit, as described in Fig. 1.

In this approach, NC receives an IHE authenticated transaction (based on XUA) and invokes the NI by using the IHE messaging interface (green arrow). If permitted by the member state local legislation, the Point of Care can connect directly using IHE messages to the NI.

A proprietary infrastructure to NCP-B

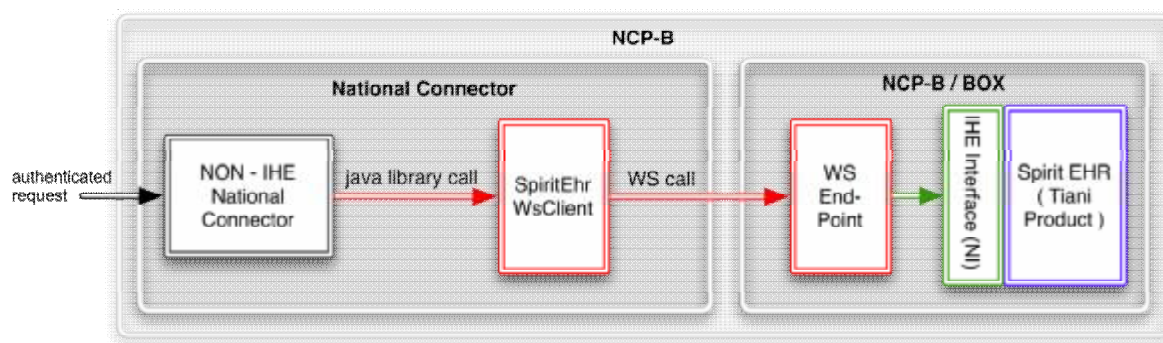


Figure 5 Non IHE-Compliant National Connector


The second possibility is to use *SpiritEhrWClient* library, as shown in Fig. 2. This library provides a set of java classes, which can be used by those national connectors, which do not implement standard IHE transactions. With this approach non-IHE compliant member states are able to communicate with the *Spirit EHR*.

In the second approach, NC receives a generic authenticated request from the PoC, as defined by the member state. NC instantiates a set of Java methods defined in the *SpiritEhrWClient* library (red arrow) that performs a web service invocation to the WS Endpoint. This actor is responsible to translate the calls into an IHE messages and route them to the NI (red box to green arrow). According to the national or regional legislation and administrative systems, this WS Endpoint might be located within the NCP, the existing infrastructure or deployed separately.

NCP-A to a national infrastructure

For communication of the *Spirit EHR* with national infrastructure in country A (NCP-A) IHE standard transactions are supported. If a national eHealth system does not provide an IHE interface, the *SpiritEhrWClient* library can be used to push data into an IHE compliant system.

The member state connects its own infrastructure to the NCP-A to give access to the epsos compliant documents of its patients. This is possible in several ways. The preferred way is the IHE compliant interface or infrastructure, since the

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

exchange of messages and documents is happening in a standardized way. In order to connect Member States, which do not provide an IHE compliant infrastructure, but do have a different solution of their eHealth system, a WS interface is provided to “translate” the standardized data used within the epSOS framework to the format used in the Member State.

The several possibilities to connect the already existing systems to the NCP-A are described below:

NCP-A to a IHE compliant interface / infrastructure

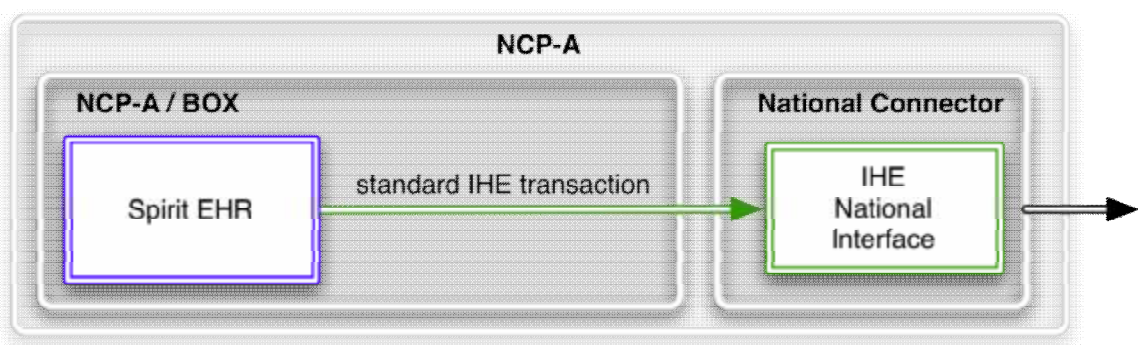


Figure 3 NCP-A to National IHE interface

In this scenario the NCP-A is connected to an IHE compliant interface or national infrastructure that supports the IHE transactions described in chapter 4. If the interface is used, it has to translate the national format to standard IHE transactions, whereas if the infrastructure is IHE compliant, the connection to NCP is possible directly.

NCP-A to a NON - IHE compliant infrastructure

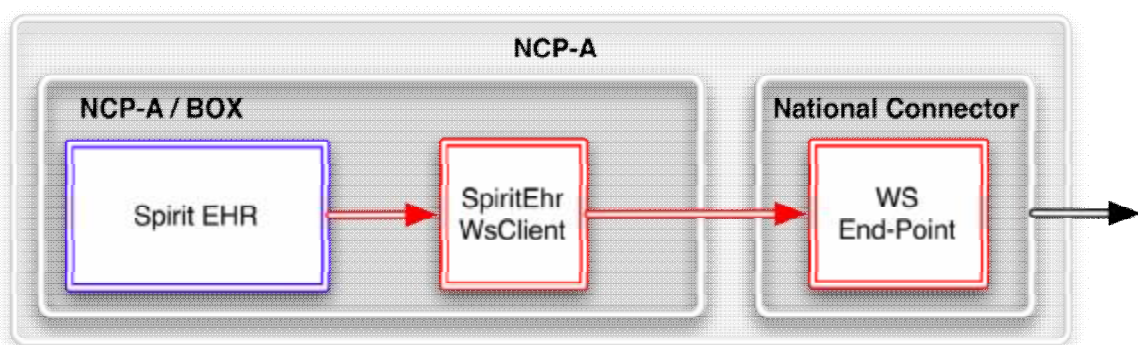


Figure 4 NCP-A to National NON - IHE infrastructure



	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

Figure 4. shows the scenario how a NCP-A Is able to be connected to a NON-IHE compliant infrastructure. In this case, the *SpiritEhrWSCClient* “translates” the standard transactions to web services, which are connected to the WS End-Point in the National Connector. So the national infrastructure is able to communicate with the epSOS framework.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

11 Front-end Portal

Common Components will develop two portal solutions for Country B and the MS can choose which one they will take over.

The Processes and workflow for the portal was described in WP 3.6 (Identity Management).

Data Elements for Patient Summary, ePrescription and eDispensation were described in D3.5.2 Appendix B and C, . D3.9.1 Appendix B1 provides the Semantic Interoperability implementation guide.

View of Patient Summary

1. HCP Authentication

MS have to provide a HCP/HCPO Database or a LDAP to authenticate the HCP as a medical doctor or as a pharmacist. The Portal defines a web service to get back from the National Infrastructure the HCP / HCPO credentials and Role

2 Selecting Country.

The portal will ask for the patient country (either by combo box or by selecting a flag).

3. Patient Identification

According to the selected country a different search form will be displayed.. Each country can have different identifiers to search for patients. Some also permit to search by demographics (name, address, birth date, gender etc).


4. Confirmation

According to a configuration parameter of NCP (MS likes to have confirmation or not) the system asks for the patient confirmation. This confirmation includes the patient Id, the organisation of doctor and a time limit (when to be invalidated).

This confirmation will be stored to the portal database. The confirmation is only valid within the same organization or for a defined time (eg 1 month)

5. Consent

Afterwards the portal application checks if a valid consent exists for this patient for Country B. The consent documents are always stored in Country A. If

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

exists the process can proceed, if not, a new consent in country B for country B will be created. This consent will be send then afterwards to Country A.

6. Printing Consent Info

This is only the case if some changes were done in the consent document. The doctor prints the consent document (based on local templates) in 2 languages (in each participant language). The consent document (information paper) is stored at NCP level.

7. Patient Summary

The portal requests the patient summary documents (max two documents) in CDA and PDF format. The portal renders the patient summary document which is in CDA format and downloads the PDF. The user can select which document he likes to see.

View of ePrescription


- 1.Steps 1-6 same as for Patient Summary
- 2.The portal asks for the list of active prescriptions and gets a list of CDA documents. Each document must have a meaningful header to display in the screen in order for the user to select it.
- 3.The user selects one prescription and gets the item or items. He can then select one by one and dispense it. The items from ePrescription will be copied to the eDispensation User interface and the pharmacist can afterwards manually change the item dispensation is different to prescription: The dispensation document will be created as a CDA document and delivered to Country A

Consent in Country A for several Country B


- 1.Patient identification
- 2.Patient confirmation
- 3.The Application receives a list of valid consents
- 4.The doctor can select to give or update or revoke a consent.
- 5.The doctor can create a new consent by selecting a country and enter the validation date for each country.
- 6.Consent Document will be stored in Country A

Web services

- HCP Authentication
- Query for a patient with demographic attributes and identifiers
- Query for a valid consent for patient in country A for country B
- Creates a consent for patient in country B for country B
- Send the consent to country A

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

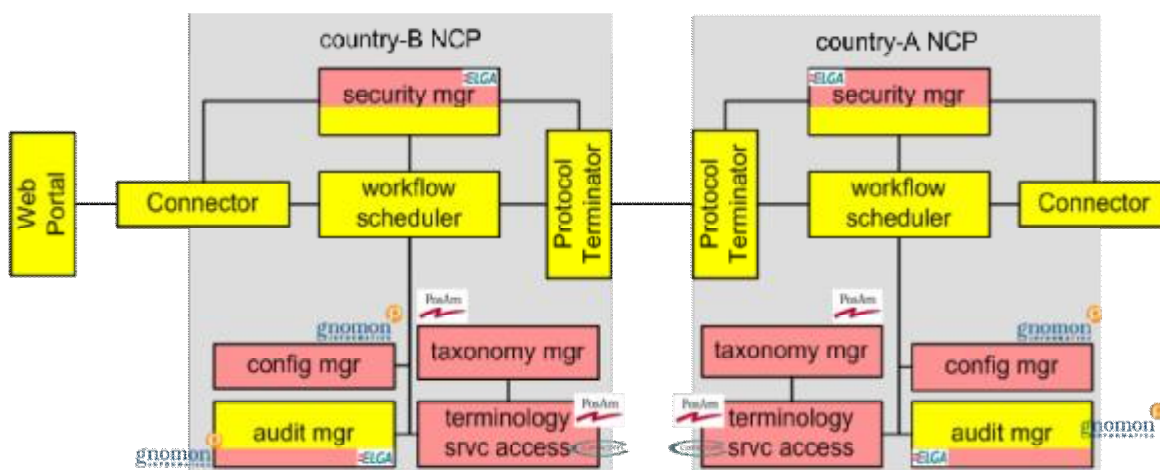
- Query for patient summary documents (without prescription content)
- Get active prescriptions for a patient
- Send the dispensation to country A
- Get list of active consents for patient in Country A for all Country B
- Send Consent document for patient in country A for country B

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

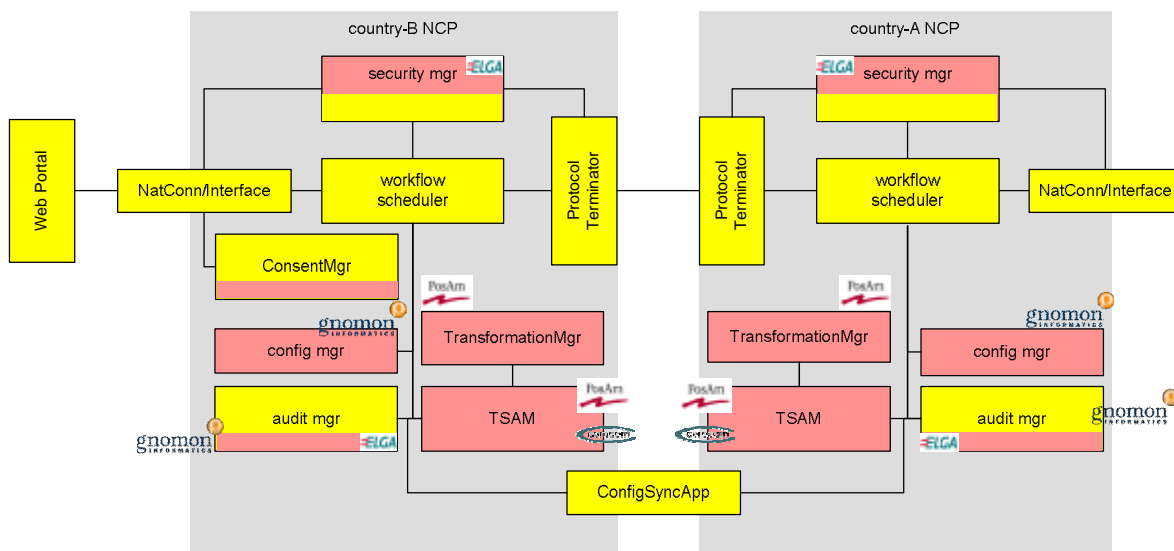
12 Change Proposal and HLDD-Delta in CCD

The goal of this chapter is to report the change requests handled and to list the major delta between the HLDD and the CCD.

The original NCP design as outlined in the HLDD is:




The NCP-in-a-Box as implemented by the CCD team is scoped as such:



Change Proposals Listing


This section provides an overview regarding the change proposals, including a brief description and scope of the concrete proposal.

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010


Accepted and quality-assured change proposals are gradually uploaded onto ProjectPlace and provided to all epSOS beneficiaries in the folder “Change Proposal”:

<https://service.projectplace.com/pp/pp.cgi/0/518757384>

CP Short Name	OK	Short Description
XCA-2T	x	Adaption of the existing epSOS solution to temporarily support a pilot operation with two XCA transaction model (query/retrieve)
XDS-PEP	x	Adaption of the epSOS access control means and policy enforcement in a decoupled two-transaction model with an additional metadata query from the XDS repository towards the XDS registry for attribute provision
BPPC-CS	x	Provision of a courtesy web-service to facilitate the creation of country-B “ad-hoc” patient privacy consents at a point of care through generic country-B client systems (portals, fat clients)
XDS-MD	x	Adaption of the set of required (mandatory) metadata for compiling a XDS submission: inclusion of ()
XDS-CC	x	Reassignment of the XDS metadata attribute “ <i>HealthCareFacilityType</i> ” to transport the country-code of the requesting country (country-B) for enhanced addressing potential and access control
CM-CC/LC	x	Inclusion of the language-code in the PoC-Registration data set and schema for facilitating transformation/translation tasks in multi-lingual countries
BPPC-BF	x	Adaption of the epSOS consents specification to include “best-before/no-valid-before” patient privacy consents time constraints by utilising and re-assigning the BPPC “service-starts/ends” attributes when transmitting a consents from country-B to country-A
CDA-ORGN	x	the epSOS-CDA has to be compiled and provided by the NI of the MS, hence is not build by any C-/Backbone-Component within the NCP
BPPC-CA	x	the interactions and transactions between NI of country-A and NCP-A with regards to the patient privacy consents are not specified in epSOS. For an adequate consents processing the minimal data and transactions are crucial. This change proposal shows possible alternatives for the MS and NI.
XCPD-PEP	x	for XCPD means, such as “ <i>findIdentitybyTraits</i> ”, a patient privacy consents is not foreseen by the specs as of now. In order to guarantee a proper access control for searching and identifying patients with no accompanying TRC, the policy enforcement is based on the role of the requester and potentially aligned with the

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

type of PoC (extracted from the PoC registration data)		
CP Short Name	OK	Short Description
CM-eCRTS	x	enable strong user authentication at eCRTS in order to safeguard the to be transmitted taxonomies
CM-IdT	x	enable the national configuration to facilitate new national identity traits with national extensions within the CCD ConfMan instead of national responsibility
Audit-SNMP	x	enable the provision of SNMP-traps for facilitating network-/systems management and the collection of operational data within the NCP
Audit-TLS	x	enable TLS safeguarded WS-connections between the subcomponents of the Audit Manager if operated on separate secure nodes/instances or in a highly distributed environment
Audit-WS	x	enable the AuditWriter subcomponent to receive relevant auditable messages from a zone outside the NCP security zone, such as an integrated portal-B running @ the HCP-O
BBC/epSOSSVC	x	extend the availability of formerly masked XDS/CDA metadata to be only accessible by the respective components in its realm for simpler and clearer service delivery – such as using healthCareFacilityType for the inclusion of a country code
BBC/LC	x	Use of language codes was initially aligned to ISO 639-1 codes that follow CC-LL, yet has been refined by using base OID for indicating country and language - formerly unknown
TM-ASSOC	x	Facilitate document consumers to be aware of relationship between CDA and PDF based on their CDA relationship (see Giorgios table in open issues). In order to be fully content neutral, the XDS MD (ebrim:assoc) must be adjusted accordingly.
TM-PDF/A	x	integrate validation functionality for a compliance check regarding a Member State provided PDF/A as one input object for the TM
CM-DZ	x	reserve baseline configuration space and national configuration drop-zone at country-code.epsos.eu in the directory “/centralservices/”
ALL-OID	x	substitute for missing OID assignment of the MS while preservice WP 3.4/WP3.5 assignments as specified in teh respective documents
ALL-Ji-WSi	x	facilitate complex deployment scenarios by enabling JAVA-API's and WS-Interfaces at all components that may be operated independently and/or separately in another secure node/instance

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010


DSig-SM	x	no processing of detached signatures by the certificate validator anymore
APP-CM	x	a new service fetches and delivers external configuration data into the secure nodes

Major Change Proposal Compilation

Major differences towards the HLDD are presented in this section individually in order to improve the readability of the delta.

Backbone Components

- XDS Submission meta-data, as defined in D3.4.2, are redefined and separated to support separation-of-concern:
 - CDA metadata exclusively for TM and semantic components
 - XDS metadata exclusively available to the backbone components / services
- ConsentManager introduced as a new component and web-service:
 - Consumable in country-B for facilitating the compilation of valid BPPC patient privacy consents
 - technology-transparent implementation and service delivery as a web-service to support thin-clients and enable integration into existing system without business-logic implementation burden
- backbone components interfaces are supporting Java-API, as well as a Web-Service Interface in order to enable more complex deployment scenarios and enable a seamless integration into existing national infrastructures
- due to a missing provision of an adapted IHE XCA profile, various work flows and transactions had to be adapted to temporarily support a 2-transaction scenario
- meta data adjustment and introduction of a new PIP (XDS registry), and an additional PDP-request in order to ensure a full policy enforcement in a 2-transaction scenario
- refinement of the BPPC meta data to fully support D3.6.2 requirement of the consents time validity (service starts/ends)
- redefinition of the D3.6.2 XDS meta data „*HealthCareFacilityType*“ holding the county-B country code now for simplified addressing and localisation functionalities
- WorkflowManager and ProtocolTerminator implemented implicitly and mapped onto the respective IHE transactions
- ProtocolTerminator is also dealing with SOAP security and SOAP stripping
- backbone components provide emergency TRC and signature functionality in case the Security Manager is unreachable

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

- a new subcomponent fetches and delivers externally stored common components local configuration data into a secure node and places the files in their respective location

Policy Manager

- this formerly common component is integrated into the backbone components:
 - PolicyManager features a highly standardised interface, so do industry products performing the very same tasks
 - the Policy Manager features little or no epSOS-specific functionality
 - easily replaceable by off-the-shelf products with Consent Manager
 - operating standard XACM policies and policy processing points
 - even as backbone component, Member States may configure the policies however required within their national domain
 - was surrendered in favour for the highly epSOS-specific Consent Manger
- access control role-mapping (HCP) is moved to this component since it is not defined in the semantic work packages

Audit Manager


- AuditWriter supports both, Java interface and WS-Interface in order to be able to intercept national auditable events with epSOS significance
- AuditWriter provides an interceptor that may compile SNMP traps for performance and operational data retrieve
- AuditWriter support log4j interface for simpler operational data collection and system state evaluation
- AuditWriter/Syslog client implements an audit cache for data safety in case that the syslog server is not accepting auditable events
- MonitoringManager implemented by a using a static stored within the AuditRepository procedure for full anonymisation-assurance and operation within the secure node

Security Manager

- in contrast to many specification document, the certificate validator of the security manager is not processing detached signatures anymore since there are none in epSOS anymore
- detached signatures of SOAP and SAML assertions are processed and validated by the respective framework and not by the certificate validator of the security manager

Transformation Manager and TSAM

- the original CDA document is to be provided by the national infrastructure and not initially build by the transformation Manager anymore

	D3.9.1: Appendix A FET Solution	Document Short name:	D3.9.1 Appendix A
		Version:	1.0
	WP3.9, JWG 3.8/3.9: Detail Specifications	Date:	01/10/2010

- the Transformation Manager is not processing and/or regulating the meta data anymore
- the Transformation Manager is validating the included PDF/A for compliance, if available
- as already mentioned regarding the XDS meta data, the semantic components are exclusively accessing the CDA meta data and no XDS meta data for full content-neutrality