Consent2Share Software Architecture

Version 1.0

17 December 2013

Revision History

Name	Description	Date
Joel Amoussou	Architecture overview of the Consent2Share v1.0 Product	12/11/2013

Table of Contents

Consent2Share Software Architecture Overview

1.Objectives

The objectives of this document are the following:

- Provide an executive overview of the Consent2Share software.
- List and explain the capabilities available in the Consent2Share software product.
- Explain the design and architectural principles that were used in building Consent2Share to ensure a high level of modularity, reusability, maintainability, testability, security, usability, scalability, and performance.
- Explain how key healthcare interoperability standards are implemented to allow Consent2Share to integrate with other software components within an HIE.

2.Executive Overview

2.1 The Goals of the Consent2Share Project

Research indicates that patients desire fine-grained control over the kind of health information they want to share, with whom, and for what purpose. Furthermore, the current paper-based consenting processes used by healthcare providers have several limitations. Paper-based patient consent directives are not executable even when they are scanned for electronic storage. Patients cannot easily access, track, update, and revoke these consents remotely.

The Consent2Share Patient Consent Management (PCM) system allows patients to capture, electronically sign, and revoke their consent directives anywhere and anytime. In addition, patients can receive data access requests and retrieve an audit trail of all data access events.

The Consent2Share Access Control System (ACS) includes a Policy Enforcement Point (PEP), a Policy Decision Point (PDP), and a Data Segmentation Engine for enforcing patient consent directives and applying privacy metadata to clinical documents. Data Segmentation consists in not only redacting information based on the patient's privacy preferences but also in applying privacy metadata tags to the clinical document. Examples of metadata include: confidentiality, applicable privacy law, refrain policy, and obligation policy.

2.2 Technical Overview of the Patient Consent Management (PCM) System

The architecture of Consent2Share PCM system is based on the Domain Driven Design (DDD) methodology to achieve a high level of modularity, loose-coupling, testability, and extensibility. The Consent2Share PCM system is built upon the open-source Java-based Spring Framework. Consent2Share PCM uses Spring Framework projects such as Spring Roo, Spring Dependency Injection (DI), Spring Aspect Oriented Programming (AOP), Spring MVC, Spring Security, and Spring Data for JPA. Data is persisted in a MySQL database.

The components of Consent2Share PCM interact with external systems through a set of well-defined SOAP-based and RESTful web service interfaces. In addition, Consent2Share uses Spring Integration and the RabbitMQ message broker to integrate the PCM system with other enterprise systems that need to be aware of patient consent directives (e.g., patient permissions for scientific research, clinical trials, and biobanks).

The user interface is based on Twitter Bootstrap, an open source Responsive Web Design (RWD) Framework using CSS3, HTML5, and JQuery to achieve cross-browser and cross-device capabilities. Consent2Share uses the Thymeleaf server-side HTML5 templating engine for rendering.

Consent2Share has undergone application-level penetration testing and can be deployed on-premise, in the cloud, or as part of a Health Information Exchange (HIE) network infrastructure.

2.3 Technical Overview of the Access Control System (ACS) System

The ACS system can segment clinical documents conforming to the HITSP C32 specification which has been selected as a Meaningful Use Stage 1 requirement. The consent documents created by the PCM system are transformed into eXtensible Access Control Markup Language (XACML) format for their automated execution by the ACS at data exchange time. Consent2Share supports the Direct Project protocol for push-based exchanges and the IHE XDS.b protocol for pull-based exchanges.

Consent2Share uses the Drools Business Rules Management System (BRMS) to allow health privacy experts to create and manage business rules for privacy metadata tagging. The following is a high level architecture diagram of Consent2Share:

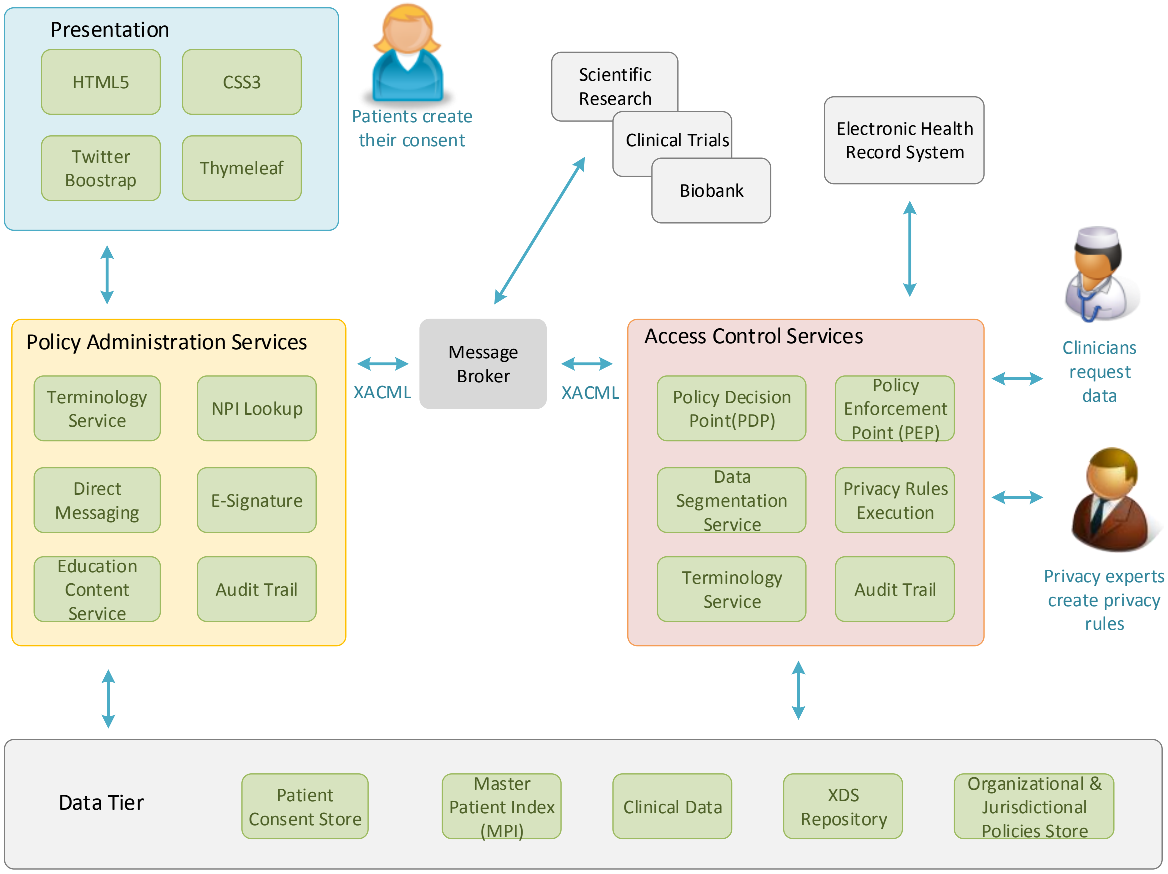


Figure 1. Architecture overview of the main components of the Consent2Share system

3.Detailed Description of the Functionalities of the Consent2Share Product

3.1 Functionalities of the Consent2Share Patient Consent Management (PCM) System

Patients can create an account, login, and use the system anywhere and anytime. To enable a seamless integration of Consent2Share into existing consent collection workflows, it is also possible to make the system available to patients at the admission or registration desk (for example on a tablet) where patient consent is traditionally collected. Alternatively, the Consent2Share PCM can be integrated into an existing healthcare provider patient portal, for example through a single sign-on mechanism. The PCM has been designed with a Responsive Web Design (RWD) approach to provide cross-browser and cross-device capabilities.

The following are the capabilities of the Consent2Share PCM system available to patients:

a) Create an account including a username and password.

b) Sign in to the Consent2Share application.

c) Create a detailed profile including demographic information.

d) Search and select healthcare providers (based on criteria such as name, telephone number, state, zip code, gender, and specialty) with whom they want to share their healthcare information.

e) Upload their C32 clinical document.

f) Create their consent documents by specifying the kind of health information they want to share, with whom, and for what purpose. There are several configurable categories for specifying the kind of health information to be shared including: sensitivity, clinical document types, clinical document section types, and specific clinical document entries. These categories use value sets from existing standards such as HITSP C32, the HL7 CCDA, the HL7 Security and Privacy Value Set, the ONC Data Segmentation for Privacy Implementation Guide, and clinical concept codes from standard vocabularies like SNOMED, ICD9, LOINC, and RxNorm.

g) View their consent document in PDF format.

h) Electronically sign their consent document using the Adobe EchoSign service.

i) Revoke their consent document.

j) Communicate with their provider through the Direct Project Secure SMTP protocol.

k) View an audit trail of all activities in the system.

The Consent2Share PCM system provides administration capabilities that are available only to designated healthcare provider personnel with Administrator privileges (typically the admission clerk). The following are the capabilities of the Consent2Share PCM system available to Administrators:

a) Search for a patient based on criteria such as first and last name.

b) Create a consent document for a patient who is not able to create their own consent. However, the consent can only be signed by the patient or her legal representative.

3.2 Functionalities of the Consent2Share Access Control Services (ACS)

The ACS Data Segmentation component can segment clinical documents conforming to the HITSP C32 specification which has been selected as a Meaningful Use Stage 1 requirement. Consent2Share conforms to the ONC Data Segmentation for Privacy (DS4P) Implementation Guide. The following are the capabilities of the Consent2Share ACS system:

a) The consent documents created by patients in the PCM system are transformed into XACML format and wrapped into CDA R2 for Consent Directives documents. For pull-based exchanges conforming to the XDS.b protocol, the CDA R2 consent document is exported from the PCM system into an XDS.b Repository through a `ProvideAndRegister` operation. To keep the consent documents in the XDS.b repository up-to-date at all time, a publish-subscribe integration pattern has been implemented between the PCM system and the XDS.b Repository based on the Spring Integration framework and the RabbitMQ message broker.

b) The PCM can integrate with an existing Master Patient Index (MPI) to obtain demographics information and a global identifier (EID) for the patient within the XDS affinity domain by using the IHE Patient Identity Cross Reference (PIX) specification.

c) For pull-based exchanges conforming to the XDS.b protocol, when a `DocQuery` or a `DocRetrieve` request is received, a policy check is performed by retrieving the consent document from the XDS.b repository and executing the policy with a XACML Policy Decision Point (PDP). The PDP can return a “deny” or “permit” response optionally with some “obligations”. These “obligations” are used to capture fine-grained patient privacy preferences such as the kind of health information the patient does not want to share. These “obligations” become directives for the data segmentation engine for redacting information out of the patient’s medical record.

d) For push-based scenarios based on the Direct Protocol, before the clinical document is sent via secure email, the ACS is invoked to obtain patient permission through a request to the XACML Policy Decision Point (PDP). If permission is granted with some obligations, then these obligations are used to segment the data before sending it through secure email.

e) The ACS system also provides the ability for privacy experts to create business rules for authoring privacy metadata tagging rules to be applied to clinical documents as part of the segmentation process. For example, the presence of sensitive health information like drug abuse in the clinical document can fire a metadata tagging rule which will instruct the segmentation engine to add the following tags to the clinical document: a confidentiality value of “`R`” (restricted), a privacy law of “`42 CFR Part 2`”, and a refrain policy of “`non-redisclosure`”. The rules authoring interface is based on Drools Guvnor.

f) The tagging rules are written based on concept codes from standard vocabularies like SNOMED, ICD9, LOINC, and RxNorm. The ACS can use a terminology service to lookup concept codes, map concept codes across vocabularies, or obtain relationships between concept codes such as subsumption relationships. A terminology service can also be used to create and manage sensitive value sets.

g) The segmentation engine is the core of the ACS. It receives segmentation directives from two sources. The first source is the XACML response from the PDP which contains a “deny” or “permit” decision optionally with some obligations. The obligations are translated into directives to redact information from the patient’s medical record. The second source of segmentation directives are the metadata tagging rules which are fired based on the presence of certain clinical concept codes (clinical facts) in the patient’s C32 document.

h) The Consent2Share ACS can also automatically enforce non-redisclosure policies based on the non-redisclosure metadata tag attached to a clinical document.

4.The Design of Consent2Share PCM

The architecture of Consent2Share PCM system is based on the Domain Driven Design (DDD) patterns language. DDD is appropriate for highly complex domains such as healthcare. It imposes a disciplined approach to software architecture based on object-oriented design patterns, application layering, the Separation of Concerns (SoC), and strategic design.

4.1 DDD Application Layering Principles

Per DDD principles, Consent2Share is designed with the following four layers: the presentation layer, the application layer, the domain layer, and the infrastructure layer (see Figure 2 on the next page). The goal is to avoid the following anti-patterns typically found in many software systems: a fat application layer, an anemic domain model, and a tangled mess in general. The diagram on the next page describes the four layers recommended by DDD and implemented in Consent2Share.

The following key principles are applied to enable a loose coupling of components:

- Repository interfaces (repository contracts) are in the domain layer, but their implementations are in the infrastructure layer to allow “Persistence Ignorance”. This is based on the “Separated Interface Pattern” defined by Martin Fowler.
- Both the interface and implementation of factories (when required) are in the domain layer.
- Dependencies are injected using dependency injection. Consent2Share uses the open source Sping Framework as its Inversion of Control (IoC) container.
- The application layer is a thin layer responsible for tasks coordination: it should not contain domain business logic. It mediates between the presentation layer and the domain layer through Data Transfer Objects (DTOs).

4.2 The Domain Layer

The domain layer contains the domain model including the domain business logic and business rules. Data validation, domain events, and domain event handling capabilities are supported in the domain layer as well. The domain layer has a modular design. Examples of modules include the following:

- The Patient Module.
- The Provider Module
- The Clinical Data Module.
- The Education Materials Module.
- The Consent Module.

An existing module can be customized or a new one can be added. The domain layer is decoupled from all the other layers and in particular from the infrastructure layer: it is “persistence ignorant” which means that you can plug in a different persistence mechanism.

4.3 The Presentation Layer

The view layer is implemented through Spring MVC controllers and a combination of Twitter Bootstrap and the Thymeleaf server-side templating engine.

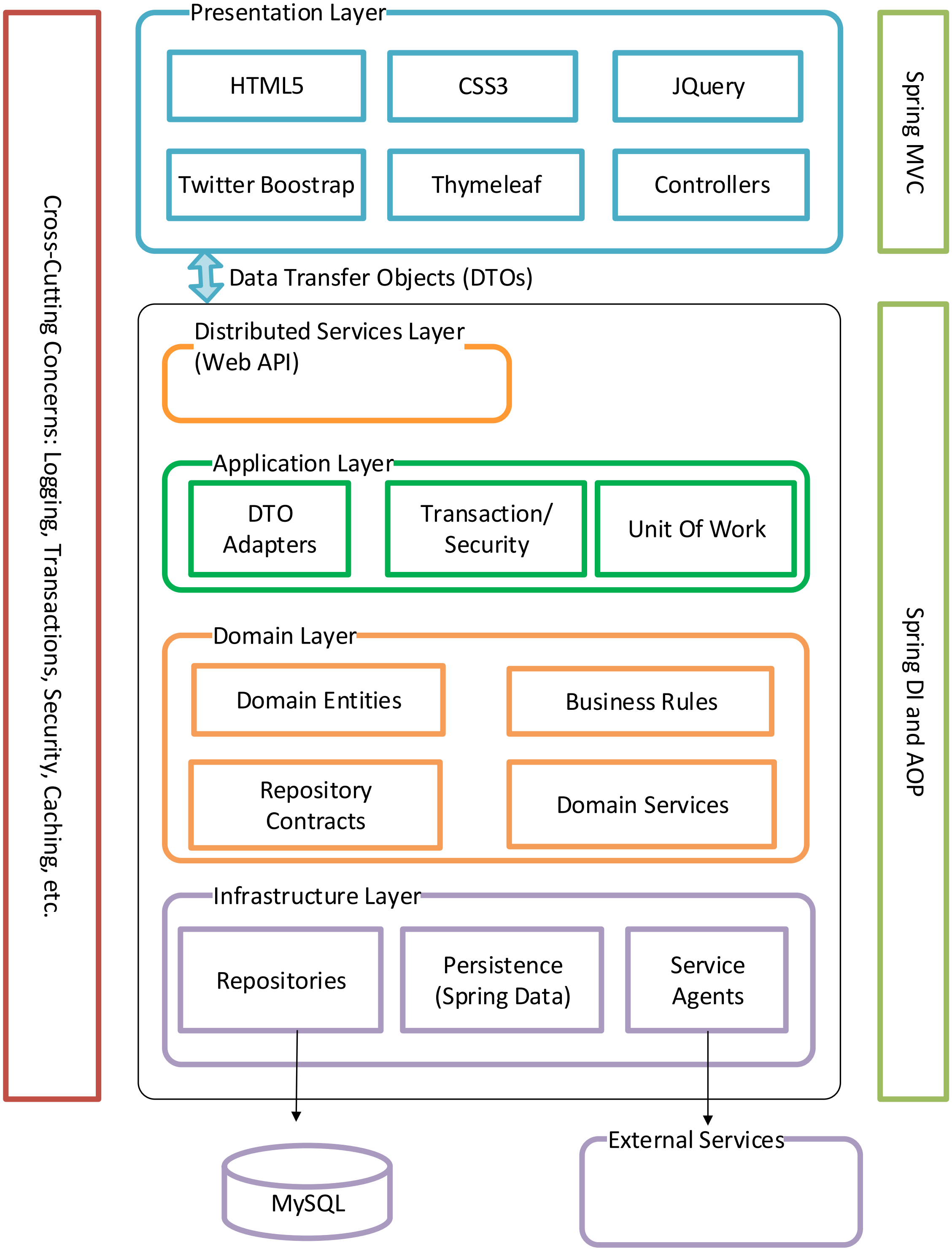


Figure 2. Architectural Layers of the Consent2Share PCM system

4.4 The Infrastructure Layer

Object-Relational Mapping (ORM) is performed based on the Spring Data repository abstraction using the Hibernate implementation of JPA 2.0. The database used is MySQL. Using Hibernate configuration one can easily replace MySQL with another database such as: MS SQL Server, Oracle, DB2, etc.

The infrastructure layer also provides the ability to communicate with external services. Examples include sending email notifications, converting an entity such as a consent object into PDF documents, or sending a request for electronic signature to a 3rd party service like Adobe EchoSign.

4.5 The Application Layer

The application layer in Consent2Share is a thin layer. It is responsible for tasks coordination and would be the right place to implement workflow capabilities. It mediates between the presentation and the domain layers through data transfer objects (DTOs). An open source library called ModelMapper is used to greatly simplify the generation of DTOs from the domain objects.

5.The Design of Consent2Share ACS

5.1 High level overview

This section describes how the Consent2Share ACS system interact with other components when the XDS.b protocol is used to exchange clinical data between healthcare providers within an HIE environment. The key components within such an environment include:

- The Master Patient Index (MPI).
- The XDS.b Repository.
- The XDS.b Registry.
- The Policy Enforcement Point (PDP).
- The XACML Policy Decision Point (PDP).
- The Business Rules Execution Engine based on Drools Expert.
- The Business Rules Authoring and Management system based on Drools Guvnor.
- The Data Segmentation Engine.
- The Terminology Service.

The following diagram shows a high level overview of the various components interacting with the ACS within an HIE using a centralized XDS.b Repository. Note that the ACS intercepts all requests coming into the XDS.b Repository such as the DocQuery and DocRetrieve operations.

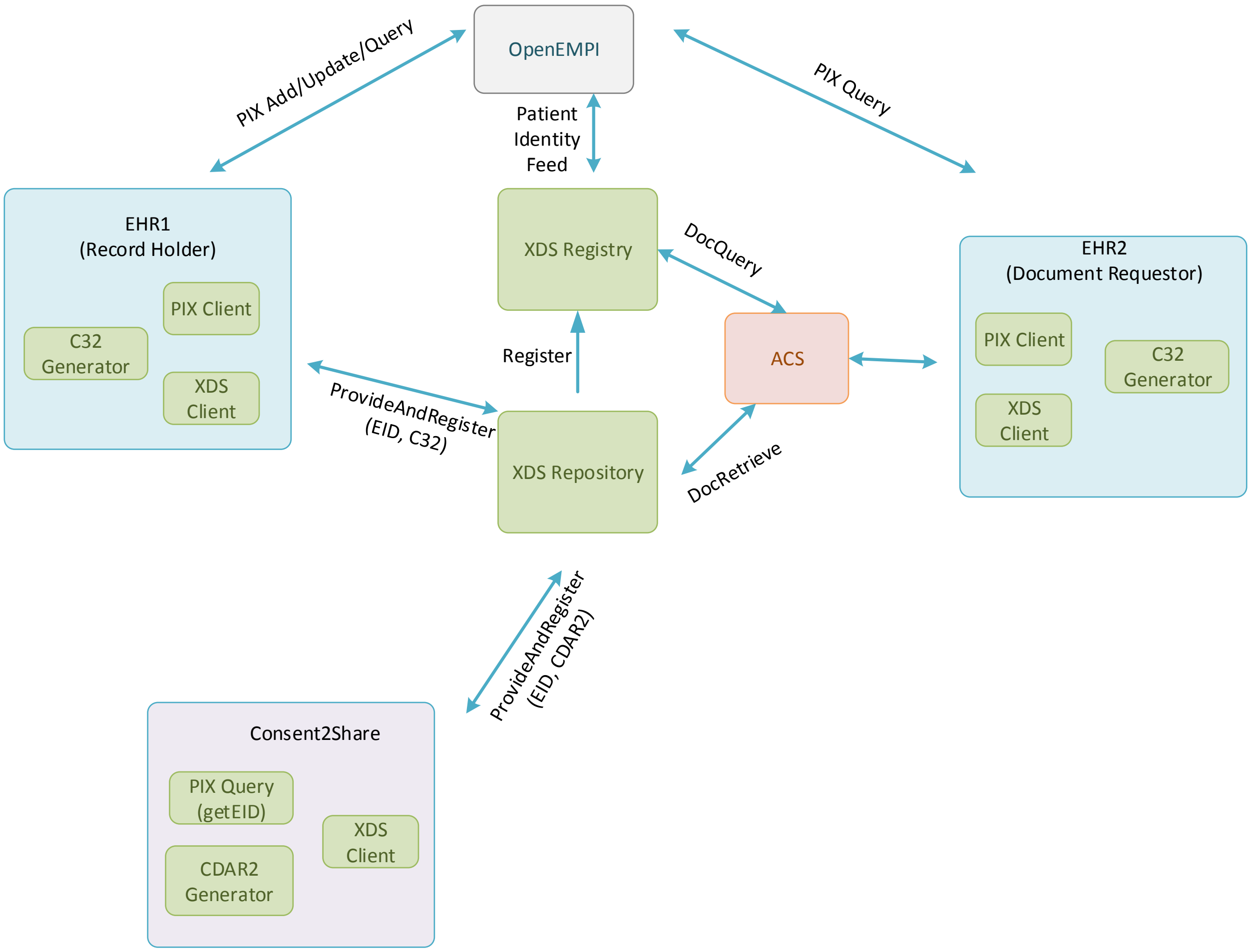


Figure 3. Document Exchange based on the XDS.b Protocol

5.2 The Master Patient Index (MPI)

The role of the MPI is to store patient demographics information and provide a global identifier (EID) for a patient within the XDS.b affinity domain. The MPI exposes web service operations like `PIX Add`, `PIX Update`, and `PIX Query` to add/update a patient record to the MPI and retrieve her EID respectively. The EID is subsequently used to do the following:

1. Establish a patient identity record in the XDS.b registry using the Patient Identity Feed (ITI-8) or Patient Identity Feed HL7v3 (ITI-44) operations.
2. Publish patient clinical documents and consent documents (from the EHR and from Consent2Share PCM respectively) to the XDS.b repository using the EID as a unique patient identifier within the XDS Affinity Domain.
3. Query and retrieve documents from the XDS.b registry and repository respectively using the EID as a unique patient identifier within the XDS Affinity Domain.

The Consent2Share PCM can use the PIX web services to query and retrieve patient demographics information and the EID from the MPI.

5.3 The XDS.b Repository and Registry

The XDS.b Repository and Registry are used to store clinical and consent documents and their associated metadata (see Figure 4 below). They expose a number of SOAP-based web service interfaces such as `ProvideAndRegister`, `Register`, `DocQuery`, and `DocRetrieve`. The Consent2Share PCM system uses the `ProvideAndRegister` operation to publish consent document into the XDS.b Repository. Service Consumers use the `DocQuery` web service to query the XDS.b Registry and the `DocRetrieve` web service to retrieve the actual clinical or consent document from the XDS.b Repository.

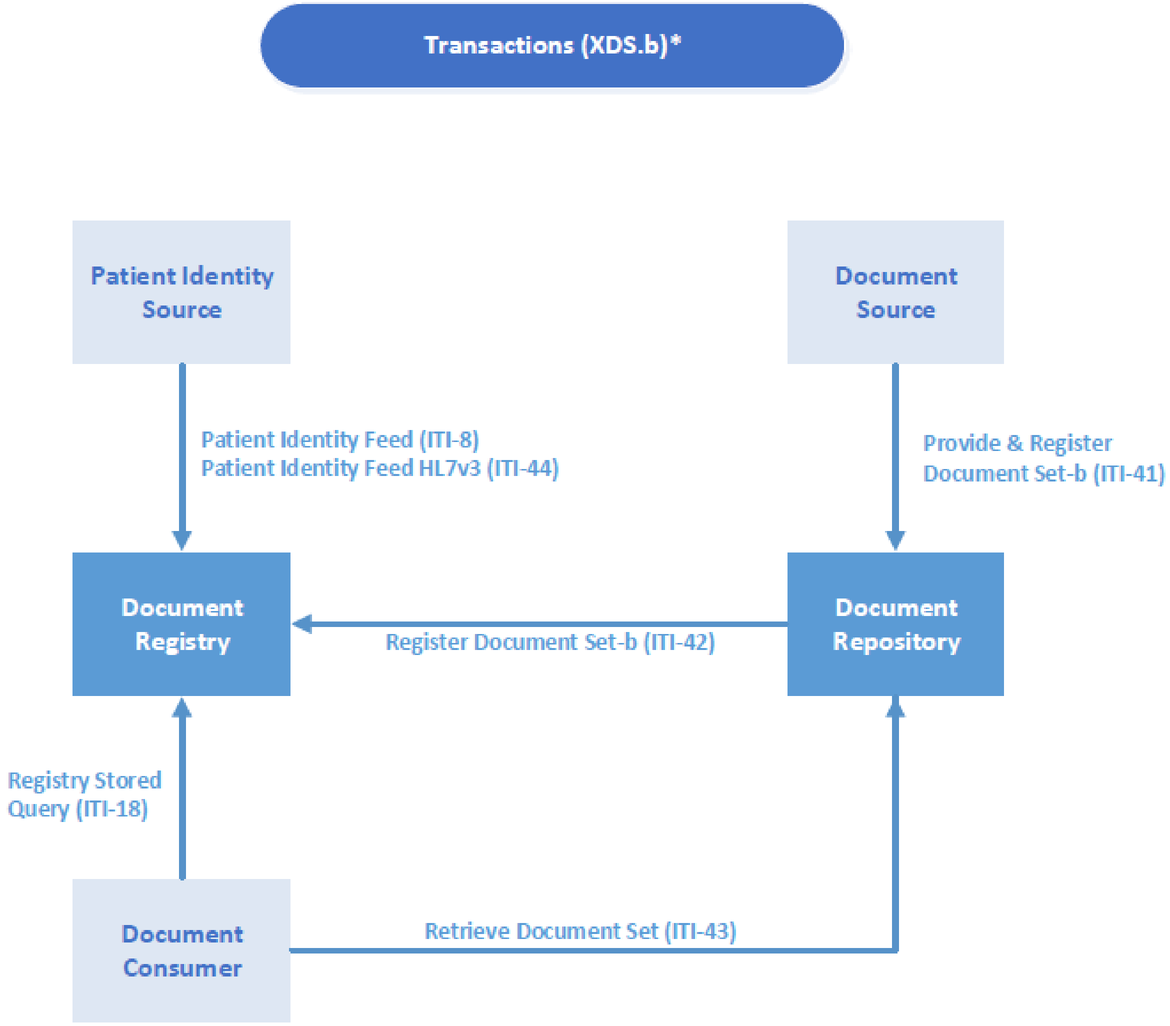

Transactions (XDS.b)*
Patient Identity Source
Document Source
Patient Identity Feed (ITI-8)
Patient Identity Feed HL7v3 (ITI-44)
Provide & Register Document Set-b (ITI-41)
Document Registry
Register Document Set-b (ITI-42)
Document Repository
Registry Stored Query (ITI-18)
Document Consumer
Retrieve Document Set (ITI-43)

Figure 4. Document Exchange based on the XDS.b Protocol

5.4 The Policy Enforcement Point (PEP)

The PEP intercepts all incoming requests into the XDS.b Repository and Registry such as the DocQuery and the DocRetrieve operations (see figures 5 and 6 below). Before attempting to respond to these requests, a policy check is first performed to determine if there is patient permission to exchange the data. The policy check is performed by retrieving the patient consent policy in XACML format from the XDS Repository.

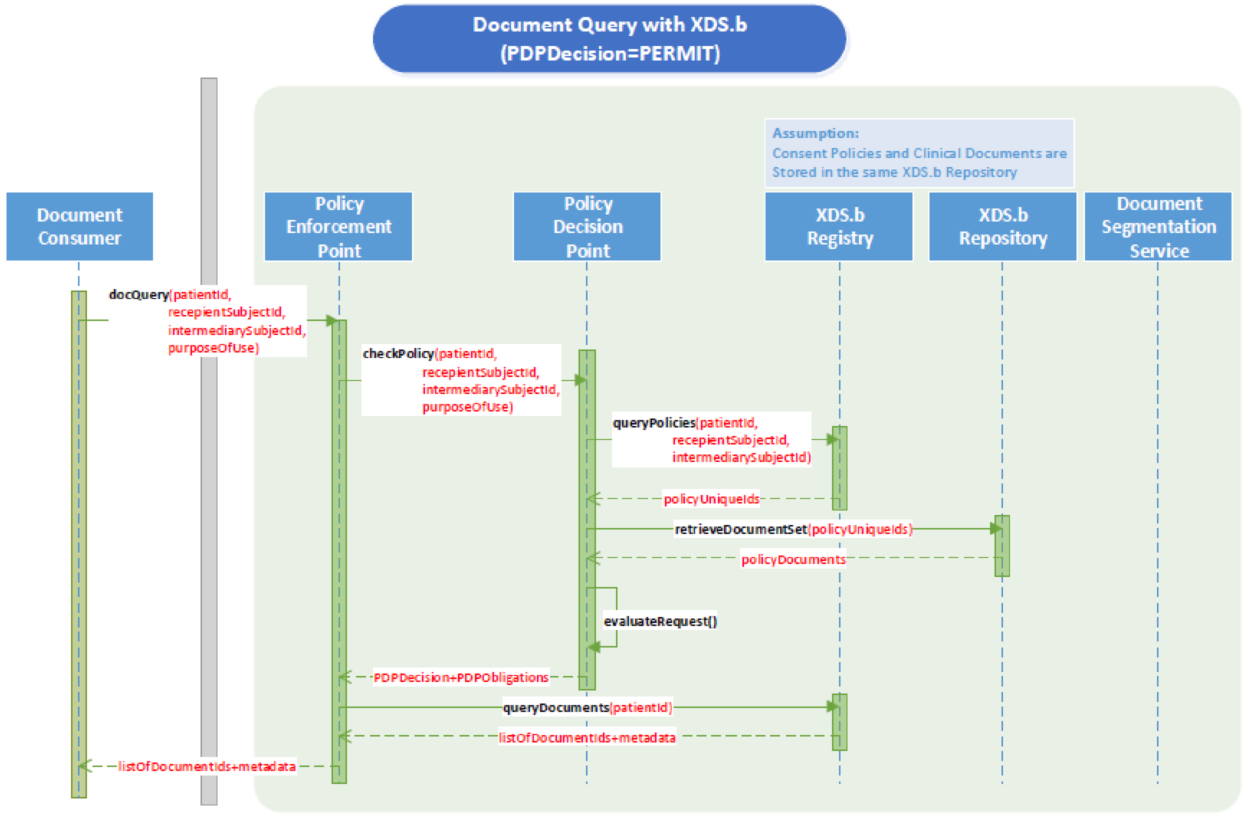

Document Query with XDS.b
(PDPDecision=PERMIT)
Assumption:
Consent Policies and Clinical Documents are Stored in the same XDS.b Repository
Document Consumer
Policy Enforcement Point
Policy Decision Point
XDS.b Registry
XDS.b Repository
Document Segmentation Service
docQuery(patientId, recepientSubjectId, intermediarySubjectId, purposeOfUse)
checkPolicy(patientId, recepientSubjectId, intermediarySubjectId, purposeOfUse)
queryPolicies(patientId, recepientSubjectId, intermediarySubjectId)
policyUniqueIds
retrieveDocumentSet(policyUniqueIds)
policyDocuments
evaluateRequest()
PDPDecision+PDPObligations
queryDocuments(patientId)
listOfDocumentIds+metadata
listOfDocumentIds+metadata

Figure 5. DocQuery Orchestration

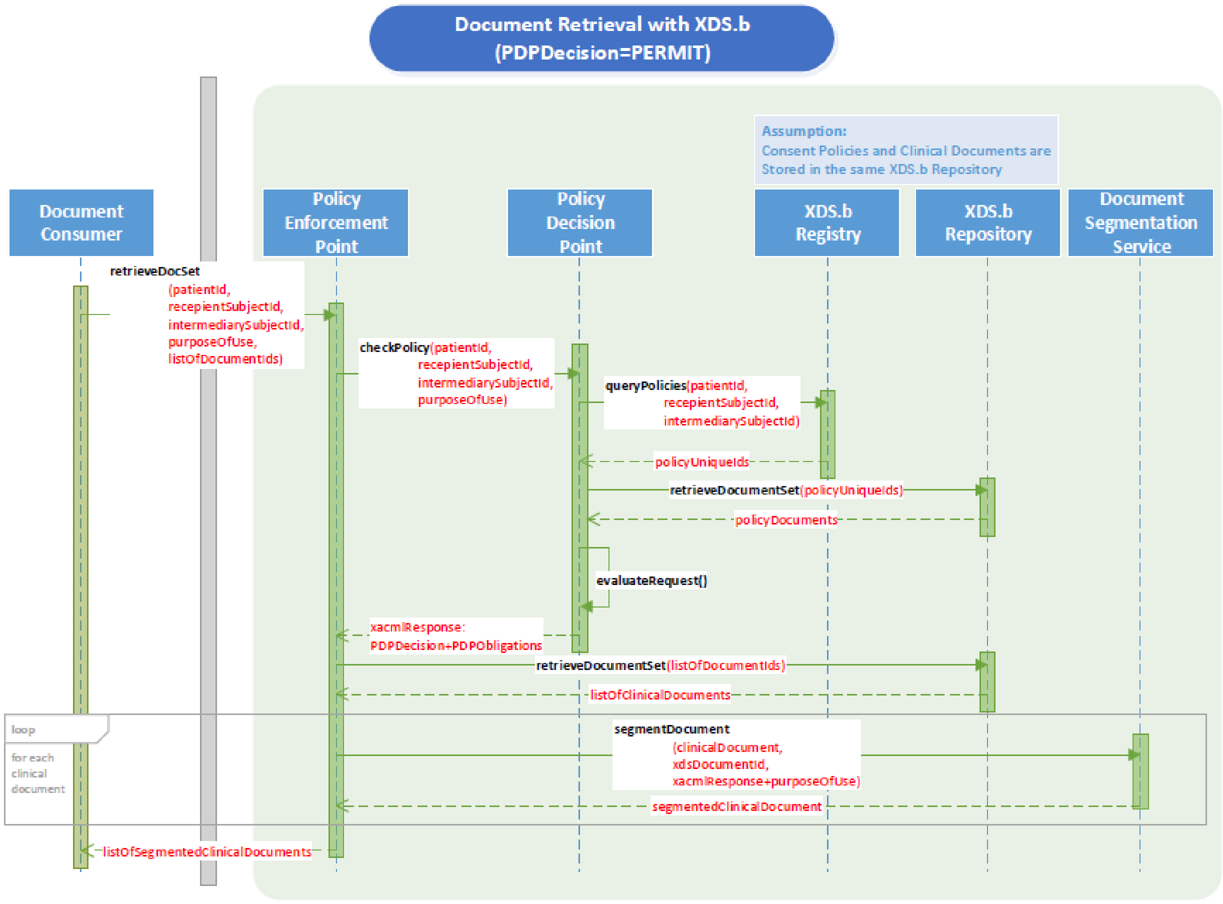

Document Retrieval with XDS.b
(PDPDecision=PERMIT)
Assumption:
Consent Policies and Clinical Documents are Stored in the same XDS.b Repository
Document Consumer
Policy Enforcement Point
Policy Decision Point
XDS.b Registry
XDS.b Repository
Document Segmentation Service
retrieveDocSet (patientId, recepientSubjectId, intermediarySubjectId, purposeOfUse, listOfDocumentIds)
checkPolicy(patientId, recepientSubjectId, intermediarySubjectId, purposeOfUse)
queryPolicies(patientId, recepientSubjectId, intermediarySubjectId)
policyUniqueIds
retrieveDocumentSet(policyUniqueIds)
policyDocuments
evaluateRequest()
xacmlResponse: PDPDecision+PDPObligations
retrieveDocumentSet(listOfDocumentIds)
listOfClinicalDocuments
loop
for each clinical document
segmentDocument (clinicalDocument, xdsDocumentId, xacmlResponse+purposeOfUse)
segmentedClinicalDocument
listOfSegmentedClinicalDocuments

Figure 6: DocRetrieve Orchestration

5.5 The XACML Policy Decision Point (PDP)

The XACML PDP evaluates the XACML policy representing the patient consent and returns a “Deny” or “Permit” decision optionally with some obligations. These obligations are used as segmentation directives to redact certain information out of the patient’s medical record based on her privacy preferences. The XACML PDP can also be used to evaluate organizational and jurisdictional policies.

5.6 The Business Rules Authoring and Management system based on Drools Guvnor

This is used to create, test, and version privacy metadata tagging rules (see Figure 7).

5.7 The Business Rules Execution Service based on Drools Expert

The metadata tagging rules are executing at runtime by the business rules execution service based on Drools Expert (see Figure 6).

5.8 The Data Segmentation Engine

The Data Segmentation Engine include the following sub-components (see Figure 7):

5.8.1 Document Fact Model Extractor

The Document Fact Model Extractor uses a combination of XPath, XSLT, and JAXB to extract clinical facts from the C32 documents to be exchanged. These clinical facts are clinical concept codes consisting of a `code`, a `displayName`, and a `code system identifier`. They are used as the fact model for the privacy metadata tagging rules.

5.8.2 Document Redactor

The Document Redactor removes nodes (typically element and text nodes) from the C32 document based on the obligations returned by the XACM PDP. It uses a combination of XPath, XSLT, and JAXB.

5.8.3 Document Tagger

The Document Tagger applies privacy metadata tags to the C32 document. The tagging directives are produced when one or more tagging rules are fired based on the presence of certain clinical facts as specified in the rule. It uses a combination of XPath, XSLT, and JAXB.

5.8.4 Metadata Generator

The Metadata Generator can generate the metadata required by specifications such as XDS.b or XDM. It uses a combination of XPath, XSLT, and JAXB

5.8.5 Document Masker

The Document Masker can mask (encrypt) specific nodes on the C32 document (as opposed to encrypting the whole document). It uses a combination of XPath, XSLT, JAXB, and the Apache Santurio XML Security framework.

5.8.6 Document Encrypter

The Document Encrypter encrypts the whole document. It uses the Apache Santurio XML Security framework.

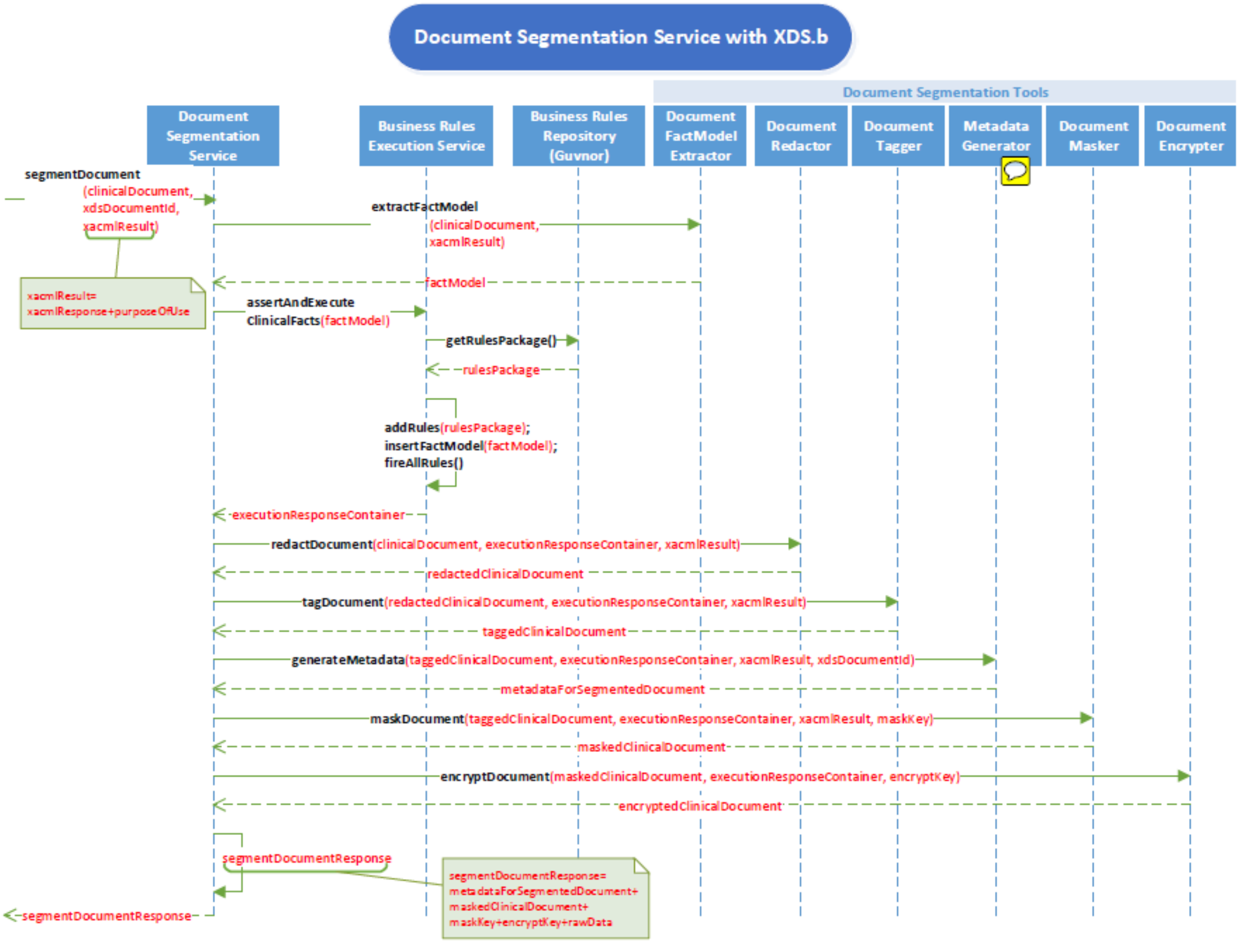
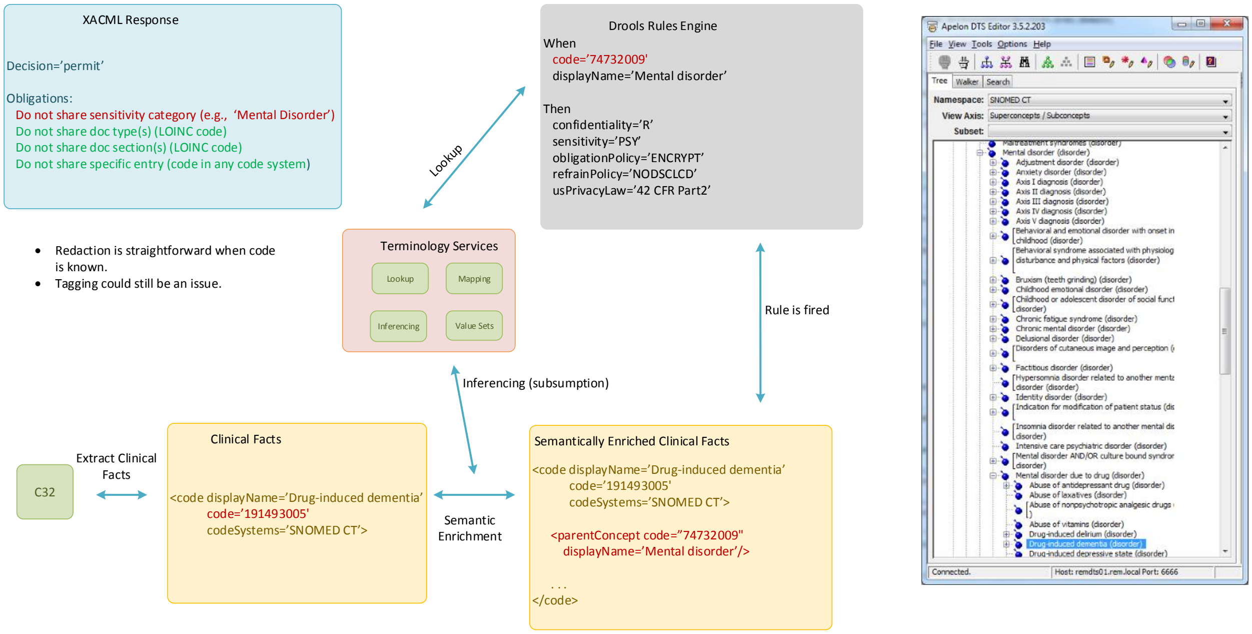

Document Segmentation Service with XDS.b
Document Segmentation Tools
Document Segmentation Service
Business Rules Execution Service
Business Rules Repository (Guvnor)
Document FactModel Extractor
Document Redactor
Document Tagger
Metadata Generator
Document Masker
Document Encrypter
segmentDocument (clinicalDocument, xdsDocumentId, xacmlResult)
xacmlResult= xacmlResponse+purposeOfUse
extractFactModel (clinicalDocument, xacmlResult)
factModel
assertAndExecute ClinicalFacts(factModel)
getRulesPackage()
rulesPackage
addRules(rulesPackage);
insertFactModel(factModel);
fireAllRules()
executionResponseContainer
redactDocument(clinicalDocument, executionResponseContainer, xacmlResult)
redactedClinicalDocument
tagDocument(redactedClinicalDocument, executionResponseContainer, xacmlResult)
taggedClinicalDocument
generateMetadata(taggedClinicalDocument, executionResponseContainer, xacmlResult, xdsDocumentId)
metadataForSegmentedDocument
maskDocument(taggedClinicalDocument, executionResponseContainer, xacmlResult, maskKey)
maskedClinicalDocument
encryptDocument(maskedClinicalDocument, executionResponseContainer, encryptKey)
encryptedClinicalDocument
segmentDocumentResponse
segmentDocumentResponse= metadataForSegmentedDocument+ maskedClinicalDocument+ maskKey+encryptKey+rawData
segmentDocumentResponse

Figure 7. The Data Segmentation Engine

5.9 The Terminology Service

The ACS can use a terminology service to lookup concept codes, map concept codes across vocabularies, or obtain relationships between concept codes such as subsumption relationships (see Figure 8). A terminology service can also be used to create and manage sensitive value sets.


XACML Response
Terminology Services
Lookup
Mapping
Inferencing
Value Sets
Lookup
Drools Rules Engine
Rule is fired
Inferencing (subsumption)
C32
Extract Clinical Facts
Clinical Facts
Semantic Enrichment
Semantically Enriched Clinical Facts
Apelon DTS Editor 3.5.2.203
File View Tools Options Help
Tree Walker Search
Maltreatment syndromes (disorder)
Mental disorder (disorder)
Adjustment disorder (disorder)
Anxiety disorder (disorder)
Axis I diagnosis (disorder)
Axis II diagnosis (disorder)
Axis III diagnosis (disorder)
Axis IV diagnosis (disorder)
Axis V diagnosis (disorder)
Behavioral and emotional disorder with onset in childhood (disorder)
Behavioral syndrome associated with physiolog disturbance and physical factors (disorder)
Bruxism (teeth grinding) (disorder)
Childhood emotional disorder (disorder)
Childhood or adolescent disorder of social funct disorder)
Chronic fatigue syndrome (disorder)
Chronic mental disorder (disorder)
Delusional disorder (disorder)
Disorders of cutaneous image and perception (
Factitious disorder (disorder)
Hypersomnia disorder related to another menta disorder (disorder)
Identity disorder (disorder)
Indication for modification of patient status (dis
Insomnia disorder related to another mental dis disorder)
Intensive care psychiatric disorder (disorder)
Mental disorder AND/OR culture bound syndror disorder)
Mental disorder due to drug (disorder)
Abuse of antidepressant drug (disorder)
Abuse of laxatives (disorder)
Abuse of nonpsychotropic analgesic drugs)
Abuse of vitamins (disorder)
Drug-induced delirium (disorder)
Drug-induced dementia (disorder)
Drug-induced depressive state (disorder)

Decision='permit'

Obligations:
- Do not share sensitivity category (e.g., 'Mental Disorder')
- Do not share doc type(s) (LOINC code)
- Do not share doc section(s) (LOINC code)
- Do not share specific entry (code in any code system)

- Redaction is straightforward when code is known.
- Tagging could still be an issue.

```
When
  code='74732009'
  displayName='Mental disorder'

Then
  confidentiality='R'
  sensitivity='PSY'
  obligationPolicy='ENCRYPT'
  refrainPolicy='NODSCLCD'
  usPrivacyLaw='42 CFR Part2'
```

```
<code displayName='Drug-induced dementia'
      code='191493005'
      codeSystems='SNOMED CT'>
```

```
<code displayName='Drug-induced dementia'
      code='191493005'
      codeSystems='SNOMED CT'>

   <parentConcept code="74732009"
     displayName='Mental disorder'/>

   . . .
</code>
```

Namespace: SNOMED CT
View Axis: Superconcepts / Subconcepts
Subset:

Connected. Host: remdts01.rem.local Port: 6666

Figure 8. Using a terminology service to manage value sets

5.10 Exchanging Documents via the Direct Protocol

The following diagram represents how document exchange is orchestrated using the Direct Protocol:

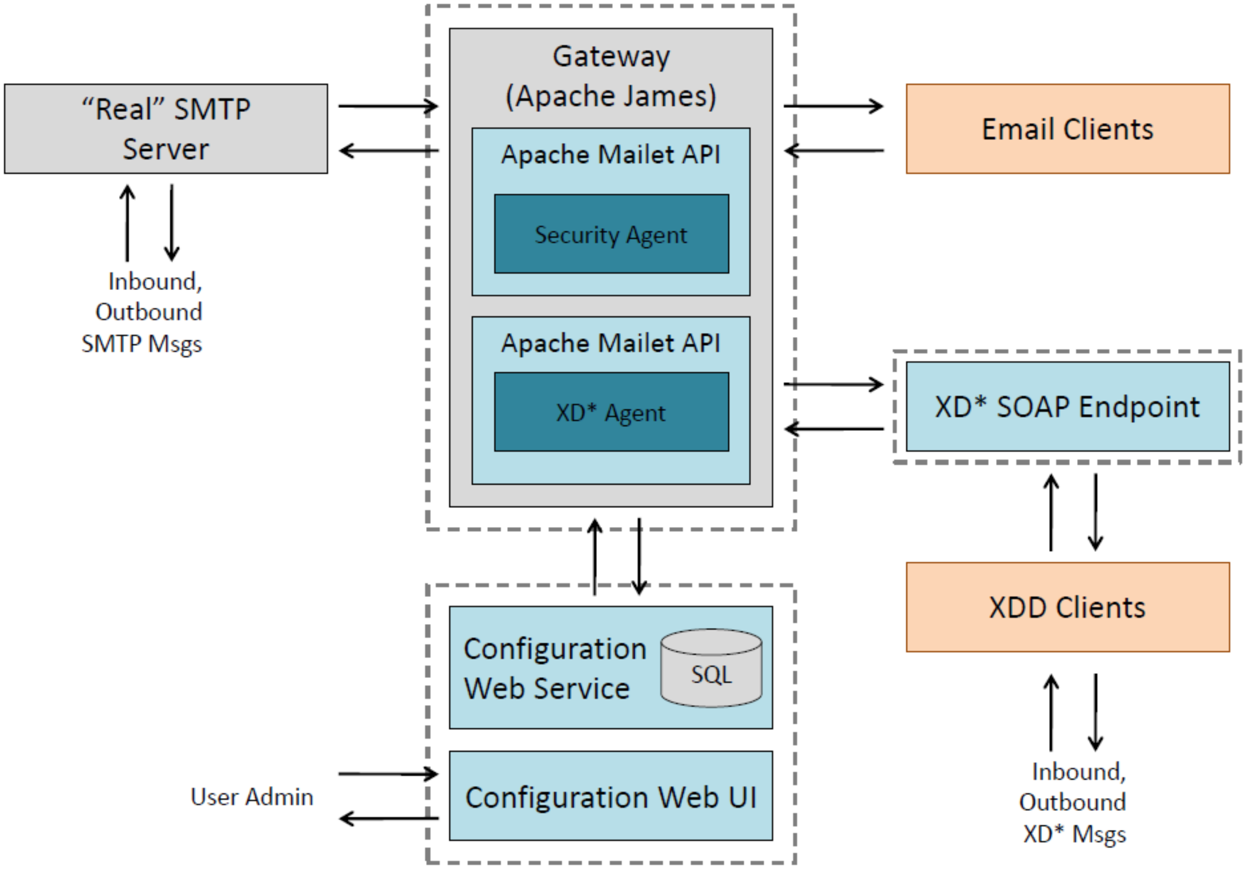


Figure 9. Diagram from the Direct Gateway documentation

6.Usability

The user experience in Consent2Share has undergone usability testing using the System Usability Score (SUS) methodology.

7.Healthcare IT Interoperability Standards

Consent2Share has been designed to implement key healthcare care IT interoperability standards such as:

- The HITSP C32 specification which is a requirement for Meaningful Use Stage 1 certification.
- The HL7 CDA R2 Implementation Guide for Consent Directives.
- The IHE Cross-Enterprise Document Sharing (XDS.b) protocol.
- The IHE Cross-Community Access (XCA).
- The IHE Patient Identity Cross Reference (PIX) specification.

- The IHE Audit Trail and Node Authentication (ATNA) specification.
- The IHE Cross-Enterprise User Assertions (XUA) specification.
- The ONC Data Segmentation for Privacy Implementation Guide.

In addition, the following industry standards are used:

- The OASIS Security Assertion Markup Language (SAML).
- The OASIS WS-Security standard.
- The OASIS eXtensible Access Control Markup Language (XACML).

8.Third Party Components

The Consent2Share product is built with best-of-breed open source components to reduce costs and minimize risks. The following are the third party components and software used in Consent2Share:

- The Spring Framework including Spring Dependency Injection, Spring AOP, Spring Data, Spring Security, Spring MVC, and Spring Integration.
- The RabbitMQ message broker.
- Thymeleaf, a server-side HTML5 templating engine which serves as the view layer for the PCM system.
- The MySQL database.
- The Hibernate ORM framework including Hibernate Envers for audit trails.
- Twitter Boostrap, an HTML5, CSS3, and Javascript library for achieving a Responsive Web Design (RWD).
- The Apache Santurio XML Security framework.
- The Apache CXF web services framework.
- The Drools Business Rules Management System (BRMS).
- The HIEOS open source XDS.b Repository.
- The Saxon XSLT processor.
- The OpenEMPI open source Master Patient Index (MPI).

9.Third Party Web Services

The following web services are used within the Consent2Share system:

- The HIPAASpace NPI lookup service is used to search healthcare providers and obtain their NPI number in the PCM system.
- The HIPAASpace terminology lookup service can also be used in the PCM to lookup clinical concept codes when creating a consent document.
- The Adobe EchoSign signature service.

10. Application Security

Security is implemented in Consent2Shae using the Spring Security 3.2 framework.

10.1 Authentication and Authorization

The authentication mechanism is based on username and password with a user store based on a MySQL database. Authorization is based on roles and groups. Spring Security also supports other authentication mechanisms such as LDAP, or third party authentication providers using OAuth or OpenID. OAuth and OpenID can be used to enable single sign on for example to integrate with an existing patient portal.

10.2 Audit Trail

Audit Trail has been implemented within the Consent2Share PCM system using the Hibernate Envers framework. All user activities in the system are recorded and displayed to the patient.

10.3 Transport-layer Security

The Consent2Share PCM is configured to support SSL.

10.4 OWASP Top Ten

Consent2Share has undergone a comprehensive web application security architecture review. The system has been tested against the OWASP Top Ten web application security vulnerabilities. In addition to authentication and authorization, Spring Security also provides the following capabilities:

1. CSRF attack prevention.
2. Session Fixation protection.
3. Security Header integration:
    - HTTP Strict Transport Security for secure requests
    - X-Content-Type-Options integration
    - Cache Control
    - X-XSS-Protection integration
    - X-Frame-Options integration to help prevent Clickjacking.

10.5 Web Services Security

For Web Services exchanges using the XDS.b SOAP-based protocol, the ACS uses SSL for transport-layer security, WS-Security with 509 certificates for authentication, and SAML tokens using a Security Token Service (STS) based on Apache CXF.

11. Software Quality Assurance

11.1 Unit Testing

Consent2Share uses JUnit and Mockito for unit testing.

11.2 Integration Testing

Consent2Share uses Spring MVC Mock for integration testing of the Spring MVC controllers.

11.3 End-to-end Functional Testing

Consent2Share uses Selenium for end-to-end automated UI testing.

11.4 Performance Testing

Consent2Share uses JMeter for performance testing.

11.5 Web Application Security Testing

Consent2Share uses Burp and IBM AppScan for application-level security testing.