Integrating the Healthcare Enterprise

IHE IT Infrastructure Technical Framework Supplement 2004-2005

Cross-Enterprise Document Sharing (XDS)

Trial Implementation Version

Publication Date: August 15, 2004

Table of Contents

Foreword

Integrating the Healthcare Enterprise (IHE) is an initiative designed to stimulate the integration of the information systems that support modern healthcare institutions. Its fundamental objective is to ensure that in the care of patients all required information for medical decisions is both correct and available to healthcare professionals. The IHE initiative is both a process and a forum for encouraging integration efforts. It defines a technical framework for the implementation of established messaging standards to achieve specific clinical goals. It includes a rigorous testing process for the implementation of this framework. And it organizes educational sessions and exhibits at major meetings of medical professionals to demonstrate the benefits of this framework and encourage its adoption by industry and users.

The approach employed in the IHE initiative is to support the use of existing standards, e.g, HL7, DICOM, IETF and others as appropriate, rather than to define new standards. IHE profiles further constrain configuration choices where necessary in these standards to ensure that they can be used in their respective domains in an integrated manner between different actors. When clarifications or extensions to existing standards are necessary, IHE refers recommendations to the relevant standards bodies.

This initiative has numerous sponsors and supporting organizations in different medical specialty domains and geographical regions. In North America the primary sponsors are the American College of Cardiology (ACC), the Healthcare Information and Management Systems Society (HIMSS) and the Radiological Society of North America (RSNA). IHE Canada has also been formed. IHE Europe (IHE-EUR) is supported by a large coalition of organizations including the European Association of Radiology (EAR) and European Congress of Radiologists (ECR), the Coordination Committee of the Radiological and Electromedical Industries (COCIR), Deutsche Röntgengesellschaft (DRG), the EuroPACS Association, Groupement pour la Modernisation du Système d'Information Hospitalier (GMSIH), Société Francaise de Radiologie (SFR), Società Italiana di Radiologia Medica (SIRM), and the European Institute for health Records (EuroRec). In Japan IHE-J is sponsored by the Ministry of Economy, Trade, and Industry (METI); the Ministry of Health, Labor, and Welfare; and MEDIS-DC; cooperating organizations include the Japan Industries Association of Radiological Systems (JIRA), the Japan Association of Healthcare Information Systems Industry (JAHIS), Japan Radiological Society (JRS), Japan Society of Radiological Technology (JSRT), and the Japan Association of Medical Informatics (JAMI). Other organizations representing healthcare professionals are invited to join in the expansion of the IHE process across disciplinary and geographic boundaries.

The IHE Technical Frameworks for the various domains (IT Infrastructure, Cardiology, Laboratory, Radiology, etc.) defines specific implementations of established standards to achieve integration goals that promote appropriate sharing of medical information to support optimal patient care. It is expanded annually, after a period of public review, and maintained regularly through the identification and correction of errata. The current version for these Technical Frameworks may be found at www.rsna.org/IHE or http://www.himss.org/IHE.

The IHE Technical Framework identifies a subset of the functional components of the healthcare enterprise, called IHE Actors, and specifies their interactions in terms of a set of coordinated,

standards-based transactions. It describes this body of transactions in progressively greater depth. The volume I provides a high-level view of IHE functionality, showing the transactions organized into functional units called Integration Profiles that highlight their capacity to address specific clinical needs. The subsequent volumes provide detailed technical descriptions of each IHE transaction.

This IHE IT Infrastructure Technical Framework Supplement is issued for Trial Implementation through March 2005, per the schedule announced in February 2004.

Comments and change proposals arising from Trial Implementation may be submitted to http://forums.rsna.org under the forum:

"*Integrating the Healthcare Enterprise*"

Select the sub-forum:

"*IHE IT Infrastructure 2004-2005 Supplement for Trial Implementation*"

The IHE IT Infrastructure Technical Committee will address these comments resulting from implementation, connect-a-thon testing, and demonstrations such as HIMSS 2005. Final text is expected to be published in May 2005.

This Supplement introduces a new IHE Integration Profile that facilitates the sharing across health enterprises of electronic clinical documents with textual and structured content.

This Integration Profile, referred to as XDS (for Cross-enterprise Document Sharing), is focused on providing a standards-based specification for managing the sharing of documents that healthcare enterprises (anywhere from a private physician to a clinic to an acute care in-patient facility) have decided to explicitly share. This contributes to the foundation of a shared Electronic Health Record.

Background

One of the key integration problems in the realization of the EHR for the patients' continuity of care is the inability to share patient records across enterprises. Each enterprise contributes to care delivery, and may wish to retain responsibility for storage of different cross sections of the clinical artifacts produced during patient care. Without commonly accepted mechanisms for accessing these repositories, it becomes difficult, if not impossible, to share documents from a broad variety of care delivery enterprises

Within these enterprises, multiple care delivery systems exist, each of which may be responsible for storage and retrieval of different cross sections of the clinical artifacts produced during patient care. Without commonly accepted mechanisms for accessing these repositories, it becomes difficult, if not impossible, to share documents from separate care delivery systems across a collection of cooperating enterprises.

This Integration Profile enables a number of healthcare delivery organizations belonging to a clinical affinity domain (e.g. a community of care) to cooperate in the care of a patient by sharing clinical records in the form of documents as they proceed with their patients' care delivery activities. In the context of this IHE Integration Profile, a document is a very broad concept that represents a unit of health information being shared. An XDS Document may include any type of information in any standard format. In this manner, a document-centric longitudinal record is accumulated over time for patients. Care providers are offered the means to query and retrieve specific clinical documents of interest from this longitudinal cross-enterprise record.

Proposed Solution

XDS will use the concepts of document repositories and document registries. These are distinct entities with separate responsibilities:

- The repository is responsible for storing documents in a transparent and persistent manner and responding to document retrieval requests.
- The registry is responsible for storing information about the documents so that documents of interest for the care of a patient may be easily found, selected and retrieved irrespective of the repository where they are actually stored.

XDS is document-content neutral. It will support any type of document without regard to content and format, allowing this integration profile to be able to handle documents in a wide variety of commonly accepted formats for medical records.

Scope of Supplement

This supplement extends the existing IHE IT Infrastructure Technical framework and assumes that the reader is familiar with the IHE process and methodology in defining Integration Profiles, as well as familiar with the five IHE IT Infrastructure Integration profiles defined by the IHE ITI Technical Framework Version 1.0. The IHE ITI Technical Framework Version 1.0 documentation is available at www.himss.org/ihe or www.rsna.org/ihe.

The XDS Integration Profile is not intended to addresses all cross-enterprise EHR communication needs. Some of them may require the use of other IHE Integration profiles, such as Patient Identifier Cross-Referencing, Audit Trail & Node Authentication, Retrieve Information for Display, etc. Others may be only partially supported. Others may require future IHE Integration profiles, which will be defined by IHE as soon as the necessary base standards are available.

This supplement has been issued for trial implementation in August 2004 following a Public Comment period during which over 600 comments were received, reviewed and more than 80% were included in this TI version. IHE Connect-a-thons will be held early 2005 in various regions of the world to test vendor implementations of this profile. Based on this feedback, a final text version is scheduled to be issued by May 2005, following demonstrations at various venues, including HIMSS 2005.

The development of this IHE Integration Profile relies on considerable work and standards development activities that have been performed in this domain in many countries around the world (e.g. OASIS-ebXML, HL7 CDA, CEN ENV 13606, etc.).

Editors

Volume 1: XDS Integration Profile, Actors, Transactions, Options and Process Flow.
Charles Parisot, Peggy Rands, Keith W. Boone, Didi Davis and Karima Bourquard
Volume 2: Transactions
Bill Majurski, Emmanuel Cordonnier, Fred Behlen
Volume 1 & 2: Security
Rob Horn and Glen Marshall

Volume I – Integration Profiles

Add the following bullet to the end of the bullet list in section 1.7History of Annual Changes

Added the Cross Enterprise Document Sharing Profile. This Integration Profile enables a number of healthcare delivery organizations belonging to a clinical affinity domain (e.g. a community of care) to cooperate in the care of a patient by sharing clinical records in the form of documents as they proceed with their patients' care delivery activities.

This profile is based upon ebXML Registry standards, SOAP, HTTP and SMTP. It describes the configuration of an ebXML Registry in sufficient detail to support Cross Enterprise Document Sharing.

Add the following row to Table 2-1 in section 2.1 Dependencies among Integration Profiles.

Cross Enterprise Document Sharing	Audit Trail and Node Authentication	Each XDS Actor must be grouped with the Secure Node Actor.	Required to secure PHI stored in Registries and Repositories.

Add the following section to Vol 1 of the IHE ITI Technical framework: 10 Cross-Enterprise Document Sharing (XDS).

The ***Cross-Enterprise Document Sharing*** IHE Integration Profile facilitates the registration, distribution and access across health enterprises of patient electronic health records. Cross-Enterprise Document Sharing (XDS) is focused on providing a standards-based specification for managing the sharing of documents between any healthcare enterprise, ranging from a private physician office to a clinic to an acute care in-patient facility.

The XDS IHE Integration Profile assumes that these enterprises belong to one or more clinical affinity domains. A clinical affinity domain is a group of healthcare enterprises that have agreed to work together using a common set of policies and share a common infrastructure.

Examples of affinity domains include:

- Community of Care supported by a Local Health Information Infrastructure (LHII) in order to serve all patients in a given region.
- Nationwide EHR
- Specialized or Disease-oriented Care
    - Cardiology Specialists and an Acute Cardiology Center
    - Oncology network

    - Diabetes network
- Federation of enterprises
    - A regional federation made up of several local hospitals and healthcare providers
- Government sponsored facilities (e.g., VA or Military)
- Insurance Provider Supported Communities

Within a clinical affinity domain, certain common policies and business rules must be defined. They include how patients are identified, consent is obtained, and access is controlled, as well as the format, content, structure, organization and representation of clinical information. This Integration Profile does not define specific policies and business rules, however it has been designed to accommodate a wide range of such policies to facilitate the deployment of standards-based infrastructures for sharing patient clinical documents. This is managed through federated document repositories and a document registry to create a longitudinal record of information about a patient within a given clinical affinity domain. These are distinct entities with separate responsibilities:

- A document repository is responsible for storing documents in a transparent, secure, reliable and persistent manner and responding to document retrieval requests.
- A document registry is responsible for storing information about those documents so that the documents of interest for the care of a patient may be easily found, selected and retrieved irrespective of the repository where they are actually stored.

The concept of a document in XDS is not limited to textual information. As XDS is document content neutral, any type of clinical information without regard to content and representation is supported. This makes the XDS IHE Integration Profile equally able to handle documents containing simple text, formatted text (*e.g.,* HL7 CDA Release 1), images (*e.g.*, DICOM) or structured and vocabulary coded clinical information (*e.g.*, CDA Release 2, CCR, CEN ENV 13606, DICOM SR). In order to ensure the necessary interoperability between the document sources and the document consumers, the Clinical Affinity Domain must adopt policies concerning document format, structure and content.

The XDS Integration Profile is not intended to address all cross-enterprise EHR communication needs. Some scenarios may require the use of other IHE Integration profiles, such as Patient Identifier Cross-Referencing, Audit Trail and Node Authentication, Enterprise User Authentication, and Retrieve Information for Display. Other scenarios may be only partially supported, while still others may require future IHE Integration profiles, which will be defined by IHE as soon as the necessary base standards are available. Specifically:

1. The management of dynamic information such as allergy lists, medication lists, problem lists, etc is not addressed by XDS. However, the Retrieve Information for Display Integration Profile does provide some transactions (e.g., LIST-ALLERGIES, LIST-MEDS) that may be used to provide an elementary support of such capabilities. A complementary approach to managing updates and structured application access to such dynamic clinical information may be expected as a separate Integration Profile in the future.

2. The placing and tracking of orders (e.g. drug prescriptions, radiology orders, etc.) is not supported by XDS. This does not preclude the use of XDS to store and register orders and corresponding results when such artifacts need to be recorded in the patient's health record. However, XDS provides no facilities for tracking progress of an order through its workflow, and therefore is not intended for order management. A complementary approach to cross-enterprise order workflow (ePrescription, eReferral) may be expected as separate Integration Profiles in the future.

3. The operation of any XDS Clinical Affinity Domain will require that a proper security model be put in place. It is expected that a range of security models should be possible. Although the XDS Integration Profile is not intended to include nor require any specific security model, it is expected that XDS implementers will group XDS Actors with actors from the IHE Audit Trail and Node Authentication and will need an Access Control capability that operates in such a cross-enterprise environment. New IHE Integration Profiles have been identified as candidates (e.g. Public Key Infrastructure, Access Control, etc.).

4. The establishment of independent but consistently XDS-based Affinity Domains will call for their federation, as patients expect their records to follow them as they move from region to region, or country to country. IHE foresees a need for transferring information from one Clinical Affinity Domain to another, or to allow access from one Affinity Domain to documents managed in other Affinity Domains. XDS has been designed with this extension in mind. An XDS Domains Federation Integration Profile that complements XDS may be anticipated in the future.

5. XDS does not address transactions for the management or configuration of a clinical affinity domain. For example, the configuration of network addresses or the definition of what type of clinical information is to be shared is specifically left up to the policies established by the clinical affinity domain.

Glossary

Clinical Affinity Domain

A group of healthcare enterprises that have agreed to work together using a common set of policies and which share a common infrastructure of repositories and a registry.

EHR-CR

An EHR-CR or Care-delivery Record abstracts the patient information managed by the IT system or set of systems of a Care Delivery Organization, which may support a broad variety of healthcare facilities: private practice, nursing home, ambulatory clinic, acute care in-patient facility, etc.

EHR-LR

The documents shared by the EHR-CR and tracked by the Registry form a Longitudinal Record for the patients that received care among the EHR-CRs of the Clinical Affinity Domain. This is known as the EHR-LR.

XDS Document

An XDS Document is the smallest unit of information that may be provided to a Document Repository and registered in a Document Registry. An XDS Document may contain simple text, formatted text (e.g. HL7 CDA Release 1), images (e.g. DICOM) or structured and vocabulary coded clinical information (e.g. CDA Release 2, CCR), or may be made up of a mixture of the above types of content.

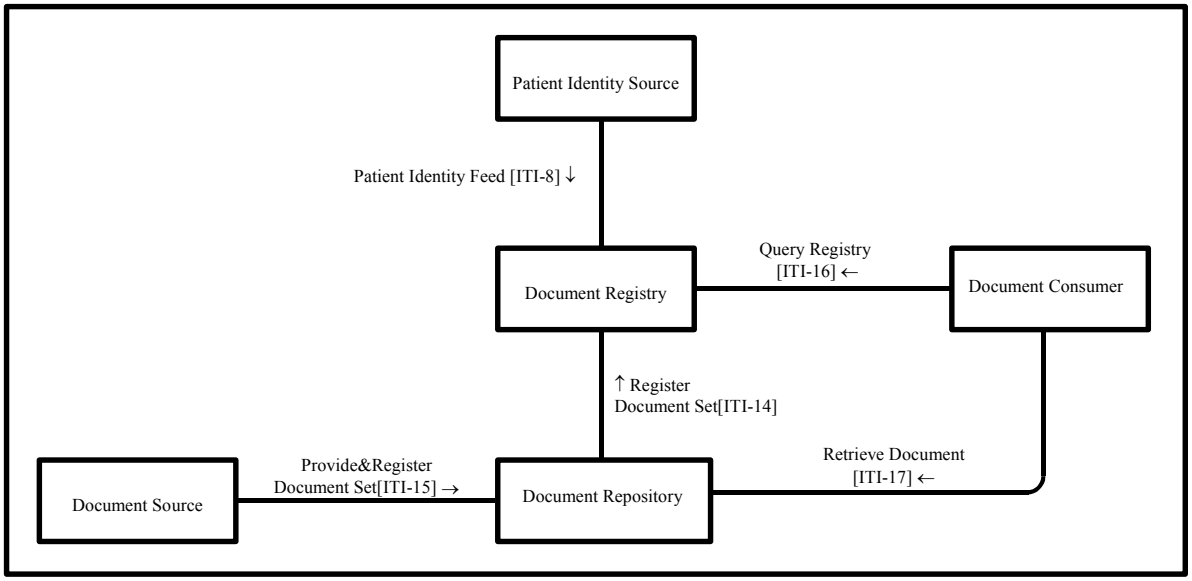
Submission Set

A set of XDS documents registered together to a Document Repository concerning information related to one care event of a single patient, provided by an EHR system.

XDS Folder

An XDS Folder allows document sources to group the documents they submit with other related documents. What constitutes a Folder and the vocabulary associated with the specific Folders used by an EHR-CR is decided by an agreement between the care delivery organization members of a Clinical Affinity Domain.

10.1 Actors/Transactions



300 **Figure 10.1-1 Cross-Enterprise Document Sharing Diagram**

Table 10.1-1 XDS - Actors and Transactions

Actors	Transactions	Optionality	Section in Vol. 2
Document Consumer	Query Registry	R	ITI TF-2:3.16
	Retrieve Document	R	ITI TF-2:3.17
Document Source	Provide and Register Document Set	R (Note 1)	ITI TF-2:3.15
Document Repository	Provide and Register Document Set	R (Note 1)	ITI TF-2:3.15
	Register Document Set	R (Note 2)	ITI TF-2:3.14
	Retrieve Document	R	ITI TF-2:3.17
Document Registry	Register Document Set	R (Note 2)	ITI TF-2:3.14
	Query Registry	R	ITI TF-2:3.16
	Patient Identity Feed	R	ITI TF-2:3.8
Patient Identity Source	Patient Identity Feed	R	ITI TF-2:3.8

Note 1: The Provide and Register Document Set is not required in implementations where the Document Source is grouped with the Document Repository Actor.

305 Note 2: The Register Document Set Transaction is not required in implementations where the Document Registry Actor is grouped with the Document Repository Actor. However, it is strongly recommended that these transactions be supported to allow for future configuration with multiple Repositories.

10.1.1 Actors

10.1.1.1 Document Source

The Document Source Actor is the producer and publisher of documents. It is responsible for sending documents to a Document Repository Actor. It also supplies metadata to the Document Repository Actor for subsequent registration of the documents with the Document Registry Actor.

10.1.1.2 Document Consumer

The Document Consumer Actor queries a Document Registry Actor for documents meeting certain criteria, and retrieves selected documents from one or more Document Repository actors.

10.1.1.3 Document Registry

The Document Registry Actor maintains metadata about each registered document in a document entry. This includes a link to the Document in the Repository where it is stored. The Document Registry responds to queries from Document Consumer actors about documents meeting specific criteria. It also enforces some healthcare specific technical policies at the time of document registration.

10.1.1.4 Document Repository

The Document Repository is responsible for both the persistent storage of these documents as well as for their registration with the appropriate Document Registry. It assigns a URI to documents for subsequent retrieval by a Document Consumer.

10.1.1.5 Patient Identity Source

The Patient Identity Source Actor is a provider of unique identifier for each patient and maintains a collection of identity traits. The Patient Identify Source facilitates the validation of patient identifiers by the Registry Actor in its interactions with other actors.

10.1.2 Transactions

10.1.2.1 Provide and Register Document Set

A Document Source Actor initiates the Provide and Register Document Set Transaction. For each document in the submitted set, the Document Source Actor provides both the documents as an opaque octet stream and the corresponding metadata to the Document Repository. The Document Repository is responsible to persistently store these documents, and to register them in the Document Registry using the Register Documents transaction by forwarding the document metadata received from the Document Source Actor.

10.1.2.2 Register Document Set

A Document Repository Actor initiates the Register Document Set transaction. This transaction allows a Document Repository Actor to register one or more documents with a Document Registry, by supplying metadata about each document to be registered. This document metadata will be used to create an XDS Document Entry in the registry. The Document Registry Actor ensures that document metadata is valid before allowing documents to be registered. If one or more documents fail the metadata validation, the Register Document Set transaction fails as a whole.

To support composite documents, an XDS Document may be a multipart document. The Document Repository must handle multi-part data sets as an "opaque entity". The Document Repository does not need to analyze or process its multi-part structure nor the content of any parts in the context of the XDS Integration Profile.

10.1.2.3 Query Registry

The Query Registry transaction is issued by the Document Consumer Actor on behalf of a care provider (EHR-CR) to a Document Registry. The Document Registry Actor searches the registry to locate documents that meet the provider's specified query criteria. It will return a list of document entries that contain metadata found to meet the specified criteria including the locations and identifier of each corresponding document in one or more Document Repositories.

10.1.2.4 Retrieve Document

A Document Consumer Actor initiates the Retrieve Document transaction. The Document Repository will return the document that was specified by the Document Consumer.

To support composite documents, an XDS Document may be a multipart document. In this case, the Document Consumer must take appropriate actions to make the multipart content accessible to the user.

10.1.2.5 Patient Identity Feed

This is IHE ITI Transaction 8, defined as part of the Patient Identifier Cross-referencing Integration Profile. It conveys the patient identifier and corroborating demographic data, captured when a patient's identity is established, modified or merged or in cases where the key corroborating demographic data has been modified. Its purpose in the XDS Integration Profile is to populate the registry with patient identifiers that have been registered for the affinity domain.

10.2 Integration Profile Options

Options that may be selected for this Integration Profile are listed in the table 10.2-1 along with the Actors to which they apply. Dependencies between options when applicable are specified in notes.

Table 10.2-1 XDS - Actors and Options

Actor	Options	Vol & Section
Document Source	*Off-Line transaction mode*	ITI TF-1:10.4.7.1
Document Repository	*Off-Line transaction mode*	ITI TF-1:10.4.7.1
Document Registry	*No options defined*	- -
Document Consumer	*Query Registry Transaction*	ITI TF-2:3.16
	Retrieve Document Transaction	ITI TF-2:3.17
Patient Identity Source	*No options defined*	- -

375

For the XDS Document Consumer Actor, either one or both of the two options shall be selected.

10.3 Integration Profile Process Flow

A typical patient goes through a sequence of encounters in different care settings. In each care setting, the resulting patient information is created and managed by multiple care delivery information systems (EHR-CRs). Through a sequence of care delivery activities, a number of clinical documents are created. The EHR-LR provides the means to share the relevant subset of these documents, as they are contributed by the various EHR-CRs that are part of the same clinical affinity domain.

Example : Cardiac Patient Management Scenario

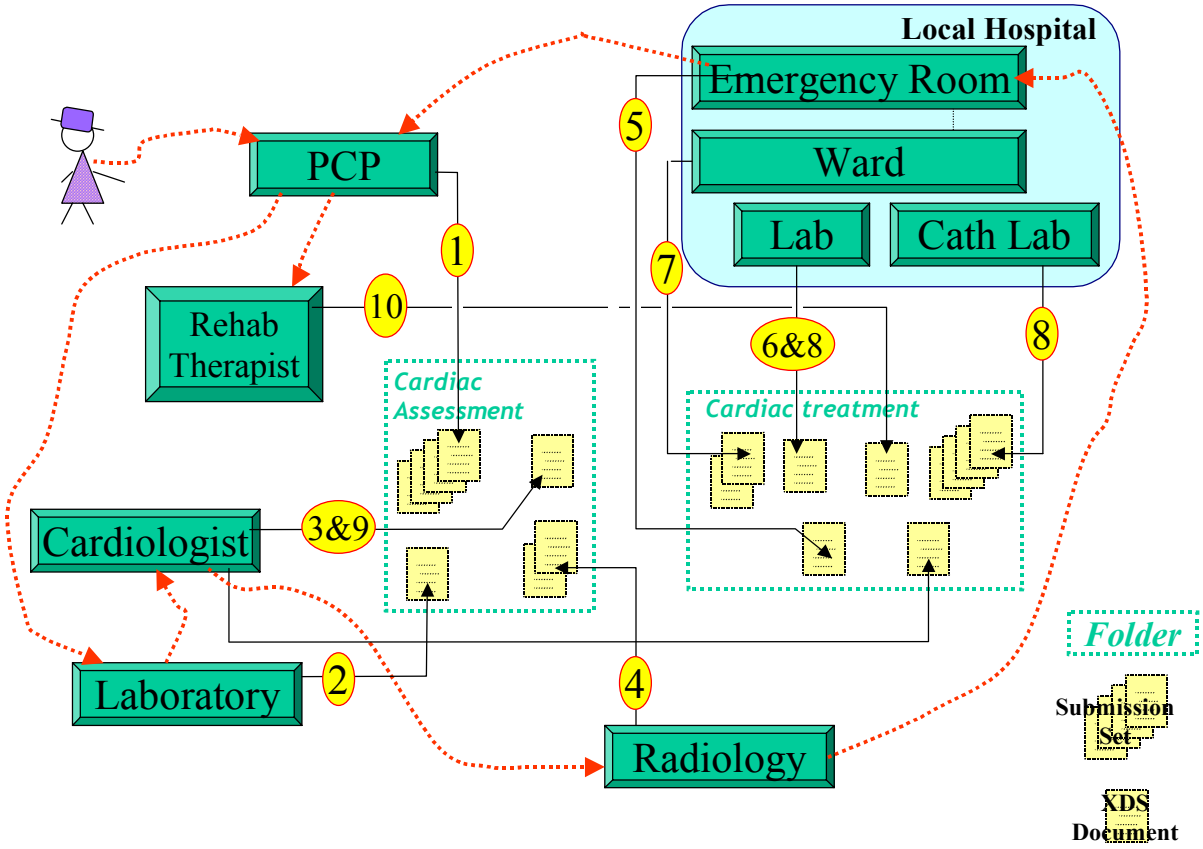


Figure 10.3-1 Cardiac Patient Management Scenario Transaction Process Flow

This scenario spans about 3 weeks of a patient's cardiac episode. The patient presents to her primary care provider (PCP) with complaints of shortness of breath, nausea, tiredness and chest pains. This doctor works closely with a local hospital that has recently established a cardiac care network that allows PCPs, cardiologists, laboratories and two local hospitals to share clinical documents to improve patient care. This cardiac network is part of a local care data exchange community that has been set-up in this community and to which the care plan to which this

patient belong has encouraged patients to subscribe. Our patient has been provided a health record account number.

1. During the patient examination, the PCP records the complaint, and determines that he should perform an ECG. He queries the cardiac care network to see if there are prior ECG reports (step 1 in Figure 10.3-2), using a coded document class "report" and a coded practice setting "cardiology" established by the cardiac care network for ECG reports. Among the matching Documents, he locates a prior ECG report which is then retrieved (step 2 in Figure 10.3-2). He compares the two results and determines that the patient should be referred to a cardiologist. He searches for additional reports in the cardiac care network (step 3 in Figure 10.3-2) for this patient, but finds none.

   Using the ambulatory EHR system, he creates a submission request onto the patients health record account number for a "PCP office visit" that includes a submission set consisting of three new documents (visit note, referral letter, new ECG report) and of one reference to the prior ECG report (step 4 in Figure 10.3-2). Following the Cardiology Network Affinity Domain policy, he creates a "cardiac assessment" Folder to contain all four documents in order to facilitate collaboration with the cardiologist.

   The repository used by the ambulatory EHR system will then register the documents that are part of this submission request (step 5 in Figure 10.3-2).

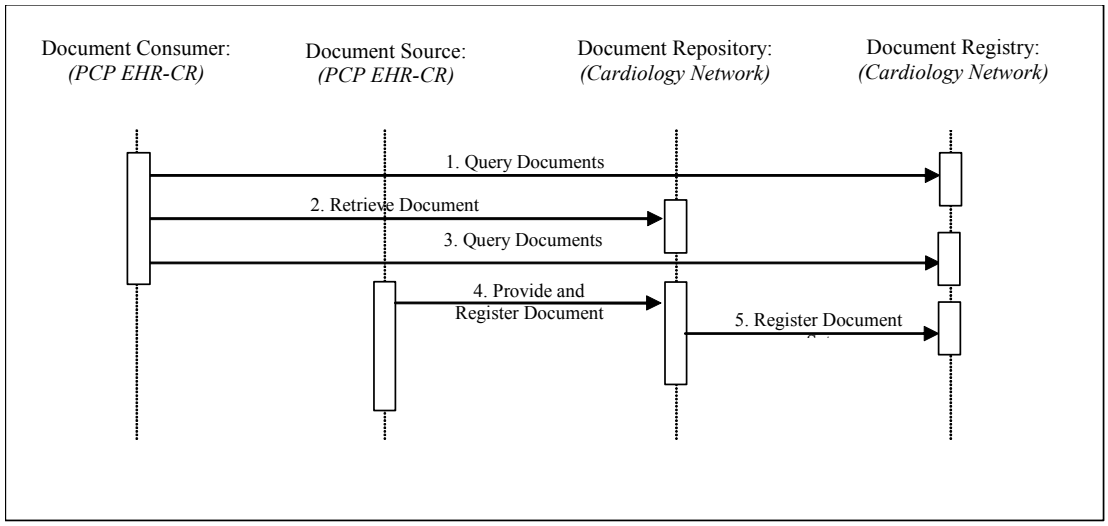


Figure 10.3-2 PCP Query Transactions Process Flow

The PCP EHR system implements the Document Consumer and Document Source actors to issue the Query, Retrieve and Provide & Register transactions as shown in Figure 10.3-2. The transactions are processed by the Document Repository and the Document Registry provided by the cardiology care network.

2. The patient appointment with the cardiologist is scheduled. The patient goes to the lab for the lab tests required before appointment. The lab creates a submission set with a clinical code of "laboratory tests" containing the lab results. The lab is not aware of the "cardiology assessment" folder.

3. The cardiologist sees the patient. He queries the repository for any patient's records in a "cardiac assessment" folder (step 1 in Figure 10.3-3). Available are the visit note from the PCP, the ECG and prior ECG, and the referral letter, which he retrieves and reviews (steps 2-5 in Figure 10.3-3). He also queries for recent lab reports, and finds the lab results (step 6 in Figure 10.3-3). This is also retrieved and reviewed (step 7 in Figure 10.3-3).

   The cardiologist performs an ultrasound, dictates a visit note, and orders a nuclear stress test. The visit note and ultrasound images and report are registered as a "cardiologist office visit" submission set and placed in the "cardiac assessment" Folder. In addition, the lab report is added to the "cardiac assessment" Folder (step 8 in Figure 10.3-3).

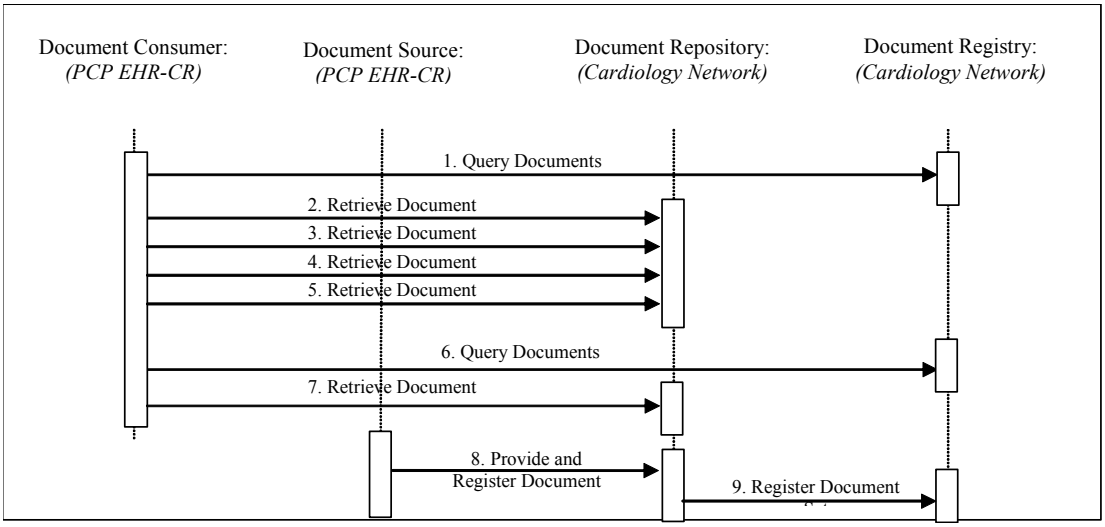


Figure 10.3-3 PCP Query Transactions Process Flow

4. The patient is seen at a radiology facility for the nuclear stress test. The test is performed, and the radiologist dictates the report. The nuclear stress test report is registered in a "radiology examination" submission set and associated with the "cardiac assessment" Folder

5. Although she has a scheduled appointment with her cardiologist in two days, she wakes up with severe chest pain. On the way to work, she decides to go to the emergency room (ER) of her local hospital.The ER doctor uses the hospital EHR system to query the cardiac care network registry and repositories for documents related to the patient in reverse chronological order (step 1 in Figure 10.3-4). Available documents from latest cardiology related Folder are the visit notes from the PCP and cardiologist, the recent and prior ECGs, the lab results, and the ultrasound images and report, and the nuclear stress test images and report.

   The ER doctor retrieves and reviews the two most relevant reports (step 2 and 3 in Figure 10.3-4).

   The ER doctor orders lab tests, ECG, and places the patient under monitoring. The lab tests and ECG are placed in the hospital EHR that acts as a Document Repository Actor

for the cardiac network. Abnormal cardiac activity requires a catheterization, diagnostics and possibly intervention. The ER doctor admits the patient to the cardiology service and contacts the cardiologist.

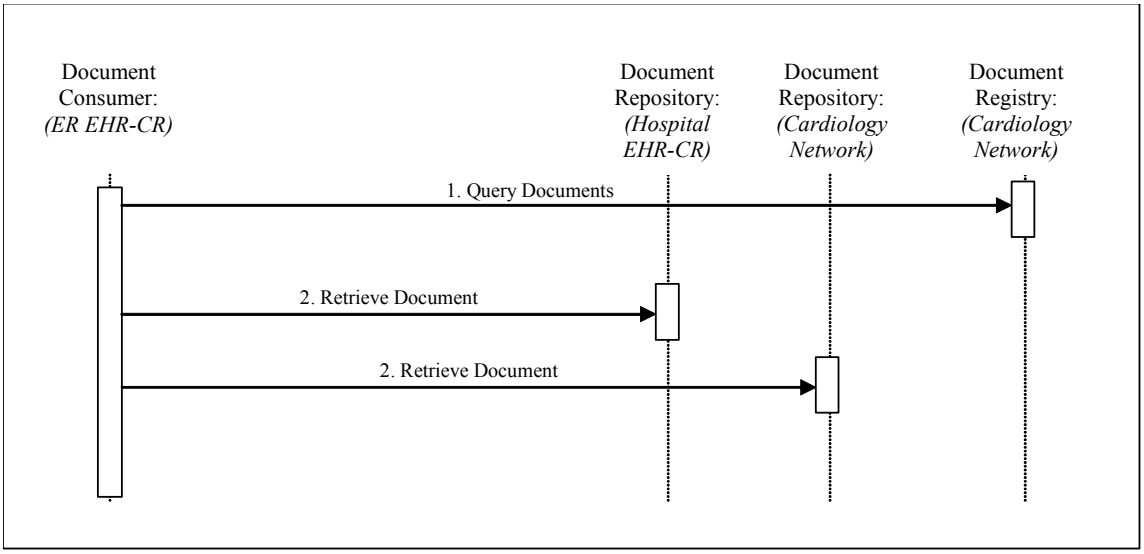


Figure 10.3-4 ER Query Transactions Process Flow

6. While talking to the ER physician, the cardiologist accesses the cardiac care network from his home office. He queries for all documents related to the patient since the last visit in his office. The nuclear stress test report that he did not previously review is available, along with lab results and ECG results from the ER. The two physicians determine a plan of care and the cardiologist makes arrangements to see the patient in the hospital.

7. As the patient is transferred from the ER, the ER visit notes are submitted as an "emergency department visit" submission set and placed in a newly created "cardiology treatment" Folder along with the earlier lab and ECG results.

8. The patient is transferred to an inpatient bed with the following sequence of events.
   - The patient is scheduled for a catheterization procedure in cath lab.
   - Additional lab tests are ordered and performed.
   - A diagnostics procedure is performed in cath lab.
   - An intervention with the placement of a stent is performed.
   - A cath intervention report is dictated.
   - Patient is returned to monitored care for recovery.
   - Education given to patient and family.
   - Discharge Summary dictated by cardiologist.
   - Cardiologist orders lab tests to be completed prior to scheduled follow-up visit.

   The admission assessment, lab results, cath intervention report and key images, and discharge summary form a "cardiology intervention" submission set, which is

registered with the cardiac care network registry in the "cardiac treatment" Folder started by the ER.

9. The patient returns to the cardiologist for the post discharge follow-up visit. The resulting visit note, cardiac rehab and summary letters are placed in a "cardiology office visit" submission set and in the "cardiac treatment" Folder.

10. The patient goes to rehab sessions as scheduled by the cardiologist. The patient recovers and is seen by the PCP and cardiologist for routine visits.

10.4 General Principles

10.4.1 EHR-CR Concept

An EHR-CR or Care-delivery Record abstracts the information system or systems of a care delivery organization, which may support a broad variety of healthcare facilities: private practice, nursing home, ambulatory clinic, acute care in-patient facility, etc.

Typically a patient goes through a sequence of encounters in different care settings as depicted in the figure below.

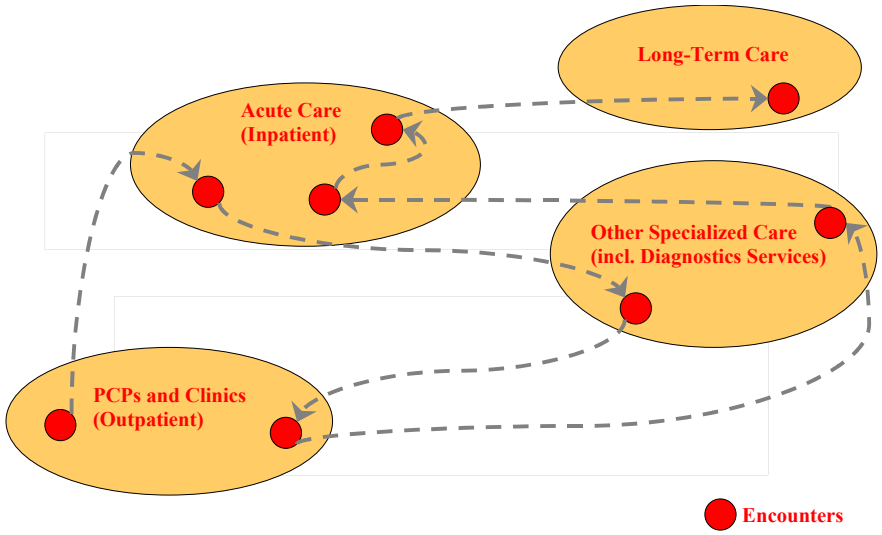


Figure 10.4.1-1 Sequence of encounters across care delivery organizations

It is out of the scope of this IHE Integration Profile to define or restrict the type of care provided, nor the internal workflow of a care delivery organization. The EHR-CR system participates only to the cross-enterprise clinical document sharing as Document Source and Document Consumer Actors according to the following principles:

1. EHR-CR as Document Source contributes documents in any one of the document formats that are supported by the XDS Affinity Domain (e.g. CDA Release 1, CDA Release 2 with specific templates, DICOM Composite SOP Classes, ASTM-CCR, CEN ENV 13606 etc).
2. This Profile does not require that the EHR-CR as Document Sources and Consumers store and manage their internal information in the form of documents as they are shared throughout the XDS Affinity Domain.

3. By grouping a Document Source with a Document Repository, an EHR-CR may leverage existing storage to provide a unified access mechanism without needing to duplicate storage.

4. EHR-CRs as Document Sources and Consumers are responsible to map their local codes into the affinity domain codes if necessary.

The XDS Documents shared by the EHR-CR and tracked by the XDS Registry form a Longitudinal Record for the patients that received care among the EHR-CRs of the XDS Affinity Domain.

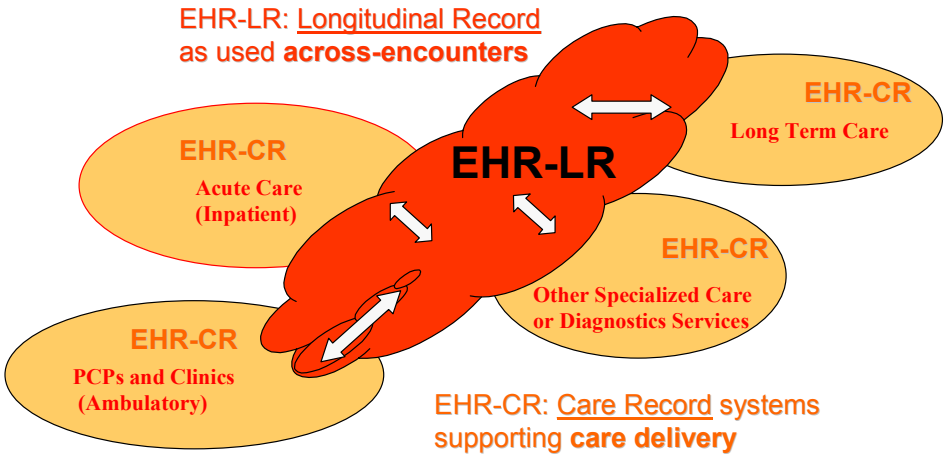


Figure 10.4.1-2 Contributing and sharing to a patients' longitudinal health record

This shared clinical record is called an EHR-LR in this Integration Profile.

10.4.2 XDS Document Concept

An XDS Document is the smallest unit of information that may be provided to a Document Repository Actor and be registered as an entry in the Document Registry Actor.

An XDS Document is a composition of clinical information that contains observations and services for the purpose of exchange with the following characteristics: Persistence, Stewardship, Potential for Authentication, and Wholeness. These characteristics are defined in the HL7 Clinical Document Architecture Release 1 specification. An XDS Document may be human readable (with the appropriate application). In any case, it should comply with a published

standard defining its structure, content and encoding. IHE intends to define content-oriented Integration Profiles relying on such content standards to be used in conjunction with XDS.

The XDS Integration Profile manages XDS Documents as a single unit of information; it does not provide mechanisms to access portions of an XDS Document. Only the Document Sources or Document Consumers have access to the internal information of the XDS Document. When submitted for sharing, an XDS Document is provided to the Document Repository Actor as an octet stream. When retrieved through the Retrieve Document transaction, it shall be unchanged from the octet stream that was submitted.

The Document Source Actor is responsible to produce the metadata that will be submitted to the Document Registry Actor to form the XDS Document Entry that will be used for query purposes by XDS Consumer Actors. The Document Source maintains responsibilities over the XDS Documents it has registered. It shall replace XDS Documents that may have been submitted in error.

XDS Documents are required to be globally uniquely identified. See ITI TF-2:Appendix B for a definition of globally unique identifiers.

10.4.3 Submission Request

An XDS Submission Request is a means to share XDS Documents. It may be conveyed:

- by a Document Source Actor in a *Provide and Register Document Set Transaction* to the Document Repository Actor, or
- by a Document Repository Actor in a *Register Document Set Transaction* to the Document Registry Actor

An XDS Submission Request contains elements of information that will ensure the proper registration of XDS Documents. These are:

1. Metadata to be placed in Document Entries for new XDS Documents being submitted,
2. A Submission Set that includes the list of all new XDS Documents and Folders being submitted and optionally a list of previously submitted XDS Documents,
3. If desired, Folders to be created with the list of included XDS Documents (new document being submitted as well as previously submitted),
4. If desired, addition to previously created Folders of lists of XDS Documents (new document being submitted as well as previously submitted), and
5. Zero or more XDS Document octet streams for the new XDS Documents being submitted.

Following a successful Submission Request, new XDS Documents, Submission Set, and Folders included in the Submission Request are available for sharing in an XDS Clinical Affinity Domain. In case of failure to process a Submission Request, the Submission Set and any XDS Documents and Folders shall not be registered.

10.4.4 Submission Set Concept

An XDS Submission Set is related to care event(s) of a single patient provided by the care delivery organization EHR-CR performing the submission request. It creates a permanent record of new XDS Documents as well as pre-existing (i.e. already registered) XDS Documents that have a relationship with the same care event(s). It also includes the record of new XDS Folders creation.

An XDS Submission Set shall be created for each submission request. It is related to a single Document Source Actor and is conveyed by a single Provide & Register Document Set Transaction or a Register Document Set Transaction.

The Document Registry may be queried to find all documents registered in the same XDS Submission Set.

The same XDS Document, initially registered as part of a Submission Set, may also be referenced by later XDS Submission Set. This allows older documents relevant to the present care of a patient to be associated with more recent Submission Sets.

XDS provides complete flexibility to EHR-CRs to relate Documents and Submission Sets to an encounter, a visit, an episode of care, or various workflow processes within EHR-CRs.

10.4.5 Concept of Folder

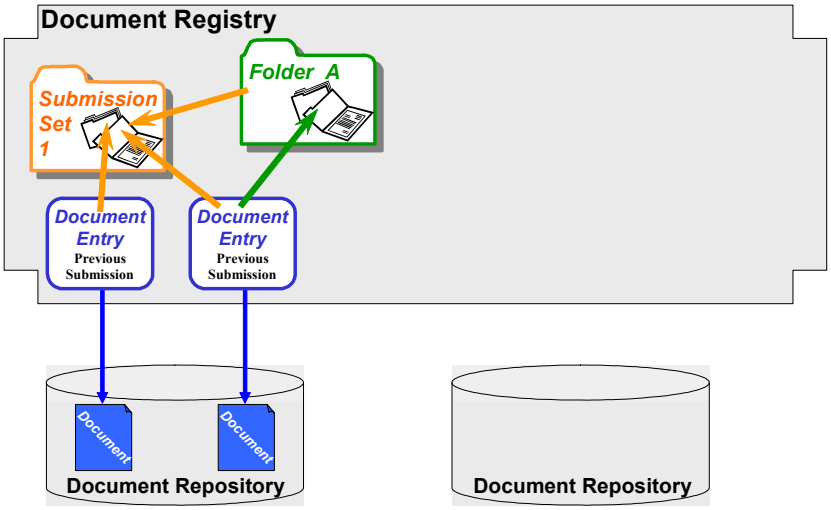
The purpose of an XDS Folder is to provide a collaborative mechanism for several XDS Document Sources to group XDS Documents for a variety of reasons (e.g. a period of care, a problem, immunizations, etc.) and to offer the Document Consumers a means to find all Document Entries placed in the same Folder. The following principles apply to an XDS Folder:

1. A Folder groups a set of XDS Documents related to the care of a single patient,
2. One or more Document Source Actors may submit documents in a given Folder,
3. A Folder may be created by a Document Source and/or predefined in an Affinity Domain,
4. The content of a Folder is qualified by a list of codes/meaning,
5. Document Source Actors may find existing Folders by querying the Document Registry or by means outside the scope of XDS (e.g. Cross-enterprise workflow, such ePrescription, eReferral, etc),
6. Once created a Folder is permanently known by the Document Registry,
7. Placing previously existing Documents in Folders is not recorded as part of the Submission Set,
8. Folders in XDS may not be nested,
9. The same documents can appear in more than one Folder, and
10. Folders have a globally unique identifier.

10.4.6 Example of use of Submission Request, Submission Set and Folder

The sequence of figures below shows an example of a submission request that includes two new documents, a reference to a pre-existing document and the use of two folders. The first figure depicts the initial state of a Document Registry in which two Documents have been submitted where one is associated with a Folder A. The second figure depicts a submission request that adds two new documents, placing one of them into a pre-existing folder and the other one into a new Folder B.

Document Repository and Registry – Initial State



Document Repository and Registry – Submission Request

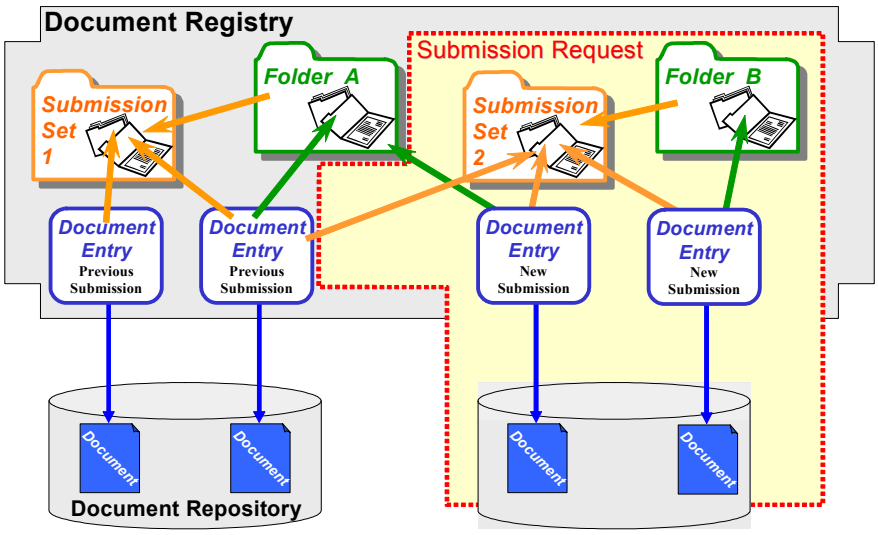


Figure 10.4.6-1 Example of a submission flow to an XDS Registry

From the above example, the contents of a Submission Set are shown by the figure below. The Document Entries associated with the Submission Set are logical part of the Submission Set.

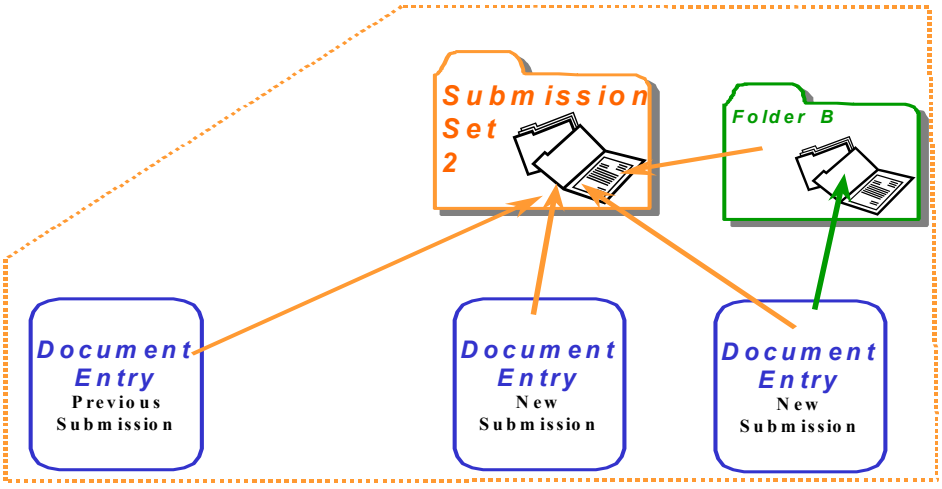


Figure 10.4.6-2 The logical content of a Submission Set

10.4.7 XDS Registry Data Model and Attributes

The XDS Integration Profile provides a means to place documents in a repository chosen by the Document Source, and also to place information about this document (or metadata) in an entry of the Document Registry that manages the Affinity Domain.

The term metadata reflects that this information is "about" the documents. The purpose of well-specified document metadata is to enable a uniform mechanism for Document Consumers to locate clinical documents of interest much in the way a card catalog in a library helps readers find the book they want.

This section addresses the high-level data model in which the metadata is registered and against which queries of the XDS registry are performed. Then it presents the specific attributes that may be registered and used to filter the document entries of the registry.

10.4.7.1 XDS Document Registry Data Model

The following entities are used in the XDS Document Registry Data Model:

XDS Document Entry: Information entity managed by a Document Registry Actor that contains a set of metadata describing the major characteristics of an XDS Document along with a link to the Document Repository Actor where the actual XDS Document may be retrieved.

XDS Document: A stream of bytes stored in a Document Repository Actor and pointed to by an XDS Document Entry.

XDS Folder: A logical container that groups one or more XDS Document Entries in any way required (e.g. by source care delivery activities, by episode, care team, clinical specialty or clinical condition). This kind of organizing structure is used variably: in some centers and

systems the Folder is treated as an informal compartmentalization of the overall health record; in others it might represent a significant legal portion of the EHR relating to the originating enterprise or team. The Folder is a means of providing organization of XDS Documents (or Composition in EHRCOM). The same XDS Document Entry may belong to zero or more Folders.

XDS Submission Set: When XDS Documents are registered by a Document Source Actor, they shall be included in one and exactly one Submission Set. An XDS Submission Set groups zero or more new XDS Documents and references to already registered XDS Documents to ensure a persistent record of their submission.

XDS Submission Request: A Submission Request includes one and only one Submission Set, zero or more new XDS Folders and assignment of XDS Documents into new or existing Folders. A Submission Request is processed in an atomic manner by the Document Repository and the Document Registry (i.e. all XDS Documents included or referenced in a Submission Set as well as the Folders and inclusion of Folders references are registered or none will). This ensures that they are all made available to Document Consumer Actors at the same time.

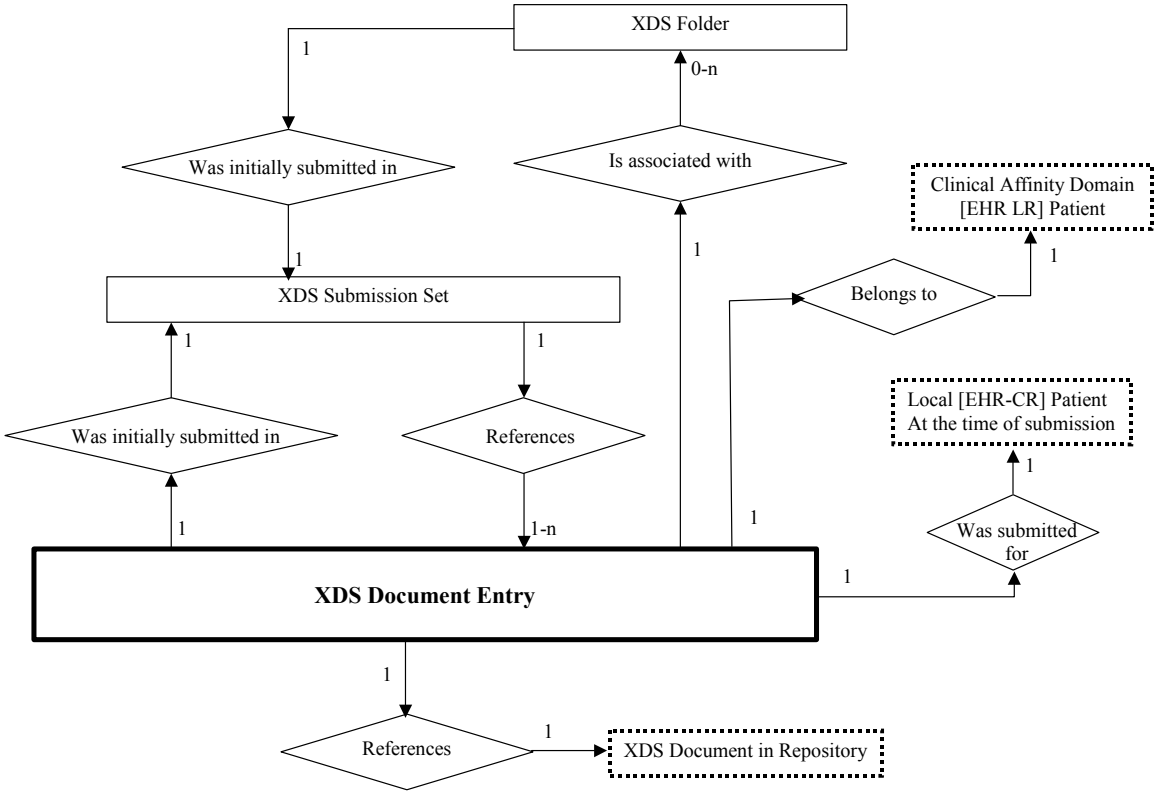


Figure 10.4.7-1 XDS Document Registry Data Model

10.4.7.2 Attributes of the XDS Document Entries

The specific attributes of each entity in the above registry data model have been selected from document header attributes from several standards (see ITI TF-2:Appendix L), including:

- ANSI/HL7 CDA R1-2000
- HL7 CDA Release 2 (draft) Document header definition (Dec 2003 Committee Ballot)
- Composition attributes from EHR ENV 13606 (draft).

XDS defines a well focused set of primary attributes that support the most common use cases to search the most relevant documents. These include:

Patient Id
Service Start and Stop Time
Document Creation Time
Document Class Code and Display Name
Practice Setting Code and Display Name
Healthcare Facility Type Code and Display Name
Availability Status (Available, Deprecated)
Document Unique Id

The three codes (Document Class, Practice Setting and Healthcare facility Type) are code set that are expected to generally include a limited number of values (between 10 and 100), thus ensuring a reasonably easy search capability.

A number of additional query attributes or attributes used to perform a secondary selection in order to decide to retrieve a specific document are also defined by this Integration Profile. At the Document Level, these include a fine grained Document Type (e.g. LOINC classification), a list of Event Code that can be used as key word, the document author and associated institution, the document relationship to manage replacement addendum and a variety of transformations, a confidentiality code, language code, etc.

The complete list of attributes and their definition is documented in the IHE ITI Register Transaction (see Volume II section 3.12).

10.4.8 Concept of an XDS Affinity Domain

An XDS Affinity Domain is an administrative structure made of a well-defined set of Document Source Actors, set of Document Repositories, set of Document Consumers organized around a single Document Registry Actor that have agreed to share clinical documents.

Note: Document Sources, Repositories and Consumers may belong to more than one Affinity Domain and share the same or different documents. This is an implementation strategy and will not be further described.

Note: The federation of Affinity Domains is not supported by the XDS Integration Profile. It is expected that a future IHE Integration Profile will address the cooperation of multiple Document Registry Actors serving different Affinity Domains.

A number of policies will need to be established in an Affinity Domain in order to ensure effective interoperability between Document Sources and Consumers. Some of the key technical policies include (A more extensive list of policy agreements that need to be made by Affinity Domains is discussed in ITI TF-1: Appendix L):

1. The document formats that will be accepted for registration
2. The various vocabulary value sets and coding schemes to be used for the submission of metadata of document, submission set and folders registration.
3. The Patient Identification Domain (Assigning Authority) used by the Document Registry.

10.4.9 Patient Identification Management

Since the central focus of the DS Integration Profile is "sharing documents", it is critical that each document be reliably associated with the corresponding patient (Patient Id).

The XDS Document Registry is not intended to be an authority for patient identification and demographics information. This Integration Profile uses a Patient Identity Source Actor as the authoritative source of Patient Identifiers (master patient ID) for the Affinity Domain.

Note: This Integration Profile can be easily extended to support a scenario where no master patient ID is defined (i.e. no Patient Identity Source for the Affinity Domain). Such option, would requiring the use of federated patient identities at the time of query of the XDS Document Registry, may be expected as a future addition to this Integration Profile.

The following principles are defined:

1. The Patient Identifier Domain managed by the Patient Identity Source Actor in the Affinity Domain is the source of patient identifiers (and merge operations) used by the XDS Document Registry to link Documents to a specific Patient. This Patient Identifier Domain is called the XDS Affinity Domain Patient Identification Domain (XAD-Pid Domain).
2. Submission Requests for Documents related to Patients with IDs not registered in the XDS Affinity Domain Patient Identifier Domain shall be rejected by the XDS Document Registry.
3. The XDS Document Registry will contain certain patient information (e.g. source patient ID, Surname, Given Name, Sex, Birthdate) for the purpose of audits and potential verification by Document Consumers. As this Integration Profile does not make any assumptions about the referential integrity and update of this information, these fields[1] shall not be used as query matching keys.
4. As XDS Document Sources and Consumers may belong to different Patient Identification Domains, these systems need to cross-reference their own local Patient ID to the

[1] It is possible to submit a new document to replace a previously submitted one, with a new document entry created in the registry to correct for errors in the submitted document in the original submission request. However this is not a mechanism that updates only the metadata, as the replaced document is only deprecated and remains pointed by the original metadata.

corresponding patient ID in the XAD-Pid Domain of the Registry. Preferably, these systems may choose to use the IHE Patient Identifier Cross-referencing Integration Profile (See Appendix D.3) for this purpose.

5. The XDS Document Registry is responsible for validating Document metadata in accordance with the XDS Affinity Domain's policies. Submissions Requests that do not conform to these policies should be rejected by the Document Registry.

The figure below depicts an example of an Affinity Domain with its Patient Identifier Domain (called XAD) and two EHR-CRs where the cross-referencing is performed internally to the Document Source and the Document Consumer Domains (Domain C and Domain D2 respectively).

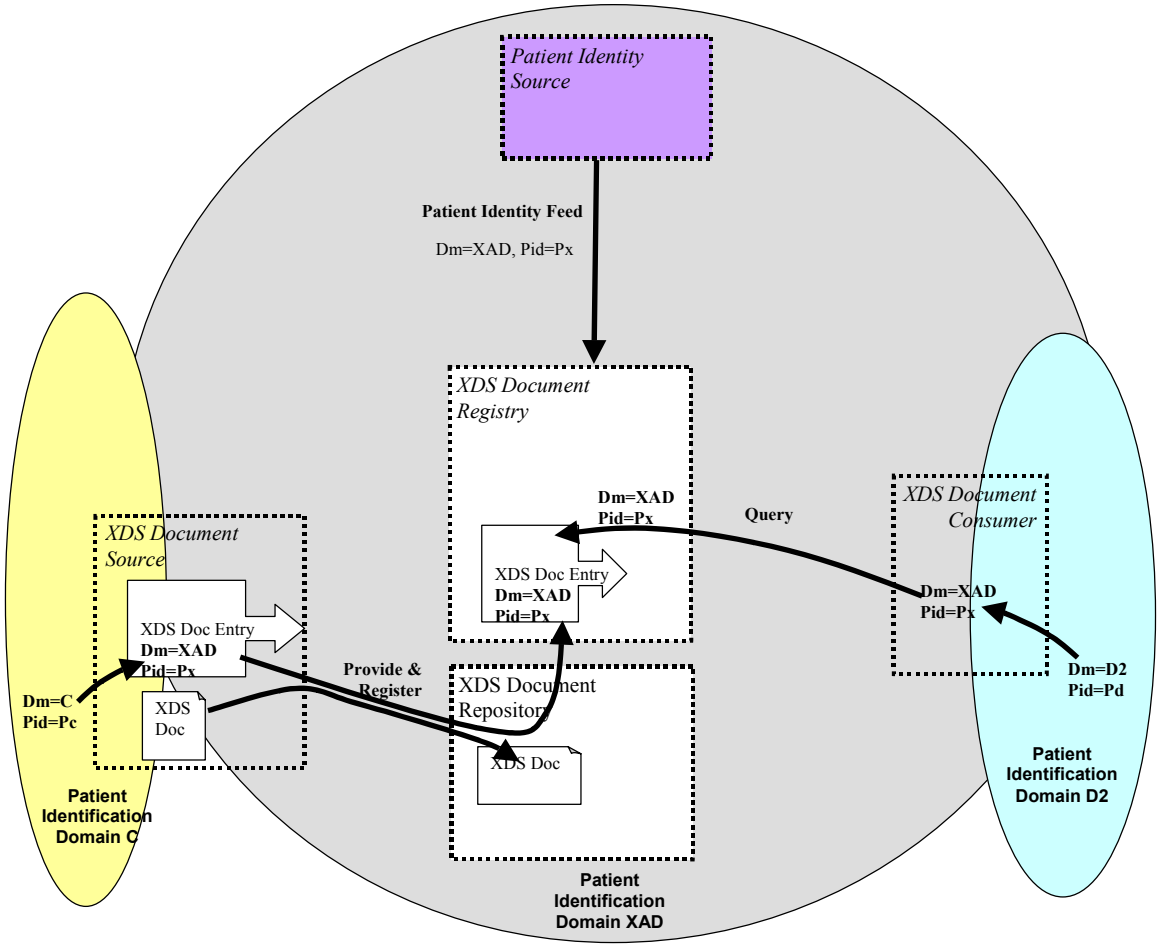


Figure 10.4.9-1 Affinity Domain with patient ID cross-referencing internal to the EHR-CRs

10.4.10 Document Lifecycle

10.4.10.1 Document Availability Status

Each XDS Document contained in a XDS Document Registry will be assigned one of the following Availability Status codes:

Approved: Available for patient care (assumes that it is authenticated, if applicable)

Deprecated: Obsolete, but may still be queried and retrieved

The XDS Document availability status is set to "approved" after the XDS Document Repository and the XDS Document Registry have successfully processed a submission request.

Note: ebXML Registry Services defines a Status of Submitted, which is used in a transient manner to provide an atomic submission. It is not significant to make this specific status externally visible.

An "approved" XDS Document may be changed to "deprecated" under the primary responsibility of its original Document Source with possible patient supervision. It is part of security policies that are beyond the scope of the XDS Integration Profile to have the XDS Repository/Registry enforce this ownership. The reason and responsible party for deprecating a document are tracked as part of the XDS Document Registry audit trail, which is a required capability. A "deprecated" Document remains available for Document Consumer queries. Except for the status change, a "deprecated" Document Entry metadata remains the same as when it was in the "approved" status.

An "approved" or "deprecated" XDS Document Entry may be deleted. This change is associated with the decision to completely remove a Document from an XDS Document Repository and the corresponding Document Entry from the XDS Document Registry. The XDS Affinity Domain shall establish the security policies associated with Document deletion. There are no transactions defined by this Integration Profile to support such operation.

10.4.10.2 Document Relationships

XDS Documents may be related to predecessor documents by one of three methods:

- Replacement,
- Addendum or
- Transformation

These relationships between XDS Documents are tracked in the XDS Document Registry. The parent relationship attribute contained in the metadata of such Documents is a coded value that describes the type of relationship. An original Document has no parent and consequently its parent Id and parent relationship are absent. XDS Document Registry shall reject submissions that contain relationships to documents that are not registered or have been "deprecated". Document stubs are supported by XDS to allow for a valid relationship to a known but not registered Document.

A replacement document is a new version of an existing document. The replacement document has a new document Id; its parent Id attribute contains the document Id of the Document Entry associated with the previous version of the XDS Document, and parent relationship contains the code "RPLC". The Document Entry for the previous version shall have its Availability Status changed to "deprecated".

An addendum is a separate XDS Document that references a prior document, and may extend or alter the observations in the prior document. It modifies the parent document, but the parent document remains a valid component of the patient record and shall remain in the state "approved" or available for care. The addendum XDS Document metadata contains the identifier of the previous XDS Document version in parent Id, and its parent relationship contains the code "APND".

A transformed document is derived by a machine translation from some other format. Examples of transformed documents could be CDA documents converted from DICOM Structured Reporting (SR) reports, or a rendering of a report into a presentation format such as PDF. The transform XDS Document contains the document Id of the previous version in parentId, and its parent relationship contains the code "XFRM". Affinity Domains may define rules that determine whether or not a transformed XDS Document replaces the source, but typically this would not be the case. If it is, an additional parent relationship of type "RPLC" is to be used.

10.4.11 Document Query

Query return info shall be either:

- a list of Registry Objects Values (e.g. XDS Document Entries)
- a list of Registry Objects UUIDs. This allows an XDS Document Consumer to receive a potentially long list of matching entries and to request them by subsets.

10.4.12 Transport Modes

The XDS Integration Profile defines an on-line mode of transport for all transactions except for the Provide & Register transactions where an off-line mode option is supported both for the Document Source and the Document Repository. In the "on-line mode" the transaction between two actors (computer applications) requires their simultaneous presence (e.g. an HTTP GET). In the "off-line mode" the transaction between the two actors (computer applications) does not require their simultaneous presence (e.g. a store and forward e-mail exchange).

1. An HTTP-based protocol (SOAP with Attachments) will be used for on-line operation.
2. The SMTP protocol will be used for off-line operation.

10.5 Implementation Strategies

The XDS Integration profile addresses the requirements of three major implementation strategies reflecting different groupings of actors within an EHR-CR as well as different configurations of the EHR-LR. This range of implementation strategies reflects the need to accommodate a

variety of workflows and configurations. These implementation strategies may coexist in some environments. Other implementation strategies are possible.

- Strategy 1: Repository at the Source. A single information system acts as both the Document Source and Document Repository for the documents it creates and registers with the Document Registry

  Upon completion of a phase of care, an EHR-CR will register a submission-set of documents in a Document Repository Actor with which it is grouped (same system). Then it registers this set of documents (newly created and priors documents of interest) with the Document Registry Actor[2].

  Any other Document Consumer Actor in the Affinity Domain may query the Document Registry Actor to find documents related to all phases of care for the patient [3]. It may choose to retrieve some of these documents from any Document Repository Actor [4].

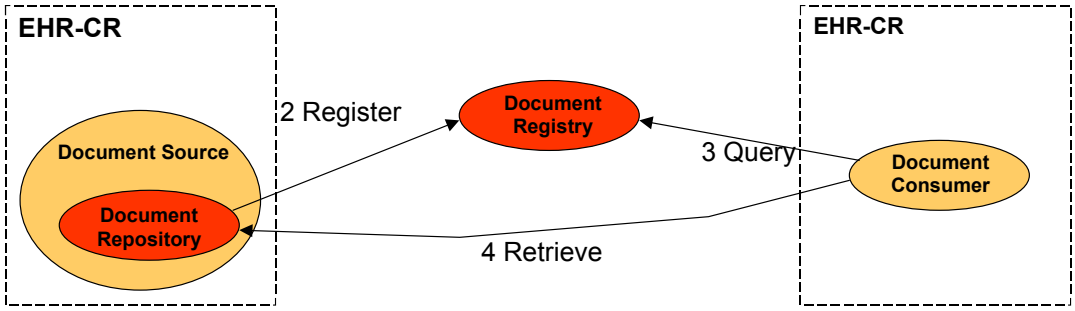


Figure 10.5-1 Implementation Strategy with Repository at the Source

- Strategy 2: Third Party Repository. The EHR-CR does not wish to be a Document Repository Actor, but rather uses the services of a third party Document Repository Actor to which it entrusts the documents it creates. First it provides both the metadata and the set of documents to this Document Repository Actor [1], which in turn forwards the registration request for the set of documents (newly created and prior documents of interest) to the Document Registry Actor [2].

  Any other Document Consumer Actor may query the Document Registry Actor to find out about documents related to all phases of care for the patient [3]. It may choose to retrieve some of these documents from any Document Repository Actor [4].

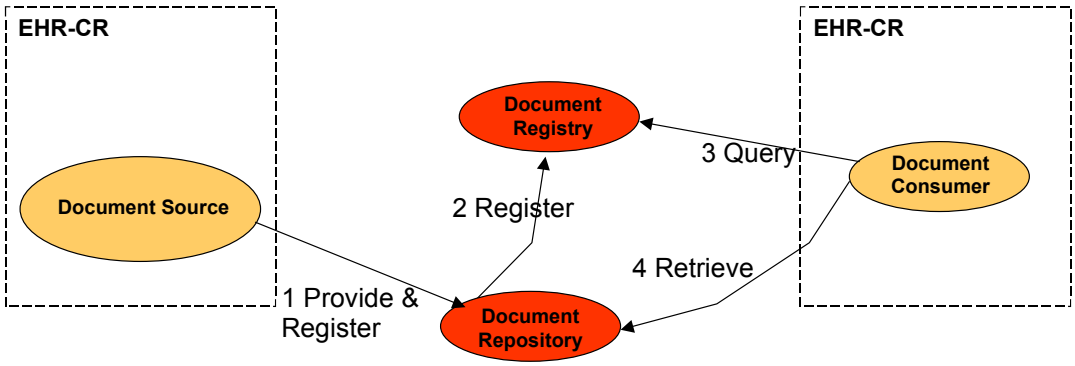


Figure 10.5-2 Implementation Strategy with 3rd party repository

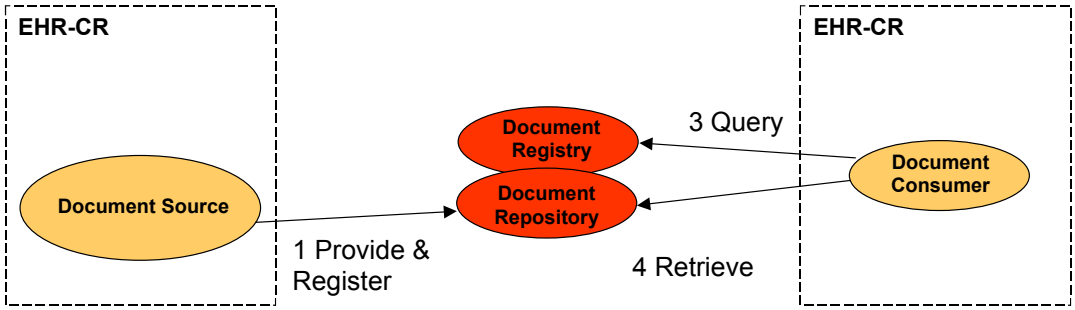


Table 10.5-3 Implementation Strategy with 3rd party central repository and registry

- Strategy 3: Direct Patient Transfer-Referral. The Document Source Actor completes a phase of care for a patient. It decides to directly provide and register [1] the set of documents (newly created and prior documents of interest) with a Document Repository [2] that has been grouped along with the Document Registry with the EHR-CR Document Consumer (Grouped Actors).

  In this case the span of the Clinical Affinity Domain may be quite limited as it could be defined to cover only the two EHR-CRs. However the same transaction [1] applies. Note that, in this implementation strategy the other transactions, although supported by the actors, are not used by the Document Consumer since the Document Registry and Document Repository reside within the Document Consumer.

Figure 10.5-4 Direct patient referral with registry and repository at consumer

Patient access to an EHR-LR may be supported by a specialized EHR-CR (i.e. a portal) implementing the Document Source and Document Consumer Actors.

Change Appendix E of Volume 1. Split the current appendix into two, adding subsections where necessary. Title the two appendices: Appendix E - Cross Profile Considerations, and Appendix G - Security Considerations.

Change title of section E to E.1. Add Section E – Cross Profile Considerations. Add the following sections to Section E of Volume 1 of the IHE ITI Technical framework

Appendix E: Cross-Profile Considerations

E.2 XDS Integration with RID

The RID Retrieve Document for Display transaction [ITI-12] is compatible with the XDS Retrieve Document transaction [ITI-17]. Thus, an RID Information Source implementing the Retrieve Document for Display transaction can be used to implement the XDS Retrieve Document transaction. In this instance, the RID Information Source must be a secure node [see ATNA].

E.3 XDS Integration with PIX

All Patient IDs managed in the XDS transactions (either in XAD-Pid Domain or in an EHR-CR Domain) shall include the related Patient Domain ID (OID of the Assigning Authority) associated with the patient ID. It is recommended that this unambiguous patient identification be used with Patient IDs within the Documents also.

Because XDS is Document content neutral, there is no verification by the XDS Repository that the Patient IDs included inside the documents are consistent with the patient IDs managed by the Registry in the document entry related to that document.

.

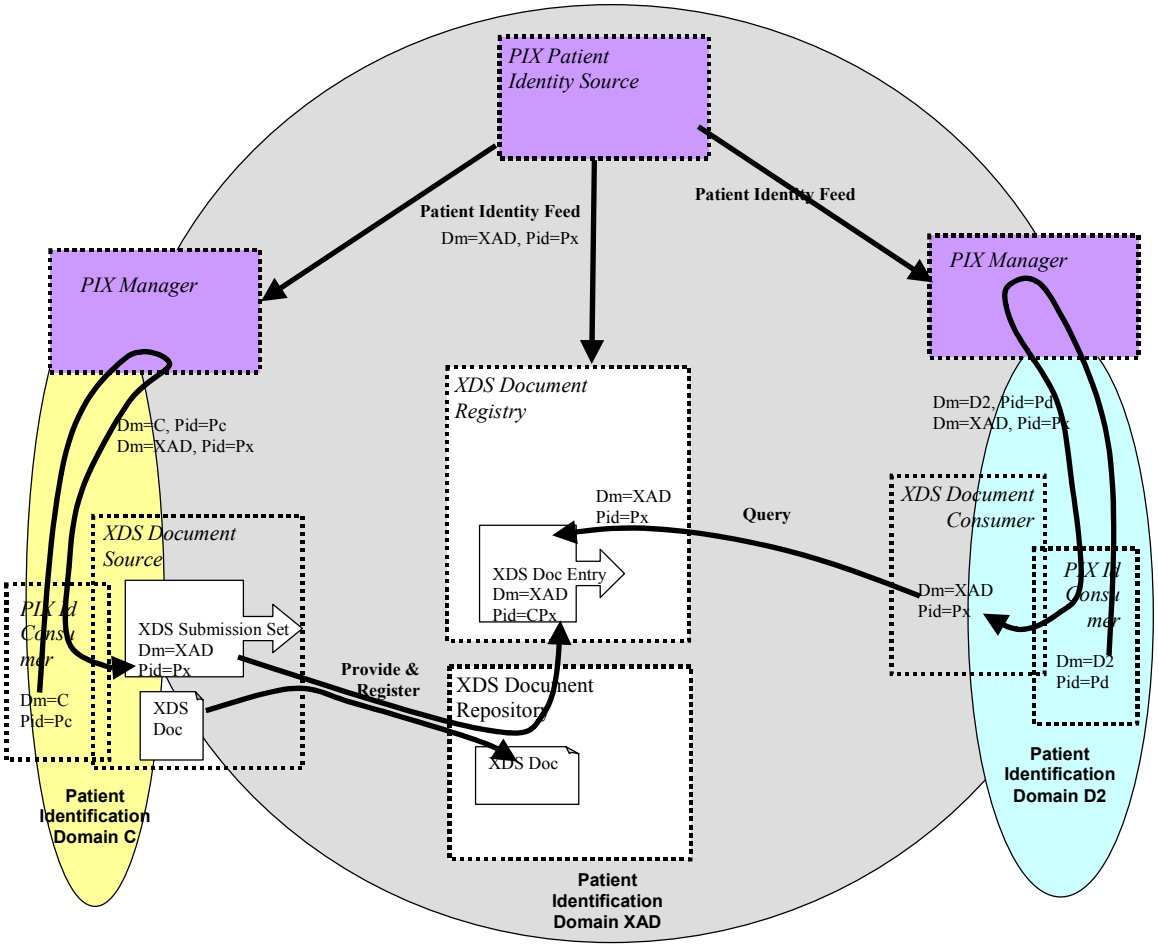


Figure E.3-1 Affinity Domain with patient ID cross-referencing with IHE PIX Managers

Figure E.3-1 depicts an example of a Affinity Domain with a Patient Identifier Domain (called XAD) and two EHR-CRs where the cross-referencing is performed by Patient Identifier Cross Referencing Managers internal to both the Document Source and the Document Consumer Domains (called C and D2 respectively).

A Document Source may choose to perform the cross-referencing of its own patient IDs in that of the XAD-Pid Domain by leveraging the IHE PIX Integration Profile (See Figure). The Patient ID Feed Transaction from the XAD Patient ID Source may be used to provide input to the Patient Identifier Cross-Referencing Manager used by the Document Source. The PIX manager may either be internal to the EHR-CRs or be shared across the XDS Affinity Domain

E.4 XDS Integration with PWP

The XDS Document Source Actor in the XDS Integration Profile may choose to utilize the PWP Query Personnel White Pages [ITI-24] transaction to obtain information needed to fill the

authorPerson and legalAuthenticatorName fields for the XDS Register Document Set [ITI-14] and Provide & Register Document Set [ITI-15] transactions.

The Personnel White Pages transaction defines, in ITI-TF 2:3.24.4.1.2.3.1, a "cn" attribute with "lang-x-ihe" that contains the information in the HL7 XCN (extended composite ID number and name for persons) format for personal information. These fields are optional in the PWP Integration Profile. A care delivery organization may choose to populate these fields in their Personnel White Pages Directory and utilize the ITI-24 transaction to support its XDS activities. This is not a required dependency, but is a possible reason to group a Document Source Actor with a Personnel White Pages Consumer Actor.

The PWP Integration Profile only provides the personnel information. Organizational information must be obtained via other means, e.g. extending the LDAP directory with organizational objects.

E.5 XDS Integration with PDQ

The Patient Demographics Query (PDQ) Integration Profile may be used in conjunction with the XDS Integration Profile to provide a lookup for XDS Affinity Domain Patient Identifiers to XDS Document Consumer and Document Source Actor. In this case a Patient Demographics Supplier Actor needs to be grouped with the XDS Patient Identifier Source Actor on one hand, and on the other hand a Patient Demographics Consumer Actor needs to be grouped with the Document Source/Consumer where one may want to query based on local patient traits and obtain a pick-list of candidate Patient Ids in the XAD Patient Identifier Domain. This offers a simpler solution that the use of the PIX Integration Profile.

Appendix F: Intentionally left blank

Appendix G: Security Considerations

G.2 XDS Security Considerations

Security and privacy

Coordinating the security and privacy policies of all the care delivery organizations in an Affinity Domain may be a challenge. An agreement is needed on security procedures, goals, auditing, record keeping, etc. This can result in changes to other enterprise policies, such as human resources procedures. Affinity Domain members are delegating full access to their published data to the other members of the Affinity Domain. This relationship requires a close ongoing partnership that ensures ongoing maintenance of policies, procedures, and activities. If laws change, relevant policies must be adjusted throughout the group. Corporate changes to group members affect the policies. Security events must be managed as a group. This must be managed as a long-term activity, not a one-time event.

Particular problem areas are likely to be:

- Authorized access and modification policies. The details of access policies are likely to have enterprise differences and conflicts that must be resolved. The Affinity Domain relationships also introduce new policy requirements. For example, changes to employment (e.g. employee hiring and firing) must now include suitably rapid notifications to other Affinity Domain members. Changes to privacy restrictions (e.g. divorces) now require full Affinity Domain notifications, not merely enterprise notifications.
- Audit trail and access record keeping are often quite sensitive internal enterprise activities that must now be appropriately coordinated with the full Affinity Domain.
- Changes to laws and regulations now affect not only the policies of the individual enterprises; they also must be reflected in the Affinity Domain relationship contracts, policies, and procedures.
- Patient access and patient identity management. Patients usually have insecure computers. Patients often object to security procedures.
- Transborder communication of Personal Health Information (PHI) often presents legal and regulatory issues.

ITI TF-2: Appendix J in volume II goes into more detail listing many of the threats, objectives, policies, and mitigations that need to be coordinated among Affinity Domain members.

The XDS Integration Profile for two main reasons does not prescribe such Security and Privacy policies. First, it is clear that the broad range of possible solutions to these policies that will depend on the legal framework and the types of healthcare system, calls for XDS to be offer such flexibility. Decisions in this domain will have some impact on the implementations of XDS Actors, but it is expected that these will be minimal.

Appendix J: Content and Format of XDS Documents

The XDS Integration Profile purposely leaves a number of policies up to the XDS Affinity Domain to decide, including the structure and format of the content of XDS Documents to be shared, the mapping of content metadata into the XDS Document Registry, the coding of XDS Document metadata, the events that trigger an XDS Submission Request, and the policies concerning the use of XDS Folders to facilitate sharing.

It is important to recognize that until sufficient experience has been gained in cross-enterprise document sharing, it is not possible to establish common or even best practices in the use of the XDS Integration Profile. IHE has therefore chosen to abstain to make recommendations in these topics at this time.

IHE also recognizes that there will be a need for content-oriented integration profiles to be used in cooperation with this Integration Profile. It is expected that in the future the various IHE Domains (Cardiology, Laboratory, Radiology, IT Infrastructure, etc.) will produce IHE Integration Profiles refining the use of XDS within the domain. These various content-oriented integration profiles may rely on XDS, but would further constrain the forms of documents to be shared, or the uses of XDS features such as Folders and Submission Sets, et cetera.

Content Neutrality

XDS is content neutral. It neither prescribes nor prohibits the format, content, structure or representation of documents that can be retrieved from an XDS Document Repository. For the XDS Integration Profile to have immediate value to a Clinical Affinity Domain, it must be able to adapt to the documents which are present and available from its members. Thus, prohibitions on content would only serve to limit the utility and adoption of the XDS Integration Profile. Similarly, Clinical Affinity Domains must be able to adapt to emerging standards, which cannot be enumerated in any list of prescribed content formats.

IHE strongly recommends that XDS Affinity Domains adopt rules that require documents to comply with widely accepted standards where possible (*e.g.*, HL7 CDA, CEN ENV 13606, ASTM CCR, DICOM Composite Object).

Document Headers and Metadata

Because XDS is content neutral, XDS cannot validate metadata contained within the body of an XDS document against the metadata supplied to the XDS Document Registry. XDS Affinity shall therefore select content where IHE has defined Integration Profiles, or until that point, the Affinity Domains shall carefully define how the attributes in the XDS Document Registry are filled.

Metadata and the Patient Record

Although metadata in the document header may be duplicated in the XDS Document Registry, the XDS Document Registry metadata has a particular role in term of being part of the legal medical record stored. It is definitively not part of the clinical record as managed by the XDS Document Repositories where documents reside. Furthermore, XDS does not provide for

transactions to "sign" or legally authenticate the content of an XDS Submission Set, although it offers the ability to track its author, if the Affinity so desires to enforce it. However, the contents of XDS Folders is not tracked, as Folders in XDS have been designed to facilitate collaboration, but not to constraint it. However, the existence of document metadata in the registry and the potential medical acts involved in creating an XDS Submission Set or XDS Folder may make the contents of the XDS Document Registry part of the patient's legal medical record. It will be up to individual XDS Affinity Domains to decide how to address the issues involved with these clinical acts and to resolve them in accord with common sense, acceptable medical practices, and local regulations.

Add the following section to Volume 1 of the IHE ITI Technical framework

Appendix K: XDS Concepts Details

K.1 XDS Document Concept

An XDS Document is the smallest unit of information that may be provided to a Document Repository Actor and be registered as an entry in the Document Registry Actor.

An XDS Document is a composition of clinical information that contains observations and services for the purpose of exchange with the following characteristics: Persistence, Stewardship, Potential for Authentication, and Wholeness. These characteristics are defined in the HL7 Clinical Document Architecture Release 1 specification.

An XDS Document may be human and/or application readable. In either cases, it shall comply with a published standard defining its structure, content and encoding. IHE intends to define content-oriented Integration Profiles relying on such content standards to be used in conjunction with XDS.

Furthermore:

1. When submitted for sharing, an XDS Document shall be provided to the Document Repository Actor as an octet stream with an associated MIME type.

2. When retrieved through the Retrieve Document transaction, an XDS Document shall be unchanged from the octet stream that was submitted (full fidelity repository).

   Note: An XDS Document may be a MIME multipart document (e.g. an HL7 CDA as its first part followed by attachments as files). The first part of the multi-part contains the primary part of the document, other parts are direct attachments to the primary part. The Document Repository handles this multi-part data set as an "opaque entity". The Document Repository does not need to analyze or process its multi-part structure nor the content of any parts in the context of the XDS Integration Profile.

   Note: An XDS Document may be retrieved using alternate methods using document specific retrieval methods. Such optional capabilities are not provided in the current specification of XDS, but are possibly candidates for addition as future options this Integration Profile.

3. An XDS Document shall be associated with metadata defined by the Document Source. This metadata information shall be placed by the XDS Registry Actor in an XDS Document Entry, and is used for query purposes by XDS Consumer Actors.

4. The XDS Integration Profile manages XDS Documents as a single unit of information, it does not provide mechanisms to access portions of an XDS Document. Only the Document Sources or Document Consumers have access to the internal information of the XDS Document.

5. An XDS Document is globally uniquely identified, so that no two XDS Documents with different content shall bear the same Unique Identifier. This identifier is unique across all Clinical Affinity Domains, which allows potential merger of XDS Document

Repositories from different domains, or exchange of XDS Documents between Clinical Affinitiy Domains, if so desired.

6. The XDS Document Registry Actor shall maintain a single document entry for each XDS Document stored in a Document Repository Actor. Duplicate copies of the same XDS Document (with the same unique identifier) may be stored and registered. Registration of an XDS Document with the same unique identifier but a different content is rejected.

7. This Integration Profile specifies the metadata required for each XDS document registered in the Document Registry. It is the responsibility of the Document Source to ensure that the XDS Document metadata reflects the actual content of the associated XDS Document. Neither the Document Repository nor the Document Registry checks this consistency.

8. The Document Source maintains the following responsibilities over the XDS Documents it has registered:

    a. It has rights to change the status of any of these Documents from "approved" to "deprecated" or to delete them outright.

    b. It has rights to submit an XDS Document with a "Parent Relationship" of replacement ("RPLC") for one of its previously submitted document[2].

Clinical Affinity Domains should have policies and procedures to provide patient access to these operations where necessary. For example, in certain regions, patients may request the removal of documents from the EHR-LR. The Registry and Repositories implementations should be ready to support these local operations although there are no IHE transactions defined at this time.

K.2 Concept of an XDS Affinity Domain

An XDS Affinity Domain is made of a well-defined set of Document Repositories and Document Consumers that have agreed to share the clinical documents. An Affinity Domain has a number of properties defined:

1. An Affinity Domain does not deliver care. Only the EHR-CRs belonging to an XDS Affinity Domain as Document Sources and Consumers do.

2. An Affinity Domain is managed by a single Document Registry Actor.

    Note: A distributed registry approach will be considered as a future and separate Integration Profile. For Document Source and Document Consumer Actors, the perception of a single Document Registry Actor hides the complexity of a distributed registry.

[2] For example, in DICOM, where the document identity does not change even though its internal patient metadata may have been updated, the Document Source would submit an updated DICOM Document as a replacement for the existing one.

3. It includes any number of Document Repository Actors (a distributed configuration is the default, however, a centralized configuration with a grouped Registry/Repository is also supported).
4. It contains an explicit list of Document Consumer and Document Repository actors that participate in document sharing. The addition of a Document Repository or Document Consumer Actor is an administrative task that requires involvement of authorities maintaining the Registry and Repositories.
5. There is a chain of trust established between the users (healthcare staff) in each EHR-CR and the Affinity Domain.
6. Document Repositories and Document Consumers may belong to more than one Affinity Domain and share the same or different documents. This is an implementation strategy and will not be further described.
7. The Affinity Domain supports a primary Patient Identification Domain that is used by the Document Source and Consumers to communicate with the Document Registry. When Document Sources and Consumers in the Affinity Domain belong to different Patient Identifier Registration Domains, the Document Source and Consumers must cross-reference their own Patient Identifier Registration Domains to that of the Registry. They may use the IHE Patient Identifier Cross-referencing Integration Profile, the IHE Patient Demographics Query Integration Profile or other Affinity Domain specific mechanisms for cross-referencing (See ITI TF-2 Appendix E Sections E.3 and E.5).
8. A Document Source may only contribute documents with Document Codes and Health Facility Codes that draw from a Vocabulary Value Set that is approved by the Affinity Domain.

K.3 Other Principles of XDS

The XDS Integration Profile has been designed with the following limitations and principles:

1. A Document may contain references to other documents in its content which are not under the management of the XDS Document Registry. Such references may be available to the EHR-CR that registered the document that includes the reference. It is beyond the scope of XDS to provide access to such documents internal to the EHR-CR.
2. The XDS Repositories are not expected to perform any processing or translations on document content. Processing and translation are the responsibility of a Source EHR-CR or Consumer EHR-CR. The analysis, cross-document combination and presentation of document content is outside the scope of the XDS Integration Profile and its actors.
3. The custodianship for the clinical information contained in a registered document remains with the Source Actor of the EHR-CR. The EHR-LR offers only a “shared space” under the responsibility of each contributing EHR-CR. Through XDS, replacement or deletion of documents in the EHR-LR may only be initiated by the corresponding EHR-CR Source.

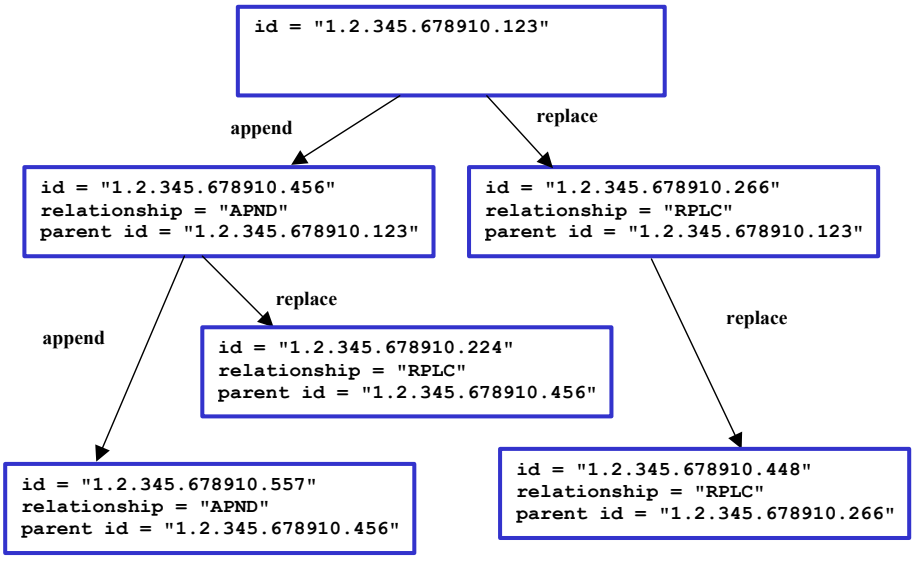
4. When an XDS Document that has already been registered in the XDS Registry of a Clinical Affinity Domain is resubmitted as if it was a new XDS Document with the same Document Unique identifier, this "duplicate submission" is detected by the Repository and/or Registry based on the fact that the XDS Document Unique Identifier already exists in a Document Entry. The submission request to which that resubmitted Document belongs shall be rejected in the case where the identifiers match but the actual content differs (detected by use of a hash key computed by the Document Repository at the time of submission).

K.4 Document Identification

In order to reduce the number of unique identifiers associated with an XDS Document, the globally unique Document Id assigned by the document source and the unique XDS Document Id used by the Repository are the same. It is strongly recommended to limit the use of the Document Entry UUID created per ebRS in order to reference the document entry for referencing internally to the encoding of the IHE transactions operations, and to encourage the use of the globally unique Document Id for all external operations (e.g. links maintained in data bases internal to the Document source Actor, links within documents, etc.).

The XDS Document Entry includes two separate attributes: an XDSDocument.uniqueId and XDSDocument.URI, a Universal Resource Identifier. The URI is a "self contained" web method that allows any Document Consumer to perform a Retrieve Document transaction (See ITI TF-2: Section 3.17). The Document Unique ID is a location independent identifier. As the result of XDS Document migration from one XDS Document Repository to another one within an Affinity Domain, the URI would be changed, but not the Document unique ID.

K.5 Example of Document Relationship



Adapted from HL7 CDA Release 2

Figure 10.4.10-5 Example of Document Relationships

These relationships are illustrated in the above figure. Typical scenarios are a simple relacement (e.g. XDSDocument.id "1.2.345.678910.266" replacing XDSDocument.id "1.2.345.678910.123") and a simple addendum (e.g. XDSDocument.id "1.2.345.678910.456" appends XDSDocument.id "1.2.345.678910.123"). More complex scenarios that might be anticipated include:

1. Replacement of an addendum (e.g. XDSDocument.id "1.2.345.678910.224" replaces XDSDocument.id "1.2.345.678910.456", which itself is an addendum to XDSDocument.id "1.2.345.678910.123") - expected behavior would be to render the replacement as the addendum (e.g. render XDSDocument.id "1.2.345.678910.224" as the addendum to XDSDocument.id "1.2.345.678910.123");
2. Addendum to a replaced document (e.g. XDSDocument.id "1.2.345.678910.456" appends XDSDocument.id "1.2.345.678910.123", which has been replaced by XDSDocument.id "1.2.345.678910.266") - expected behavior would be to render the addendum along with the replacement (e.g. render XDSDocument.id "1.2.345.678910.456" as an addendum to XDSDocument.id "1.2.345.678910.266").

K.6 Off-Line transaction mode

Document Source Actors are allowed to be off-line part of the time, as in the case of a doctor's office system connected only by a dial-up line acting as a Document Source.

The Document Registry and Document Repositories should be designed to be on-line all the time (see note for exception).

Note: The Document Repository may be off-line also in the degenerate case of point-to-point e-mail transmission, where the affinity domain is made only of two systems; on one hand a document source and on the other a document repository grouped with the Document registry and Document Consumer (See ITI TF-1: 10;5 Strategy 3).

Information sent to off-line systems will be supported through Internet e-mail protocols. E-mail protocols provide mechanisms for sending acknowledgments:

(1) Delivery receipts from the end-user, and

(2) Delivery failure notices from intermediate store-and-forward SMTP servers.

When using e-mail protocols, the asynchronous nature of the acknowledgments, which are delivered by e-mail messages, requires that the Send and Acknowledge components of the transaction be separated into distinct messages.

Body of the e-mail message should contain a simple notice (in English/ASCII), fixed subject line, address should be used for automated processing. An attachment formatted in the local language should contain instructions. Transaction should be included in a separate attachment.

Add the following section to Volume 1 of the IHE ITI Technical framework

Appendix L: XDS Affinity Domain Definition Checklist

The concept of an XDS Affinity Domain is defined in ITI TF-1:10 and Appendix K. This informative appendix summarizes the key policies that need to be agreed to in order to deploy a EHR-LR document sharing environment.

L.1 Configuration of an XDS Affinity Domain

A number of systems implementing IHE Actors defined in the XDS Integration Profile need to be identified and configured to communicate. This includes defining addressing information and ATNA node certificate:

1. Identify the system that will support the Document Registry Actor.
2. Identify the systems that will support the Document Repository Actors.
3. Identify the systems that will support Document Source and/or Document Consumer Actors.

L.2 Patient Identification

Initialize the XDS Document Registry (See ITI TF-2:Appendix H) with the proper patient identification information:

1. Assign an Assigning Authority (OID) for the XDS Affinity Domain Patient Id Domain.
2. Assign an Assigning Authorities (OID) for the each one the Local Patient Id Domains in which the EHR-CR Document Source and/or Document Consumer operate.
3. Identify the system that will support the Patient Identity Source and if some of the systems that support Document Source and/or Document Consumer Actors also support a Patient Identity Cross-reference Manager (needs to receive a patient identity feed Transaction).

L.3 XDS Registry Related Vocabularies

Initialize the XDS Document Registry (See ITI TF-2:Appendix H) with the proper vocabulary information:

1. Select and initialize the XDS Document Registry as well as the Document Sources and/or Document Consumers with the vocabulary definitions specified in Registry Enforcement (ITI TF-2:3.14.4.1.2.9) where either the Coding Scheme or the Coding Scheme/Code Values are enforced.

L.4 Document Sharing Practice Policies

1. Define the care events and the corresponding expected level of information that is expected to be shared within the EHR-LR.

2. Define the usage policies for XDS Folder (creation and update) in the selected care pathways supported.

L.5 XDS Document Content

1. For each Document Format Code Value, establish the necessary interoperability agreements (e.g. by selecting IHE Document Content Profiles) to ensure that the Document Consumers may find (e.g. Document UniqueId structure) and process the XDS Documents content (e.g. MIME type, template definitions, archetypes, etc.) they retrieve from the XDS Repositories of the XDS Affinity Domain.

L.6 Document Update and Maintenance Policies

Document Sources are responsible for the on-going accuracy (custodianship) of the XDS Documents they have elected to shared in the EHR-LR supported by the Affinity Domain. This includes:

1. Replacement of documents in the EHR-LR
2. Cases and means to possibly delete documents in the EHR-LR

L.7 Security and Privacy Policies

1. Establish agreed policies and procedures among care delivery organizations in the Affinity Domain. In particular address security considerations in ITI TF-2:Appendix K.
2. Establish operational security infrastructure, including certificate exchange.
3. Maintain operational security infrastructure, configuration management, audit management, periodic inspections, etc.

Add the following section to Volume 1 of the IHE ITI Technical framework

Appendix M: Cross-Enterprise Document Sharing and IHE Roadmap

The IHE Cross-Enterprise Document Sharing Integration Profile is part of a family of IHE Integration Profiles grouped in a number of domain-specific Technical Frameworks (Cardiology, Laboratory, Radiology, IT Infrastructure, etc.). XDS is a central foundation for Cross-Enterprise interoperability that may be combined with a number of the existing IHE Integration Profiles (See ITI TF-1:Appendix E). However a number of new IHE Integration Profiles need to be developed, pending the availability of the relevant base standards.

M.1 Document Content Integration Profiles for XDS

It is expected that the various IHE Domains (Cardiology, Laboratory, Radiology, IT Infrastructure, etc.) will produce new IHE Integration Profiles addressing the content of the documents that need to be shared. These various "content-oriented" Integration Profiles will rely on the XDS Integration Profile for managing the registration, discovery and access processes in a common manner.

Such an effort is underway with the ASTM CCR (Continuity of Care Record) care in the community general documents (e.g. referrals).

M.2 Cross-Enterprise Dynamic Information Sharing

The management of dynamic information (non-document-oriented) such as allergy lists, medication lists, problem lists, etc is not addressed by XDS. However, a means to access this information in a structured form and to manage updates to such dynamic clinical information is a candidate for a specific Integration Profile.

M.3 Collaborative Workflow Process Management

There is a wide array of shared care delivery collaborative processes such as the placing and tracking of orders (e.g. drug prescriptions, radiology orders, etc.) for which XDS provides only a partial solution (the creation of the patient record with the resulting persistent artifacts). XDS offers a critical infrastructure for ePrescribing and eReferral in that it can ensure that the various providers share access to orders, prescriptions, dispensations, and results. The means to interoperate on the command/control part of these collaborative workflow processes is a candidate for specific Integration Profiles in the future.

M.4 Security and Privacy Management

The operation of any XDS Clinical Affinity Domain will require that a proper security model be put in place. It is expected that a range of security models should be possible. Although the XDS Integration Profile is not intended to include nor require any specific security model, it is expected that XDS implementers will group XDS Actors with actors from the IHE Audit Trail and Node Authentication and will need an Access Control capability that operates in such a

cross-enterprise environment. New IHE Integration Profiles have been identified as candidates (e.g. Public Key Infrastructure, Access Control, etc.).

M.5 Federation of Affinity Domains

XDS is an effective means to establish Affinity Domains that include care delivery organizations at any level, local, regional or national. However, the establishment of independent but consistently XDS-based Affinity Domains will call for their federation, as patients expect their records to follow them as they move from region to region, or country to country. IHE foresees a need for transferring information from one Clinical Affinity Domain to another, or to allow access from one Affinity Domain to documents managed in other Affinity Domains. XDS has been designed with this extension in mind. An XDS Domains Federation Integration Profile that complements XDS may be anticipated in the future.

Volume 2 – Transactions

Add the following section to Volume 2 of the IHE ITI Technical framework

3.14 Register Document Set

This section corresponds to Transaction ITI-14 of the IHE IT Infrastructure Technical Framework. Transaction ITI-14 is used by the Document Repository Actor to register a set of documents with the Document Registry.

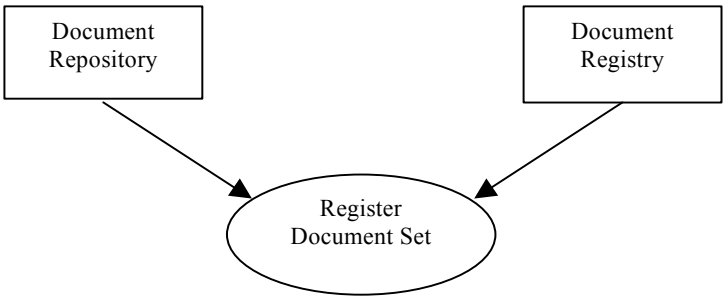
3.14.1 Scope

The Register Document Set transaction passes a Submission Request from a Document Repository Actor to a Document Registry Actor.

A Register Document Set transaction carries:

- Metadata describing zero or more documents
- XDS Submission Set definition along with the linkage to new documents and references to existing documents
- XDS Folder definitions along with linkage to new or existing documents

3.14.2 Use Case Roles



Actor: Document Repository

Role: A document storage system that submits document metadata to a Document Registry.

Actor: Document Registry

Role: A document indexing system that receives and stores document metadata.

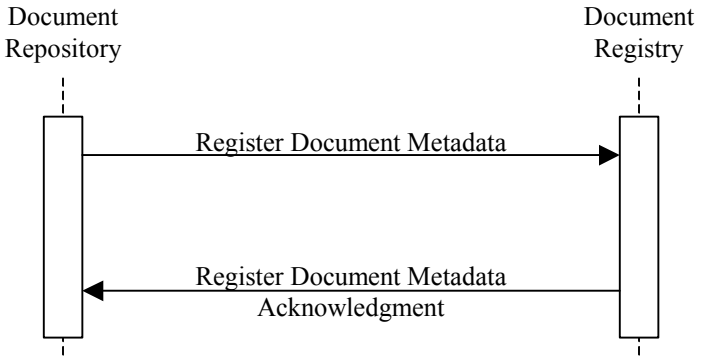
3.14.3 Referenced Standards

ebRIM OASIS/ebXML Registry Information Model v2.0

ebRS OASIS/ebXML Registry Services Specifications v2.0

1280	HTTP	HyperText Transfer Protocol HTTP/1.1 (IETF RFC2616)
	CDA	HL7 Clinical Document Architecture (ANSI/HL7 CDA R1-2000)
	HL7V2	HL7 Version 2.5

3.14.4 Interaction Diagram



1285 3.14.4.1 Register Document Metadata

The Document Repository sends metadata for a set of documents to the Document Registry.

3.14.4.1.1 Trigger Events

The Register Document Metadata message is triggered when:

1. A Document Repository wants to register metadata for a set of documents it holds.
1290 2. A Document Repository receives a Provide and Register Document Set transaction (ITI-15)

3.14.4.1.2 Message Semantics

The following sections specify the mapping of XDS concepts to ebRS and ebRIM semantics:

1. XDS Document
1295 2. XDS Submission Request
3. XDS Submission Set
4. XDS Folder
5. Document Relationships

Metadata definitions to support the above are discussed as follows:

1300 6. XDS Document
7. XDS Submission Request

8. XDS Submission Set
9. XDS Folder

In addition the following topics are discussed:

10. Protocol requirements
11. XDS registry adaptor function
12. Security Requirements

3.14.4.1.2.1 Class Diagram

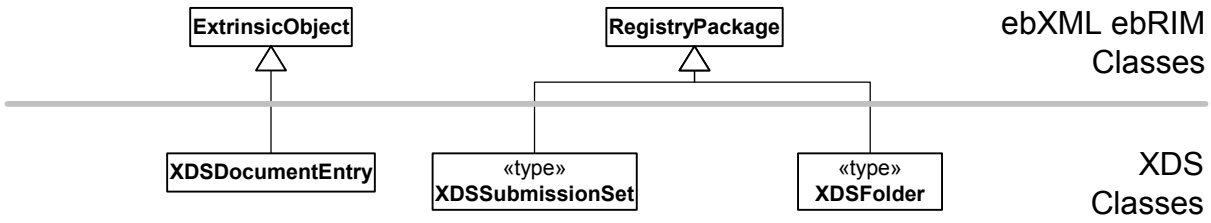


Figure 3.14.4.1-1 ebXML Class Diagram of the Register Document Metadata

The XDSDocumentEntry class is derived from the ebXML ExtrinsicObject class. The XDSSubmissionSet and XDSFolder classes are derived from the ebXML[3] RegistryPackage class. Since the ebXML Registry standard does not allow for subclassing the RegistryPackage class, these two classes are implemented as ebXML RegistryPackages. Type information (submission set vs. folder) is coded as an ebXML Classification against two object types created by the XDS profile, XDSSubmissionSet and XDSFolder.

3.14.4.1.2.1 Document Specification

A new registry object type is declared as a subclass of ebXML ExtrinsicObject. Its name is XDSDocumentEntry. An object of this type in the XDS registry is used to represent a document in an XDS repository.

An XDSDocumentEntry object in the registry contains a reference to a single document in a single repository.

Note: A repository may hold documents that are not indexed in the registry.

Appendix H defines the metadata to initialize an ebXML registry to serve as an XDS Document Registry.

[3] ebXML Registry terms such as RegistryPackage are shown with an ebXML prefix to help distinguish ebXML Registry terms from XDS terms. Unless otherwise indicated, references to 'ebXML' in XDS refer to the ebXML Registry specifications as opposed to other ebXML specifications. The short term is used for readability.

3.14.4.1.2.2 XDS Submission Request Specification

A Submission Request is the collection of information that is transferred to an XDS Document Registry or Repository.

There are two types of submission requests: XDS Registry Submission Request and XDS Repository Submission Request. Both are described below.

Appropriate protocol bindings are used to transfer this content between systems when the actors are not implemented together on the same system. The bindings are described in section 3.14.4.1.2.10.

The two types of XDS Submission Requests are described next.

3.14.4.1.2.2.1 XDS Registry Submission Request

An XDS Registry Submission Request is the collection of metadata transferred between a Document Repository and a Document Registry in a single ebXML SubmitObjectsRequest. This request contains:

- A collection of metadata to be stored in the registry including:
  - Metadata for new documents
  - Folders to be created
  - Documents to be added to folders
- A single XDS Submission Set, contained within the metadata, organizing the metadata

This request is part of the Register Document Set transaction.

3.14.4.1.2.2.2 XDS Repository Submission Request

An XDS Repository Submission Request is the collection of metadata and documents transferred between a Document Source and a Document Repository using a single ebXML SubmitObjectsRequest. This request contains:

- Metadata
- Zero or more documents; each document is represented by an XDSDocumentEntry object in the metadata. Submissions that add metadata to the registry without adding documents to the repository are possible.

This request is the information payload of the Provide and Register Document Set message of the Provide and Register Document Set transaction ITI-15.

Unless otherwise stated, the XDS Submission Set requirements specified hereafter apply to both types of XDS Submission Requests

3.14.4.1.2.2.3 Atomicity Requirements for XDS Submission Requests

XDS Submission requests shall be atomic operations. The result of a Submission Request is to update either:

- a Registry or
- a Registry and a Repository.

All changes requested are successfully applied or no net changes are made. More specifically:

1. Atomicity shall be managed by an XDS registry adaptor. (see section 3.14.4.1.2.11 for details on registry adaptor.addressing the fact that the ebXML Registry specification does not guarantee that a SubmitObjectsRequest is atomic). XDS specifies the mechanism through which atomicity is to be implemented and where it is needed.
2. All objects shall have their Status attribute set to Submitted when the objects are first created in the ebXML registry. An ebXML ApproveObjectsRequest, shall be issued within the XDS Registry Adaptor to change the Status attribute to Approved. This completes the transaction.
3. The following types of objects shall be have their status set to Approved to be considered publicly available:
    - XDSSubmissionSet (ebXML RegistryPackage)
    - XDSFolder (ebXML RegistryPackage)
    - XDSDocumentEntry (subclass of ebXML ExtrinsicObject)
4. If an error occurs storing documents in the repository then all documents stored as part of the Repository Submission Request shall be removed.
5. If an error occurs storing metadata in the registry, then the following actions are performed:
    - All metadata stored as part of the Registry Submission Request shall be removed from the registry
    - All documents stored as part of the Repository Submission Request shall be removed. This only applies if the Registry Submission Request is a result of a Repository Submission Request.
6. Registry queries from the Registry Query transaction shall not find XDS Submission Sets, XDS Folders or XDSDocumentEntry objects until after the above atomic operation that creates them has completed successfully and the status attributes have been set to Approved.

3.14.4.1.2.2.4 Other Properties of Submission Requests

A Submission Request may contain metadata beyond the XDS Submission Set, XDS Folder, and XDSDocumentEntry objects. These are:

- ebXML Associations linking XDSDocumentEntry objects to XDSFolder objects. There are no restrictions on whether the XDSDocumentEntry objects or XDSFolder objects are in this Submission Request. Such an Association is the ebXML mechanism for including objects in an ebXML RegistryPackage (the basis of XDSFolder).

- Associations linking existing (already contained in the registry) XDSDocumentEntry objects to the XDSSubmissionSet RegistryPackage contained in this Submission Request. This option is discussed in the next section.

3.14.4.1.2.3 Submission Set Specifications

Submission Sets exist for two reasons:

1. To support atomic submission to the registry
2. To make a permanent record in the registry of
   - The existence and status of the submission
   - The contents of the submission including XDS Folders and XDSDocumentEntry objects.

An XDS SubmissionSet is an ebXML RegistryPackage, classified as XDSSubmissionSet that is used to bundle XDSDocumentEntry objects.

A Submission Set has a set of attributes that are described in section 3.14.4.1.2.7 Submission Set Metadata.

Documents may be included in a Submission Set in two ways: inclusion by value and inclusion by reference.

Inclusion by value: A new document is being submitted to the registry. The Submission Set contains the XDSDocumentEntry object with associated attributes.

Inclusion by reference: Existing documents in the registry can be referenced by a Submission Set. These documents are included because of their clinical relevance to the rest of the Submission Set.

Linking document metadata to submission set: An XDSSubmissionSet shall be represented by an ebXML RegistryPackage. Document metadata (XDSDocumentEntry objects) shall be linked to the RegistryPackage via ebXML Associations according to the ebXML Registry standard.

For documents included by reference, the Submission Request shall include the Association object used to link the document. For documents included by value, the Submission Request shall include the XDSDocumentEntry object and the Association object used to link the document.

Submission Set Association labeling: Two types of association labels are defined: original (submission by value), or reference (Submission by reference). This allows finding the submission set that first submitted any document. It also supports proper rollback in case of a submission error. For document metadata included by value, a rollback of the submission shall delete the document metadata and the association. For document metadata included by reference, a rollback of the submission shall not delete the document metadata but shall still delete the association. (The document whose association is being deleted existed before this submission and shall be maintained.) The following labeling of the Associations is required.

.**Table 3.14.4.1-1 Submission Set Association Labeling**

Inclusion type	Rollback	Association Labeling
By Value	Yes	Slot: Name=SubmissionSetStatus Value=Original
By Reference	No	Slot: Name=SubmissionSetStatus Value=Reference

Submission Sets and patients: A Submission Set is restricted in terms of mixing documents from different patients. All documents included by value in a Submission Set shall have their patientId attribute set to the same value. This restriction does not apply to documents included by reference.

Document metadata duplication: There are several conditions regarding the duplication of document metadata that can occur.

- Duplicate registration of a document - A document and its metadata are submitted to the repository as part of a Repository Submission Request. This document already exists in one or more repositories and is already represented in the registry. It is submitted with a new (not previously used) UUID for the XDSDocumentEntry and associated ancillary objects. The registry shall accept such duplicate registration of the documents.
- Duplicate document id submitted to repository - A document with its associated metadata is part of a Repository Submission Request. A document with the same XDSDocumentEntry.uniqueID is present in the repository but the XDSDocumentEntry.hash is different. This is an error and the Submission Request shall be rejected by the repository.

  Note: There are two approaches to detecting this fault. First, this can be detected at the repository if repository logic can validate the hashes and has record of the document id to compare. Otherwise the request can be forwarded on to the registry and let the fault be detected by the registry (see next bullet). The repository then deals with the error returned by the registry.

- Duplicate document ID submitted to registry - Metadata representing a document (XDSDocumentEntry) is part of a Registry Submission Request. An XDSDocumentEntry object with the same uniqueID is present in the registry but, the hash is different. This is an error and the Submission Request shall be rejected by the XDS registry adaptor.

3.14.4.1.2.4 Folder Specification

An XDS Folder is an ebXML RegistryPackage classified as XDSFolder. This folder is used to bundle XDSDocumentEntry objects. Folders shall not be nested inside other folders. The patientId attribute of the XDSDocumentEntry objects it contains shall match the patientId attribute on the folder itself. This shall be enforced by the Registry Actor.

Note: The nesting of folders may be considered as a future extension to this transaction.

3.14.4.1.2.5 Document Relationships

XDS Documents may be derived from predecessor documents by one of three methods: replacement, Addendum or Transformation. These document relationship in the XDS Registry

are tracked in a manner similar to that defined by HL7 CDA. Two pieces of information shall be tracked by theXDS Registry: the link between documents (XDSDocumentEntry.parentDocumentId) and the method (or reason or code) for the link (XDSDocumentEntry.parentDocumentRelationship).

The linkage between documents is implemented via an ebXML Association object. New Association Types are defined to describe these relationships. The new values are listed in Table 3.14.4.1-2. The metadata to add these values to an existing registry is described in Appendix H.

The case when the XDS Registry does not know the predecessor document shall be supported by either the Document Source Actor (Provide&Register Transaction) or the Document Repository Actor (Register Transaction). In this case, the source object is known in the registry but not the target object (the parent object). For this situation, this Actor creates a "document stub" as a placeholder in the registry and links it to other metadata as necessary. A document stub is a registry object of type XDSDocumentStub.with only one XDS required attribute, its XDSDocumentEntry.uniqueId. That is all that is known about it.

The source of the metadata (Document Source or Document Repository) is responsible for detecting the need for such a document replacement and coding the above metadata to represent it in the registry.

During a Registry Query transaction, the Document Consumer can detect the document stub by its object type. This shall be interpreted as the document exists but the registry does not know how to retrieve it. Future versions of this profile may add to the collection of attributes stored in a document stub.

Table 3.14.4.1-2 Value set for relatedDocument.typeCode (CNE)

Code	Definition
APND (append)	The current document is an addendum to the parent document.
RPLC (replace)	The current document is a replacement of the parent document.
XFRM (transform)	The current document is a transformation of the parent document.

Adapted from HL7 CDA Release 2, Committee Ballot 2

See ITI Vol-1: 10.4.11.1 for further detail on the use and meaning of document relationships.

3.14.4.1.2.6 Document Definition Metadata

The XDSDocumentEntry object type is created in ebXML Registry by extending the ebXML Registry ObjectType Classification Scheme[4].

[4] The specific requirement in ebRIM that object types be user extendable was introduced after version 2.0.

The following metadata elements shall be used to describe an XDS Document. They shall be provided by the Document Repository Actor in the Register Document Set Transaction either directly if grouped with a Document Source Actor or forwarded from a Provide and Register Document Set Transaction.

The XDSDocumentEntry.URI shall be supplied by the Document Repository Actor. Its value is dependent on how the repository stores the document.

Each attribute shown below is an attribute on the XDSDocumentEntry object. The attribute name is defined with a prefix of the object type of XDSDocumentEntry when referenced by other objects, for example XDSDocumentEntry.patientId.

The column “source requirement” indicates if the Document Source is required to provide the attribute at the time of submission if it is known. There will be some instances where a required attribute value is not known. Allowable values: R – required, R2 – required if available, O – optional.

The column “query key” indicates if the attribute is required to be supported by the Document Registry as a filtering query key when XDS Documents are queried by a Document Consumer. Allowable values: R – required to be supported as a query key, O – may be ignored in the query, P – prohibited from use as a query key.

In the attribute tables below, when an OID format is specified, it shall follow the assignment and format rules defined for document UID in ITI TF-2 : Appendix B.

Table 3.14.4.1-3 Document Metadata Attribute Definition

XDSDocumentEntry Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
authorDepartment	Represents a specific department within a healthcare facility under which the human and/or machines authored the document.	R2	O	Slot	HL7V2 XON data type (Slot object limits length)
authorInstitution	Represents a specific healthcare facility under which the human and/or machines authored the document. A specific case is that of homecare.	R2	R	Slot	HL7V2 XON data type (Slot object limits length)
authorPerson	Represents the human and/or machines that authored the document within the authorInstitution. The document author may be the patient itself.	R2	R	Slot	HL7V2 XCN data type (Slot object limits length)
availabilityStatus	An XDS Document shall have one of two availability statuses: • Approved – available for patient care • Deprecated – obsolete This attribute is always set to Approved as part of the submission of new XDS Documents. It may be changed to Deprecated under the primary responsibility of the Document Source with possible patient supervision. Although XDS supports the ability to delete documents, there is no such state as "the Document Entry is removed" (only an audit trail is kept if such a deletion is allowed). This list may be extended in the future.	Computed by Registry	R	ebRIM Status attribute	
classCode	The code specifying the particular kind of document (e.g. Prescription, Discharge Summary, Report). It is	R	R	External Classification	XDS Affinity Domain specific

XDSDocumentEntry Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
	suggested that the XDS Affinity Domain draws these values from a coding scheme providing a coarse level of granularity (about 10 to 100 entries).				
classCodeDisplayName	The name to be displayed for communicating to a human the meaning of the classCode	R	P	Name attribute on Classification object, coding scheme name saved as codingScheme slot on Classification	XDS Affinity Domain specific
confidentialityCode	The code specifying the level of confidentiality of the XDS Document. These codes are specific to an Affinity Domain. Enforcement and issues related to highly sensitive documents are beyond the scope of XDS (see security section). These issues are expected to be addressed in later years. confidentialityCode is part of a codification scheme and value set enforced by the Document Registry.	R	P	External Classification	Further restricted by the XDS Affinity Domain
creationTime	Represents the time the author created the document in the Document Source.	R	R	Slot	Time encoding format is HL7V2 Date/Time (DTM) data type. Example: 20041225T23:50:50.
entryUUID	The globally unique identifier (may be assigned by either by Source, Repository, or Registry), which is primarily intended for use as a management identifier. It shall not be used as a way to externally reference the XDS Document. The uniqueId shall be used for that purpose.	Assigned by Document Source or Registry	P	ebRIM specified internal UUID	

XDSDocumentEntry Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
eventCodeList	This list of codes represents the main clinical acts, such as a colonoscopy or an appendectomy, being documented. In some cases, the event is inherent in the typeCode, such as a "History and Physical Report" in which the procedure being documented is necessarily a "History and Physical" act. An event can further specialize the act inherent in the typeCode, such as where it is simply "Procedure Report" and the procedure was a "colonoscopy". If one or more eventCodes are included, they shall not conflict with the values inherent in the classCode, practiceSettingCode or typeCode, as such a conflict would create an ambiguous situation. This short list of codes is provided to be used as “key words” for certain types of queries.	O	R (Useful for Document Consumer application sub-selection)	External Classification	XDS Affinity Domain specific
eventCodeDisplayNameList	The list of names to be displayed for communicating to human reader the meaning of the eventCode.	R (if event Code is valued	P	Name attribute on Classification object, coding scheme name saved as codingScheme slot on Classification	XDS Affinity Domain specific
formatCode	Code globally uniquely specifying the format of the document. Along with the typeCode, it should provide sufficient information to allow any potential XDS Document Consumer to know if it will be able to process the document. The formatCode shall be sufficiently	R	O	External Classification	XDS Affinity Domain specific

XDSDocumentEntry Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
	specific to ensure processing/display by identifying a document encoding, structure and template (e.g. for a CDA Document, the fact that it complies with a CDA schema, possibly a template and the choice of a content-specific style sheet).				
hash	Hash key of the XDS Document itself. This value is computed by the Document Repository and used by the Document Registry for detecting the improper resubmission of XDS Documents.	Computed by Repository	P	Slot	Uses SHA1 hash algorithm.
healthcareFacilityTypeCode	This code represents the type of organizational setting of the clinical encounter during which the documented act occurred. In some cases, the setting of the encounter is inherent in the typeCode, such as "Diabetes Clinic Progress Note". healthcareFacilityTypeCode shall be equivalent to or further specialize the value inherent in the typeCode; for example, where the typeCode is simply "Clinic Progress Note" and the value of healthcareFacilityTypeCode is "private clinic". The value shall not conflict with the value inherent in the typeCode, as such a conflict would create an ambiguous situation.	R	R	External Classification	XDS Affinity Domain specific
healthcareFacilityTypeCodeDisplayName	The name to be displayed for communicating to a human the meaning of the healthcareFacilityTypeCode	R	P	Slot	XDS Affinity Domain specific
languageCode	Specifies the human language of character data in the document. The values of the attribute are language identifiers as described by the IETF (Internet Engineering Task Force) RFC 3066.	R	P	Slot	Further restricted by the registry according to XDS Affinity Domain specific policy

XDSDocumentEntry Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
legalAuthenticator	Represents a participant who has legally authenticated or attested the document within the authorInstitution. Legal authentication implies that a document has been signed manually or electronically by the legalAuthenticator. This attribute may be absent if not applicable.	O	O	Slot	Multi-valued slot. Substructure comes from HL7 V2 serialization. (XCN)
mimeType	MIME type of the document in the Repository.	R	P	MimeType Attribute on extrinsic object	
parentDocument Relationship	The type of relationship that the document has with the parentDocument (e.g. Replace, addendum, or transformation).	R (when applicable)	P	Association Type	Extends Association types with 3 new values (APND, RPLC, XFRM).
parentDocumentId	The identifier of the parentDocument entry that represents the source of a document replacement, addendum, or transformation. May identify a document which is unknown by the Document Registry.	R (when parent Document Relationship is present)	P	Association	If parent document is in registry then code as association. If not then create stub document object to anchor other end of association. Create new object type XDSDocumentStub to support this. Meaning: document stub represents document metadata that is not in registry but need an object to point at to support parentDocumentRelationship. This association is coded with a type from parentDocumentRelationship.
patientId	The patientId represents the subject of care medical record identifier as selected by the Document Source. This identifier shall be from the Assigning Authority Domain supporting the Affinity Domain in which the Document Registry operates. It shall contain two parts:	R	R	External Id	This is the XDS Affinity Domain patient identifier. patientID values shall be formatted as HL7V2 CX data type (including assigning authority) according to the requirements specified for the Patient

XDSDocumentEntry Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
	• Authority Domain Id (enforced by the Registry) • An Id in the above domain. The value of the patientId shall be the same for all new documents of a Submission Set.				Identity Feed transaction (see Section 3.8.4.1.2.3). The assigning authority component must include the OID of the assigning authority in the second subcomponent, and the third subcomponent must be ISO.
practiceSettingCode	The code specifying the clinical specialty where the act that resulted in the document was performed (e.g. Familly Practice, Laboratory, Radiology). It is suggested that the XDS Affinity Domain draws these values from a coding scheme providing a coarse level of granularity (about 10 to 100 entries	R	R	External Classification	XDS Affinity Domain specific
practiceSettingCode DisplayName	The name to be displayed for communicating to a human the meaning of the practiceSettingCode	R	P	Name attribute on Classification object	XDS Affinity Domain specific
serviceStartTime	Represents the start time the service being documented took place (clinically significant, but not necessarily when the document was produced or approved). This may be the same as the encounter time in case the service was delivered during an encounter. This time is expressed as (date/time/UTC). Note: Other times, such as document creation or approval are to be recorded, if needed, within the document.	R2	R	Slot	For point in time this holds serviceTime. Time encoding format must sort correctly as a string. Time encoding format is HL7V2 Date/Time (DTM) data type.
serviceStopTime	Represents the stop time the service being documented took place (clinically significant, but not necessarily when the document was produced or approved). This	R2	R	Slot	For point in time this holds serviceTime, the value shall be same as serviceStartTime. XDS registry adaptor

XDSDocumentEntry Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
	may be the same as the encounter time in case the service was delivered during an encounter. This time is expressed as (date/time/UTC). If the Service happens at a point in time, this attribute shall contain the same value as the serviceStartTime.				function verifies serviceStartTime less than or equal to serviceStopTime. Time encoding format is HL7V2 Date/Time (DTM) data type.
size	Size in bytes of the byte stream that was provided in the Register and Provide Transaction and stored by the XDS Document Repository. This value is computed by the Document Repository and included in the Register Documents Set Transaction.	R (in the Register Trans-action)	P	Slot	Size in Bytes of the document as retrieved from the URI.
sourcePatientId	The sourcePatientId represents the subject of care medical record Identifier (e.g. Patient Id) in the local patient Identifier Domain of the Document Source. It shall contain two parts: • Authority Domain Id • An Id in the above domain (e.g. Patient Id). This sourcePatientId is not intended to be updated once the Document is registered (just as the Document content and metadata itself will not be updated without replacing the previous document). As this sourcePatientId may have been merged by the source actor, it may no longer be in use within the Document Source (EHR-CR). It is only intended as an audit/checking mechanism and has occasional use for Document Consumer Actors.	R	P	Slot	sourcePatientID values shall be formatted as HL7V2 CX data type (including assigning authority) according to the requirements specified for the Patient Identity Feed transaction (see Section 3.8.4.1.2.3). The assigning authority component must include the OID of the assigning authority in the second subcomponent, and the third subcomponent must be ISO.
sourcePatientInfo	This attribute contains demographics information of the patient to whose medical record this document belongs, as the Document Source knew it at the time of Submission.	R	P	Slot	UTF-8, Multi-valued, each value up to 256 bytes. Each value will contain an HL7V2 Message Field Identifier (e.g., PID-1)

XDSDocumentEntry Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
	This information typically includes: Patient Name (Last & First), Patient Sex, and Patient Birth Date. The Clinical Affinity Domain policies may require specific information and format. This patient information is not intended to be updated once the Document is registered (just as the Document content and metadata itself will not be updated without replacing the previous document). As sourcePatientInfo may have been updated by the source actor, it may no longer be in use within the Document Source (EHR-CR). It is only intended as an audit/checking mechanism and has occasional use for Document Consumer actors.				Message Field Identifier (e.g., PID-1), followed by a vertical bar (|), followed by the HL7V2 defined data for that field.
title	Represents the title of the document. Clinical documents often do not have a title, and are collectively referred to by the display name of the classCode (e.g. a "consultation" or "progress note"). Where these display names are rendered to the clinician, or where the document has a unique title, the title component shall be used. Max length, 128 bytes, UTF-8.	O	P	EbRIM Name attribute	
typeCode	The code specifying the precise kind of document (e.g. Pulmonary History and Physical, Discharge Summary, Ultrasound Report). It is suggested that the XDS Affinity Domain draw these values from a coding scheme providing a fine level of granularity.	R	R	External Classification	XDS Affinity Domain specific
typeCodeDisplay Name	The name to be displayed for communicating to a human the meaning of the typeCode	R	P	Name attribute on Classification object	XDS Affinity Domain specific

XDSDocumentEntry Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
uniqueId	The globally unique identifier assigned by the document creator to this document. This unique identifier may be used in the body of other XDS Documents to reference this document. The length of Unique Identifier shall not exceed 128 bytes. The structure and format of this Id shall be consistent with the specification corresponding to the format attribute. (e.g. for a DICOM standard document a 64 character numeric UID, for an HL7 CDA format a serialization of the CDA Document uniqueId plus extension in the form OID plus ID, where OID is a 64 digits max, and the ID is a 16 UTF-8 char max). This uniqueId is intended to respond to the following types of usage: 1. The means to reference this XDS document from within the content of another document. Neither the XDS Registry nor the Repository is aware of such references, but the Document Sources and Consumers are. 2. The means to ensure that when a XDS Document is retrieved from the XDS Document Repository using the URI component, the selected XDS Document is the correct one.	R	R	External Id	Assigned by Document Source. Size limited to 128 bytes (printable characters only). See section 3.14.4.1.2.6.2 for the specification of the format and encoding depending on the document standard used for the XDS Document content.
URI	The URI of the XDS Document to be used for retrieval.	Computed by Repository	P	External Link	External Link name is URI to be used for retrieving the document from the Repository (ITI-17 Retrieve Document transaction) XDS does not constraint the format of this URI beyond RFC 2616. However,

XDSDocumentEntry Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
					the IHE Retrieve Information for Display Integration Profile defined format may be used in cases where the Document repository is grouped with a RID Information Source Actor (See ITI TF-1:Appendix E.5)

3.14.4.1.2.6.1 XDSDocumentEntry.formatCode

In general, the repository holds an octet stream representing the document. The registry metadata describes, among other things, the format of the document. This is coded in XDSDocumentEntry.formatCode. This code will identify document format parameters necessary for interoperability. Rules about handling the formatCode are necessary but are not imposed by XDS. In the future. IHE content specific Integration Profiles may be created that specify these rules.

Note: Although only a small number of document standards may be used, a large number of code values may be defined to point to specific templates and archetypes structuring specific document content.

3.14.4.1.2.6.2 XDSDocumentEntry.uniqueId

The specification of the format and encoding for this attribute depends on the document standard defining the content of the XDS Document (*e.g.* OID with optional extension ID for HL7 CDA, UUID in some cases, SOP Instance UID for DICOM composite objects. Format is: OID^Extension). This attribute shall not exceed 128 bytes in size. It shall be used as an opaque and globally unique identifier for the XDS Document. Document Consumers, Registries, Repositories shall not attempt to interpret its content.

3.14.4.1.2.7 Submission Set Metadata

The following metadata elements shall be used to describe an XDS Submission Set. They shall be provided by the Document Source Actor in the Provide and Register Document Set transaction. They shall be provided by the Document Repository Actor in the Register Document Set Transaction either directly if grouped with a Document Source Actor or forwarded from a Provide and Register Document Set Transaction.

Each of the attributes listed below is an attribute on the RegistryPackage object defining the Submission Set. The attribute name is defined with a prefix of the object type of XDSSubmissionSet when referenced by other objects, for example XDSSubmissionSet.sourceId.

The column “source requirement” states if the attribute is required to be provided by the Document Source at the time of submission. There will be some instances where a required attribute value is not known. Allowable values: R – required, R2 – required if available, O – optional.

The column “query key” states if the attribute is required to be supported by the Document Registry as a filtering query key when XDS Documents are queried by a Document Consumer. Allowable values: R – required to be supported as a query key, O – may be ignored in the query, P – prohibited from use as a query key.

In the attribute tables below, when an OID format is specified, it shall follow the assignment and format rules defined for document UID in ITI TF-2 : Appendix B.

Table 3.14.4.1-4 Submission Set Metadata Attribute Definitions

XDSSubmission Set Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
authorDepartment	Represents a specific department within a healthcare facility under which the human and/or machines authored the document.	R2	O	Slot	HL7V2 XON data type (Slot object limits length)
authorInstitution	Represents a specific healthcare facility under which the human and/or machines authored the Submission Set.	R2	R	Slot	HL7V2 XON data type (Slot object limits length)
authorPerson	Represents the human and/or machines that authored the Submission Set. The document author may be the patient itself.	O	R	Slot	HL7V2 XCN data type (Slot object limits length)
comments	Comments associated with the Submission Set. Free form text with an Affinity Domain specified usage.	R2	R	ebRIM Description attribute	Use specific to XDS Affinity Domain.
contentTypeCode	The code specifying the type of clinical activity that resulted in placing these XDS Documents in this XDS-Submission Set. These values are to be drawn for a vocabulary defined by the Affinity Domain.	R	R	External Classification	XDS Affinity Domain specific
contentTypeCode DisplayName	The name to be displayed for communicating to a human the meaning of the contentTypeCode.	R	P	Slot	XDS Affinity Domain specific
sourceId	Globally unique identifier for the instance of the Document Source that contributed the Submission Set. The assigning authority for these identifiers is specified by the XDS Affinity Domain.	R	R	External Id	SourceId values shall be formatted as HL7V2 CX data type. The first three subcomponents shall be populated, the first subcomponent shall reference the same entity as is referenced by the second and third components. The assigning authority component must include the OID of the assigning authority in the second subcomponent,

XDSSubmission Set Attribute	Definition	Source Reqmt	Query Key	EbRIM Attribute Type	Comments
					and the third subcomponent must be ISO.
submissionTime	Point in Time at the Document Source when the Submission Set was created and issued for registration to the Document Registry. This shall be provided by the Document Source (in case of e-mail with significant delay).	R	R	Slot	Time encoding format is HL7V2 Date/Time (DTM) data type.
uniqueId	Globally unique identifier for the submission-set instance assigned by the Document Source in OID format.	R	R	External Id	

1550

3.14.4.1.2.8 Folder Metadata

The following metadata elements shall be used to describe an XDS Folder. They shall be provided by the Document Source Actor in the Provide and Register Document Set transaction. They shall be provided by the Document Repository Actor in the Register Document Set transaction if this transaction is used outside the context of a Provide and Register Document Set transaction.

Each of the attributes listed below is an attribute on the RegistryPackage object defining the Folder. The attribute name is defined with a prefix of the object type of XDSFolder when referenced by other objects, for example XDSFolder.patientId.

The column “source requirement” states if the attribute is required to be provided by the Document Source at the time of submission. Allowable values: R – required, R2 – required if available, O – optional.

The column “query key” states if the attribute is required to be supported by the Document Registry as a filtering query key when XDS Documents are queried by a Document Consumer. Allowable values: R – required to be supported as a query key, O – may be used as a query key, but can be ignored, P – prohibited from use as a query key.

In the attribute tables below, when an OID format is specified, it shall follow the assignment and format rules defined for document UID in ITI TF-2 : Appendix B.

Table 3.14.4.1-5 Folder Metadata Attribute Definitions

XDSFolder Attribute	Definition	Source Reqmt	Query Key	Attribute Type	Comments
codeList	The list of codes specifying the type of clinical activity that resulted in placing these XDS Documents in this XDSFolder. These values are to be drawn for a vocabulary or coding scheme defined by the Clinical Affinity Domain. When a new submission request associates XDS Documents (new submission or previously submitted) to an XDS Folder, the Code included in the codeList is appended to the existing list of codes for this Folder (if any) unless this code is already present in the list managed by the Registry for the same XDS-Folder. Only one code may be assigned to the Folder when a XDS Document is placed in a Folder	R	R	External Classification (s)	Multi-Valued. XDS Affinity Domain specific
codeDisplayName List	The list of human readable descriptions of the meaning of each on of the codes present in the codeList. Only one code may be assigned to the Folder when a XDS Document is placed in such a Folder	R	P	Slot	Multi-valued. Each value is coded as first three components of CWE datatype. CWE datatype includes code from above codeList.
comments	Comments associated with the Folder. Free form text with an Affinity Domain specified usage.	R2	R	ebRIM Description attribute	Use specific to XDS Affinity Domain. Length of 256 characters.
lastUpdateTime	Point in time at the Document Registry when an XDS Document was registered and placed in the XDS Folder.	Valued by Registry Adaptor	R	Slot	Value maintained by XDS registry adaptor function. Time encoding format is HL7V2 Date/Time (DTM) data type.
patientId	The patientId represents the subject of care medical record Identifier as defined by the Document Source. This identifier shall be from the Assigning Authority Domain supporting the Affinity Domain in which the	R	R	External Id	patientID values shall be formatted as HL7V2 CX data type (including assigning authority) according to the requirements specified for the Patient

XDSFolder Attribute	Definition	Source Reqmt	Query Key	Attribute Type	Comments
	Domain supporting the Affinity Domain in which the Document Registry operates. It shall contain two parts: • Authority Domain Id (enforced by the Registry) • An Id in the above domain. The value of the patientId shall be the same for all new documents of a Folder.				Identity Feed transaction (see Section 3.8.4.1.2.3). The assigning authority component must include the OID of the assigning authority in the second subcomponent, and the third subcomponent must be ISO.
uniqueId	Globally unique identifier for the XDS-Folder in which one or more XDS Documents are placed. It is assigned by the Document Source at the time the XDS Folder is created in OID format.	R	R	External Id	

3.14.4.1.2.9 Registry Enforcement of Attributes

1570

Table 3.14.4.1-6 Document Metadata Attribute Enforcement

XDSDocumentEntry Attribute	Registry Enforcement
patientId	Authority Domain Id Patient Id (known from patient identity feed)
creationTime	No enforcement
serviceStartTime	No enforcement
serviceStopTime	Verifies serviceStartTime <= serviceStopTime
classCode	Coding Scheme and Code Value.
classCodeDisplayName	No enforcement
practiceSettingCode	Coding Scheme and Code Value
practiceSettingCode DisplayName	No enforcement
healthcareFacilityTypeCode	Coding Scheme and Code Value
healthcareFacilityTypeCodeDisplayName	No enforcement
availabilityStatus	No enforcement
uniqueId	No enforcement
typeCode	Coding Scheme
typeCodeDisplayName	No enforcement
formatCode	Coding Scheme and Code Value
eventCodeList	Coding Scheme and Code Value
eventCodeDisplayNameList	No enforcement
title	No enforcement
authorInstitution	No enforcement
authorDepartment	No enforcement
authorPerson	No enforcement
legalAuthenticator	No enforcement
URI	No enforcement
parentDocumentRelationship	One of three values
parentDocumentId	Existing UUID
confidentialityCode	Coding Scheme and Code Value
languageCode	Coding Scheme and Code Value
sourcePatientId	No enforcement
sourcePatientInfo	No enforcement
size	No enforcement
hash	No enforcement
mimeType	Code Value
entryUUID	No enforcement

Table 3.14.4.1-7 SubmissionSet Metadata Attribute Enforcement

XDSSubmissionSet Attribute	Registry Enforcement
uniqueId	No identical existing Submission Set uniqueId. OID format (See Appoendix B).
sourceId	Coding Scheme and Code value

submissionTime	No enforcement
authorInstitution	No enforcement
authorDepartment	No enforcement
authorPerson	No enforcement
contentTypeCode	Coding Scheme and Code value
contentTypeCodeDisplayName	No enforcement
comments	No enforcement

Table 3.14.4.1-8 Folder Metadata Attribute Enforcement

XDSFolder Attribute	Registry Enforcement
patientId	The value of the patientId shall be the same for all new documents of a Folder.
uniqueId	No identical existing Submission Set uniqueId. OID format (See Appoendix B).
lastUpdateTime	No enforcement
codeList	Coding Scheme and Code value
codeDisplayNameList	No enforcement
comments	No enforcement

3.14.4.1.2.10 Protocol Requirements

SOAP with Attachments shall be used as the protocol between the Document Repository and the Document Registry when these two actors are implemented separately. The protocol is specified in ITI TF-2 : 3.15.4.1.2.3.1 (On-line protocol binding).

3.14.4.1.2.11 XDS Registry Adaptor

The XDS Registry Adaptor is a set of functionality that is not provided for in the ebXML registry standard, but is instead specified by XDS to support integration into the healthcare environment. This adaptor includes the following functionality:

- Validate patient ID – patient IDs (XDSDocumentEntry.patientId attribute) shall be a known patient ID and registered against the Patient ID Domain of the XDS Affinity Domain managed by the patient Identity Source Actor.
- Validate submitted metadata – the adaptor shall verify that submitted metadata meets XDS Registry metadata specification
- Verify coded values – the adaptor shall verify that coded fields (ebXML external classifications) contain valid XDS specified values or where the Affinity Domain constraints code values, to verify them (See Section 3.14.4.1.2.9).
- The adaptor shall make submission to registry an atomic operation – see section 3.14.4.1.2.2.3 Atomicity Requirements for Submission Requests for atomicity requirements.

- If the registry submission is successful then the adaptor shall issue an ebXML ApproveObjectsRequest to label the appropriate collection of objects Approved. See atomicity requirements section for details.
- If the registry submission fails then the adaptor shall issue an ebXML RemoveObjectsRequest to remove all objects included in this submission.

- When documents are replaced, the registry adaptor shall detect it and shall generate the appropriate metadata changes for the registry. Specifically, when a document replacement is specified using the document attributes parentDocumentRelationship and parentDocumentId, the adaptor:
  - shall mark the replaced document metadata as deprecated by setting the XDSDocumentEntry attribute Status to Deprecated
  - shall link the new version to the old version with Association type "RPLC".
- The adaptor shall verify that all documents in a folder are for the same patient. Specifically, verify that the patientId attribute of the folder matches the patientId attribute of each document in the folder.
- The adaptor shall validate that the mimeType document attribute for all documents received is on the approved list for this Affinity Domain.
- The XDS Folder attribute lastUpdateTime shall be updated by the adaptor every time a new document is added to an XDS Folder.
- The patientId attribute of an XDSDocumentEntry object shall match the patientId attribute on any folder that holds it.
- The adaptor shall enforce the number of classifications offered against a document. Code lists are allowed to be multiples. Codes are required to be singular.

Implementation Tip: There are two common implementation scenarios for rolling back a Submission Request atomic transaction. 1) The registry adaptor (internal to the XDS Document Registry) generating the submission has maintained a list of object UUIDs that make up the Submission Request and can use the list to delete the metadata from the registry and 2) registry adaptor (internal to the Document Registry) generating the submission has not maintained state and must discover through query the UUIDs of the Submission Set and its contents.

3.14.4.1.2.12 Security Requirements

This profile requires all actors be grouped with a Secure Node Actor as defined in the IHE Audit Trail and Node Authentication Integration profile. This use of the ATNA profile in an XDS Affinity Domain does not require a centralized affinity domain Audit Repository Actor.

The use of ATNA along with XDS does require that each member of the Affinity Domain does have audit and security mechanisms in place. See appendix ITI TF-1: Appendix G and ITI-TF-2: Appendix K.

The individual actors involved are often members of different secure domains, as illustrated in Figure XXX. The data transfers between different secure domains need different protection than transfers within a secure domain. They shall be either:

- Encrypted, with TLS authentication of both hosts, for online transfers, or
- Encrypted, using S/MIME secure encoding and digital signature by the sender, for offline transfers.

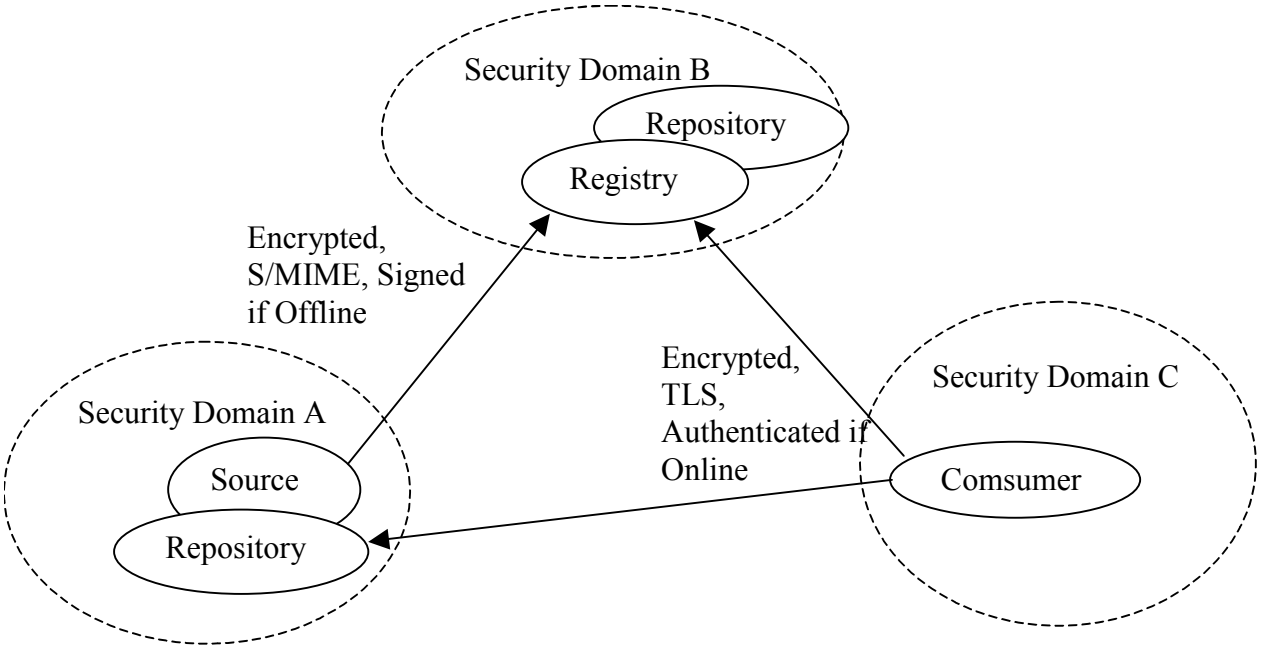
Transfers within a single secure domain may choose to omit encryption if it is unnecessary, so it is recommended that the online transfer security mechanisms be configurable. Certificate management and exchange is defined as part of the affinity domain business relationships and no IHE Integration Profile is specified at this time, see ITI TF-1: Appendix L.

Each transaction will result in audit records describing the transaction. Each secure domain has its own audit server to capture the records for the actors that are within that domain. Access to audit records by other enterprises within the affinity domain is managed and controlled by the business relationship terms of the affinity domain. There is no automatic IHE transaction for such access.

The audit records that should be generated (referfences IHE ATNA Integration Profile) by normal XDS activities are:

- For the Register Document Set, and the Provide and Register Document Set Transactions:
  - The Source Actor shall generate "Export" events describing the export of PHI from the Source to the Registry Actor. There should be one report for each transaction.
  - The Registry Actor shall generate "Import" events describing the import of PHI from the Source to the Registry Actor. There should be one report for each transaction.
- For the Query Documents Transaction:
  - The Registry Actor shall generate a "Query" event describing the query, and shall generate an "Export" event if the query results in a reply that contains PHI.
- For the Retrieve Document Transaction:
  - The Repository Actor shall generate an "Export" event. This may be an event for each Retrieve Document Transaction, or multiple transactions for the same patient may be heuristically combined. The heuristics for this combination are not specified by IHE. It is intended to reduce the volume of audit records. Combination is permitted when the active participants and patient are the same, and the time difference is considered insignificant.
  - The Document Consumer Actor shall generate an "Import" event. This may be one event per transaction, or multiple transactions may be reported as a single event using a heuristic for combining transactions. Combination is permitted when the active participants and patient are the same, and the time difference is considered insignificant.

Figure 3.14.4.1-2 - Example Security Domain Relationships



All Actors are part of the same Clinical Affinity Domain

3.14.4.1.3 Expected Actions

Upon receipt of a Register Document Metadata message, the Document Registry with the aid of the Registry Adaptor shall do the following:

- Accept all valid SubmitObjectsRequests.
- Perform validations
- Update the registry with the contained metadata
- Return a RegistryResponse message given the status of the operation.

If the registry rejects the metadata, then, the following occurs:

- An error is returned
- The error status includes an error message
- The request is rolled back

3.14.4.2 Register Document Metadata Acknowledgment

3.14.4.2.1 Trigger Events

The Document Registry finishes processing a Register Document Metadata request and shall respond with:

- Register Document Metadata Acknowledgment

This message corresponds to the ebXML RequestResponse message.

3.14.4.2.2 Message Semantics

The ebXML RequestResponse message carries the status of the requested operation and an error message if the requested operation failed. The conditions of failure and possible error messages are given in the ebRS standard.

3.14.4.2.3 Expected Actions

The Document Repository now knows that the transaction succeeded/failed and can continue. The metadata added to the registry as a result of this transaction is now available for discovery via query transactions.

Add the following section to Volume 2 of the IHE ITI Technical framework

3.15 Provide and Register Document Set

This section corresponds to Transaction ITI-15 of the IHE Technical Framework. Provide and Register Document Set is used by the Document Source to provide a set of documents to the Document Repository, and to request that the repository store these documents and then register them with the Document Registry.

The Provide and Register Document Set transaction describes only the interaction between the Document Source and Document Repository actors. The interaction between the Document Repository and the Document Registry is described separately in the Register Document Set Transaction (ITI-14).

This transaction aligns with the Registry Services standard (ebRS). The ebRS standard covers the interaction with a service that includes a registry with integrated repository. From the point of view of the Document Source, the separate nature of the XDS Document Registry and Repository actors is hidden. This transaction exactly matches the registry service for submitting registry/repository content found in ebRS.

By specifying separate registry and repository actors, XDS offers additional flexibility of having a single registry index content for multiple repositories. The ebRIM portion of the registry standard supports this possibility though the ExternalLink object type.

The documents and metadata go to the repository actor and then the metadata is forwarded on to the registry actor. They move in this direction for several reasons:

- Allows best reuse of ebXML Registry specified protocols
- Document Source only needs to know the identity of the Document Repository. Repository knows the identity of the registry. If Provide and Register Document Set transaction were sent to the registry then routing decisions for documents would be more complex.
- Resulting protocols are simpler
- Simplifies the common case where the Document Source and the Document Repository are grouped.

3.15.1 Scope

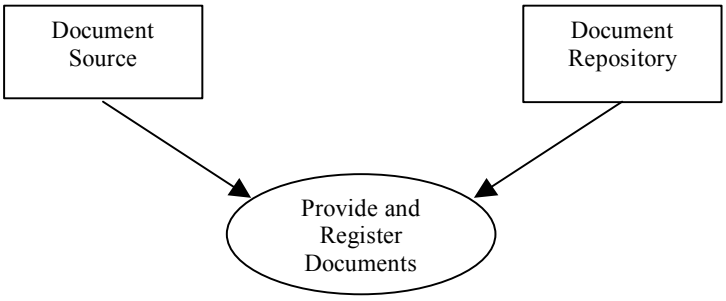
The Provide Register Document Set transaction passes a Repository Submission Request (see ITI TF-2: 3.14.4.1.2.2.2) from a Document Source to a Document Registry.

A Provider and Register Document Set transaction carries:

- Metadata describing zero or more new documents
- Submission Set definition along with the linkage to new documents and references to existing documents
- Zero or more XDS Folder definitions along with linkage to new or existing documents

- Zero or more documents

3.15.2 Use Case Roles



Actor: Document Source

Role: A system that submits documents and associated metadata to a Document Repository.

Actor: Document Repository

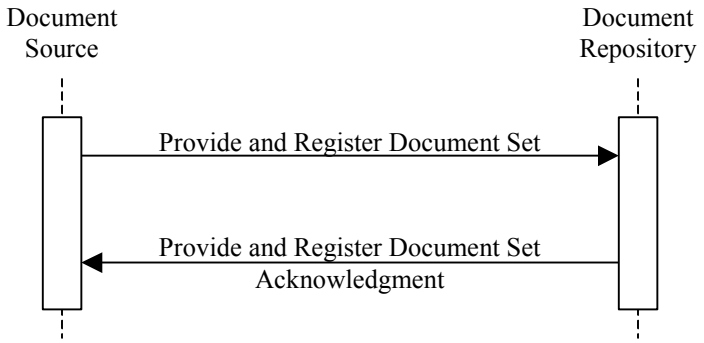
Role: A document storage system that receives documents and associated metadata and:

- Stores the documents
- Enhances submitted metadata with repository information to enable later retrieval of documents
- Forwards the enhanced metadata to the Document Registry.

3.15.3 Referenced Standards

ebMS	OASIS/ebXML Messaging Services Specifications v2.0
ebRIM	OASIS/ebXML Registry Information Model v2.0
ebRS	OASIS/ebXML Registry Services Specifications v2.0
HTTP	HyperText Transfer Protocol HTTP/1.1 (IETF RFC2616)
MIME	Multipurpose Internet Message Extensions (RFC 2045 to RFC 2049)
SMTP	Simple Mail Transfer Protocol (RFC2821)
multipart/related	The MIME Multipart/Related Content-type (RFC2387)

3.15.4 Interaction Diagram



3.15.4.1 Provide and Register Document Set Message

A Document Source sends documents and associated metadata to a Document Repository that has an associated Document Registry. This message corresponds to an ebRS SubmitObjectsRequest with associated documents.

3.15.4.1.1 Trigger Events

The Document Source, based on a human decision or the application of a certain rule of automatic operation, wants to submit

- A set of one or more documents to the Document Repository and
- The associated metadata to the Document Registry.

3.15.4.1.2 Message Semantics

Message semantics are discussed as follows:

1. Metadata
2. Security Requirements
3. Protocol Selection (On-Line Protocol binding and Off-Line Protocol binding)

3.15.4.1.2.1 Metadata

The Register Document Set message shall include the metadata attributes (as defined in section 3.14.4.1.2.6) that will be forwarded by the Document Repository to the Document Registry using the Register Document Set Transaction (ITI-14).

The Document Source supplies all necessary registry object attributes with the exception of the URI attribute of an XDSDocumentEntry that must be assigned by the Document Repository. Therefore, the Document Repository must add this attribute to the metadata before initiating the Register Document Set transaction to the registry.

3.15.4.1.2.2 Security Requirements

Relevant security requirements are discussed in the Register Document transaction (see ITI TF-1: 3.14.4.1.2.12).

3.15.4.1.2.3 Protocol Selection

There are two types of network relationships between the Document Source and Document Repository:

- On-line – the Document Source constructs a direct connection (i.e, socket) to the Document Repository.
- Off-line – the Document Source connects to the Document Repository via SMTP.

3.15.4.1.2.3.1 On-Line Protocol Binding

3.15.4.1.2.3.1.1 General structure and header

This is a MIME multipart/related message. The first attachment inside the payload of the SOAP request bears the registry metadata in an XML file containing the SubmitObjectsRequest.

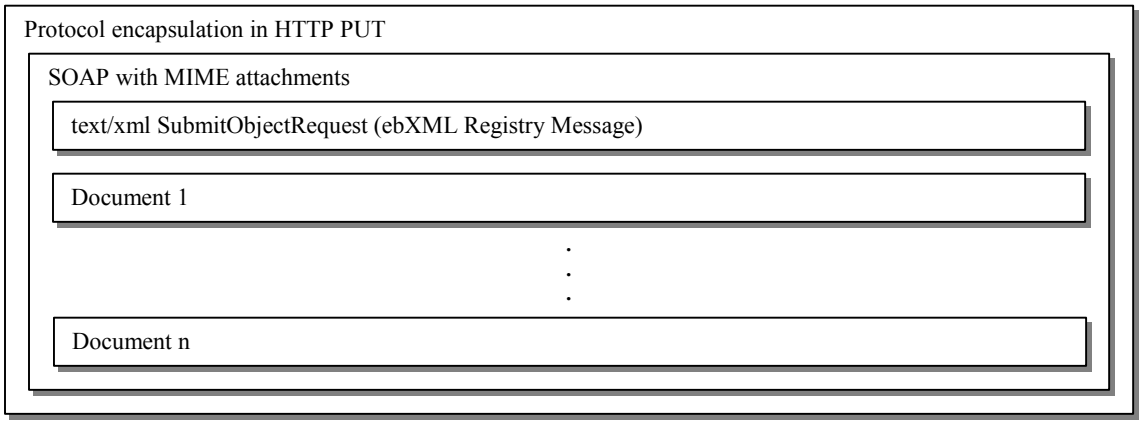


Figure 3.15.4.1-1 General Diagram of the Main message composing the On-Line Provide and Register Document Set Transaction

3.15.4.1.2.3.1.1 Associated Documents

The next attachments will contain the document(s) to be provided and registered, as MIME parts. There are one or more parts that contain byte streams representing documents[5].

[5] This section is written independent of which protocol binding is used to package this multipart message. The protocol choice is documented elsewhere in this profile.

The multipart packaging transmits the MIME-type of each part. The metadata part shall be of type text/xml. Parts containing documents destined for the Document Repository can have any MIME type, either single part or multipart. Each part containing a document has associated with it a document ID that is unique within the scope of this message. The Registry Metadata contained within one part of this message uses these document IDs to bind pieces of metadata to documents.

The registry metadata will be valid according to ebRIM and will contain the definition of one or more ebXML ExtrinsicObjects. An ExtrinsicObject is a registry object that represents a repository document within the registry. Each ExtrinsicObject will contain an **id** attribute. The value of the **id** attribute will match exactly one Content-Id header in one part of the multipart package. This is how metadata is linked to an associated document within the message. The format of this **id** follows the ebXML Registry definition. It is either a valid UUID or a symbolic name.

3.15.4.1.2.3.2 Off-Line Protocol Binding

3.15.4.1.2.3.2.1 General structure and header

As shown on Figure 3.15.4.1-2, the Off-Line transaction will be based on the ebXML Message Service Binding, as defined in the ebXML Registry Service (ebRS), with an Asynchronous Message and responses as defined in ebXML Messaging Services (ebMS). The re-use of ebXML enables implementers to integrate the Provide and Register Document Set transaction into a server which supports more comprehensive services, including some using Collaboration-Protocol Profiles (CPP) and Collaboration-Protocol Agreement (CPA) as supported by ebXML. Because IHE is aiming to specify such as plug-and-play mechanisms, the Off-Line Protocol Binding is entirely defined into the present document. This specification does not mandate the use of a CPA between the Document Repository acting as "ebRS Registry" and the Document Source acting as "ebRS Registry Client". Such protocol agreement aspects are beyond the scope of the XDS Profile. The Document Source has only to know the Document Repository e-mail

address to be able to provide and register a document set.

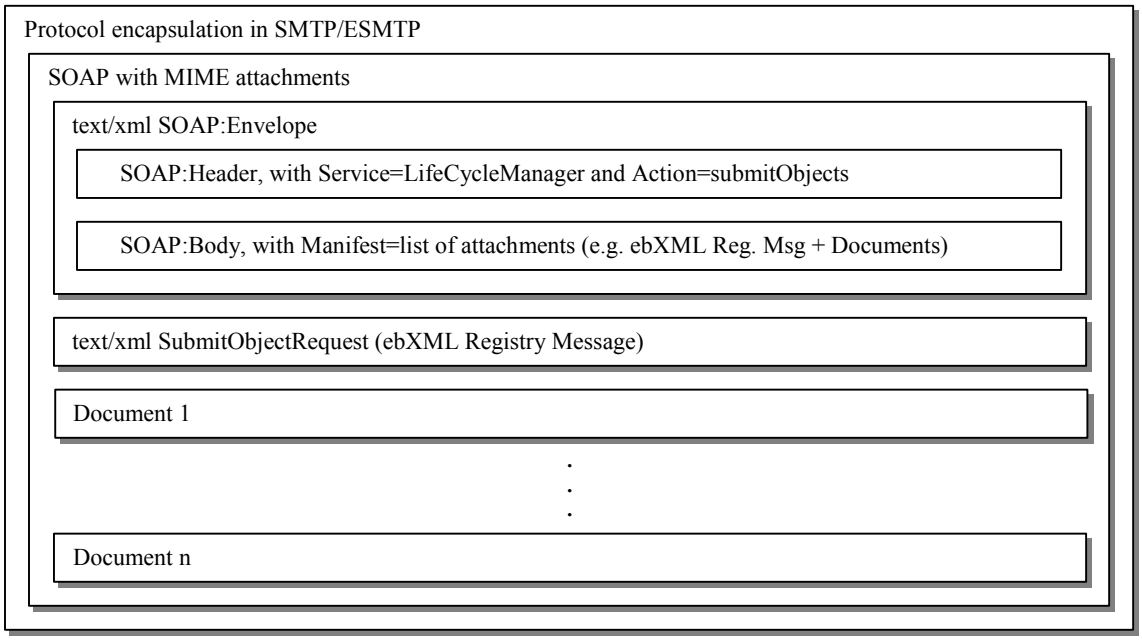


Figure 3.15.4.1-2 General Diagram of the Main message composing the Off-Line Provide and Register Document Set Transaction

The message is an e-mail message (which the ebXML Messaging Services can split into several messages if a single message would be too big) containing the following fields:

- The **From:** e-mail address of the sender (Document Source).
- Optionally, a **Reply-to:** address if the Document Source wants the response messages to be sent to another e-mail address.
- The **To:** e-mail address of the recipient (Document Repository). In case the Document Repository is able to register a document set to more than one Document Registry, it will have a different e-mail address for each one of the Repository-Registry peer.
- **Date:** is the date and time of the Provide and Register Document Set Transaction.
- **Subject: XDS/1.0/PnR/** (followed optionally by indication of XDS "subprofile" name. It SHALL NOT contain any Patient related information)
- **MIME-Version: 1.0**.
- **SOAPAction: "ebXML"**.

This is a MIME multipart/related message. The first attachment is the text/xml SOAP:Envelope part containing the ebMS header. The character set of the ebMS header is UTF-8.

The Header is described in the ebMS standard. It contains the following ebRS tag values:

- The header of the message, in /SOAP:Envelope/ SOAP:Header/eb:MessageHeader/ as shown in the table below.

Table 3.15.4.1-1 ebXML Message Header

Location ("@" for attributes)	Description
eb:From/eb:PartyId	Identification of the message sender (its email address, preceded by **mailto:**)
eb:From/eb:Role	String indicating the authorized role of the sender formatted as a URI per ebXML messaging specification: http://www.ihe.net/roles/iti/xds/DocumentSource
eb:To/eb:PartyId	Identification of intended recipient of the message (its email address, preceded by **mailto:**)
eb:To/eb:Role	String indicating the authorized role of the sender formatted as a URI per ebXML messaging specification: http://www.ihe.net/roles/iti/xds/DocumentRepository
eb:CPAId	Identification of a Collaboration Protocol Agreement between the sender and receiver. This shall contain the trading partner agreed CPA text reference, if it exists (e.g., the URI of the XML file describing the partnership agreement). If there is no CPA, this element shall be the concatenation of eb:From/eb:PartyId and the eb:To/eb:PartyId, separated by the hyphen character (-).
eb:ConversationId	In the absence of a local trading partner agreement, shall be CCYYMMDD-HHMMSS-mmmmm based upon the sending ebXML message generation. When generating responses the eb:ConversationID is taken from the original message.
eb:Service	Shall be LifeCycleManager
eb:Action	Shall be submitObjects
eb:MessageData/eb:MessageId	A unique message identifier generated by the sender: either a concatenation of message elements to create a globally unique identifier, or a single message element if that element is globally unique.
eb:MessageData/eb:Timestamp	UTC Time that the message header was created in XMLSchema dateTime format. Example: 2004-12-25T23:50:50
eb:DuplicateElimination	If present, duplicate messages should be eliminated.
eb:Description	Description of the submissionSet (equivalent to the XDSSubmissionSet Title).

eb:AckRequested	Optional in ebMS, required here to indicate that the repository shall acknowledge the message. This element has the following attributes: SOAP:mustUnderstand="1" eb:version="2.0" eb:signed="false"

- List of references to document, in /SOAP:Envelope/ SOAP:Header/eb:Manifest/eb:Reference as shown in the table below.

Table 3.15.4.1-2 ebXML Message References

Location ("@" for attributes)	Description
@eb:id	Identification of the document, which is the OID of the XDSDocument. However, the first reference shall be to the SubmitObjectsRequest XML file, with id set to SubmitObjectsRequest.
@xlink:href	The relative URI of the document in the payload of the ebMS message, cid: followed by the OID. Used only for a newly submitted XDS Document.
@xlink:role	Shall be present only for the first reference, and be set to **http://www.ihe.net/roles/iti/xds/SubmitObjectsRequest**
eb:Schema	Shall be present only for the first reference, and has following attributes: eb:location=**http://www.ihe.net/schemas/iti/xds/SubmitObjectsRequest** eb:version=1.0
eb:Description	To be set to the XDSDocumentEntry.title. However, for the first reference, shall be set to the meaning of SubmitObjectsRequest in the local language (i.e. lang="en-GB", "Provide and Register Document Set Metadata").

The following attachment inside the payload of the SOAP request bears the registry metadata in an XML file containing the SubmitObjectsRequest.

3.15.4.1.2.3.2.2 Associated Documents

See the subsection "Associated Documents" in the On-Line Binding section (ITI TF-2: 3.15.4.1.2.3.1.1). Any document that has a reference xlink:href and contains a URI that is a content id (URI scheme "cid") shall be included in the payload.

3.15.4.1.3 Expected Actions

The Document Repository will receive this message. Each document within the message will be stored into the repository. A detected failure will result in an error result message being returned to the Document Source thus terminating this transaction.

As each document is stored into the repository an identifier must be created:

- A URI that can be used by a Document Consumer to reference the document.

The Document Repository will modify the received registry metadata. Each ExtrinsicObject, which represents a document deposited in the repository, will have added to it the above URI. A Register Document Set transaction will be issued to the XDS Document Registry.

3.15.4.2 Provide and Register Document Set Acknowledgment

The Document Repository sends a Provide and Register Document Set Acknowledgment when the processing of a Provide and Register Document Set is complete. This message is identical to the RegistryResponse message specified in ebRS. It shall be conveyed in the same protocol as the request.

3.15.4.2.1 Trigger Events

The following events can trigger this message:

- Documents stored to repository successfully and metadata stored to registry successfully (The registry part is carried out as part of a Register Document Set transaction)
- Documents stored to repository successfully but an error occurred in storing the metadata to the registry
- Documents were not successfully stored to the repository

3.15.4.2.2 Message Semantics

An ebRS RegistryResponse message is returned containing status and an error message if necessary.

Additional relevant semantics for both the repository and registry are described in the Register Document Set transaction.

3.15.4.2.3 Expected Actions

The Document Source now knows that the transaction succeeded/failed and continue. The metadata added to the registry as a result of this transaction is now available for discovery via query transactions. The document(s) added to the repository are now available for retrieval.

Add the following section to Volume 2 of the IHE ITI Technical framework

3.16 Query Registry

This section corresponds to Transaction ITI-16 of the IHE Technical Framework. Transaction ITI-16 is used by the Document Consumer to query the Document Registry for information about documents indexed in the registry.

Note: This is a very general query mechanism that allows very broad use. Future extensions to XDS may introduce restrictions or specified the use of canned queries. Proposals for restricting the search mechanism are requested.

3.16.1 Scope

The Query Registry Transaction supports a variety of types of queries. Examples include the following:

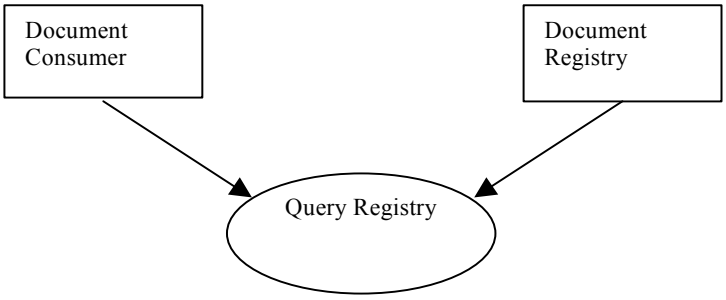
- Query by patient (Id) for a time interval, by document type(s), by practice setting(s), by author person
- Query by Document Source
- Query for XDS Folders updated during a time interval
- Query for all documents in a Folder or Submission Set
- Query by time of submission

The list of XDS DocumentEntry attributes that can be the target of a query are defined in section ITI TF-1: 10.4.10.1, XDS Registry Entry Attributes. This transaction will document the basic syntax and semantics of XDS Document Registry queries.

All queries return:

- Metadata for one or more registry objects, or
- Object references for one or more registry objects (registry UUIDs).

3.16.2 Use Case Roles



Actor: Document Consumer

Role: Generates Query Registry messages and sends them to the Document Registry.

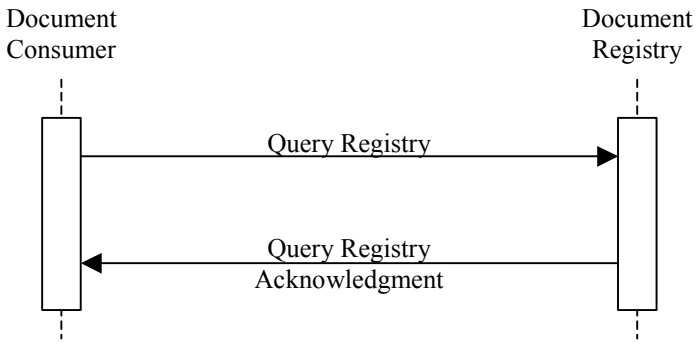
Actor: Document Registry

Role: Receives Query Registry messages and executes a query against registry metadata to select and return matching data to the Document Consumer.

3.16.3 Referenced Standard

ebRS	OASIS/ebXML Registry Services Specifications v2.0
SQL	ISO/IEC 9075 Database Language SQL

3.16.4 Interaction Diagram



3.16.4.1 Query Registry

This is the query request to the registry from a Document Consumer.

3.16.4.1.1 Trigger Events

This message is initiated when the Document consumer wants to retrieve document metadata.

3.16.4.1.2 Message Semantics

XDS specifies the use of SQL as a query language to the registry. There are 2 significant parameters to an AdHocQueryRequest:

- returnType
- SQL query text

3.16.4.1.2.1 Parameter returnType

XDS supports the following values for the parameter returnType:

- ObjectRef – a list of object UUIDs (references)
- LeafClass – list of XML elements representing the leaf class of the object returned

3.16.4.1.2.2 SQL query text

SQL queries submitted to an XDS Document Registry shall conform to the ebRS Registry Services specification, which maps elements of the information model (ebRIM) into a collection of SQL views.

The next sections show the details of several useful queries. This is not an exhaustive list. Any valid SQL query written against the registry information model (ebRIM+XDS specialization) may be used. The specific SQL subset used by registry is specified in Appendix D of ebRS.

3.16.4.1.3 Expected Actions

The registry returns a Query Registry Acknowledgment message.

3.16.4.1.4 Example Queries

3.16.4.1.4.1 Query for Patient Documents in Date Range

The following query searches for Approved documents with patientId equal to 123 and creationTime between 1 January 2004 and 1 May 2004.

SELECT * FROM ExtrinsicObject doc, ExternalIdentifier ei, Slot dt WHERE	The select must include ExtrinsicObject representing the document and attributes representing the patientId and creationTime
doc.objectType=XDSDocumentEntry AND	The objectType must be XDSDocumentEntry. XDSDocumentEntry will be coded as a UUID.
	Match the patient id (patientId)
ei.identificationScheme=XDSPatientId AND	XDSPatientId is the classification scheme that defines the patientId attribute in XDS. This symbolic name will be replaced by a UUID in the actual query.
ei.registryObject=doc.id AND	This external Id belongs to this document.
ei.value='123' AND	The patient Id matches
	Validate the time range
dt.parent=doc.id AND	The document time is stored in a slot. Check the linkage between the slot and the document object.
dt.name='creationTime' AND	The slot name is serviceTime.

dt.value > '20040101' AND dt.value < '20040501' AND	The creationTime must be in requested range. Note the encoding of less-than using the < sequence so it does not conflict with the XML within which this query is embedded. The time is coded as a string and not a number so sort order determines performance of comparison operators.
doc.Status='Approved'	Find only approved documents.

3.16.4.1.4.2 Query for all Discharge Summaries for a Patient

In this query we want all discharge summaries for a patient whose patientId is123. It is presumed we have already established a Classification Scheme representing document types and there is a node within the Classification Scheme with the name 'Discharge Summary' that identifies the document type we want.

SELECT * FROM ExtrinsicObject doc WHERE	
doc.id in (SELECT doc.id FROM ExtrinsicObject doc, ExternalIdentifier ei WHERE doc.objectType=XDSDocumentEntry AND ei.identificationScheme=XDSPatientId AND ei.registryObject=doc.id AND ei.value='123') AND	Select all documents for our patient. This subordinate query is constructed from the elements of the previous query example. Again, XDSPatientId must be replaced by the UUID. The query could be extended to include discovering the UUID for XDSPatientId.
doc.id in (SELECT classifiedObject FROM Classification WHERE classificationScheme=ClassCode AND nodeRepresentation=xxx) AND	Document is classified as a Discharge Summary – ClassCode is replaced by its UUID – xxx is represented by the correct LOINC Code (for example)
doc.Status='Approved'	Only find approved documents.

3.16.4.1.4.3 Query the contents of an XDS Folder

An XDS Folder can hold 3 distinct types of objects:

1. XDS Folders (Registry Package object)
2. XDSDocumentEntry objects
3. Submission Sets (Registry Package object)

For later processing, such as creating a display of the contents, it is necessary to know the type of all the objects returned since each type is displayed differently. A separate query will be used for each type.

XDS Folders and Submission Sets are similar in that they are implemented as Registry Packages and their type is determined by classifications. XDSDocumentEntry objects are a subtype of ExtrinsicObjects, their typed coded via the objectType attribute.

The identity of the target XDS Folder is shown symbolically as currentXDSFolder. In a real application, this would be a raw UUID from a prior query. Also shown symbolically are XDSFolder and XDSSubmissionSet, the classification nodes representing Folders and Submission Sets. These would normally be coded as raw UUIDs from a prior query or have a query coded as an IN predicate to find them.

Query for XDS Folders

SELECT * FROM RegistryPackage folder, Classification c, Association a WHERE	
a.sourceObject=currentXDSFolder AND	Folder is a member of currentXDSFolder
a.associationType='HasMember' AND	
a.targetObject=folder.id AND	
c.classifiedObject=folder.id AND	Folder is classified as XDSFolder
c.classificationNode=XDSFolder AND	XDSFolder is replaced by the raw UUID
Folder.status='Approved'	Folders must be approved.

Query for XDSDocumentEntry objects

SELECT eo.id FROM ExtrinsicObject eo, Association a WHERE	
a.sourceObject=currentXDSFolder AND	ExtrinsicObject is a member of currentXDSFolder

a.associationType='HasMember' AND	
a.targetObject=eo.id AND	
eo.objectType=XDSDocumentEntry AND	Extrinsic Object is type XDSDocumentEntry where XDSDocumentEntry is replaced by the raw UUID
eo.status='Approved'	

Query for Submission Set objects

SELECT eo.id FROM ExtrinsicObject eo, Association a WHERE	
a.sourceObject=currentXDSFolder AND	ExtrinsicObject is a member of currentXDSFolder
a.associationType='HasMember' AND	
a.targetObject=eo.id AND	
eo.objectType=XDSSubmissionSet	ExtrinsicObject is type XDSSubmissionSet where XDSSubmissionSet is replaced by the raw UUID
eo.status='Approved'	

3.16.4.1.4.4 Query for all XDS Folders for a patient

In this query we want to find all XDS Folders for a patient whose patientId is123. In this example we show the raw UUIDs that must be used as part of a real query. These UUIDs are established when the registry is initialized for XDS used as described in Appendix J1.

SELECT * FROM RegistryPackage folder, ExternalIdentifier ei WHERE	
folder.objectType= 'urn:uuid:d9d542f3-6cc4-48b6-8870-ea235fbc94c2' AND	Object of type XDSFolder (This UUID represents XDSFolder objectType)
ei.identificationScheme=	Patient id is 123 (This UUID represents the

'urn:uuid:58a6f841-87b3-4a3e-92fd-a8ffeff98427' AND ei.value='123') AND	Classification Scheme that the External Identifer patientId is based on)
ei.registryObject=folder.id AND	The External Identifier belongs to the folder
folder.status='Approved'	Folders must be approved.

3.16.4.2 Query Registry Acknowledgement

This is the response to the Query Registry message.

3.16.4.2.1 Trigger Events

Completion of query initiated by a Query Registry message.

3.16.4.1.2 Message Semantics

The Query Registry Acknowledgement (AdhocQueryResponse) is returned in one of three forms:

1. List of ObjectRefs
2. Registry metadata describing objects found by query
3. Error message

3.16.4.2.3 Expected Actions

The Document Consumer may process the returned registry data, retrieve documents based on the metadata if the necessary metadata was returned, or handle returned errors

Add the following section to Volume 2 of the IHE ITI Technical framework

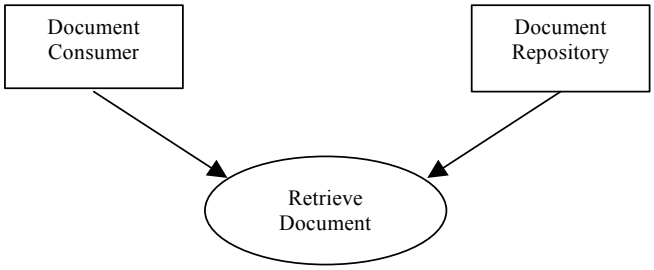
3.17 Retrieve Document

This section corresponds to Transaction ITI-17 of the IHE Technical Framework. The Document Consumer and Document Repository actors use transaction ITI-17.

3.17.1 Scope

This transaction is used by the Document Consumer to retrieve a document from the Document Repository. The Document Consumer has already obtained the URI information from the Document Registry by means of the Query Registry transaction.

3.17.2 Use Case Roles



Actor: Document Consumer

Role: Obtains document.

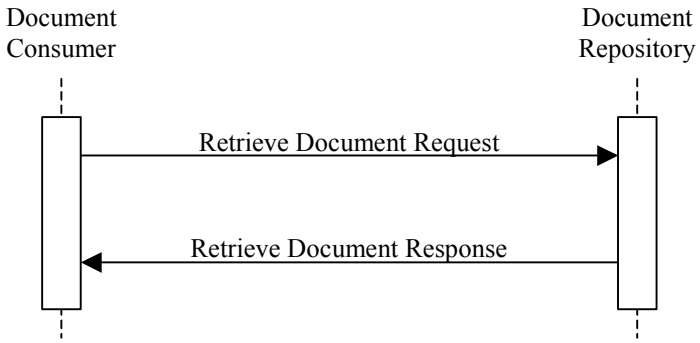
Actor: Document Repository

Role: Provides documents.

3.17.3 Referenced Standard

HTTP	Hyper Text Transfer Protocol HTTP 1.1 (RFC 2616)
MIME	Multipurpose Internet Message Extensions (RFC 2045 to RFC 2049)
SMTP	Simple Mail Transfer Protocol (RFC 2821)
Multipart/Related	The MIME Multipart/Related Content-type (RFC 2387)

3.17.4 Interaction Diagram



3.17.4.1 Retrieve Document Request

3.17.4.1.1 Trigger Events

The Document Consumer obtains document URIs via the

- Query Registry transaction.

3.17.4.1.2 Message Semantics

The URI specifies the protocol and protocol parameters that are to be used to retrieve the document. The Document Repository shall support the following parameters for protocol in the URI:

- HTTP

The details of URI handling are specified in the HTTP standard (RFC 2616).

The Document Repository shall fully implement support for any protocol parameters that are required by the HTTP standard.

3.17.4.1.3 Expected Actions

A Retrieve Document Response will be generated in return. Details are specified in the HTTP standard.

3.17.4.2 Retrieve Document Response

3.17.4.2.1 Trigger Events

This message is triggered by the:

- Retrieve Document Request.

3.17.4.2.2 Message Semantics

The message semantics are specified in the HTTP standard.

3.17.4.2.3 Expected Actions

The Document Consumer now has the content of the document to process.

This is an extension (addition of a new actor and expected action) for a transaction initially defined in the context of the PIX Integration profile.

Replace section 3.8 with the following section of Volume 2 of the IHE ITI Technical framework.

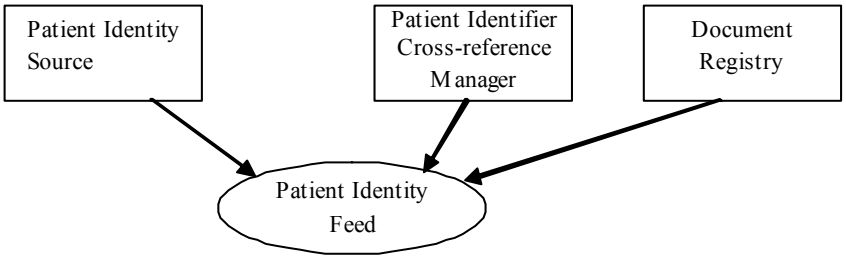
3.8 Patient Identity Feed

This section corresponds to Transaction 8 of the IHE IT Infrastructure Technical Framework. Transaction 8 is used by the Patient Identity Source, Patient Identifier Cross-reference Manager and Document Registry actors.

3.8.1 Scope

This transaction communicates patient information, including corroborating demographic data, after a patient's identity is established, modified or merged or after key corroborating demographic data has been modified.

3.8.2 Use Case Roles



Actor: Patient Identity Source

Role: Provides notification to the Patient Identifier Cross-reference Manager for any patient identification related events including: creation, updates, merges, etc.

Actor: Patient Identifier Cross-reference Manager

Role: Serves a well-defined set of Patient Identification Domains. Based on information provided in each Patient Identification Domain by a Patient Identification Source Actor, it manages the cross-referencing of patient identifiers across Patient Identification Domains.

Actor: Document Registry

Role: Uses patient identifiers provided to ensure that XDS Documents metadata registered is associated with a known patient and tracks any changes relevant to its operation (e.g. merge).

3.8.3 Referenced Standards

HL7 Version 2.3.1 Chapter 2 – Control, Chapter 3 – Patient Administration

- HL7 version 2.3.1 was selected for this transaction because it provides a broader potential base of Patient Identity Source Actors capable of participating in the profiles associated with this transaction.
- It allows existing ADT Actors from within IHE Radiology to participate as Patient Identity Source Actors.

3.8.4 Interaction Diagram

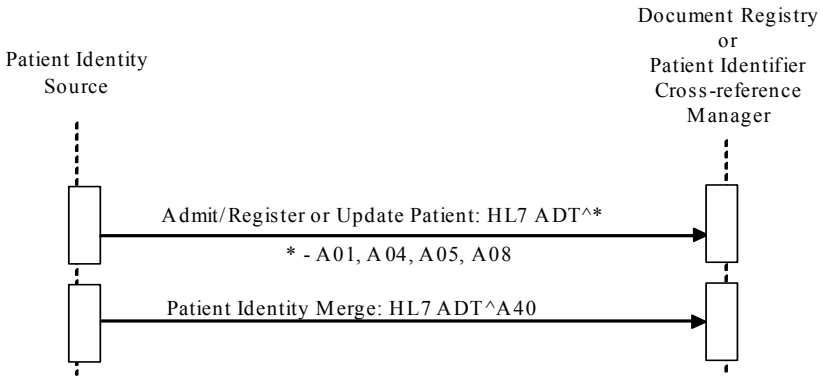


Figure 3.8-1 Patient Identity Sequence

3.8.4.1 Patient Identity Management – Admit/Register or Update Patient

3.8.4.1.1 Trigger Events

The following events from a Patient Identity Source Actor will trigger one of the Admit/Register or Update messages:

- A01 – Admission of an inpatient into a facility
- A04 – Registration of an outpatient for a visit of the facility
- A05 – Pre-admission of an inpatient (i.e., registration of patient information ahead of actual admission).

Changes to patient demographics (e.g., change in patient name, patient address, etc.) shall trigger the following Admit/Register or Update message:

- A08 – Update Patient Information

The Patient Identifier Cross-reference Manager shall only perform cross-referencing logic on messages received from Patient Identity Source Actors. For a given Patient Identifier Domain there shall be one and only one Patient Identity Source Actor, but a given Patient Identity Source Actor may serve more than one Patient Identifier Domain.

3.8.4.1.2 Message Semantics

The Patient Identity Feed transaction is conducted by the HL7 ADT message, as defined in the subsequent sections. The Patient Identity Source Actor shall generate the message whenever a

patient is admitted, pre-admitted, or registered, or when some piece of patient demographic data changes. Pre-admission of inpatients shall use the A05 trigger event. The segments of the message listed below are required, and their detailed descriptions are provided in the following subsections.

Note: Conventions used in this section as well as additional qualifications to the level of specification and HL7 profiling are stated in Appendix C and C.1 in this Volume.

Required segments are defined below. Other segments are optional

Table 3.8-1 ADT Patient Administration Messages

ADT	Patient Administration Message	Chapter in HL7 2.3.1
MSH	Message Header	2
EVN	Event Type	3
PID	Patient Identification	3
PV1	Patient Visit	3

Each message shall be acknowledged by the HL7 ACK message sent by the receiver of ADT message to its sender. See Appendix C.1.3, "Acknowledgement Modes", for definition and discussion of the ACK message.

This transaction does not require Patient Identity Source Actors to include any attributes not already required by the corresponding HL7 message (as is described in the following sections). This minimal set of requirements enables inclusion of the largest range of Patient Identity Source Actor systems.

This transaction **does** place additional requirements on the Patient Identifier Cross-reference Manager and Document Registry Actors, requiring them to accept a set of HL7 attributes beyond what is required by HL7. (See Section 3.8.4.1.3 for a description of these additional requirements).

3.8.4.1.2.1 MSH Segment

The MSH segment shall be constructed as defined in Appendix C.1.2 "Message Control".

Field *MSH-9 Message Type* shall have at least two components. The first component shall have a value of **ADT**; the second component shall have values of **A01**, **A04**, **A05** or **A08** as appropriate. The third component is optional; however, if present, it shall have a value of **ADT_A01**.

3.8.4.1.2.2 EVN Segment

The Patient Identity Source Actor is not required to send any attributes within the EVN segment beyond what is specified in the HL7 standard. See Table C.1-4 in Appendix C.1.4 "Common Segment Definitions" for the specification of this segment.

3.8.4.1.2.3 PID Segment

The Patient Identity Source Actor is not required to send any attributes within the PID segment beyond what is specified in the HL7 standard.

This message shall use the field PID-3 Patient Identifier List to convey the Patient ID uniquely identifying the patient within a given Patient Identification Domain.

The Patient Identity Source Actor shall provide the patient identifier in the ID component (first component) of the PID-3 field (PID-3.1). If the Patient Identity Source Actor provides component PID-3.4, Assigning Authority, then either the first subcomponent (namespace ID) or the second and third subcomponents (universal ID and universal ID type) shall be populated. If all three subcomponents are populated, the first subcomponent shall reference the same entity as is referenced by the second and third components.

3.8.4.1.2.4 PV1 Segment

The Admit/ Register or Update Patient message is not required to include any attributes within the PV1 segment beyond what is specified in the HL7 standard.

3.8.4.1.3 Expected Actions -- Patient Identifier Cross-reference Manager

The Patient Identifier Cross-reference Manager shall be capable of accepting attributes in the PID segment as specified in Table 3.8-2. This is to ensure that the Patient Identifier Cross-reference Manager can handle a sufficient set of corroborating information in order to perform its cross-referencing function.

Table 3.8-2 IHE Profile - PID segment

SEQ	LEN	DT	OPT	TBL#	ITEM#	ELEMENT NAME
1	4	SI	O		00104	Set ID - Patient ID
2	20	CX	O		00105	Patient ID
3	250	CX	R		00106	Patient Identifier List
4	20	CX	O		00107	Alternate Patient ID
5	250	XPN	R		00108	Patient Name
6	250	XPN	R+		00109	Mother's Maiden Name
7	26	TS	R+		00110	Date/Time of Birth
8	1	IS	R+	0001	00111	Administrative Sex
9	250	XPN	O		00112	Patient Alias
10	250	CE	O	0005	00113	Race
11	250	XAD	R2		00114	Patient Address
12	4	IS	O	0289	00115	County Code
13	250	XTN	R2		00116	Phone Number - Home
14	250	XTN	R2		00117	Phone Number - Business
15	250	CE	O	0296	00118	Primary Language
16	250	CE	O	0002	00119	Marital Status
17	250	CE	O	0006	00120	Religion

SEQ	LEN	DT	OPT	TBL#	ITEM#	ELEMENT NAME
18	250	CX	O		00121	Patient Account Number
19	16	ST	R2		00122	SSN Number – Patient
20	25	DLN	R2		00123	Driver's License Number - Patient
21	250	CX	O		00124	Mother's Identifier
22	250	CE	O	0189	00125	Ethnic Group
23	250	ST	O		00126	Birth Place
24	1	ID	O	0136	00127	Multiple Birth Indicator
25	2	NM	O		00128	Birth Order
26	250	CE	O	0171	00129	Citizenship
27	250	CE	O	0172	00130	Veterans Military Status
28	250	CE	O	0212	00739	Nationality
29	26	TS	O		00740	Patient Death Date and Time
30	1	ID	O	0136	00741	Patient Death Indicator

Adapted from the HL7 standard, Version 2.3.1

Note: This table reflects attributes required to be handled by the Patient Identifier Cross-reference Manager (receiver). It is likely that not all attributes marked as R2 or R+ above will be sent in some environments.

If the PID-3.4 (assigning authority) component is not included in the message (as described in Section 3.8.4.1.2.3) the Patient Identifier Cross-reference Manager shall fill PID-3.4 prior to storing the ID information and performing its cross-referencing activities. The information filled by the Patient Identifier Cross-reference Manager is based on the configuration associating each of the Patient Identity Source actors with the subcomponents of the correct assigning authority (namespace ID, UID and UID type). (See 3.8.4.1.3.1 below for a list of required Patient Identifier Cross-reference Manager configuration parameters).

A single Patient Identity Source Actor can serve multiple Patient Identification domains as long as it explicitly provides a fully qualified assigning authority. The Patient Identifier Cross-reference Manager Actor shall only recognize (by configuration) a single Patient Identity Source Actor per domain. (See 3.8.4.1.3.1 below for a list of required Patient Identifier Cross-reference Manager configuration parameters).

The cross-referencing process (algorithm, human decisions, etc.) is performed within the Patient Identifier Cross-reference Manager Actor, but its specification is beyond the scope of IHE.

Once the Patient Identifier Cross-reference Manager has completed its cross-referencing function, it shall make the newly cross-referenced identifiers available to PIX queries and send out notification to any Patient Identifier Cross-reference Consumers that have been configured (as being interested in receiving such notifications) using the PIX Update Notification transaction (see Section 3.10 for the details of that transaction).

3.8.4.1.3.1 Required Patient Identifier Cross-reference Manager Configuration

The following items are expected to be parameters that are configurable on the Patient Identifier Cross-reference Manager Actor. For each Identification Domain included in the Identification

Cross-reference Domain managed by a Patient Identifier Cross-reference Manager Actor, the following configuration information is needed:

- Identifier of the Identification Domain itself. This identifier shall specify the 3 components of the HL7 assigning authority (including the namespace ID and/or both the universal ID and universal ID type subcomponents) of the PID-3 field for the identification domain/source.
- Patient Identity Source Actor for that domain. This is expected to be the MSH-3 Sending Application field in the HL7 ADT message. (Alternative identification schemes might include the IP address of the Patient Identity Source Actor or Node Authentication if the Basic Security Profile defined by IHE Radiology is used.)
- Details about where in the HL7 ADT message (Identity Feed transaction) that the Source Actor will provide the patient identifier for the domain (e.g., first component of the PID-3 field). If a Source Actor is managing multiple Patient Identifier Domains then the Source Actor is required to include the assigning authority in its Identity Feed Transactions.

3.8.4.1.4 Expected Actions – Document Registry

The Document Registry shall be capable of accepting attributes in the PID segment as specified in Table 3.8-2. The patient Id feed transaction contains more than what the XDS Document Registry needs for its operation.

Table 3.8-2 IHE Profile - PID segment

SEQ	LEN	DT	OPT	TBL#	ITEM#	ELEMENT NAME
3	250	CX	R		00106	Patient Identifier List

Adapted from the HL7 standard, Version 2.3.1

Note: This table reflects only the attributes required to be handled by the Document Registry (receiver). Other attributes of the PID Segment may be ignored.

If the PID-3.4 (assigning authority) component is not included in the message (as described in Section 3.8.4.1.2.3) the Document Registry shall fill PID-3.4 prior to storing the ID information and performing its cross-referencing activities. The information filled by the Document Registry is based on the configuration associating each of the Patient Identity Source actors with the subcomponents of the correct assigning authority (namespace ID, UID and UID type). (See 3.8.4.1.3.1 below for a list of required Document Registry configuration parameters).

A single Patient Identity Source Actor can serve multiple Patient Identification domains as long as it explicitly provides a fully qualified assigning authority. The Document Registry Actor shall only recognize (by configuration) a single Patient Identity Source Actor per domain. (See 3.8.4.1.3.1 below for a list of required Document Registry configuration parameters).

The cross-referencing process (algorithm, human decisions, etc.) is performed within the Document Registry Actor, but its specification is beyond the scope of IHE.

Once the Document Registry has completed its cross-referencing function, it shall make the newly cross-referenced identifiers available to PIX queries and send out notification to any Patient Identifier Cross-reference Consumers that have been configured (as being interested in

receiving such notifications) using the PIX Update Notification transaction (see Section 3.10 for the details of that transaction).

3.8.4.1.4.1 Required Document Registry Configuration

The following items are expected to be parameters that are configurable on the Document Registry Actor:

- Identifier of the PatientIdentification Domain itself. This identifier shall specify the 3 components of the HL7 assigning authority (including the universal ID-OID and universal ID type subcomponents-ISO) of the PID-3 field for the identification domain/source.

3.8.4.2 Patient Identity Management –Patient Identity Merge (Merge Patient ID)

3.8.4.2.1 Trigger Events

When two patients' records are found to identify the same patient by a Patient Identity Source Actor in a Patient Identifier Domain and are merged, the Patient Identity Source shall trigger the following message:

- A40 – Merge Patient – Internal ID

An A40 message indicates that the Patient Identity Source Actor has done a merge within a specific Patient Identification Domain. That is, MRG-1 (patient ID) has been merged into PID-3 (Patient ID).

3.8.4.2.2 Message Semantics

The Patient Identity Feed transaction is an HL7 ADT message. The message shall be generated by the system (Patient Identity Source Actor) that performs the update whenever two patient records are found to reference the same person.

Note: Conventions used in this section as well as additional qualifications to the level of specification and HL7 profiling are stated in Appendix C and C.1 in this Volume.

The segments of the HL7 Merge Patient message listed below are required, and the detailed description of the message is provided in Section 3.8.4.2.2.1–3.8.4.2.2.6. The PV1 segment is optional.

Table 3.8-3 ADT A40 Patient Administration Message

ADT A40	Patient Administration Message	Chapter in HL7 v2.3.1
MSH	Message Header	2
EVN	Event Type	3
PID	Patient Identification	3
MRG	Merge Information	3
[PV1]	Patient Visit	3

Each message shall be acknowledged by the HL7 ACK message sent by the receiver of ADT message to its sender. See Appendix C.1.3 “Acknowledgement Modes” for definition and discussion of the ACK message.

A separate merge message shall be sent for each pair of patient records to be merged. For example, if Patients A, B, and C are all to be merged into Patient B, two ADT^A40 messages would be sent. In the first ADT^A40 message, patient B would be identified in the PID segment and Patient A would be identified in the MRG segment. In the second ADT^A40 message, patient B would be identified in the PID segment, and Patient C would be identified in the MRG segment.

Modification of any patient demographic information shall be done by sending a separate Update Patient Information (A08) message for the current Patient ID. An A40 message is the only method that may be used to update a Patient ID.

3.8.4.2.2.1 MSH Segment

MSH segment shall be constructed as defined in the Appendix C.1.2 “Message Control”.

Field *MSH-9 Message Type* shall have at least two components. The first component shall have a value of **ADT**; the second component shall have value of **A40**. The third component is optional; however, if present, it shall have a value of **ADT_A39**.

3.8.4.2.2.2 EVN Segment

See Appendix C.1.4 for the list of all required and optional fields within the EVN segment.

3.8.4.2.2.3 PID Segment

The PID segment shall be constructed as defined in Section 3.8.4.1.2.3.

3.8.4.2.2.4 MRG Segment

The PID and PV1 segments contain the dominant patient information, including Patient ID (and Issuer of Patient ID). The MRG segment identifies the “old” or secondary patient records to be de-referenced. HL7 does not require that the ”old” record be deleted; it does require that the "old" identifier shall not be referenced in future transactions following the merge.

IHE does not require that the Patient Identity Source Actor send any attributes within the MRG segment beyond what is specified in the HL7 standard. If the assigning authority component of MRG-1 is present (MRG-1.4), it shall be equal to the assigning authority component of PID-3 (PID-3.4)

3.8.4.2.2.5 PV1 Segment

PV1 segment shall be constructed as defined in Section 3.8.4.1.2.4.

3.8.4.2.3 Expected Actions -- Patient Identifier Cross-reference Manager

The Patient Identifier Cross-reference Manager shall be capable of accepting attributes in the MRG segment as specified in Table 3.8-4.

Table 3.8-4 IHE Profile - MRG segment

SEQ	LEN	DT	OPT	TBL#	ITEM#	ELEMENT NAME
1	250	CX	R		00211	Prior Patient Identifier List
2	250	CX	O		00212	Prior Alternate Patient ID
3	250	CX	O		00213	Prior Patient Account Number
4	250	CX	R2		00214	Prior Patient ID
5	250	CX	O		01279	Prior Visit Number
6	250	CX	O		01280	Prior Alternate Visit ID
7	250	XPN	R2		01281	Prior Patient Name

Adapted from the HL7 Standard, Version 2.3.1

In addition, the Patient Identifier Cross-reference Manager shall perform the Expected Actions as specified in Section 3.8.4.1.3.

When the Patient Identifier Cross-reference Manager receives the ADT^A40 message of the Patient Identity Feed transaction, it shall cross-reference the patient identifiers provided in the PID-3 and MRG-1 fields of the message by replacing any references it is maintaining internally to the patient ID provided in the MRG-1 field by the patient ID included in the PID-3 field. The Document Registry shall merge the Document Entries and Folders under the new patient Id.

3.8.4.2.4 Expected Actions – Document Registry

The Document Registry shall be capable of accepting attributes in the MRG segment as specified in Table 3.8-4.

Table 3.8-4 IHE Profile - MRG segment

SEQ	LEN	DT	OPT	TBL#	ITEM#	ELEMENT NAME
1	250	CX	R		00211	Prior Patient Identifier List
2	250	CX	O		00212	Prior Alternate Patient ID
3	250	CX	O		00213	Prior Patient Account Number
4	250	CX	R2		00214	Prior Patient ID
5	250	CX	O		01279	Prior Visit Number
6	250	CX	O		01280	Prior Alternate Visit ID
7	250	XPN	R2		01281	Prior Patient Name

Adapted from the HL7 Standard, Version 2.3.1

In addition, the Document Registry shall perform the Expected Actions as specified in Section 3.8.4.1.3.

When the Document Registry receives the ADT^A40 message type of the Patient Identity Feed transaction, it shall cross-reference the patient identifiers provided in the PID-3 and MRG-1

fields of the message by replacing any references it is maintaining internally to the patient ID provided in the MRG-1 field by the patient ID included in the PID-3 field. After the identifier references are replaced, the Document Registry shall reapply its internal cross-referencing logic/ policies before providing the updated information via either the PIX Query or PIX Notification Transactions.

Appendix H: Required Registry Initialization

The following XML is to be used to initialize an ebXML Registry as an XDS registry. It includes:

- Classification Schemes that anchor the definition of ExternalIdentifiers
- Additions to the ObjectType ClassificationScheme that introduces a general **XDS** ClassificationNode that anchors these additions. The usable new ClassificationNodes are: XDSDocumentEntry, XDSDocumentEntryStub, XDSFolder, and XDSSubmissionSet. XDSDocumentEntry and XDSDocumentEntryStub are used as new objectTypes for use in an ExtrinsicObjectto create XDS specific object types. XDSFolder and XDSSubmissionSet are used to classify RegistryPackage objects to label them as XDS Folders or XDS SubmissionSets.
- External Classification Schemes to support attribute coding.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
This submission initializes a ebXML Registry version 2.0 to act as an XDS Registry.
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<rs:SubmitObjectsRequest
    xmlns:query="urn:oasis:names:tc:ebxml-regrep:query:xsd:2.0"
    xmlns:rim="urn:oasis:names:tc:ebxml-regrep:rim:xsd:2.0"
    xmlns:rs="urn:oasis:names:tc:ebxml-regrep:registry:xsd:2.0"
    xmlns:tns="urn:oasis:names:tc:ebxml-regrep:rim:xsd:2.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="urn:ihe:xds:xsd:1.0"
    xsi:schemaLocation="urn:oasis:names:tc:ebxml-regrep:registry:xsd:2.0 rs.xsd">


    <rim:LeafRegistryObjectList>

        <!-- ++++++++++++++++++++++++++++++++++++++++++++++++
        Classification Schemes to anchor External Identifiers
        ++++++++++++++++++++++++++++++++++++++++++++++++ -->

        <rim:ClassificationScheme id="urn:uuid:58a6f841-87b3-4a3e-92fd-a8ffeff98427"
isInternal="true" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSDocument.recordTargetId"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = ""/>
            </rim:Description>
        </rim:ClassificationScheme>

        <rim:ClassificationScheme id="urn:uuid:2e82c1f6-a085-4c72-9da3-8640a32e42ab"
isInternal="true" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSDocument.uniqueId"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = ""/>
            </rim:Description>
        </rim:ClassificationScheme>


        <rim:ClassificationScheme id="urn:uuid:4b052cba-b03b-4233-8b27-e8d5e3f8d3e4"
isInternal="true" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSSubmissionSet.id"/>
            </rim:Name>
            <rim:Description>
```

```
                <rim:LocalizedString value = ""/>
            </rim:Description>
        </rim:ClassificationScheme>

        <rim:ClassificationScheme id="urn:uuid:554ac39e-e3fe-47fe-b233-965d2a147832"
isInternal="true" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSSubmissionSet.sourceId"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = ""/>
            </rim:Description>
        </rim:ClassificationScheme>

        <rim:ClassificationScheme id="urn:uuid:f64ffdf0-4b97-4e06-b79f-a52b38ec2f8a"
isInternal="true" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSFolder.id"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = ""/>
            </rim:Description>
        </rim:ClassificationScheme>

        <!-- ++++++++++++++++++++++++++++++++++++++++++++++++
        Additions to the ObjectType classification scheme. This adds:
        XDS ++
            +++ XDSDocumentEntry
            +++ XDSSubmissionSet
            +++ XDSFolder
            +++ XDSDocumentEntryStub

        XDS becomes a child of ObjectType
        ++++++++++++++++++++++++++++++++++++++++++++++++ -->

        <rim:ObjectRef id="urn:uuid:3188a449-18ac-41fb-be9f-99a1adca02cb"/>
        <rim:ClassificationNode
            id="urn:uuid:415715f1-fc0b-47c4-90e5-c180b7b82db6"
            parent="urn:uuid:3188a449-18ac-41fb-be9f-99a1adca02cb"
            code="XDS">
                <rim:Name>
                    <rim:LocalizedString value = "XDS"/>
                </rim:Name>

                <rim:ClassificationNode
                    id="urn:uuid:7edca82f-054d-47f2-a032-9b2a5b5186c1"
                    code="XDSDocumentEntry">
                    <rim:Name>
                        <rim:LocalizedString value = "XDSDocumentEntry"/>
                    </rim:Name>
                </rim:ClassificationNode>

                <rim:ClassificationNode id="urn:uuid:a54d6aa5-d40d-43f9-88c5-b4633d873bdd"
                    code="XDSSubmissionSet">
                    <rim:Name>
                        <rim:LocalizedString value = "XDSSubmissionSet"/>
                    </rim:Name>
                </rim:ClassificationNode>

                <rim:ClassificationNode
                    id="urn:uuid:d9d542f3-6cc4-48b6-8870-ea235fbc94c2"
                    code="XDSFolder">
                    <rim:Name>
                        <rim:LocalizedString value = "XDSFolder"/>
                    </rim:Name>
                </rim:ClassificationNode>
                <rim:ClassificationNode
                    id="urn:uuid:96fdda7c-d067-4183-912e-bf5ee74998a8"
```

```
                    code="XDSDocumentEntryStub">
                    <rim:Name>
                        <rim:LocalizedString value = "XDSDocumentEntryStub"/>
                    </rim:Name>
                </rim:ClassificationNode>
        </rim:ClassificationNode>

        <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++
        Additions to the AssociationType classification scheme. This adds:
        XDS ++
            +++ APND
            +++ RPLC
            +++ XFRM

        XDS becomes a child of AssociationType
        ++++++++++++++++++++++++++++++++++++++++++++++++++ -->

        <rim:ObjectRef id="urn:uuid:6902675f-2f18-44b8-888b-c91db8b96b4d"/>  <!-- AssociationType
-->
         <!-- Parent is AssociationType -->
        <rim:ClassificationNode
            id="urn:uuid:f9653189-fdd2-4c31-afbc-86c96ac8f0ad"
            parent="urn:uuid:6902675f-2f18-44b8-888b-c91db8b96b4d"
            code="XDS">
                <rim:Name>
                    <rim:LocalizedString value = "XDS"/>
                </rim:Name>

                <rim:ClassificationNode
                    id="urn:uuid:917dc511-f7da-4417-8664-de25b34d3def"
                    code="APND">
                    <rim:Name>
                        <rim:LocalizedString value = "APND"/>
                    </rim:Name>
                </rim:ClassificationNode>

                <rim:ClassificationNode
                    id="urn:uuid:60fd13eb-b8f6-4f11-8f28-9ee000184339"
                    code="RPLC">
                    <rim:Name>
                        <rim:LocalizedString value = "RPLC"/>
                    </rim:Name>
                </rim:ClassificationNode>

                <rim:ClassificationNode
                    id="urn:uuid:ede379e6-1147-4374-a943-8fcdcf1cd620"
                    code="XFRM">
                    <rim:Name>
                        <rim:LocalizedString value = "XFRM"/>
                    </rim:Name>
                </rim:ClassificationNode>
        </rim:ClassificationNode>

        <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++
        External Classification Schemes
        ++++++++++++++++++++++++++++++++++++++++++++++++++ -->

        <rim:ClassificationScheme id="urn:uuid:41a5887f-8865-4c09-adf7-e362475b143a"
isInternal="false" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSDocument.classCode"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = "An XDSDocumentEntry must have exactly one
Classification of this type."/>
            </rim:Description>
        </rim:ClassificationScheme>
```

```
        <rim:ClassificationScheme id="urn:uuid:f4f85eac-e6cb-4883-b524-f2705394840f"
isInternal="false" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSDocument.confidentialityCode"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = "An XDSDocumentEntry must have exactly one
Classification of this type."/>
            </rim:Description>
        </rim:ClassificationScheme>

        <rim:ClassificationScheme id="urn:uuid:2c6b8cb7-8b2a-4051-b291-b1ae6a575ef4"
isInternal="false" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSDocument.eventCodeList"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = "An XDSDocumentEntry may have zero or more
Classification of this type."/>
            </rim:Description>
        </rim:ClassificationScheme>

        <rim:ClassificationScheme id="urn:uuid:a09d5840-386c-46f2-b5ad-9c3699a4309d"
isInternal="false" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSDocument.formatCode"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = "An XDSDocumentEntry must have exactly one
Classification of this type."/>
            </rim:Description>
        </rim:ClassificationScheme>

        <rim:ClassificationScheme id="urn:uuid:f33fb8ac-18af-42cc-ae0e-ed0b0bdb91e1"
isInternal="false" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSDocument.healthCareFacilityTypeCode"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = "An XDSDocumentEntry must have exactly one
Classification of this type."/>
            </rim:Description>
        </rim:ClassificationScheme>

        <rim:ClassificationScheme id="urn:uuid:cccf5598-8b07-4b77-a05e-ae952c785ead"
isInternal="false" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSDocument.practiceSettingCode"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = "An XDSDocumentEntry must have exactly one
Classification of this type."/>
            </rim:Description>
        </rim:ClassificationScheme>

        <rim:ClassificationScheme id="urn:uuid:f0306f51-975f-434e-a61c-c59651d33983"
isInternal="false" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSDocument.typeCode"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = "An XDSDocumentEntry must have exactly one
Classification of this type."/>
            </rim:Description>
        </rim:ClassificationScheme>

        <rim:ClassificationScheme id="urn:uuid:aa543740-bdda-424e-8c96-df4873be8500"
isInternal="false" nodeType="EmbeddedPath">
```

```
            <rim:Name>
                <rim:LocalizedString value = "XDSSubmissionSet.contentTypeCode"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = ""/>
            </rim:Description>
        </rim:ClassificationScheme>

        <rim:ClassificationScheme id="urn:uuid:1ba97051-7806-41a8-a48b-8fce7af683c5"
isInternal="false" nodeType="EmbeddedPath">
            <rim:Name>
                <rim:LocalizedString value = "XDSFolder.codeList"/>
            </rim:Name>
            <rim:Description>
                <rim:LocalizedString value = ""/>
            </rim:Description>
        </rim:ClassificationScheme>

    </rim:LeafRegistryObjectList>
</rs:SubmitObjectsRequest>
```

Appendix I: Required Initialization of the Affinity Domain

This initialization supports the operation of the Registry Adaptor. The following information must be provided by the Affinity Domain administrator and loaded into the Registry Adaptor. This supports the functionality specified for the Registry Adaptor in section 3.14.4.1.2.11. How this information is loaded into the Registry Adaptor or how the Registry Adaptor is implemented is not defined by this profile.

1. List of acceptable mimeTypes for documents indexed by the registry.
2. PIX domain name (Assigning Authority) for XDS Affinity Domain. PatientIds attached to metadata submitted to this registry must come from this PIX Assigning Authority.
3. Acceptable values for all coded attributes represented in the registry by ebXML External Classifications. These include classCode, eventCode, confidentialityCode, healthCareFacilityTypeCode, formatCode for XDS Document and XDSSubmissionSet.code and XDSFolder.codeList.

Appendix J: Example Submissions and Query Results

J.1 Sample submission

This XML submits a new XDS document and a new XDS folder contained in an XDS Submission Set. The document is placed inside the folder.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
This example of a XDS Registry submission includes a:
    XDS Submission Set containing a,
    XDS Folder containing a,
    Single XDS Document.

This example covers the content for metadata for both the Provide and Register
transaction and the Register Transaction.  From the view of the metadata, the only
difference is that the Register transaction contains information on how thec
repository stored the document. Attributes/elements below that only apply to the
Register transaction are labeled "Register Transaction Only".  Otherwise, everything
applies to both transactions.

This applies to only one attribute, XdsDocument.URI.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<rs:SubmitObjectsRequest
    xmlns:query="urn:oasis:names:tc:ebxml-regrep:query:xsd:2.0"
    xmlns:rim="urn:oasis:names:tc:ebxml-regrep:rim:xsd:2.0"
    xmlns:rs="urn:oasis:names:tc:ebxml-regrep:registry:xsd:2.0"
    xmlns:tns="urn:oasis:names:tc:ebxml-regrep:rim:xsd:2.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="urn:ihe:xds:xsd:1.0"
    xsi:schemaLocation="urn:oasis:names:tc:ebxml-regrep:registry:xsd:2.0 rs.xsd">


    <rim:LeafRegistryObjectList>
        <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
             ++ Multiple ebXML Registry objects can be coded here, limited only by the
             ++ capacity of the XML parser and transforms
             ++ within the registry server.
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->

        <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
             ++ Here starts the coding of a single XDS Documnet
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->

        <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            ExtrinsicObject is a registry object that represents a document in a Document
            Repository. The UUID in the objectType attribute below identifies the document
            as being of type XDSDocumentEntry.

            The UUID/Name linkage (XdsDocumentEntry vs. urn:uuid:7edca82f-054d-47f2-a032-
9b2a5b5186c1) for
            the objectType is created as part of a calledScheme called ObjectType (defined by
ebXML
            Registry standard) which is extended by XDS.

            The id "theDocument" is symbolic within this file.
            Other elements within this file can reference this element by this symbolic name.
```

```
            The registry server creates a unique UUID to replace the symbolic name when this
            file is submitted.

            The ObjectRef declares that UUID as being already in the registry and not
            defined in this file.

            XdsDocumentEntry.title is coded as the Name element under the ExtrinsicObject.

            XdsDocumentEntry.availabilityStatus is coded as the status element of the
ExtrinsicObject below.
            When an object is created the status element is automatically generated internally by
t
            he registry with a value of "submitted" so it is not shown in this submission.

            XdsDocumentEntry.EntryUUID is implemented by the "id" attribute on the
ExtrinsicObject below. As
            mentioned above, it is specified symbolically below and will be assigned by the
registry.
         ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
        <rim:ObjectRef id="urn:uuid:7edca82f-054d-47f2-a032-9b2a5b5186c1"/>
        <rim:ObjectRef id="urn:uuid:58a6f841-87b3-4a3e-92fd-a8ffeff98427"/>
        <rim:ObjectRef id="urn:uuid:2e82c1f6-a085-4c72-9da3-8640a32e42ab"/>
        <rim:ObjectRef id="urn:uuid:41a5887f-8865-4c09-adf7-e362475b143a"/>
        <rim:ObjectRef id="urn:uuid:92d03292-84a0-4b86-8139-dd244173ddbb"/>
        <rim:ObjectRef id="urn:uuid:cccf5598-8b07-4b77-a05e-ae952c785ead"/>
        <rim:ObjectRef id="urn:uuid:2c6b8cb7-8b2a-4051-b291-b1ae6a575ef4"/>
        <rim:ObjectRef id="urn:uuid:f4f85eac-e6cb-4883-b524-f2705394840f"/>
        <rim:ObjectRef id="urn:uuid:f33fb8ac-18af-42cc-ae0e-ed0b0bdb91e1"/>
        <rim:ObjectRef id="urn:uuid:a09d5840-386c-46f2-b5ad-9c3699a4309d"/>

        <rim:ExtrinsicObject
            id="theDocument"
            objectType="urn:uuid:7edca82f-054d-47f2-a032-9b2a5b5186c1"
            mimeType="application/pdf">

            <rim:Name>
                <rim:LocalizedString value = "XdsDocumentEntry.title goes here"/>
            </rim:Name>

            <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                An ExternalIdentifier creates an optimized search key for a document. The
                attribute identificationScheme gives this external identifier a type.
                An object can have multiple ExternalIdentifiers. A name for this type
                (and possibly other details) can be found by finding the registry object
                with this UUID. It will be a Classification Scheme object.
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.patientId - assign patient id
                6578946^^^&7898372243&ISO to this document.
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:ExternalIdentifier
                identificationScheme="urn:uuid:58a6f841-87b3-4a3e-92fd-a8ffeff98427"
                value="6578946^^^%267898372243%26ISO">
                <rim:Name>
                    <rim:LocalizedString value = "patientId"/>
                </rim:Name>
            </rim:ExternalIdentifier>

            <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.uniqueId (document id)
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:ExternalIdentifier
                identificationScheme="urn:uuid:2e82c1f6-a085-4c72-9da3-8640a32e42ab"
                value="36784664">
                <rim:Name>
                    <rim:LocalizedString value = "uniqueId"/>
                </rim:Name>
            </rim:ExternalIdentifier>
```

```
<!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    XDSDocumentEntry.authorDepartment
 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<rim:Slot name="authorDepartment">
    <rim:ValueList>
        <rim:Value></rim:Value>
    </rim:ValueList>
</rim:Slot>

<!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    XDSDocumentEntry.authorInstitution
 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<rim:Slot name="authorInstitution">
    <rim:ValueList>
        <rim:Value>Fairview Hospital</rim:Value>
    </rim:ValueList>
</rim:Slot>

<!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    XdsDocumentEntry.authorPerson  (single value)
 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<rim:Slot name="authorPerson">
    <rim:ValueList>
        <rim:Value>^Welby^Marcus^^^Dr^MD</rim:Value>
    </rim:ValueList>
</rim:Slot>

<!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    XdsDocumentEntry.serviceStartTime  (single value)
 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<rim:Slot name="serviceStartTime">
    <rim:ValueList>
        <rim:Value>20041225T21:20:10</rim:Value>
    </rim:ValueList>
</rim:Slot>
<!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    XdsDocumentEntry.serviceStopTime  (single value)
 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<rim:Slot name="serviceStopTime">
    <rim:ValueList>
        <rim:Value>20041225T23:50:50</rim:Value>
    </rim:ValueList>
</rim:Slot>

<!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    XdsDocumentEntry.languageCode  (single value)
 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<rim:Slot name="languageCode">
    <rim:ValueList>
        <rim:Value>en-us</rim:Value>
    </rim:ValueList>
</rim:Slot>

<!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    XdsDocumentEntry.creationTime  (single value)
 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<rim:Slot name="creationTime">
    <rim:ValueList>
        <rim:Value>20041225T21:20:10</rim:Value>
    </rim:ValueList>
</rim:Slot>

<!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    XdsDocumentEntry.legalAuthenticator  (multi value)
 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<rim:Slot name="legalAuthenticator">
    <rim:ValueList>
```

```
                    <rim:Value>^Welby^Marcus^^^Dr^MD</rim:Value>
                </rim:ValueList>
            </rim:Slot>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.sourceRecordTargetId  (single value) oid+ext
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="sourcePatientId">
                <rim:ValueList>
                    <rim:Value>65789946+3982</rim:Value>
                </rim:ValueList>
            </rim:Slot>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.sourceRecordTargetInfo  (multi value)
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="sourcePatientInfo">
                <rim:ValueList>
                    <rim:Value></rim:Value>
                </rim:ValueList>
            </rim:Slot>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.size  (single value)
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="size">
                <rim:ValueList>
                    <rim:Value>3654</rim:Value>
                </rim:ValueList>
            </rim:Slot>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.hash  (single value)
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="hash">
                <rim:ValueList>
                    <rim:Value>da39a3ee5e6b4b0d3255bfef95601890afd80709</rim:Value>
                </rim:ValueList>
            </rim:Slot>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.classCode - an external classification.  An external
classification
                has a registry enforced type but a value enforced by some agent outside
                the registry, in this case the XDS Registry Adaptor. A document must have exactly
                one Classification of this type.

                Within the Classification object we encode the code, code display name, and
                coding scheme name. The following XML can be interpreted as:

                classificationScheme - identifies this Classification as classCode

                classifiedObject - object being classified

                nodeRepresentation - code value

                Name - code name

                codingScheme Slot - name of coding scheme
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
            <rim:Classification
                classificationScheme="urn:uuid:41a5887f-8865-4c09-adf7-e362475b143a"
                classifiedObject="theDocument"
                nodeRepresentation="18748-4">
                <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                    XdsDocumentEntry.classCodeDisplayName
                 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
                <rim:Name>
```

```
                    <rim:LocalizedString value="Diagnostic Imaging Report"/>
                </rim:Name>
                <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                    Coding scheme for classCode
                 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
                <rim:Slot name="codingScheme">
                    <rim:ValueList>
                        <rim:Value>LOINC</rim:Value>
                    </rim:ValueList>
                </rim:Slot>

            </rim:Classification>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.typeCode - an external classification.  An external
classification
                has a registry enforced type but a value enforced by some agent outside
                the registry, in this case the XDS Registry Adaptor. A document must have exactly
                one Classification of this type.

                Within the Classification object we encode the code, code display name, and
                coding scheme name. The following XML can be interpreted as:

                classificationScheme - identifies this Classification as typeCode

                classifiedObject - object being classified

                nodeRepresentation - code value

                Name - code name

                codingScheme Slot - name of coding scheme
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
            <rim:Classification
                classificationScheme="urn:uuid:41a5887f-8865-4c09-adf7-e362475b143a"
                classifiedObject="theDocument"
                nodeRepresentation="18748-4">
                <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                    XdsDocumentEntry.typeCodeDisplayName
                 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
                <rim:Name>
                    <rim:LocalizedString value="name"/>
                </rim:Name>
                <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                    Coding scheme for classCode
                 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
                <rim:Slot name="codingScheme">
                    <rim:ValueList>
                        <rim:Value>a coding scheme</rim:Value>
                    </rim:ValueList>
                </rim:Slot>

            </rim:Classification>

           <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.practiceSettingCode (exactly one classification of this type)
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Classification
                classificationScheme="urn:uuid:cccf5598-8b07-4b77-a05e-ae952c785ead"
                classifiedObject="theDocument"
                nodeRepresentation="Laboratory">
                <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                    XdsDocumentEntry.practiceSettingCodeDisplayName
                 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
                <rim:Name>
                    <rim:LocalizedString value="name"/>
                </rim:Name>
                <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
                Coding scheme for practiceSettingCode
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="codingScheme">
                <rim:ValueList>
                    <rim:Value>a coding scheme</rim:Value>
                </rim:ValueList>
            </rim:Slot>
        </rim:Classification>

        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            XdsDocumentEntry.eventCodeList (zero or more classifications of this type)
         ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
        <rim:Classification
            classificationScheme="urn:uuid:2c6b8cb7-8b2a-4051-b291-b1ae6a575ef4"
            classifiedObject="theDocument"
            nodeRepresentation="eventCode 1">
            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.eventCodeListDisplayName
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Name>
                <rim:LocalizedString value="name"/>
            </rim:Name>
            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                Coding scheme for eventCodeList
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="codingScheme">
                <rim:ValueList>
                    <rim:Value>a coding scheme</rim:Value>
                </rim:ValueList>
            </rim:Slot>
        </rim:Classification>
        <rim:Classification
            classificationScheme="urn:uuid:2c6b8cb7-8b2a-4051-b291-b1ae6a575ef4"
            classifiedObject="theDocument"
            nodeRepresentation="eventCode 2">
            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.eventCodeListDisplayName
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Name>
                <rim:LocalizedString value="name"/>
            </rim:Name>
            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                Coding scheme for eventCodeList
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="codingScheme">
                <rim:ValueList>
                    <rim:Value>a coding scheme</rim:Value>
                </rim:ValueList>
            </rim:Slot>
        </rim:Classification>

        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              XdsDocumentEntry.confidentialityCode (value = "protected")
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
        <rim:Classification
            classificationScheme="urn:uuid:f4f85eac-e6cb-4883-b524-f2705394840f"
            classifiedObject="theDocument"
            nodeRepresentation="protected">
            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.confidentialityCodeDisplayName
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Name>
                <rim:LocalizedString value="name"/>
            </rim:Name>
            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                Coding scheme for confidentialityCode
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="codingScheme">
```

```
                    <rim:ValueList>
                        <rim:Value>a coding scheme</rim:Value>
                    </rim:ValueList>
                </rim:Slot>
            </rim:Classification>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.healthCareFacilityTypeCode - an external classification.  The
classification
                has a registry enforced type but a  value enforced by the XDS Registry
                Adaptor (external to the registry). The classificationScheme identifies this as
the
                healthCareFacilityTypeCode. The nodeRepresentation attribute is
                the "value".
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
            <rim:Classification
                classificationScheme="urn:uuid:f33fb8ac-18af-42cc-ae0e-ed0b0bdb91e1"
                classifiedObject="theDocument"
                nodeRepresentation="clinic">
                <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                    XdsDocumentEntry.healthCareFacilityTypeCodeDisplayName
                 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
                <rim:Name>
                    <rim:LocalizedString value="name"/>
                </rim:Name>
                <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                    Coding scheme for healthCareFacilityTypeCode
                 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
                <rim:Slot name="codingScheme">
                    <rim:ValueList>
                        <rim:Value>a coding scheme</rim:Value>
                    </rim:ValueList>
                </rim:Slot>
            </rim:Classification>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsDocumentEntry.formatCode - an external classification.
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
            <rim:Classification
                classificationScheme="urn:uuid:a09d5840-386c-46f2-b5ad-9c3699a4309d"
                classifiedObject="theDocument"
                nodeRepresentation="">
                <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                    XdsDocumentEntry.formatCodeDisplayName
                 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
                <rim:Name>
                    <rim:LocalizedString value="name"/>
                </rim:Name>
                <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                    Coding scheme for formatCode
                 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
                <rim:Slot name="codingScheme">
                    <rim:ValueList>
                        <rim:Value>a coding scheme</rim:Value>
                    </rim:ValueList>
                </rim:Slot>
            </rim:Classification>

        </rim:ExtrinsicObject>

        <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            There are ancilliary objects that must be created to complete the definition
            of the XDSDocumentEntry.

            The ExternalLink object(below) defines a URI. There can be multiple ExternalLinks
associated
            with an object so assigning a name is important. XDS requires that the name
```

```
            of the ExternalLink referencing the document in the document repository be
"Document".
            Separately, an Association is used to link the ExternalLink object to the
            ExtrinsicObject.
         ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->


        <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            XdsDocumentEntry.URI - link to document in repository.

            This object has a symbolic id - documentURI which is used below in the Association.

            The Name element included since an object can have multiple ExternalLinks
            associated with it.

            Register Transaction Only

            The ExternalLink object is defined outside
            of the ExtrinsicObject above.  The ExternalLink and the ExtrinsicObject are linked
            by the the Association below.
         ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
        <rim:ExternalLink
            id="documentURI"
            externalURI="http://www.google.com">
            <rim:Name>
                <rim:LocalizedString value="URI"/>
            </rim:Name>
        </rim:ExternalLink>

        <rim:Association
            associationType="urn:uuid:92d03292-84a0-4b86-8139-dd244173ddbb"
            sourceObject="documentURI"
            targetObject="theDocument">
                <rim:Name>
                    <rim:LocalizedString value="URI"/>
                </rim:Name>
        </rim:Association>


        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            XdsDocumentEntry.parentDocumentId and parentDocumentRelationship are together
            coded as an Association.  As part of a CDA, the parentDocumentId is an OID.
            For registry, that OID must be translated into a registry UUID. A document
            OID is found in the attribute XdsDocumentEntry.uniqueId.

            The parentDocumentId is the targetObject. The UUID references an object already
            in the registry, so it must be coded as a raw UUID and an ObjectRef
            is required to declare to the parser that this UUID exists in the registry.

            The parentDocumentRelationship is the associationType.

            The base document (the one with a relationship with the parent) is
            coded as the sourceObject, "theDocument" in this example.

            The parent document is coded below as the targetObject.

            This code is commented out since no parent document exists when this
            exmaple is loaded.
         ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
    <!--    <rim:ObjectRef id="urn:uuid:a6e06ca8-0c75-4064-9e5c-88b9045a96f6"/>
        <rim:Association
            associationType="RPLC"
            sourceObject="theDocument"
            targetObject="urn:uuid:a6e06ca8-0c75-4064-9e5c-88b9045a96f6"/>
    -->
        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              ++ Here ends the coding of a single XDS Document.
```

```
            ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->


        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              ++   Here starts the coding of the XDS Submission Set.
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->

        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            The id="SubmissionSet" is used instead of a pre-assigned UUID for
            references between objects in this submission.
         ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
        <rim:ObjectRef id="urn:uuid:4b052cba-b03b-4233-8b27-e8d5e3f8d3e4"/>
        <rim:ObjectRef id="urn:uuid:554ac39e-e3fe-47fe-b233-965d2a147832"/>
        <rim:RegistryPackage id="SubmissionSet" description="Annual physical">
            <rim:Name>
                <rim:LocalizedString value = "Submission Set"/>
            </rim:Name>
        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            XdsSubmissionSet.comments is coded in the description attribute.
         ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
            <rim:Description>
                <rim:LocalizedString value = "Describe submission set here"/>
            </rim:Description>
            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XDSSubmissionSet.uniqueId - an ExternalIdentifier.
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:ExternalIdentifier
                identificationScheme="urn:uuid:4b052cba-b03b-4233-8b27-e8d5e3f8d3e4"
                value="3670984664"/>
            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XDSSubmissionSet.sourceId
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:ExternalIdentifier
                identificationScheme="urn:uuid:554ac39e-e3fe-47fe-b233-965d2a147832"
                value="8449607624"/>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XDSSubmissionSet.submissionTime  (single value)
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="submissionTime">
                <rim:ValueList>
                    <rim:Value>20041225T23:50:50</rim:Value>
                </rim:ValueList>
            </rim:Slot>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XDSSubmissionSet.authorDepartment
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="authorDepartment">
                <rim:ValueList>
                    <rim:Value></rim:Value>
                </rim:ValueList>
            </rim:Slot>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XDSSubmissionSet.authorInstitution
             ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="authorInstitution">
                <rim:ValueList>
                    <rim:Value>Fairview Hospital</rim:Value>
                </rim:ValueList>
            </rim:Slot>

            <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
                XDSSubmissionSet.authorPerson
             +++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:Slot name="authorPerson">
                <rim:ValueList>
                    <rim:Value></rim:Value>
                </rim:ValueList>
            </rim:Slot>
        </rim:RegistryPackage>

        <!--  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            XDSSubmissionSet.contentTypeCode - an external classification.
         +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
        <rim:ObjectRef id="urn:uuid:aa543740-bdda-424e-8c96-df4873be8500"/>
        <rim:Classification
            classificationScheme="urn:uuid:aa543740-bdda-424e-8c96-df4873be8500"
            classifiedObject="SubmissionSet"
            nodeRepresentation="Routine Physical"/>

        <!--  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            This classification labels the above RegistryPackage as an XDSSubmissionSet.
            The classificaitonNode shown exists under the ClassificationScheme ObjectTypes.
         +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
        <rim:ObjectRef id="urn:uuid:a54d6aa5-d40d-43f9-88c5-b4633d873bdd"/>
        <rim:Classification
            classifiedObject="SubmissionSet"
            classificationNode="urn:uuid:a54d6aa5-d40d-43f9-88c5-b4633d873bdd"/>

        <!--  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            Link the document above and the Folder below into the Submission Set with an
Association of type "HasMember"
            The UUID for HasMember is urn:uuid:2d03bffb-f426-4830-8413-bab8537a995b
         +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
        <rim:Association
            associationType="urn:uuid:2d03bffb-f426-4830-8413-bab8537a995b"
            sourceObject="SubmissionSet"
            targetObject="theDocument"/>
        <rim:Association
            associationType="urn:uuid:2d03bffb-f426-4830-8413-bab8537a995b"
            sourceObject="SubmissionSet"
            targetObject="Folder"/>

        <!--  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              ++ Here ends the coding of the XDS Submission Set.
              +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              +++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->

        <!--  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              ++ Here starts the coding of the XDS Folder.
              +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              +++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
        <rim:ObjectRef id="urn:uuid:f64ffdf0-4b97-4e06-b79f-a52b38ec2f8a"/>
        <rim:RegistryPackage id="Folder">
            <rim:Name>
                <rim:LocalizedString value = "FOLDER"/>
            </rim:Name>
        <!--  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            XDSFolder.comments
         +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
            <rim:Description>
                <rim:LocalizedString value = "comments go here"/>
            </rim:Description>
            <!--  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XDSFolder.uniqueId
             +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:ExternalIdentifier
```

```
                identificationScheme="urn:uuid:f64ffdf0-4b97-4e06-b79f-a52b38ec2f8a"
                value="367084664"/>

            <!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                XdsFolder.patientId - assign patient id
                6578946+312 to this Folder. It must match the patientId on all documents
                include in the Folder.  It uses the same ClassificationScheme to
                support the ExternalIdentifier as XDSDocumentEntry.patientId.
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
            <rim:ExternalIdentifier
                identificationScheme="urn:uuid:58a6f841-87b3-4a3e-92fd-a8ffeff98427"
                value="6578946+312"/>

        </rim:RegistryPackage>

        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            XDSFolder.codeList - a collection (multiple) external classifications.
            Shown below is a single value.
         ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
        <rim:ObjectRef id="urn:uuid:1ba97051-7806-41a8-a48b-8fce7af683c5"/>
        <rim:Classification
            classificationScheme="urn:uuid:1ba97051-7806-41a8-a48b-8fce7af683c5"
            classifiedObject="Folder"
            nodeRepresentation="11548-7"/>

        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            This classification labels the above RegistryPackage as an XDS
            Folder. The classificaitonNode shown exists under the
            ClassificationScheme ObjectTypes.
         ++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
        <rim:ObjectRef id="urn:uuid:d9d542f3-6cc4-48b6-8870-ea235fbc94c2"/>
        <rim:Classification
            classifiedObject="Folder"
            classificationNode="urn:uuid:d9d542f3-6cc4-48b6-8870-ea235fbc94c2"/>

        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            Link the document into the Folder with an Association of
            type "HasMember"
         ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->
        <rim:Association
            associationType="urn:uuid:2d03bffb-f426-4830-8413-bab8537a995b"
            sourceObject="Folder"
            targetObject="theDocument"/>
        <!--  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              ++ Here ends the coding of the XDS Folder.
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              ++++++++++++++++++++++++++++++++++++++++++++++++++++++++  -->

    </rim:LeafRegistryObjectList>
</rs:SubmitObjectsRequest>
```

J.2 Example Query Response Returning Object References

Given the query:

```
<?xml version="1.0"?>
<AdhocQueryRequest>
  <ResponseOption returnType="ObjectRef" returnComposedObjects="true"/>
  <SQLQuery>
    SELECT *
      FROM ExtrinsicObject e
      WHERE e.id='urn:uuid:fbeacdb7-5421-4474-9267-985007cd8855'
  </SQLQuery>
</AdhocQueryRequest>
```

Which asks for ObjectRefs (list of object UUIDs) instead of object content to be returned. The result is:

```
<?xml version="1.0"?>
<RegistryResponse
    status="Success"
    xmlns="urn:oasis:names:tc:ebxml-regrep:registry:xsd:2.1">
  <AdhocQueryResponse
      xmlns="urn:oasis:names:tc:ebxml-regrep:query:xsd:2.1">
    <SQLQueryResult>
      <ObjectRef id="urn:uuid:fbeacdb7-5421-4474-9267-985007cd8855"/>
    </SQLQueryResult>
  </AdhocQueryResponse>
</RegistryResponse>
```

J.3 Example Query Response Returning Object Content

Given the query:

```
<?xml version="1.0"?>
<AdhocQueryRequest>
  <ResponseOption returnType="LeafClass" returnComposedObjects="true"/>
  <SQLQuery>
    SELECT *
      FROM ExtrinsicObject e
      WHERE e.id='urn:uuid:fbeacdb7-5421-4474-9267-985007cd8855'
  </SQLQuery>
</AdhocQueryRequest>
```

which asks for object content to be returned. The result is:

```
<?xml version="1.0"?>

<RegistryResponse
   status="Success"
   xmlns="urn:oasis:names:tc:ebxml-regrep:registry:xsd:2.1">
   <AdhocQueryResponse
      xmlns="urn:oasis:names:tc:ebxml-regrep:query:xsd:2.1">
      <SQLQueryResult>
         <ExtrinsicObject
            id="urn:uuid:fbeacdb7-5421-4474-9267-985007cd8855"
            objectType="urn:uuid:7edca82f-054d-47f2-a032-9b2a5b5186c1"
            majorVersion="1"
            minorVersion="0"
            status="Submitted"
            mimeType="application/octet-stream"
            isOpaque="false">
            <Name>
               <LocalizedString
                  lang="en-us"
                  charset="UTF-8"
                  value="XdsDocumentEntry.title goes here"/>
            </Name>
            <Description/>
            <Slot
               name="authorInstitution">
               <ValueList>
                  <Value>
                  </Value>
               </ValueList>
            </Slot>
            <Slot
               name="authorPerson">
               <ValueList>
                  <Value>
                  </Value>
               </ValueList>
            </Slot>
            <Slot
               name="codeDisplayName">
               <ValueList>
```

```
            <Value>Value of codeDisplayName
            </Value>
         </ValueList>
      </Slot>
      <Slot
         name="eventCodeDisplayName">
         <ValueList>
            <Value>Event Code Display Name
            </Value>
         </ValueList>
      </Slot>
      <Slot
         name="hash">
         <ValueList>
            <Value>da39a3ee5e6b4b0d3255bfef95601890afd80709
            </Value>
         </ValueList>
      </Slot>
      <Slot
         name="languageCode">
         <ValueList>
            <Value>en-us
            </Value>
         </ValueList>
      </Slot>
      <Slot
         name="legalAuthenticator">
         <ValueList>
            <Value>
            </Value>
            <Value>
            </Value>
         </ValueList>
      </Slot>
      <Slot
         name="serviceStartTime">
         <ValueList>
            <Value>2004-12-25T21:20:10
            </Value>
         </ValueList>
      </Slot>
      <Slot
         name="serviceStopTime">
         <ValueList>
            <Value>2004-12-25T23:50:50
            </Value>
         </ValueList>
      </Slot>
      <Slot
         name="size">
         <ValueList>
            <Value>3654
            </Value>
         </ValueList>
      </Slot>
      <Slot
         name="recordTargetIdAssignBySource">
         <ValueList>
            <Value>65789946+3982
            </Value>
         </ValueList>
      </Slot>
      <Slot
         name="recordTargetInfoAssignBySource">
         <ValueList>
            <Value>Name=Joe Smith
            </Value>
            <Value>Dob=4/13/44
```

```
                </Value>
                <Value>Sex=M
                </Value>
             </ValueList>
          </Slot>
          <Classification
             id="urn:uuid:d15c410b-b8c5-4619-b47b-c957c8b5e8da"
             objectType="Classification"
             classificationScheme="urn:uuid:41a5887f-8865-4c09-adf7-e362475b143a"
             classifiedObject="urn:uuid:fbeacdb7-5421-4474-9267-985007cd8855"
             nodeRepresentation="theCode">
             <Name/>
             <Description/>
          </Classification>
          <Classification
             id="urn:uuid:3b198e2f-cba4-4f85-b5c6-d957f6b0bc80"
             objectType="Classification"
             classificationScheme="urn:uuid:2c6b8cb7-8b2a-4051-b291-b1ae6a575ef4"
             classifiedObject="urn:uuid:fbeacdb7-5421-4474-9267-985007cd8855"
             nodeRepresentation="theEventCode">
             <Name/>
             <Description/>
          </Classification>
          <Classification
             id="urn:uuid:308411da-e85b-4b98-948a-886045b3e32b"
             objectType="Classification"
             classificationScheme="urn:uuid:f4f85eac-e6cb-4883-b524-f2705394840f"
             classifiedObject="urn:uuid:fbeacdb7-5421-4474-9267-985007cd8855"
             nodeRepresentation="protected">
             <Name/>
             <Description/>
          </Classification>
          <Classification
             id="urn:uuid:5c31436a-5fe6-4b58-8b80-89ae62e283f4"
             objectType="Classification"
             classificationScheme="urn:uuid:f33fb8ac-18af-42cc-ae0e-ed0b0bdb91e1"
             classifiedObject="urn:uuid:fbeacdb7-5421-4474-9267-985007cd8855"
             nodeRepresentation="clinic">
             <Name/>
             <Description/>
          </Classification>
          <Classification
             id="urn:uuid:ac7be990-11e2-47d0-ba48-a880c448dbc7"
             objectType="Classification"
             classificationScheme="urn:uuid:a09d5840-386c-46f2-b5ad-9c3699a4309d"
             classifiedObject="urn:uuid:fbeacdb7-5421-4474-9267-985007cd8855"
             nodeRepresentation="">
             <Name/>
             <Description/>
          </Classification>
          <ExternalIdentifier
             id="urn:uuid:cc69926e-0ca1-4121-88cc-b17c29557748"
             objectType="ExternalIdentifier"
             identificationScheme="urn:uuid:2e82c1f6-a085-4c72-9da3-8640a32e42ab"
             value="36784664">
             <Name/>
             <Description/>
          </ExternalIdentifier>
          <ExternalIdentifier
             id="urn:uuid:35a325a1-16ba-4b79-b2d8-a64af471eaa6"
             objectType="ExternalIdentifier"
             identificationScheme="urn:uuid:58a6f841-87b3-4a3e-92fd-a8ffeff98427"
             value="6578946+312">
             <Name/>
             <Description/>
          </ExternalIdentifier>
       </ExtrinsicObject>
       <ObjectRef
```

```
            id="urn:uuid:41a5887f-8865-4c09-adf7-e362475b143a"/>
         <ObjectRef
            id="urn:uuid:fbeacdb7-5421-4474-9267-985007cd8855"/>
         <ObjectRef
            id="urn:uuid:2c6b8cb7-8b2a-4051-b291-b1ae6a575ef4"/>
         <ObjectRef
            id="urn:uuid:f4f85eac-e6cb-4883-b524-f2705394840f"/>
         <ObjectRef
            id="urn:uuid:f33fb8ac-18af-42cc-ae0e-ed0b0bdb91e1"/>
         <ObjectRef
            id="urn:uuid:a09d5840-386c-46f2-b5ad-9c3699a4309d"/>
         <ObjectRef
            id="urn:uuid:2e82c1f6-a085-4c72-9da3-8640a32e42ab"/>
         <ObjectRef
            id="urn:uuid:58a6f841-87b3-4a3e-92fd-a8ffeff98427"/>
      </SQLQueryResult>
   </AdhocQueryResponse>
</RegistryResponse>
```

Appendix K : XDS Security Environment

This Appendix expands on the summary provided in the XDS Volume 1 specification (ITI TF-1: Appendix H).

The XDS operations assume that a suitable security and privacy environment has been established. Almost all of the relevant threats will be managed by agreements, policies, and technologies that are external to the XDS transactions. The few that affect the XDS transactions will be managed by generic security mechanisms that are not unique to XDS. The threats and security objectives that must be addressed are described in sections 1 and 2 below. Only a few of these have issues that are unique to the XDS application.

Section 3 discusses these few threats and objectives in terms of the agreements and policies that need to be established to create a suitable environment for XDS. Establishing these agreements often involves business agreement discussions that are part of establishing the XDS affinity group. These agreements are necessary because the exchange of documents implies agreeing to the delegation of responsibility for maintaining the security of these documents and for providing the necessary audit and record keeping facilities.

K.1 Security Environment

K.1.1 Threats

Specific threats to the overall XDS system are listed below. These threats are identified using the Common Criteria nomenclature defined by ISO 17799. Most of these are mitigated by policies, procedures, and technologies that are not unique to XDS and do not require any special XDS considerations. Many of these mitigations do require that the parties within the XDS affinity group have agreement on details of how they will work together.

- **T.ADMIN_ERROR** Improper administration may result in defeat of specific security features.
- **T.ADMIN_ROGUE** Authorized administrator's intentions may become malicious resulting in TSF data to be compromised.
- **T.AUDIT_CORRUPT** A malicious process or user may cause audit records to be lost or modified, or prevent future records from being recorded by taking actions to exhaust audit storage capacity, thus masking an attacker's actions.
- **T.CONFIG_CORRUPT** A malicious process or user may cause configuration data or other trusted data to be lost or modified.
- **T.DISASTER** System or network may failure due to disaster (e.g. fire, earthquake).
- **T.DOS** A malicious process or user may block others from system resources via a resource exhaustion denial of service attack.

- **T.EAVESDROP** A malicious process or user may intercept transmitted data inside or outside of the enclave. Some of the XDS environments are not concerned with eavesdrop exposure. They may employ external protective mechanisms such as physical network security or VPNs to protect against eavesdropping.
- **T.HARDWARE** Hardware may malfunction.
- **T.IMPROPER_INSTALLATION** XDS components may be delivered, installed, or configured in a manner that undermines security.
- **T.INSECURE_START** Reboot may result in insecure state of the operating system.
- **T.INTRUSION** Malicious software (e.g. virus) may be introduced into the system.
- **T.MASQUERADE** A malicious process or user on one machine on the network may masquerade as an entity on another machine on the same network.
- **T.OBJECTS_NOT_CLEAN** Systems may not adequately remove the data from objects between usage by different users, thereby releasing information to a user unauthorized for the data. This also includes swapping hard disk with PHI during service and repair.
- **T.POOR_DESIGN** Unintentional or intentional errors in requirement specification, design or development of the TOE components may occur.
- **T.POOR_IMPLEMENTATION** Unintentional or intentional errors in implementing the design of the XDS environment may occur.
- **T.POOR_TEST** Incorrect system behavior may result from inability to demonstrate that all functions and interactions within the XDS operation are correct.
- **T.REPLAY** A malicious process or user may gain access by replaying authentication (or other) information.
- **T.SPOOFING** A hostile entity may masquerade itself as part of the XDS affinity group and communicate with authorized users who incorrectly believe they are communicating with authorized members.
- **T.SYSACC** A malicious process or user may gain unauthorized access to the administrator account, or that of other trusted personnel.
- **T.UNATTENDED_SESSION** A malicious process or user may gain unauthorized access to an unattended session.
- **T.UNAUTH_ACCESS** Unauthorized access to data by a user may occur. This includes access via direct user interaction with the device, access via network transactions, and access via removable electronic and printed media.
- **T.UNAUTH_MODIFICATION** Unauthorized modification or use of XDS attributes and resources may occur.

- **T.UNDETECTED_ACTIONS** Failure of the XDS components to detect and record unauthorized actions may occur.
- **T.UNIDENTIFIED_ACTIONS** Failure of the administrator to identify and act upon unauthorized actions may occur.
- **T.UNKNOWN_STATE** Upon failure of XDS components, the security of the XDS environment may be unknown.
- **T.USER_CORRUPT** User data may be lost or tampered with by other users.

K.1.2 Security and Privacy Policy

There are a wide variety of security and privacy regulations established by law and regulation. These are interpreted and extended to create individual enterprise policies. This equipment will be installed into a variety of enterprises that are subject to a variety of laws and regulations. The XDS environment will provide support for the common aspects of these enterprise policies. The policy statements whose enforcement must be provided by the XDS security mechanisms are:

- **P.ACCOUNT** The users of the system shall be held accountable for their actions within the system.
- **P.AUTHORIZATION** The system must limit the extent of each user's abilities in accordance with the TSPP. (See P.PATIENT_CARE)
- **P.AUTHORIZED_USERS** Only those users who have been authorized to access the information within the system may access the system. (See P.PATIENT_CARE)
- **P.CRYPTOGRAPHY** The system shall use standard approved cryptography (methods and implementations) for key management (i.e., generation, access, distribution, destruction, handling, and storage of keys) and cryptographic services (i.e., encryption, decryption, signature, hashing, key exchange, and random number generation services).
- **P.DECLARATIVE_SECURITY** The system shall allow the administrator to define security related rules. Examples include defining access control policies and password expiration restriction.
- **P.I_AND_A** All users must be identified and authenticated prior to accessing any controlled resources with the exception of public objects.
- **P.OBJECTAUTHORIZATION** The XDS components must enforce the policy regarding how authorization is established for protected objects. The policy determines how access control and other policies are enforced. (This is often considered part of P.Authorization, but in the XDS context it may make sense to consider this as a separate policy.)
- **P.PATIENT_CARE** The security and privacy measures should not prevent patient care. In particular, there should be emergency bypass mechanisms to override security when necessary to provide patient care.

- **P.SYSTEM_INTEGRITY** The system must have the ability to periodically validate its correct operation and, with the help of Administrators, Backup and Restore Operators, and Service Personnel, it must be able to recover from any errors that are detected.
- **P.TRACE** The primary method for enforcing the security and privacy policy is the use of auditing. The XDS components must have the ability to review the actions of individuals. The XDS environment must provide sufficient audit information to external audit and monitoring systems to permit the review of actions of individuals by that other system.
- **P.TRUSTED_RECOVERY** Procedures and/or mechanisms shall be provided to assure that, after a system failure or other discontinuity, recovery without a protection compromise is obtained
- **P.VULNERABILITY_SEARCH** The XDS environment must undergo an analysis for vulnerabilities beyond those that are obvious.

K.1.3 Security Usage Assumptions

Assumptions of the use of the XDS environment:

- **A.PHYSICAL** It is assumed that appropriate physical security is provided within the domain for the value of the IT assets and the value of the stored, processed, and transmitted information.
- **A. AUDIT_REVIEW** It is assumed that there will be audit repository and review services provided that can accept audit information from the XDS components in real time.
- **A.OPERATION** It is assumed that networks, firewalls, etc. are deployed and maintained to meet appropriate network security levels.
- **A.PERSONNEL** It is assumed that the organization can assure IT user & other workforce personal integrity/trustworthiness.
- **A.PKI** It is assumed that there will be a facility to provide signed certificates as needed for node and user authentication. The key management maybe done manually or automatically depending on the availability of appropriate technology.

K.2 Security Objectives

This section defines the security objectives for the XDS environment. These objectives are suitable to counter all identified threats and cover all identified organizational security policies and assumptions. Common Criteria nomenclature is used. The XDS component security objectives are identified with “O.” appended to at the beginning of the name and the environment objectives are identified with “OE.” appended to the beginning of the name.

K.2.1 XDS Component Security Objectives

- **O.ACCESS** The XDS components will ensure that users gain only authorized access to it and to the resources that it controls. (See O.EMERGENCY_BYPASS)
- **O.ACCESS_HISTORY** The XDS components will display information (to authorized users) related to previous attempts to establish a session.
- **O.ADMIN_ROLE** The XDS components will provide separate administrator roles to isolate administrative actions. These include a General Administrator role, a Backup and Restore Operator role, a Cryptographic Administrator role, and a Service Personnel role. Additional roles can be defined. These roles are collectively called Administrators.
- **O.ADMIN_TRAINED** The XDS components will provide authorized Administrators with the necessary information for secure management and operation.
- **O.AUDIT_GENERATION** The XDS components will provide the capability to detect and create records of security and privacy relevant events associated with users. The XDS components will reliably transmit this information to the central audit repository, and provide reliable local storage of events until the central audit repository has confirmed receipt. (See OE.AUDIT_REVIEW)
- **O.AUDIT_PROTECTION** Each XDS component will provide the capability to protect audit information within its scope of control.
- **O.AUDIT_REVIEW** If an external central audit repository is not part of the environment, the components will be configured to provide limited capability to analyze and selectively view audit information. (See OE.AUDIT_REVIEW)
- **O.CONFIG_MGMT** All changes to the components and its development evidence will be tracked and controlled.
- **O.DECLARATIVE_SECURITY** The components will allow security functions and access control to be defined by the authorized administrator.
- **O.DISASTER_RECOVERY** The components should allow the authorized Administrators to perform backup and restore of electronic data, and rapid configuration and reconfiguration of device operation. In addition, the TOE should support administrative procedures to restore operation after disasters that may have substantially destroyed portions of the hospital operation and where substitute temporary systems are in place.
- **O.DISCRETIONARY_ACCESS** The components will control accesses to resources based upon the identity of users and the role of users. (See O.EMERGENCY_BYPASS)
- **O.DISCRETIONARY_USER_CONTROL** The components will allow authorized users to specify which resources may be accessed by which users and groups of users. (See O.EMERGENCY_BYPASS)
- **O.EMERGENCY_BYPASS**The XDS components should allow access to any secured data during a declared medical emergency.

- **O.ENCRYPTED_CHANNEL** Based on the environmental policies, encryption may be used to provide confidentiality of protected data in transit over public network.
- **O.INSTALL** The XDS components will be delivered with the appropriate installation guidance in the form of installation manuals and training to establish and maintain component security.
- **O.INTRUSION_DETECTION** The XDS components will ensure intrusion of malicious software (e.g. virus) is detected.
- **O.MANAGE** The XDS components will provide all the functions and facilities necessary to support the authorized Administrators in their management of the security of the TOE.
- **O.PROTECT** The XDS components will provide means to protect user data and resources.
- **O.RECOVERY** Procedures and/or mechanisms will be provided to assure that recovery is obtained without a protection compromise, such as from system failure or discontinuity.
- **O.REMOTE_SERVICE** The XDS components will provide the means for remote service without sacrificing security or privacy policy.
- **O.RESIDUAL_INFORMATION** The XDS components will ensure that any information contained in a protected resource is not released when the resource is reallocated. Information on permanent media such as hard disk shall be secured during service and repair.
- **O.RESOURCE_SHARING** No user will block others from accessing resources.
- **O.SELF_PROTECTION** Each XDS component will maintain a domain for its own execution that protects itself and its resources from external interference, tampering, or unauthorized disclosure.
- **O.TRAINED_USERS** The XDS environment will provide authorized users with the necessary guidance for secure operation.
- **O.TRUSTED_PATH** **The TOE** will provide a means to ensure users are not communicating with some other entity pretending to be the TOE. This covers entity authentication. (See O.USER_AUTHENTICATION.)
- **O.TRUSTED_SYSTEM_OPERATION** The XDS components will function in a manner that maintains security.
- **O.USER_AUTHENTICATION** The XDS components will verify the claimed identity of the interactive user. (See O.ENTITY_AUTHENTICATION.)
- **O.USER_IDENTIFICATION** The XDS components will uniquely identify the interactive users.

K.2.2 Environment Security Objectives

- **OE.PHYSICAL** Physical security will be provided within the domain for the value of the IT assets protected by the XDS environment and the value of the stored, processed, and transmitted information.
- **OE.AUDIT_REVIEW** There may be an audit repository and review service provided that can accept audit information from the XDS environment in real time. This facility will provide review and analysis functions. (See O.AUDIT_GENERATION, O.AUDIT_REVIEW)
- **OE.OPERATION** Networks, firewalls, etc. are deployed and maintained to meet appropriate network security levels.
- **OE.PERSONNEL** Assure IT user & other workforce personal integrity/trustworthiness.
- **OE.PKI** There will be a facility to provide signed certificates as needed for node and user authentication.

K.3 Functional Environment

The XDS can be modelled as having four different organizations that have a delegated responsibility relationship where each organization has a different functional responsibility. In some configurations a single organization is responsible for two or more of these functions, which makes delegation much easier. This section discusses the major areas that must be solved.

The four functions are:

1. **Creator** – This functional organization has created the PHI and is legally responsible to the patient and others for providing healthcare and for protecting this data.
2. **Repository** – This functional organization is responsible for providing access to persistent documents to readers. The creator has delegated responsibility to the repository to provide adequate protection for a subset of the PHI. This subset is called the document.
3. **Registry** - This functional organization is responsible for providing query services to readers. The creator has delegated responsibility to the to the registry to provide adequate protection for a subset of the PHI. This subset is called the metadata.
4. **Reader** – This functional organization is providing healthcare services that make use of data that is contained in the metadata and the documents.

There are three levels of difficulty in delegation.

"**Trivial**" delegation is that where it is not necessary to delegate the responsibility for implementing the threat mitigation. In those cases it does not matter whether the organizations have the same policy or mitigations. For example, if the registry provides adequate mitigation

against the threat of disaster, it need not be concerned with the disaster related policies of the reader.

"**Easy**" delegation is that where the two organizations have the equivalent policies. In those cases there is an initial difficult phase of discovering that the policies are the same and evaluating that the mitigation strategies are acceptable. This results in a simple binary decision to approve or disapprove a business relationship permitting the exchange of data. With the exception of the three policy classes described as "hard" below, the details of policies are likely to differ, but the goals are sufficiently uniform that a simple business decision can be made.

For the "easy" delegation, the IHE transactions must provide adequate mitigations for the threats so that the business decision to exchange data can be made based simply on review of the partners policies and mitigations. This means that some IHE transactions will have additional security requirements attached. For example, encryption to avoid the threat of eavesdropping may be required. These requirements are not unique to XDS and will be able to use standardized security features like TLS and VPN tools. These requirements may be significantly different from the usual practice within an enterprise, because of the differences in the environment.

"**Hard**" delegation is that where the two organizations have different policies or inconsistent/incompatible mitigation strategies. These are likely to occur for the following policies, where organizations often disagree on the details of the policy goals, and where policies often change:

a. **P.Authorization** – The authorized access policies and authorized modification policies often differ, and are often subject to change. The changes that occur are often at a detailed level, e.g. access rights to a particular patient information may change. This means that either there is an agreed mechanism to propogate changes, or an acceptance that policy changes may not be enforced, or there will be restrictions on the data exchange to avoid delegating responsibility for data that is subject to change.

b. **P.Account and P.trace** – The policies for accountability and traceability often differ. These are much less subject to change, but it is often difficult to reconcile delegation when these policies differ. This will be an especially difficult issue for repository and registry functions that support multiple different creator organizations.

c. **P.ObjectAuthorization** – The policies regarding creation and modification of access rights often differ.

In addition, any of the policy and threat mitigations may be determined to be unacceptable by creator, registry, or repository. In the simple situation where there are only four real world participants this simply means that there is no business relationship. In the more complex world where the registry or repository are in many relationships with many creators and readers it introduces a serious problem. Either the registry and repository must limit its relationship to that small set of creators and readers that mutually accept all the policies and mitigations of all the other organizations, or there must be a mitigation strategy so that creators can restrict delegations

by the registry and repository to only those readers that have policies and mitigations that are acceptable to the creator.

Mitigations for differences include the following:

a. Limit the data exchange to that data where the differences are not significant. For example, highly sensitive data like psychiatric notes might not be shared, while relatively insignificant data like allergy information is shared.

b. Provide a revocation mechanism to deal with policy changes, so that future delegations can be prohibited. It is often impractical to revoke past delegations because the PHI has already been disclosed. But the revocation mechanism can stop further delegation from taking place. This revocation mechanism must be part of the P.Authorization and P.ObjectAuthorization policies and must be mutually acceptable for this mitigation to be effective.

c. Trusted third party inspections and audits can sometimes deal with reconciliation of differences in P.Account and P.Trace.

d. An "approved delegation" list identifying acceptable and unacceptable creator/reader pairs can mitigate the repository and registry issues when the reader has incompatible policies with the creator. This does require the creator to accept the approved delegation policy and implementation of the repository and registry, but it reduces the combinatorial explosion of policy combinations between creators, repositories, registries, and readers into a linear growth in complexity.

e. The "approved delegation" may go further into identification of persons, but this is only a viable path when all parties have policies the easily support delegation of personal responsibility. Persons are usually required to comply with organizational policies, and organizations generally use roles rather than persons to establish policies. The often viable exception is the special case of the "deny access to person X". This can be a viable means of dealing with situations involving a conflict of interest. This kind of access denial may be applicable to just a particular subset of the PHI exchanged, (e.g. denying access to an ex-spouse).

These mitigations do not directly change the technical requirements for the XDS transactions. They are policy decisions that may affect how particular actors are configured. The implementation of XDS actors will need to be aware that this kind of site-specific configuration management and policy control will be routinely required.

Appendix L:Relationship of Document Entry Attributes and Document Headers

XDS Document Entry attributes, placed in the XDS Document Registry by Document Sources, may be derived from header data present in the document content. Although the XDS Integration Profile does not mandate a strict relationship, this appendix illustrates sample mappings of XDS Document Entry attributes to header fields of some standard document formats. This relationship does not imply that values are mapped or copied directly as transformations may be needed between conventions in the EHR-CR and EHR-LR (e.g. vocabulary mappings).

Table L-1 Relationship of XDS Document Attributes to Document header fields

Attribute	CDA R1-2000	CDA R2 Draft Aug 2004	EHRCOM
patientId	levelone >clinical_document_header >>patient >>>id *mapped into XDS Affinity Domain patient id domain*	ClinicalDocument >recordTarget >>patientRole >>>id *mapped into XDS Affinity Domain patient id domain*	Class: EHR_EXTRACT attribute: subject_of_care[1]: II *mapped into XDS Affinity Domain patient id domain*
serviceStartTime	levelone >clinical_document_header> >patient_encounter >>>encounter_tmr	ClinicalDocument >documentationOf >>event >>>effectiveTime low=	Class: CLINICAL_SESSION attribute: session_time[1]: IVL<TS>
serviceStopTime	levelone >clinical_document_header >>patient_encounter >>>encounter_tmr	ClinicalDocument >documentationOf >>event >>>effectiveTime high=	
classCode	*Inferred from* levelone >clinical_document_header >>document_type_cd RT= EX=	*Inferred from* ClinicalDocument >code codeSystem= code=	Class COMPOSITION Attribute: to be added.
classCodeDisplay Name	*Inferred from* levelone >clinical_document_header >>document_type_cd DN=	*Inferred from* ClinicalDocument >code codeSystem= code=	

Attribute	CDA R1-2000	CDA R2 Draft Aug 2004	EHRCOM
practiceSettingCode	levelone >clinical_document_header >>patient_encounter >>>practice_setting_cd V= S=	*Inferred from* ClinicalDocument >code codeSystem= code=	(need input from CEN TC 251)
practiceSettingCode DisplayName	levelone >clinical_document_header >>patient_encounter >>>practice_setting_cd DN=	*Inferred from* ClinicalDocument >code codeSystem= code=	
healthcareFacility TypeCode	*Inferred from* levelone >clinical_document_header >>patient_encounter >>>practice_setting_cd V= S=	*Inferred from* ClinicalDocument >code codeSystem= code=	Class CLINICAL_SESSION attribute: healthcare_facility[0..1]: II
healthcareFacility TypeCodeDisplay Name	*Inferred from* levelone >clinical_document_header >>patient_encounter >>>practice_setting_cd DN=	*Inferred from* ClinicalDocument >code codeSystem= code=	
availabilityStatus	N/A (Generated and maintained by the Registry)	N/A (Generated and maintained by the Registry)	N/A (Generated and maintained by the Registry)
uniqueId	levelone >clinical_document_header >>id	ClinicalDocument >id	Class RECORD_COMPONENT attribute: rc_id[1]: II
typeCode	levelone >clinical_document_header >>document_type_cd RT= EX=	ClinicalDocument >code codeSystem= code=	Class RECORD_COMPONENT attribute: meaning[0..1]: CV
typeCodeDisplay Name	levelone >clinical_document_header >>document_type_cd DN=	ClinicalDocument >code displayName=	
formatCode		ClinicalDocument >typeId	Class EHR_EXTRACT attribute: rm_id[1]: String
eventCode	*Inferred from* levelone >clinical_document_header >>document_type_cd	*Inferred from* ClinicalDocument >code codeSystem=	(need input from CEN TC 251)

Attribute	CDA R1-2000	CDA R2 Draft Aug 2004	EHRCOM
	RT= EX=	code=	
eventCodeDisplay Name	*Inferred from* levelone >clinical_document_header >>document_type_cd RT= EX=	*Inferred from* ClinicalDocument >code codeSystem= code=	(need input from CEN TC 251)
title	*Inferred from* levelone >clinical_document_header >>document_type_cd DN=	ClinicalDocument >title	Class: RECORD_COMPONENT attribute: name[1]: TEXT
authorInstitution	levelone >clinical_document_header >>originating_organization >>>organization	ClinicalDocument >author >>assignedAuthor >>>representedOrganization >>>>name	Class CLINICAL_SESSION attribute: healthcare_facility[0..1]: II
authorPerson	levelone >clinical_document_header >>originator >>>person	ClinicalDocument >author >>assignedAuthor >>>assignedAuthorChoice >>>>person	Class: COMPOSITION attribute: composer[0..1]: II
legalAuthenticator	levelone >clinical_document_header >>legal_authenticator >>>person	ClinicalDocument >legalAuthenticator >>assignedEntity >>>person	Class FUNCTIONAL_ROLE (association from class ATTESTATION) attribute: performer[1]: II
URI	N/A	N/A	N/A
parentDocument Relationship	levelone >clinical_document_header >>document_relationship >>>document_relationship.type_cd	ClinicalDocument >relatedDocument typeCode=	IN THE CASE OF REPLACEMENT Class: AUDIT_INFO attribute: revision_status CS_REV_STAT IN THE CASE OF ADDENDUM or TRANSFORM Class LINK attribute nature: CV
parentDocumentId	levelone >clinical_document_header >>document_relationship	ClinicalDocument >relatedDocument >>parentDocument	IN THE CASE OF REPLACEMENT attribute:

Attribute	CDA R1-2000	CDA R2 Draft Aug 2004	EHRCOM
	>>>related_document >>>>id	>>>id	previous_version[0..1]: II This attribute uniquely identifies the RECORD_COMPONENT of which the current RECORD_COMPONENT is a revision (null for the first ever version). IN THE CASE OF ADDENDUM or TRANSFORM Class LINK Attribute: target[1]: II
confidentialityCode	levelone >clinical_document_header >>confidentiality_cd RT= EX=	ClinicalDocument >confidentialityCode	Class RECORD_COMPONENT attribute: sensitivity[1]: CS_SENSITIVITY
languageCode	xml:lang attribute	ClinicalDocument >relatedDocument typeCode=	This attribute is a property of all text data types in CEN, and so we have not defined a separate overall language to govern the whole document. It might be reasonable to assume that the natural language used for the name attribute is considered to be a reasonable guide to the value of this attribute.
patientId AssignBySource	levelone >clinical_document_header >>patient >>>person >>>>id	ClinicalDocument >recordTarget >>patientRole >>>id	Class: EHR_EXTRACT attribute: subject_of_care[1]: II
patientInfo AssignBySource	levelone >clinical_document_header >>patient >>>person >>>>person_name	ClinicalDocument >recordTarget >>patientRole >>>patientPatient >>>>name	
size	N/A Total length of submitted document.	N/A Total length of submitted document.	N/A Total length of submitted document.

Attribute	CDA R1-2000	CDA R2 Draft Aug 2004	EHRCOM
hash	N/A Hash of submitted document.	N/A Hash of submitted document.	N/A Hash of submitted document.
entryUUID	N/A Generated by registry	N/A Generated by registry.	N/A Generated by registry.