Integrating the Healthcare Enterprise

IHE IT Infrastructure
Technical Framework Supplement

Restricted Metadata Update
(RMU)

Revision 1.0 – Draft for Public Comment

Date: May 23, 2018

Author: IHE ITI Technical Committee

Email: iti@ihe.net

Please verify you have the most recent version of this document. See here for Trial Implementation and Final Text versions and here for Public Comment versions.

Copyright © 2018: IHE International, Inc.

Foreword

This is a supplement to the IHE IT Infrastructure Technical Framework V14.0. Each supplement undergoes a process of public comment and trial implementation before being incorporated into the volumes of the Technical Frameworks.

This supplement is published on May 23, 2018 for public comment. Comments are invited and can be submitted at http://www.ihe.net/ITI_Public_Comments. In order to be considered in development of the trial implementation version of the supplement, comments must be received by June 22, 2018.

This supplement describes changes to the existing technical framework documents.

"Boxed" instructions like the sample below indicate to the Volume Editor how to integrate the relevant section(s) into the relevant Technical Framework volume.

Amend section X.X by the following:

Where the amendment adds text, make the added text **bold underline**. Where the amendment removes text, make the removed text ~~**bold strikethrough**~~. When entire new sections are added, introduce with editor's instructions to "add new text" or similar, which for readability are not bolded or underlined.

General information about IHE can be found at www.ihe.net.

Information about the IHE ITI domain can be found at ihe.net/IHE_Domains.

Information about the organization of IHE Technical Frameworks and Supplements and the process used to create them can be found at http://ihe.net/IHE_Process and http://ihe.net/Profiles.

The current version of the IHE V14.0 Technical Framework can be found at http://ihe.net/Technical_Frameworks.

CONTENTS

Introduction to this Supplement

Attention Readers:

Material originally released in the XDS Metadata Update Supplement is used for the development of a new transaction in this profile that includes the attributes and semantics used for versioning metadata.

The ITI Technical Committee determined that it was appropriate to reuse the work already developed as it met the criteria for the proposed use case(s).

By only introducing the Update Document Set [ITI-57] transaction's Update DocumentEntry Metadata operation for this RMU Profile, complexity would be reduced for implementers that do not need full administrative capabilities included in the XDS Metadata Update Supplement. This profile does not include support for updating Submission Set, Folder, and Association objects.

While the RMU Profile depends on metadata versioning semantics for the new transaction, this does not preclude using it in environments that are not capable of storing and maintaining full metadata history.

As this RMU Profile supports updating only DocumentEntry metadata objects, the receiving actor will be expected to perform Association Propagation to maintain the view of the metadata. How the implementer interprets this requirement depends on the actor option selected.

The ITI Technical Committee intends on maintaining compatibility with the feature set between both this profile and the Metadata Update Supplement.

Public comments on this strategy are welcomed.

The Restricted Metadata Update (RMU) Profile provides a mechanism for changing Document Sharing metadata both within and across community boundaries in a controlled manner. Certain metadata attributes have been restricted from modification to allow for predictable and safe use in a wide range of operating environments. Communities can use this Profile as part of meeting legislative requirements for patient control of their distributed medical information.

Open Issues and Questions

RMU_001: Do we need to define a new or extra metadata attribute that indicates the DocumentEntry object was modified by an actor from another community? Should this be left for implementers to decide?

Resolution: The Technical Committee will await community feedback before making a decision.

RMU_018: Should this profile include any discussion or support for the DocumentAvailability attribute?

Resolution: The profile currently includes this attribute being carried forward from the Metadata Update Supplement. During TCON, it was agreed this attribute cannot be change using ITI-X1.

RMU_021: Should we bring forward ITI TF-1 Section 10.4.14 and ITI TF-2a Section 3.18.4.1.2.3.5.1 from XDS Metadata Update?

Resolution: It is the Technical Committee's intent to eventually bring the underlying content together for both profiles.

RMU_022: Should the metadata attributes, hash and size, be restricted? Do we have the list of allowed and restricted attributes correct?

Resolution: At this time, we have allowed modification of the hash and size attributes.

Closed Issues

RMU_002: Are DocumentEntry objects the only objects which shall be modifiable by this profile?

Resolution (F2F, 14-Feb-2018): yes, because use cases relate only to DocumentEntry objects.

RMU_003: Do we need a white list/a black list of DocumentEntry attributes which can be modified by this profile?

Resolution (F2F, 14-Feb-2018): yes, white list is created.

RMU_004: Shall DocumentEntry attributes creationTime, serviceStartTime, serviceStopTime be modifiable?

Resolution (F2F, 14-Feb-2018): yes. These attributes may have a special meaning e.g., in the BPPC context, but in this case the receiving actor can reject the change.

RMU_005: Shall DocumentEntry attributes patientId, sourcePatientId, sourcePatientInfo be modifiable?

Resolution (F2F, 14-Feb-2018):

- *patientId is not modifiable because this is a structural attribute.*
- *sourcePatientId is not modifiable because it contains the patient ID which was valid at the submission time.*
- *sourcePatientInfo is modifiable because its value does not influence anything.*

RMU_006: Do we need a new actor “RMU Document Administrator” in the Initiating Community?

Resolution (TCON, 04-April-2018): No. The Update Initiator Actor provides similar capabilities.

RMU_007: Do we need the Asynchronous WS option?

Resolution: Not at this time but could be added in the future. At this time, most use cases indicate that synchronous web services would be used.

RMU_008: Do we need two transactions or in one enough?

Resolution (F2F, 14-Feb-2018): Because of their similarity, the intra and cross-community transactions have been coalesced into a single transaction being differentiated by the use of the homeCommunityId.

RMU_010: Should the Update Initiator be grouped with an XDS Document Consumer with Metadata Update Option instead of embedding the transaction within the RMU actor? This could allow for the use of X-Community Retrieve Document set, as well.

Resolution: While likely needed for the successful execution of the RMU, it was determined that the formal grouping of an XDS Document Consumer distracted from the central functionality being added by the profile. Instead, it was added as Cross Profile Consideration.

RMU_011: Is UpdateAvailabilityStatus operation in or out?

Resolution: No – It was determined that to keep implementation simple, this operation would not be introduced into the transaction.

RMU_012: Should the homeCommunityId be set in @home attribute, RequestList slot, and/or SOAP header?

Resolution: The decision was made to use the @home attribute following the pattern established by XCA. It was felt that this would allow the Update Initiator to more easily compose the request message using the previously obtained response. The XCDR approach was discussed, but participants agreed that verifying the homeCommunityId in both the SOAP header and body was not desirable as currently written.

RMU_014: Should a workflow that uses Remove Metadata and Document Profile be included?

Resolution: Out of scope. A new work item maybe submitted for consideration next year.

RMU_015: Should the XCA with XDS Affinity Domain Option be included in the profile?

Resolution: No-See RMU_010.

RMU_016: Should Restricted Metadata Update be considered a replacement for the Metadata Update Supplement?

Resolution: The Restricted Metadata Update (RMU) Profile currently supports only the updating of metadata for DocumentEntry objects. The primary goal of this profile was to enable cross-community support for metadata updates, but still relies on nearly all of its mechanics previously *established in the Metadata Update Supplement. The mechanics apply whether used within a community or across multiple communities.*

As a result, this profile was written in way so that either this profile or the supplement could reach final text independent of one another. Thus, it should be considered as a refinement of the existing Update Document Set transaction, but not as a direct replacement.

RMU_017: Can Restricted Metadata Update be used for updates within a "federated" community (e.g.: Update Responder forward the request to all known holders of the data)?

Resolution: No – The remote community must be identified prior to issuing the Restricted Update Metadata transaction.

RMU_019: How should we incorporate the requirements for the XDS option, Document Metadata Update Option? If brought into this profile, this could introduce a required grouping with XDS for the RMU actors.

Resolution: No – The Technical Committee did not want to introduce a mandatory requirement for implementers to use full metadata versioning as described in the XDS Metadata Update Supplement.

RMU_020: Should this profile allow the Update Initiator to set the value for Association Propagation to "no"? If not, should this profile address this concept directly? Is it sufficient to leave the requirements as expected actions on the Update Responder?

Resolution: In order to maintain consistency with the XDS Metadata Update supplement, the Association Propagation attribute will remain in this Profile, but be force to "yes".

Editor Note: Comment numbers RMU_009 & RMU_013 were unintentionally skipped.

General Introduction and Shared Appendices

The IHE Technical Framework General Introduction and Shared Appendices are components shared by all of the IHE domain technical frameworks. Each technical framework volume contains links to these documents where appropriate.

*Update the following appendices to the General Introduction as indicated below. Note that these are **not** appendices to Volume 1.*

Appendix A – Actor Summary Definitions

Add the following actors to the IHE Technical Frameworks General Introduction Appendix A:

Actor Name	Definition
Update Initiator	An Update Initiator creates requests to update previously stored metadata.
Update Responder	An Update Responder processes requests to update previously stored metadata.

Appendix B – Transaction Summary Definitions

Add the following transactions to the IHE Technical Frameworks General Introduction Appendix B:

Transaction Name and Number	Definition
Restricted Update Document Set [ITI-X1]	The Restricted Update Document Set is used to request updates to a restricted set of metadata attributes for a document in a community.

Appendix D – Glossary

*Add the following **new** glossary terms to the IHE Technical Frameworks General Introduction Appendix D.*

Glossary Term	Definition
Metadata Object Instance	An object representing a single version of a logical metadata object.
Logical Metadata Object	A collection of metadata object instances describing a single object. Each instance represents a different version of the object's metadata.
Metadata Annotation	An XML element used in a submission to further describe or trigger additional behavior by a receiving actor for a submitted metadata object.

Volume 1 – Profiles

Add new Section X

X Restricted Metadata Update (RMU) Profile

The Restricted Metadata Update (RMU) Profile provides a mechanism for changing Document Sharing metadata both within and across community boundaries in a controlled manner. RMU enables modification of a restricted set of Document Sharing metadata attributes (see Section X.4.1) to allow for predictable and safe use in a wide range of operating environments. Communities can use this Profile as part of meeting legislative requirements for patient control of their distributed medical information

X.1 RMU Actors, Transactions, and Content Modules

Figures X.1-1 show the actors directly involved in the RMU Profile and the relevant transactions between them. If needed for context, other actors that may be indirectly involved due to their participation in other related profiles are shown in dotted lines. Actors which have a required grouping are shown in conjoined boxes (see Section X.3).

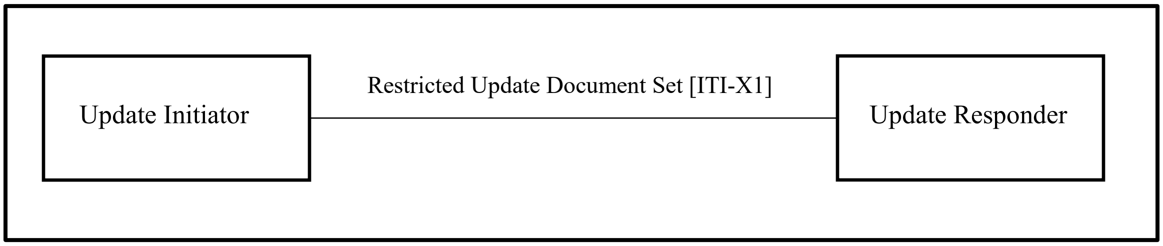


Figure X.1-1: RMU Actor Diagram

Table X.1-1 lists the transactions for each actor directly involved in the RMU Profile. To claim compliance with this profile, an actor shall support all required transactions (labeled "R") and may support the optional transactions (labeled "O").

Table X.1-1: RMU Profile - Actors and Transactions

Actors	Transactions	Initiator or Responder	Optionality	Reference
Update Initiator	Restricted Update Document Set [ITI-X1]	Initiator	R	ITI TF-2c: 3.X1
Update Responder	Restricted Update Document Set [ITI-X1]	Responder	R	ITI TF-2c: 3.X1

X.1.1 Actor Descriptions and Actor Profile Requirements

Most requirements are documented in ITI TF-2 Transactions. This section documents any additional requirements on profile's actors.

X.1.1.1 Update Initiator

The Update Initiator creates requests for updating metadata previously stored either within or outside the Document Sharing community. This update may have been initiated by an automated mechanism or manual administrative procedure, in accordance with a predetermined agreement or policy.

The Update Initiator will perform the update to the most recent version of the Document Sharing metadata using the patient identifier and terminology known to the storing community. This profile does not specify how the Update Initiator obtains the metadata; potential mechanisms are the results from a prior query where the data is stored, or grouping with an XDS Document Source that published the metadata originally.

The Restricted Update Document Set [ITI-X1] transaction is used by the Update Initiator to transmit the update request to the Update Responder for processing. The success or failure of the transaction is returned in the response.

X.1.1.2 Update Responder

The Update Responders accepts a Restricted Update Document Set [ITI-X1] transaction from an Update Initiator. If the Update Responder services the target community for the request, the Update Responder will persist the request. Otherwise, the Update Responder may optionally forward the request to destination community for processing. The Update Responder will indicate the success or failure of the transaction and the response will be augmented with the homeCommunityId of the community servicing the request.

X.2 RMU Actor Options

Options that may be selected for each actor in this profile, if any, are listed in the Table X.2-1. Dependencies between options, when applicable, are specified in notes.

Table X.2-1: RMU - Actors and Options

Actor	Option Name	Reference
Update Initiator	None	
Update Responder	Forward Update Option (Note 1)	ITI TF-1: X.2.1
	Persistence Option (Note 1)	ITI TF-1: X.2.2
	XDS Persistence Option (Note 2)	ITI TF-1: X.2.3
	XCA Persistence Option (Note 2)	ITI TF-1: X.2.4

Note 1: An actor shall claim support for at least one of these two options.

Note 2: An actor may claim support for one of these options only when the Persistence Option is also supported.

X.2.1 Forward Update Option

The Update Responder supporting the Forward Update Option shall be grouped with an Update Initiator. When it receives a Restricted Update Document Set [ITI-X1] transaction request for a remote Document Sharing community, the grouped Update Initiator will forward the request to the remote community's Update Responder.

X.2.2 Persistence Option

An Update Responder that supports the Persistence Option shall process the Restricted Update Document Set [ITI-X1] request and store the updated metadata.

X.2.3 XDS Persistence Option

An Update Responder that supports the XDS Persistence Option is capable of processing the Restricted Update Document Set [ITI-X1] transaction request and storing the results with full metadata versioning semantics as described in ITI TF-3: 4.1.5 – Metadata Object Versioning Semantics (currently in the XDS Metadata Update Trial Implementation Supplement).

The Update Responder that supports the XDS Persistence Option is capable of returning the updated metadata in a Registry Stored Query [ITI-18] transaction response including metadata versioning semantics as described in ITI TF-2a: 3.18.4.1.2.3.5.1 - Compatibility Issues (currently in the XDS Metadata Update Trial Implementation Supplement)..

X.2.4 XCA Persistence Option

The Update Responder that supports the XCA Persistence Option is capable of processing the Restricted Update Document Set [ITI-X1] request and storing the updated metadata.

The Update Responder that supports the XCA Persistence Option is capable of returning the updated metadata in Cross Gateway Query [ITI-38] responses according to requirements found in ITI TF-2b: 3.38.4.1.2.3 - Special handling of some stored queries.

X.3 Required Actor Groupings

An actor from this profile (Column 1) shall implement all of the required transactions and/or content modules in this profile ***in addition to*** ***<u>all</u>*** of the requirements for the grouped actor (Column 2) (Column 3 in alternative 2).

Section X.5 describes some optional groupings that may be of interest for security considerations and Section X.6 describes some optional groupings in other related profiles.

Table X.3-1: RMU - Required Actor Groupings

RMU Actor	Grouping Condition	Actor to be grouped with	Reference
Update Initiator	Required	ATNA / Secure Node or Secure Application	ITI TF-1: 9.1.1
	Required	CT / Time Client	ITI TF-1: 7.1

RMU Actor	Grouping Condition	Actor to be grouped with	Reference
Update Responder	Required	ATNA / Secure Node or Secure Application	ITI TF-1: 9.1.1
	Required	CT / Time Client	ITI TF-1: 7.1
	Forward Update Option	RMU / Update Initiator	ITI TF-1: X.1.1.1

X.4 Overview

X.4.1 Concepts

The Restricted Metadata Update (RMU) Profile provides a mechanism for updating Document Sharing metadata both within and across community boundaries in a controlled manner. By restricting which structural metadata attributes can be changed, this Profile can be applied in situations where legislation grants patients control of their medical information.

This Profile can be combined with other IHE profiles, such as XCA and XDS, to build patient longitudinal records that can be maintained both within and across medical communities and national borders.

The Profile maintains the stewardship of the patient record in the community where patient record was stored. When patient records are shared across communities, it is expected that the record will contain the patient identifier and coding system values from the community where the record originated.

The Profile expects that the latest version of the stored DocumentEntry metadata object(s) will be used as the basis for the update submission.

In order to maintain interoperability among participating communities, certain metadata attributes are restricted from modification as they describe the current state of DocumentEntry object, or the stored physical document.

This Profile permits updating the following metadata attributes:

- author
- classCode
- comments
- confidentialityCode
- creationTime
- eventCodeList
- formatCode
- hash

- healthcareFacilityTypeCode
- languageCode
- legalAuthenticator
- mimeType
- practiceSettingCode
- referenceIdList
- serviceStartTime
- serviceStopTime
- size
- sourcePatientInfo
- title
- typeCode
- URI

Affinity Domain policies may impose additional restrictions on the metadata attributes included in the preceding list. It is expected that a document sharing agreement will be established among domain participants prior to deploying the capabilities in this Profile.

X.4.2 Use Cases

X.4.2.1 Use Case #1 Restricting Document Access

X.4.2.1.1 Use Case Description

A patient believes that certain documents may have been either incorrectly classified or contain confidential information that accessible by non-physicians within a community.

The patient makes a request through the local Hospital Information System (HIS) patient portal for a list of documents and related metadata stored in the community. Upon review, the patient identifies several records that should be restricted from general view.

On behalf of the patient, the practitioner triggers an action in the local HIS to update the classification of a metadata attribute which restricts these documents from non-physician use.

X.4.2.1.2 Process Flow

The practitioner uses the local HIS to obtain the latest version of the metadata for each document, and then prepares the updated document metadata, reflecting the requested changes. The HIS uses the homeCommunityId metadata attribute to determine where the documents are stored. When the community is the same community where the documents are stored, this value may not be populated.

Having prepared the metadata, the HIS, as an Update Initiator, issues a metadata update request to Update Responder to process the request against the locally stored metadata.

The Update Responder returns a response to the Update Initiator indicating the success or failure 435 of the request.

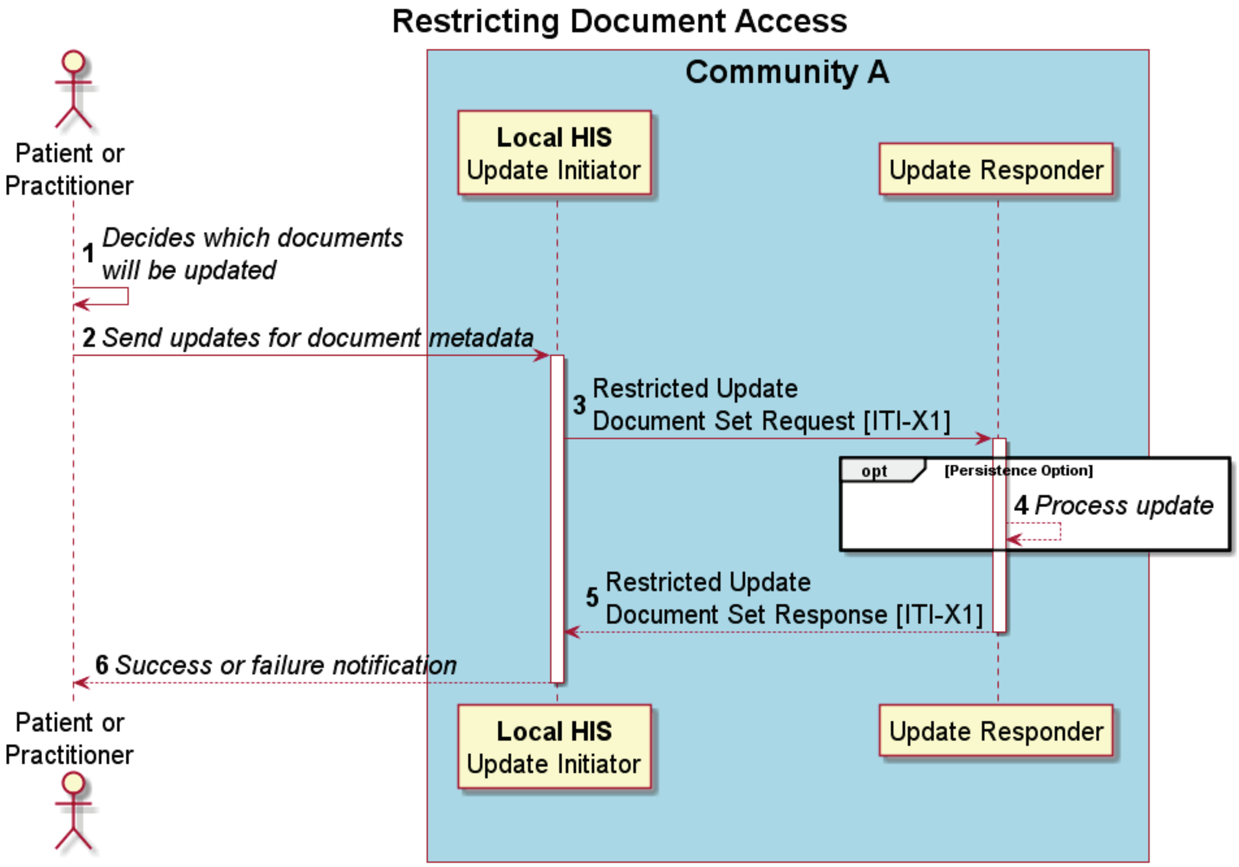

Restricting Document Access
Community A
Patient or Practitioner
Local HIS
Update Initiator
Update Responder
1 Decides which documents will be updated
2 Send updates for document metadata
3 Restricted Update Document Set Request [ITI-X1]
opt
[Persistence Option]
4 Process update
5 Restricted Update Document Set Response [ITI-X1]
6 Success or failure notification
Patient or Practitioner
Local HIS
Update Initiator
Update Responder

Figure X.4.2.1-1: Restricting Document Access Process Flow

The text in Figure X.4.2.1-2 below was used to generate the diagram in Figure X.4.2.1-1. Readers will generally find the diagram more informative. The text is included here to facilitate 440 editing.

```
@startuml
'Version 1.0 2018-05-08

title Restricting Document Access

autonumber
skinparam sequence {
TitleFontSize 25
ArrowFontSize 20
ParticipantFontSize 20
ActorFontSize 20
BoxFontSize 25
ParticipantPadding 20
}
skinparam BoxPadding 10

actor "Patient or\nPractitioner" as USER

box "Community A" #LightBlue
participant "**Local HIS**\nUpdate Initiator" as UIA
participant "Update Responder" as URA
end box


USER->USER: //Decides which documents//\n//will be updated//

USER->UIA: //Send updates for document metadata//
activate UIA
UIA->URA:Restricted Update\nDocument Set Request [ITI-X1]
activate URA
opt Persistence Option
URA-->URA: //Process update//
end opt
URA-->UIA:Restricted Update\nDocument Set Response [ITI-X1]
deactivate URA
UIA-->USER: //Success or failure notification//

deactivate UIA

@enduml
```

Figure X.4.2.1-2: Restricting Document Access Pseudocode

X.4.2.2 Use Case #2: Allowing Document Access

X.4.2.2.1 Use Case Description

Three communities have reached an agreement allowing for sharing documents among its facilities. Communities B and C have agreed to make available to practitioners, documents that are classified with both "normal" and "restricted" confidentiality. Community A has only agreed to only share records classified as "normal." As part of this agreement, a provision allows patients to modify the confidentiality classification in accordance with local law.

A patient has received treatment in two communities, A and B, for a disease requiring specialized treatment. As part of this treatment, these medical records were stored in these communities with confidentiality code classified as "restricted".

Later, the patient travels to Community C to receive treatment using a specialized procedure not available in the other communities. In order to treat the patient, the physician retrieves the medical records stored in Community B where access has already been granted. The physician reviews these records and has determined that they now need access to the restricted record stored in Community A to complete the diagnosis.

Utilizing the provision provided in the community's document sharing agreement, the patient changes the classification of their medical records in Community A from "restricted" to "normal". This allows their attending physician to access these records and begin treatment.

X.4.2.2.2 Process Flow

Starting with the latest version of the metadata for each document, the local HIS prepares the updated document metadata, reflecting the requested changes. The HIS determines where the documents are stored by examining the metadata for the homeCommunityId of the community.

With the metadata prepared, the HIS issues a metadata update request to Community A's Update Responder. The Update Responder uses the homeCommunityId to lookup the destination address for Community B and interacts with the grouped Update Initiator to forward the metadata to Community B for processing.

The Update Responder in the Community B processes the request. The Update Responder will return the response to the Community A's grouped Update Responder/Initiator which returns the response to the Update Initiator. The Update Initiator returns the response to the Local HIS which provides the user the success or failure of their request.

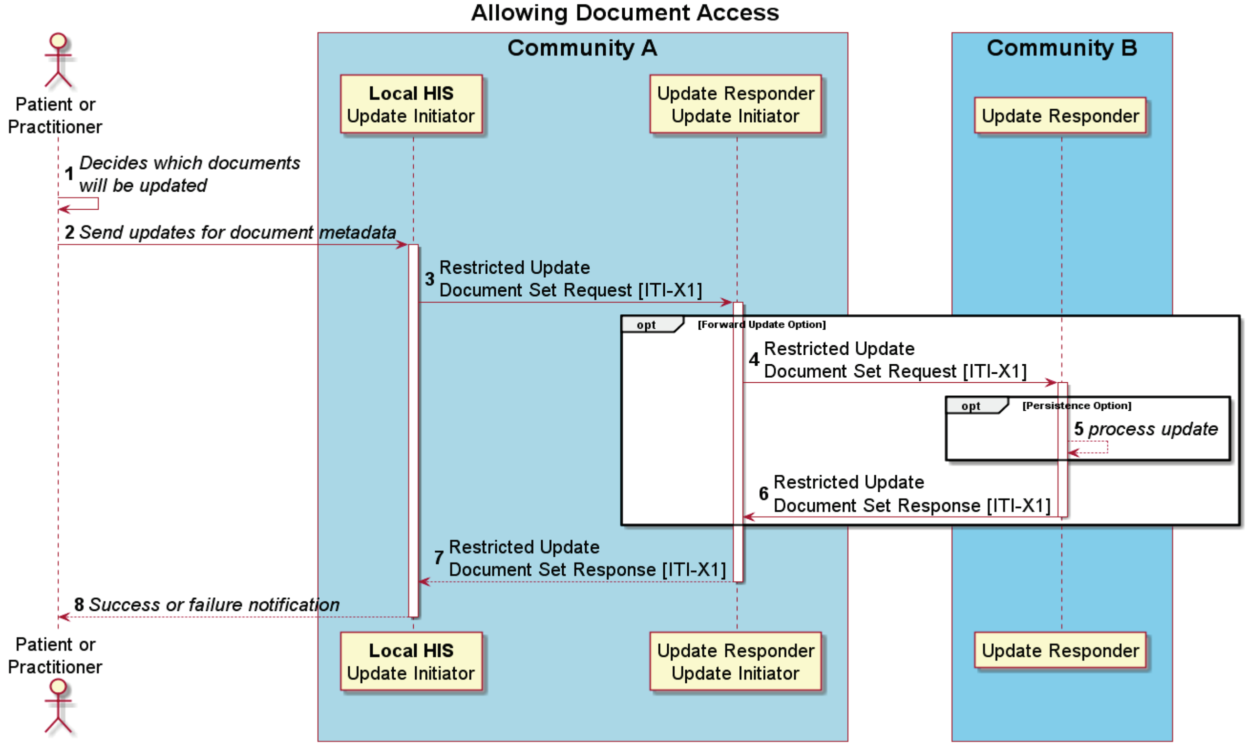

Allowing Document Access
Community A
Community B
Patient or Practitioner
Local HIS Update Initiator
Update Responder Update Initiator
Update Responder
1 Decides which documents will be updated
2 Send updates for document metadata
3 Restricted Update Document Set Request [ITI-X1]
opt [Forward Update Option]
4 Restricted Update Document Set Request [ITI-X1]
opt [Persistence Option]
5 process update
6 Restricted Update Document Set Response [ITI-X1]
7 Restricted Update Document Set Response [ITI-X1]
8 Success or failure notification
Patient or Practitioner
Local HIS Update Initiator
Update Responder Update Initiator
Update Responder

Figure X.4.2.2-1: Allowing Document Access Process Flow

The text in Figure X.4.2.2-2 below was used to generate the diagram in Figure X.4.2.2-1. Readers will generally find the diagram more informative. The text is included here to facilitate editing.

```
@startuml
'Version 1.0 2018-05-08

title Allowing Document Access

autonumber
skinparam sequence {
TitleFontSize 25
ArrowFontSize 20
ParticipantFontSize 20
ActorFontSize 20
BoxFontSize 25
ParticipantPadding 20
}
skinparam BoxPadding 10

actor "Patient or\nPractitioner" as USER

box "Community A" #LightBlue
participant "**Local HIS**\nUpdate Initiator" as UI
participant "Update Responder\nUpdate Initiator" as URA
end box

box "Community B" #SkyBlue
participant "Update Responder" as URB
end box

USER->USER: //Decides which documents//\n//will be updated//

USER->UI: //Send updates for document metadata//
activate UI
UI->URA:Restricted Update\nDocument Set Request [ITI-X1]
activate URA
opt Forward Update Option
URA->URB:Restricted Update\nDocument Set Request [ITI-X1]
activate URB
opt Persistence Option
URB-->URB: //process update//
end opt
URB->URA:Restricted Update\nDocument Set Response [ITI-X1]
deactivate URB
end opt
URA-->UI:Restricted Update\nDocument Set Response [ITI-X1]
deactivate URA
UI-->USER: //Success or failure notification//

deactivate UI

@enduml
```

Figure X.4.2.2-2: Allowing Document Access Pseudocode

X.4.2.3 Use Case #3: Event Reclassification and Distribution

X.4.2.3.1 Use Case Description

Continuing from the previous use case, the physician in Community C has diagnosed the patient's disease as a form of bacteria which is resistant to many known antibiotics commonly used for this type of disease.

After learning of several other such cases, the physician later determines that the bacteria causing the disease can now be classified and a treatment plan developed to help other physicians recognize and treat these pathogens.

All three communities have agreed to a use a common set of event classification codes to facilitate document sharing among their communities. Each community may introduce additional classifications and coding systems within their community which can be adopt by the other participating members.

After consulting with boards in all three communities, it has been determined that this new strain of bacteria should be separately classified and all metadata describing the medical records for these cases be updated to include the new classification.

The physician in Community C proceeds with updating the records in both Community A and B with the new classification. This allows Community A and B to cross-reference other similar case within their populations and promote earlier identification of these symptoms in other patients.

Finally, all applicable records associated with the patient's treatment for this event are updated with an additional accession number for purposes of describing this as a single event for billing and insurance-related uses.

X.4.2.3.2 Process Flow

Starting from the latest version of the metadata for each document, the local HIS prepares the updated document metadata, reflecting the requested changes. The HIS determines where the documents are stored by examining the metadata for the community's homeCommunityId. In this use case, the HIS will need to issue three separate request, one for each community.

With the metadata prepared, the HIS issues a metadata update request to the Community C's Update Responder. By checking the homeCommunityId attribute, the Update Responder determines it is responsible for persisting the update in the request. The Update Responder processes the request and returns the success or failure in the response.

The HIS next issues an update request to the Community C's Update Responder. Again, by examining the homeCommunityId, the Update Responder determines the request will be forwarded for processing to the Community A. The Update Responder uses the grouped Update Initiator to forward the request to Community A. Community A's Update Responder processes the request and returns the success or failure through the actor chain to the Local HIS.

This same process is now repeated for Community B.

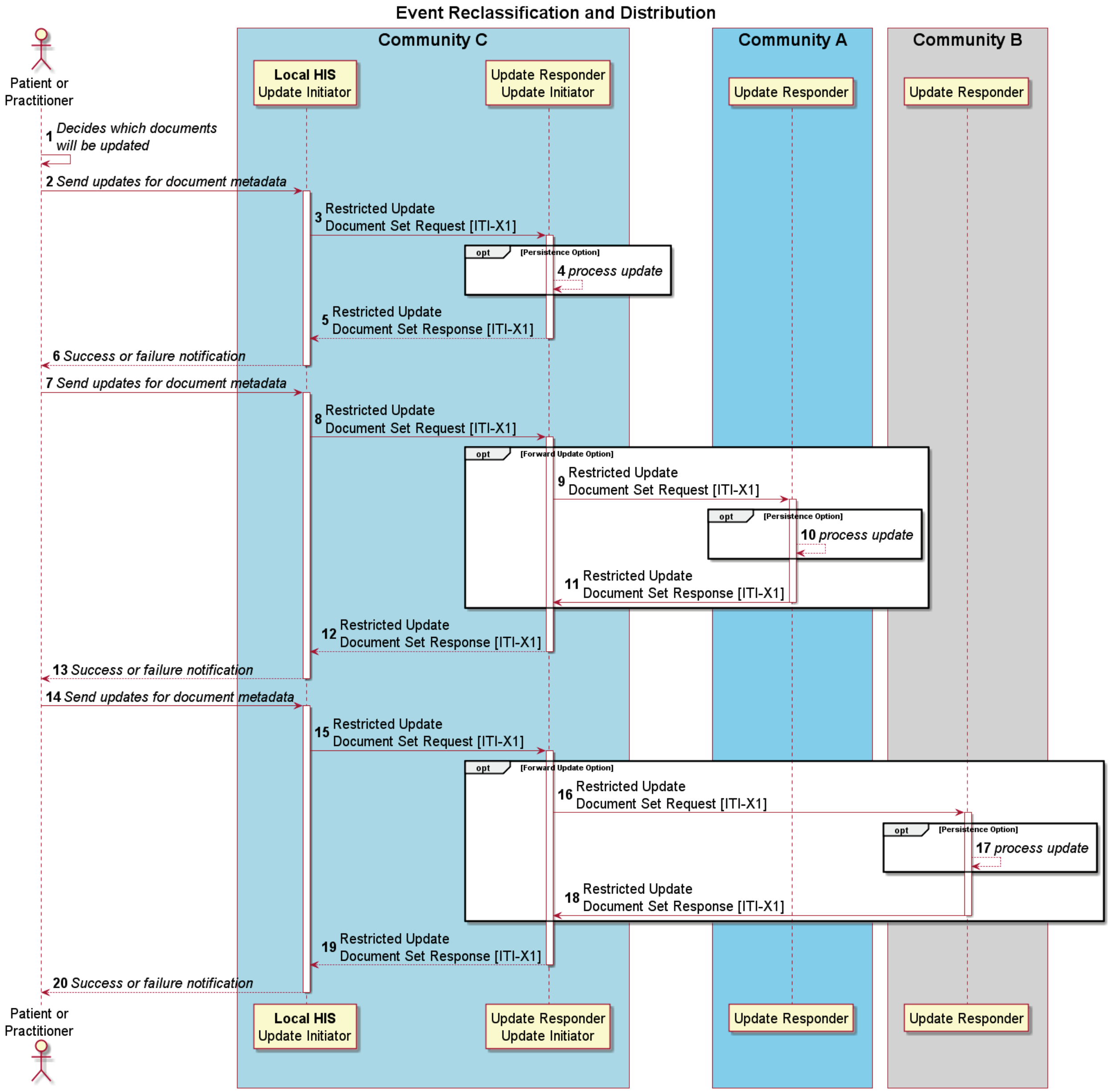

Event Reclassification and Distribution
Community C
Community A
Community B
Patient or Practitioner
Local HIS
Update Initiator
Update Responder
Update Initiator
Update Responder
Update Responder
1 Decides which documents will be updated
2 Send updates for document metadata
3 Restricted Update Document Set Request [ITI-X1]
opt [Persistence Option]
4 process update
5 Restricted Update Document Set Response [ITI-X1]
6 Success or failure notification
7 Send updates for document metadata
8 Restricted Update Document Set Request [ITI-X1]
opt [Forward Update Option]
9 Restricted Update Document Set Request [ITI-X1]
opt [Persistence Option]
10 process update
11 Restricted Update Document Set Response [ITI-X1]
12 Restricted Update Document Set Response [ITI-X1]
13 Success or failure notification
14 Send updates for document metadata
15 Restricted Update Document Set Request [ITI-X1]
opt [Forward Update Option]
16 Restricted Update Document Set Request [ITI-X1]
opt [Persistence Option]
17 process update
18 Restricted Update Document Set Response [ITI-X1]
19 Restricted Update Document Set Response [ITI-X1]
20 Success or failure notification
Patient or Practitioner
Local HIS
Update Initiator
Update Responder
Update Initiator
Update Responder
Update Responder

Figure X.4.2.3-1: Event Reclassification and Distribution Process Flow

The text in Figure X.4.2.3-2 below was used to generate the diagram in Figure X.4.2.3-1. Readers will generally find the diagram more informative. The text is included here to facilitate editing. 525

```
@startuml
'Version 1.0 2018-05-08

title Event Reclassification and Distribution

autonumber
skinparam sequence {
TitleFontSize 25
ArrowFontSize 20
ParticipantFontSize 20
ActorFontSize 20
BoxFontSize 25
ParticipantPadding 20
}
skinparam BoxPadding 10

actor "Patient or\nPractitioner" as USER

box "Community C" #LightBlue
participant "**Local HIS**\nUpdate Initiator" as UI
participant "Update Responder\nUpdate Initiator" as URA
end box

box "Community A" #SkyBlue
participant "Update Responder" as URB
end box

box "Community B" #LightGray
participant "Update Responder" as URC
end box

USER->USER: //Decides which documents//\n//will be updated//

'Update local community
USER->UI: //Send updates for document metadata//
activate UI
UI->URA:Restricted Update\nDocument Set Request [ITI-X1]
activate URA
opt Persistence Option
URA-->URA: //process update locally//
end opt
URA-->UI:Restricted Update\nDocument Set Response [ITI-X1]
deactivate URA
UI-->USER: //Success or failure notification//
deactivate UI
```

```
'Update Community A
USER->UI: //Send updates for document metadata//
activate UI
UI->URA:Restricted Update\nDocument Set Request [ITI-X1]
activate URA
opt Forward Update Option
URA->URB:Restricted Update\nDocument Set Request [ITI-X1]
activate URB
opt Persistence Option
URB-->URB: //process update locally//
end opt
URB->URA:Restricted Update\nDocument Set Response [ITI-X1]
deactivate URB
end opt
URA-->UI:Restricted Update\nDocument Set Response [ITI-X1]
deactivate URA
UI-->USER: //Success or failure notification//
deactivate UI

'Update Community B
USER->UI: //Send updates for document metadata//
activate UI
UI->URA:Restricted Update\nDocument Set Request [ITI-X1]
activate URA
opt Forward Update Option
URA->URC:Restricted Update\nDocument Set Request [ITI-X1]
activate URC
opt Persistence Option
URC-->URC: //process update locally//
end opt
URC->URA:Restricted Update\nDocument Set Response [ITI-X1]
deactivate URC
end opt
URA-->UI:Restricted Update\nDocument Set Response [ITI-X1]
deactivate URA
UI-->USER: //Success or failure notification//
deactivate UI


@enduml
```

Figure X.4.2.3-2: Event Reclassification and Distribution Pseudocode

X.5 Security Considerations

For general security considerations, see ITI TF-1: Appendix G - Security Considerations and ITI TF-2x: Appendix K - XDS Security Environment. Transaction specific security considerations are presented in the Security Considerations section of each transaction in Volume 2.

Updating patient care records is subject to local policies and government restrictions. As the sensitivity of the data may vary, the XDS Affinity Domain deployment will need to perform its own risk analysis and establish a mitigation strategy when enabling the transactions from this profile.

In most cases, appropriate precautions should be taken to restrict use of this profile's transactions to users with sufficient privileges.

Please see ITI TF-1: Appendix L - XDS Affinity Domain Definition Checklist for more information.

X.6 Cross Profile Considerations

X.6.1 Grouping with XDS Document Consumer and Registry

The Update Initiator may be grouped with an XDS Document Consumer to obtain the latest version of the DocumentEntry object metadata.

When the Update Initiator, is grouped with a Document Consumer, the Document Consumer will query the XDS Document Registry for the latest version of the metadata using the Registry Stored Query [ITI-18] transaction.

An Update Responder will likely be grouped with an XDS Document Registry that supports metadata versioning semantics.

The Update Responder will be capable of receiving the Restricted Update Document Set [ITI-X1] and provide any transformation, if needed, to allow the Document Registry to process the request.

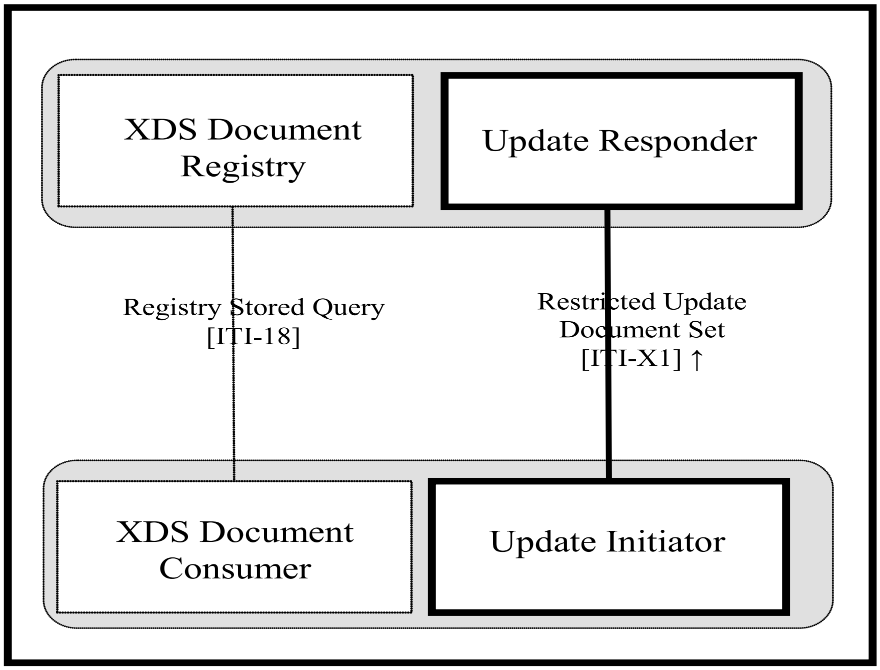


Figure X.6.1-1: RMU and XDS Actor Groupings

X.6.2 Grouping within an XCA Environment

The Update Initiator may be grouped with an XDS Document Consumer to obtain the latest versions of the DocumentEntry object metadata.

For remote communities, the XDS Document Consumer will use the Registry Stored Query [ITI-18] transaction. This query is sent the query to an XDS Initiating Gateway supporting the XDS Affinity Domain Option. The Initiating Gateway will use the Cross Gateway Query [ITI-38] transaction to contact Responding Gateways for the remote communities for servicing the request.

The metadata in the Cross-Gateway Query responses contain the patient identifier and coded values from Community B. These are used by the Community A's Update Initiator to form the request for the Restricted Update Document Set [ITI-X1] transaction.

The Update Initiator will send the update request to the Community A's Update Responder. The Update Responder will be grouped with another Update Initiator responsible for forwarding the request to the Community B's Update Responder.

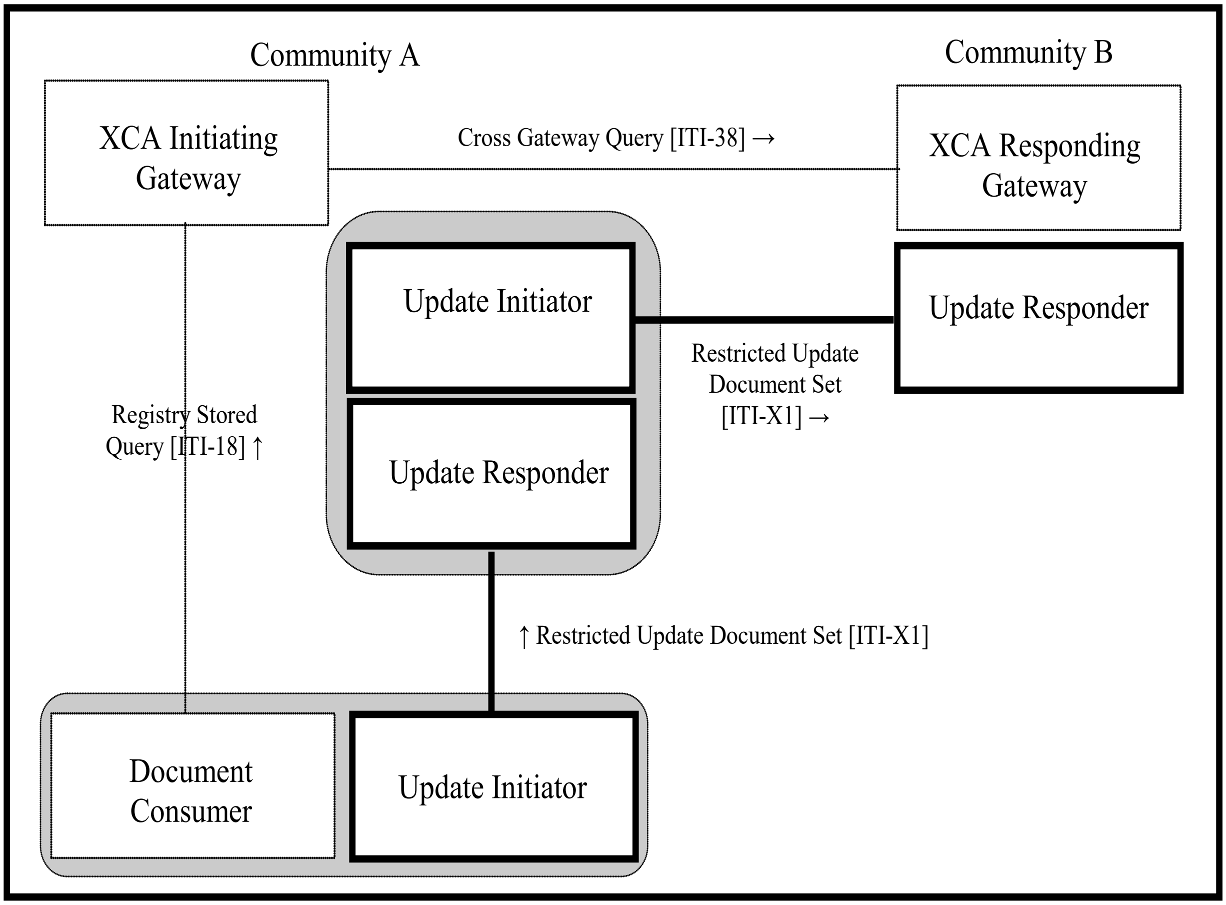


Figure X.6.2-1: RMU and XCA Actor Groupings

Volume 2 – Transactions

Add Section 3.X1

3.X1 Restricted Update Document Set

This section corresponds to the Restricted Update Document Set [ITI-X1] transaction of the IHE Technical Framework. Restricted Update Document Set [ITI-X1] is used by the Update Initiator and Update Responder.

3.X1.1 Scope

The Restricted Update Document Set [ITI-X1] transaction is used to update DocumentEntry metadata objects from the Update Initiator to the Update Responder.

3.X1.2 Actor Roles

The Roles in this transaction are defined in the following table and may be performed by the actors shown here:

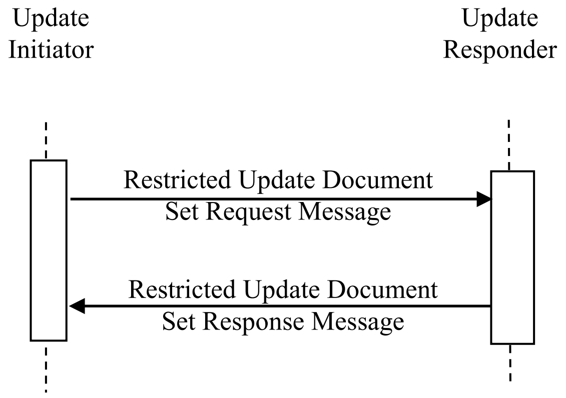
Table 3.X1.2-1: Actor Roles

Actor:	Update Initiator
Role:	Prepares and issues an update to DocumentEntry metadata objects.
Actor:	Update Responder
Role:	Accepts requests for updates to DocumentEntry metadata objects.

3.X1.3 Referenced Standard

ebRIM	OASIS/ebXML Registry Information Model v3.0 This model defines the types of metadata and content that can be stored in an ebXML Registry and forms the basis for the Document Sharing metadata model.
ebRS	OASIS/ebXML Registry Services Specifications v3.0 This defines the services and protocols for an ebXML Registry, used as the basis for the XDS Document Registry
ITI TF-2x: Appendix V	Web Services for IHE Transactions
ITI TF-3:4	Metadata Used in Document Sharing Profiles

3.X1.4 Interaction Diagram


Update Initiator
Update Responder
Restricted Update Document Set Request Message
Restricted Update Document Set Response Message

3.X1.4.1 Restricted Update Document Set Request Message

The Restricted Update Document Set Request message provides the ability to submit updated attributes for a DocumentEntry object in a community.

3.X1.4.1.1 Trigger Events

The Restricted Update Document Set Request message is triggered when an Update Initiator needs to transmit updated DocumentEntry metadata in a community.

3.X1.4.1.2 Message Semantics

The Restricted Update Document Set Request message shall use SOAP 1.2 and Simple SOAP (see ITI TF-2x: V.8.1). Implementers of this transaction shall comply with all requirements described in: ITI TF-2x: Appendix V: Web Services for IHE Transactions.

XML namespace prefixes used in text and in examples below are for informative purposes only and are documented in ITI TF-2x: Appendix V, Table 2.4-1.

The requirements for the request message are:

- <wsa:Action/> shall contain the value “urn:ihe:iti:2018:RestrictedUpdateDocumentSet”
- <soap12:Body/> shall contain one <lcm:SubmitObjectsRequest/> element representing the Submission Request (see ITI TF-3: 4.2.1.4 for details of expressing a Submission Request).
- <lcm:SubmitObjectsRequest/> shall contain one <rim:RegistryObjectList/>
- <rim:RegistryObjectList/> element shall include the following objects:
    - One <rim:RegistryPackage/> classified as a Submission Set object (See ITI TF-3: 4.2.1.2.

  - For each DocumentEntry metadata object:
    - One <rim:ExtrinsicObject/> that includes all required metadata attributes (see Section 3.X1.4.1.2.1 Content) and ebRIM attribute, @lid representing the logicalID attribute which is the entryUUID of the initial version of the object.
    - One matching <rim:Association/> that represents a SS-HM HasMember relationship between the Submission Set and DocumentEntry Object. This Association shall include two instances of <rim:Slot/> for the SubmissionSetStatus and PreviousVersion metadata annotations. The Association may optionally include a <rim:Slot/> for the AssociationPropagation metadata annotation (see Section 3.X1.4.1.2.2) with a value set to "yes".

A full XML Schema Document for the XDS types is available online on the IHE FTP site, see ITI TF-2x: Appendix W.

Below is an example of the SOAP Body for a Restricted Update Document Set Request message.

```
<soap12:Body>
    <lcm:SubmitObjectsRequest>
         <!-- Submission Request contents – See ITI TF-3: 4.2.1.4 -->
         <rim:RegistryObjectList>
              <rim:RegistryPackage id="SubmissionSet01" home="urn:oid:1.2.3.4.5.6.2333.23">
                 <!-- Submission Set goes here -->
              </rim:RegistryPackage>
              <rim:ExtrinsicObject id="Document01" lid="urn:uuid:0000-0-0000000"
                 home="urn:oid:1.2.3.4.5.6.2333.23">
                 <!—- DocumentEntry metadata goes here -->
              </rim:ExtrinsicObject>
              <rim:Association
                associationType=”urn:oasis:names:tc:ebxml-regrep:AssociationType:HasMember”
                sourceObject=”SubmissionSet01”
                targetObject=”Document01”>
                  <rim:Slot name=”SubmissionSetStatus”>
                     <rim:ValueList>
                        <rim:Value>Original</rim:Value>
                     </rim:ValueList>
                  </rim:Slot>
                  <rim:Slot name=”PreviousVersion”>
                     <rim:ValueList>
                        <rim:Value>1</rim:Value>
                     </rim:ValueList>
                  </rim:Slot>
                  <rim:Slot name=”AssociationPropagation”>
                     <rim:ValueList>
                        <rim:Value>yes</rim:Value>
                     </rim:ValueList>
                  </rim:Slot>
              </rim:Association>
       </rim:RegistryObjectList>
   </lcm:SubmitObjectsRequest>
</soap12:Body>
```

A full example of document metadata submission can be found available online on the IHE FTP site. See ITI TF-2x: Appendix W.

3.X1.4.1.2.1 Content

The Update Initiator shall assure all that metadata attributes restrictions are consistent with the Document Sharing metadata requirements as specified in ITI TF-3: Table 4.3.1-3: "Sending Actor Metadata Attribute Optionality" column "RMU-UI".

The Update Initiator shall use the latest version of the DocumentEntry object metadata as the basis for the update submission.

These metadata attributes shall not be modified by the Update Initiator:

- availabilityStatus
- entryUUID
- homeCommunityId
- logicalID
- version
- patientId
- sourcePatientId
- documentAvailability
- uniqueId
- repositoryUniqueId
- objectType

These metadata attributes describes the current state of DocumentEntry object, or physical document stored in a repository.

Affinity Domain policies may impose restrictions on updating metadata attributes not included in the preceding list.

3.X1.4.1.2.2 Metadata Annotations

Metadata Annotations are added to Submission Set SS-DE HasMember Association to specify the expected state or indicate the new state of a target object instance. The Update Responder supporting shall use these annotations when processing the request for storage.

Metadata Annotations are coded as <rim:Slot/> child elements within a <rim:Association/> object.

3.X1.4.1.2.2.1 PreviousVersion

The PreviousVersion annotation is constructed as a <rim:Slot/> with a name attribute equals to *PreviousVersion.* This slot contains a <rim:ValueList/> element containing a single <rim:Value/>, the version number of the existing object that will be updated by the submission.

If processed successfully, the submitted version will get the value of the PreviousVersion annotation plus one.

```
<rim:Association
  associationType=”urn:oasis:names:tc:ebxml-regrep:AssociationType:HasMember”
  sourceObject=”SubmissionSet01”
  targetObject=”Document01”>
    <rim:Slot name=”SubmissionSetStatus”>
       <rim:ValueList>
          <rim:Value>Original</rim:Value>
       </rim:ValueList>
    </rim:Slot>
    <rim:Slot name=”PreviousVersion”>
       <rim:ValueList>
          <rim:Value>1</rim:Value>
       </rim:ValueList>
    </rim:Slot>
</rim:Association>
```

Figure 3.X1.4.1.2.2.1-1: PreviousVersion Example

3.X1.4.1.2.2.2 AssociationPropagation

The AssociationPropagation annotation triggers the receiving actor to create Association objects for the updated metadata object based the Association objects link to the existing metadata object version. The default behavior for Association Propagation is "yes".

This example shows a SubmissionSet HasMember Association for a new version of a DocumentEntry where the AssociationPropagation annotation triggers association propagation by the receiving actor for the updated metadata object, Document01.

```
<rim:Association
  associationType=”urn:oasis:names:tc:ebxml-regrep:AssociationType:HasMember”
  sourceObject=”SubmissionSet01”
  targetObject=”Document01”>
    <rim:Slot name=”SubmissionSetStatus”>
       <rim:ValueList>
          <rim:Value>Original</rim:Value>
       </rim:ValueList>
    </rim:Slot>
    <rim:Slot name=”PreviousVersion”>
       <rim:ValueList>
          <rim:Value>1</rim:Value>
       </rim:ValueList>
    </rim:Slot>
    <rim:Slot name=”associationPropagation”>
       <rim:ValueList>
          <rim:Value>yes</rim:Value>
       </rim:ValueList>
    </rim:Slot>
</rim:Association>
```

Figure 3.X1.4.1.2.2.2-1: Example SubmissionSet – DocumentEntry HasMember Association as part of an update including AssociationPropagation annotation

3.X1.4.1.3 Expected Actions

The Update Responder shall process the Restricted Update Document Set Request message according the capabilities of the receiving system described in the following sections.

The Update Responder shall return the status and any error codes incurred during the processing of the request in its response message.

3.X1.4.1.3.1 Forward Update Option

The Update Responder shall validate the value of the homeCommunityId.

If the value is omitted, the Update Responder shall return the error code, *XDSMissingHomeCommunityId*. If the value is not known, the Update Responder shall return the error code, *XDSUnknownCommunity*,

If valid, the request shall be forwarded to the Update Responder servicing the community configured for the homeCommunityId contained in the request. The Update Responder may return the error code, *XDSUnavailableCommunity*, if it is unable to process the request.

3.X1.4.1.3.2 Persistence Option

The Update Responder shall verify that all metadata objects contained in the request have the same value for the homeCommunityId and it matches the value configured for that community. If this value does not match, the Update Responder shall return the error code, *XDSUnknownCommunity*. If the value is omitted, the Update Responder may return the error code, *XDSMissingHomeCommunityId*.

The request shall be validated according to the requirements in Section 3.X1.4.1.3.5 - Request Validation.

If the request is valid, the Update Responder shall store the updated metadata.

3.X1.4.1.3.3 XDS Persistence Option

If the request is valid, the Update Responder shall persist the data according to the requirements in Section 3.X1.4.1.3.3.1 - XDS Storage Requirements.

The Update Responder shall ensure that future Registry Stored Query [ITI-18] transaction responses including metadata versioning semantics as described in ITI TF-2a: 3.18.4.1.2.3.5.1 - Compatibility Issues.

3.X1.4.1.3.3.1 XDS Storage Requirements

The Update Responder updates the metadata of a DocumentEntry object by persisting a new version of the DocumentEntry object.

The Update Responder shall be capable of storing multiple versions of a logical DocumentEntry metadata object.

Prior to the update, the current version of the stored logical metadata object shall always have an availabilityStatus equal to `urn:oasis:names:tc:ebxml-regrep:StatusType:Approved`. All prior versions shall have their availabilityStatus attribute set to `urn:oasis:names:tc:ebxml-regrep:StatusType:Deprecated`.

The Update Responder shall be capable of processing the Association Propagation metadata annotation. See ITI TF-1: 10.4.14 – Metadata Update (currently in the XDS Metadata Update Trial Implementation Supplement) for more information.

The Update Responder shall identify the condition where the request contains DocumentEntry object updates that have the value set for Association Propagation equaled to "no". If detected, the Update Responder shall send the error code, *XDSMetadataUpdateAnnotationError*.

For each DocumentEntry object, the following actions are performed:

1. Store the new DocumentEntry
   a. Set the version attribute value to PreviousVersion plus one.
2. Change the availabilityStatus attribute for the existing DocumentEntry object to Deprecated.
3. If Association Propagation slot is missing or has a value equaled to ‘yes’, the Update Responder shall perform the following actions:
   a. Scan for existing approved HasMember Associations for the existing DocumentEntry object. For each HasMember Association found:
      i. If the Association links the existing DocumentEntry object to a Folder, create a new FD-DE HasMember Association linking the new DocumentEntry object with

the Folder. The existing FD-DE Association availabilityStatus will remain approved.

In addition, create an additional SS-HM HasMember Association linking the created FD-DE Association to the submitted SubmissionSet object.

ii. If the Association links the existing DocumentEntry object to an existing SubmissionSet with the SubmissionSetStatus attribute set to 'Reference', then deprecate this Association and create a new Association between the existing SubmissionSet and the submitted version of the DocumentEntry.

Note: In this case, the associated objects are not required to have the same PatientId attribute value.

b. Scan for non-deprecated relationship associations linked to the existing DocumentEntry (see ITI TF-3: 4.2.2.2). When found, these associations are replicated referencing the submitted DocumentEntry instead of the existing DocumentEntry.

c. Verify that a generated Association object has not been created more than once during processing (a.k.a.: overlapping updates). If so, the duplicate Association object shall be discarded. This condition is detected by checking the sourceObject, targetObject, and associationType attributes of the Association object.

d. Store the generated Associations

Any error that occurs during the processing of the Restricted Update Document Set Request message shall cause the entire transaction to fail. All errors detected during processing shall be returned in the response.

3.X1.4.1.3.4 XCA Persistence Option

If the request is valid, the Update Responder shall persist the data according to the requirements in Section 3.X1.4.1.3.4.1 - XCA Storage Requirements.

The Update Responder shall ensure that future Cross Gateway Query [ITI-38] transaction responses reflect the updated metadata consistent with the requirement in ITI TF-2b: 3.38.4.1.2.3 - Special handling of some stored queries.

3.X1.4.1.3.4.1 XCA Storage Requirements

If accepted, the Update Responder shall store the updated DocumentEntry objects contained in the request. If supported, the Updated Responder shall the store the Submission Set and SS-DE HasMember Association objects.

If the Update Responder supports persisting other metadata objects such as Folder or Submission Sets, the following actions shall be taken when updating an existing Document Entry object:

1. FD-DE HasMember and any Relationship Associations that link the existing DocumentEntry object shall be updated to now refer to the updated Document Entry object.

2. For existing HasMember Association linking the existing Document Entry object to its original Submission Set, this system is allowed to remove the Association, and the Submission Set when the Submission Set is no longer related to any other object included in the original submission.

If an error occurs during processing and storing of the updated metadata object, the Update Responder shall return the error code, *InvalidDocumentContent*.

Any error that occurs during the processing of the Restricted Update Document Set Request message shall cause the entire transaction to fail. All errors detected during processing shall be returned in the response.

3.X1.4.1.3.5 Request Validation

The following rules shall be used by the Update Responder to validate the submission.

Unless a more specific code is provided within the validation, the Update Responder shall return the error code, *XDSMetadataUpdateError*, for any error returned during processing:

1. Verify the request does not include initial versions of DocumentEntry objects (i.e.: entryUUID does not equal the logicalID or the logicalID is missing). If an initial version of a DocumentEntry is received, the error code, *XDSInvalidRequestException*, should be returned.
2. Verify that the metadata object submitted for updating is a DocumentEntry object. Otherwise, the error code, *XDSObjectTypeError*, should be returned.
3. The system contains an existing DocumentEntry (Stable or On-Demand) metadata object instance with status of Approved. If the DocumentEntry cannot be located, the error code, *UnresolvedReferenceException*, shall be returned.
4. The SubmissionSet to DocumentEntry HasMember Association has a Slot with name PreviousVersion. The value of the PreviousVersion Slot shall match the version number of the existing DocumentEntry. Otherwise, the error code, *XDSMetadataVersionError*, shall be returned.
5. Verify the submitted and existing DocumentEntry have the same values for both the logicalID and uniqueId attribute. If these values are not identical, the error code, *XDSMetadataIdentifierError*, should be returned.
6. Verify the submitted and existing DocumentEntry have the same values for the patientId attributes. If these values are not identical, the error code, *XDSPatientIDReconciliationError*, should be returned.
7. Check the submitted DocumentEntry metadata and determine if it contains any change to an unmodifiable attribute, as described in Section 3.X1.4.1.2.1. If so, the error code, *UnmodifiableMetadataError*, should be returned.

8. Check the submitted DocumentEntry metadata and determine if it contains any modifications to an attribute that violates local policy. If so, the error code, *LocalPolicyRestrictionError*, should be returned.
9. All metadata objects must conform to the rules for a content and format defined in ITI TF-3: 4.2 and 4.3. These are the same rules defined for the Register Document Set-b [ITI-42] transaction. If an error is encountered during request validation, the appropriate error code, as defined in ITI TF-2b: 3.42 for the Register Document Set-b [ITI-42] transaction, shall be returned.

3.X1.4.2 Restricted Update Document Set Response Message

The Restricted Update Document Set Response is sent when the processing of the Restricted Update Document Set Request is complete.

3.X1.4.2.1 Trigger Events

The Restricted Update Document Set Request message processing has been completed.

3.X1.4.2.2 Message Semantics

The Restricted Update Document Set Response message shall use SOAP 1.2 and Simple SOAP (See ITI TF-2x: V.8.1). Implementers of this transaction shall comply with all requirements described in: ITI TF-2x: Appendix V: Web Services for IHE Transactions.

XML namespace prefixes used in text and in examples below are for informational purposes only and are documented in ITI TF-2x: Appendix V, Table V.2.4-1.

The requirements for the response message are as follows:

- <wsa:Action/> SOAP element shall contain the value urn:ihe:iti:2018:RestrictedUpdateDocumentSetResponse
- <soap12:Body/> SOAP element shall contain one <rs:RegistryResponse/> element

The rs:RegistryResponse/@status attributes provides the overall status of the request. It shall contain one of two values:

- If all metadata in the request was updated successfully, the Document Registry shall set the status equal to `urn:oasis:names:tc:ebxml-regrep:ResponseStatusType:Success`.
- If the metadata could not be updated successfully, then the Document Registry shall set the status equal to `urn:oasis:names:tc:ebxml-regrep:ResponseStatusType:Failure`.

If an error occurs when removing a metadata object, then a rs:RegistryResponse/rs:RegistryErrorList/rs:RegistryError element shall be returned in the response with:

- @severity is `urn:oasis:names:tc:ebxml-regrep:ErrorSeverityType:Error`.

- @errorCode contains an error code from ITI TF-3: Table 4.2.4.1-2.
- @codeContext contains the error message and the entryUUID for the object that caused the error.

See ITI TF-3: 4.2.4.1 for examples of response messages.

3.X1.4.2.3 Expected Actions

When the Update Initiator receives a success response, the metadata objects were successfully updated and the transaction is complete. The Update Initiator can continue processing normally.

If an error response was received, the Update Initiator may need to perform additional steps to determine the cause and correct the error. These steps are not specified by this transaction.

3.X1.5 Security Considerations

Additional security considerations that may apply are discussed in ITI TF-1: X.5 - RMU Security Considerations.

3.X1.5.1.4.1 Audit Record Considerations

The Restricted Update Document Set [ITI-X1] transaction is PHI-Patient Record event, as defined in ITI TF-2a: Table 3.20.4.1.1.1-1 with the following exceptions:

3.X1.5.1.4.1.1 Update Initiator Audit Message

	Field Name	Opt	Value Constraints
Event **AuditMessage/ EventIdentification**	EventID	M	EV(110110, DCM, “Patient Record”)
	EventActionCode	M	“U” (Update)
	EventDateTime	*M*	*not specialized*
	EventOutcomeIndicator	*M*	*not specialized*
	EventTypeCode	M	EV(“ITI-X1”, “IHE Transactions”, “Restricted Update Document Set”)
Source (Update Initiator) (1)			
Human Requestor (0..n)			
Destination (Update Responder) (1)			
Audit Source (Update Initiator) (1)			
Patient (1)			
SubmissionSet (1)			

Where:

Source **AuditMessage/ ActiveParticipant**	UserID	*U*	*not specialized*
	AlternativeUserID	M	Process ID as used within the local operating system in the local system logs.
	UserName	*U*	*not specialized*
	UserIsRequestor	*U*	*not specialized*
	RoleIDCode	M	EV(110153, DCM, “Source”)
	NetworkAccessPointTypeCode	M	“1” for machine (DNS) name, “2” for IP address

	NetworkAccessPointID	M	The machine name or IP address

Human Requestor (if known) **AuditMessage/ ActiveParticipant**	UserID	M	Identity of the human that initiated the transaction.
	AlternativeUserID	*U*	*not specialized*
	UserName	*U*	*not specialized*
	UserIsRequestor	*U*	*not specialized*
	RoleIDCode	U	Access Control role(s) the user holds that allows this transaction.
	NetworkAccessPointTypeCode	*U*	*not specialized*
	NetworkAccessPointID	*U*	*not specialized*

Destination **AuditMessage/ ActiveParticipant**	UserID	M	SOAP endpoint URI.
	AlternativeUserID	*U*	*not specialized*
	UserName	*U*	*not specialized*
	UserIsRequestor	M	“false”
	RoleIDCode	M	EV(110152, DCM, “Destination”)
	NetworkAccessPointTypeCode	M	“1” for machine (DNS) name, “2” for IP address
	NetworkAccessPointID	M	The machine name or IP address

Audit Source **AuditMessage/ AuditSourceIdentification**	*AuditSourceID*	*U*	*not specialized*
	AuditEnterpriseSiteID	*U*	*not specialized*
	AuditSourceTypeCode	*U*	*not specialized*

Patient **(AuditMessage/ ParticipantObjectIdentification)**	ParticipantObjectTypeCode	M	“1” (Person)
	ParticipantObjectTypeCodeRole	M	“1” (Patient)
	ParticipantObjectDataLifeCycle	*U*	*not specialized*
	ParticipantObjectIDTypeCode	*M*	*not specialized*
	ParticipantObjectSensitivity	*U*	*not specialized*
	ParticipantObjectID	M	The patient ID in HL7®[1] CX format.
	ParticipantObjectName	*U*	*not specialized*
	ParticipantObjectQuery	*U*	*not specialized*
	ParticipantObjectDetail	*U*	*not specialized*

[1] HL7 is the registered trademark of Health Level Seven International.

Submission Set **(AuditMessage/ ParticipantObjectIdentification)**	ParticipantObjectTypeCode	M	"2" (System)
	ParticipantObjectTypeCodeRole	M	"20" (job)
	ParticipantObjectDataLifeCycle	*U*	*not specialized*
	ParticipantObjectIDTypeCode	*M*	EV("urn:uuid:a54d6aa5-d40d-43f9-88c5-b4633d873bdd", "IHE XDS Metadata", "submission set classificationNode")
	ParticipantObjectSensitivity	*U*	*not specialized*
	ParticipantObjectID	M	SubmissionSet uniqueID
	ParticipantObjectName	*U*	*not specialized*
	ParticipantObjectQuery	*U*	*not specialized*
	ParticipantObjectDetail	C	If known, set the "urn:ihe:iti:xca:2010:homeCommunityId" as the value of the attribute type and the value of the homeCommunityId as the value of the attribute value.

3.X1.5.1.4.1.2 Update Responder Audit Message

	Field Name	Opt	Value Constraints
Event **AuditMessage/ EventIdentification**	EventID	M	EV(110110, DCM, "Patient Record")
	EventActionCode	M	"U" (Update)
	EventDateTime	*M*	*not specialized*
	EventOutcomeIndicator	*M*	*not specialized*
	EventTypeCode	M	EV("ITI-X1", "IHE Transactions", "Restricted Update Document Set")
Source (Update Initiator) (1)			
Destination (Update Responder) (1)			
Audit Source (Update Responder) (1)			
Patient (1)			
SubmissionSet (1)			

Where:

Source **AuditMessage/ ActiveParticipant**	UserID	*U*	*not specialized*
	AlternativeUserID	*U*	*not specialized*
	UserName	*U*	*not specialized*
	UserIsRequestor	*U*	*not specialized*
	RoleIDCode	M	EV(110153, DCM, "Source")
	NetworkAccessPointTypeCode	M	"1" for machine (DNS) name, "2" for IP address
	NetworkAccessPointID	M	The machine name or IP address

Destination **AuditMessage/ ActiveParticipant**	UserID	M	SOAP endpoint URI
	AlternativeUserID	M	Process ID as used within the local operating system in the local system logs.
	UserName	*U*	*not specialized*
	UserIsRequestor	M	"false"
	RoleIDCode	M	EV(110152, DCM, "Destination")
	NetworkAccessPointTypeCode	M	"1" for machine (DNS) name, "2" for IP address
	NetworkAccessPointID	M	The machine name or IP address

Audit Source **AuditMessage/ AuditSourceIdentification**	*AuditSourceID*	*U*	*not specialized*
	AuditEnterpriseSiteID	*U*	*not specialized*
	AuditSourceTypeCode	*U*	*not specialized*

Patient **(AuditMessage/ ParticipantObjectIdentification)**	ParticipantObjectTypeCode	M	"1" (Person)
	ParticipantObjectTypeCodeRole	M	"1" (Patient)
	ParticipantObjectDataLifeCycle	*U*	*not specialized*
	ParticipantObjectIDTypeCode	*M*	*not specialized*
	ParticipantObjectSensitivity	*U*	*not specialized*
	ParticipantObjectID	M	The patient ID in HL7 CX format.
	ParticipantObjectName	*U*	*not specialized*
	ParticipantObjectQuery	*U*	*not specialized*
	ParticipantObjectDetail	*U*	*not specialized*

Submission Set **(AuditMessage/ ParticipantObjectIdentification)**	ParticipantObjectTypeCode	M	"2" (System)
	ParticipantObjectTypeCodeRole	M	"20" (job)
	ParticipantObjectDataLifeCycle	*U*	*not specialized*
	ParticipantObjectIDTypeCode	*M*	EV("urn:uuid:a54d6aa5-d40d-43f9-88c5-b4633d873bdd", "IHE XDS Metadata", "submission set classificationNode")
	ParticipantObjectSensitivity	*U*	*not specialized*
	ParticipantObjectID	M	SubmissionSet uniqueID
	ParticipantObjectName	*U*	*not specialized*
	ParticipantObjectQuery	*U*	*not specialized*
	ParticipantObjectDetail	C	If known, set the "urn:ihe:iti:xca:2010:homeCommunityId" as the value of the attribute type and the value of the homeCommunityId as the value of the attribute value.

Volume 3 – Cross-Transaction and Content Specifications

4.1 Abstract Metadata Model

Editor: Update Table 4.1.3.2-1: DocumentEntry Metadata Attribute Definition as shown below

Reviewers: *These are the same changes as shown in the XDS Metadata Update Supplement-Rev. 1.9.*

Table 4.1.3.2-1: DocumentEntry Metadata Attribute Definition

DocumentEntry Metadata Attribute	Description	Patient identity	Provenance	Security &Privacy	Descriptive	Object Lifecycle	Exchange
documentAvailability	**The status of the Document in the Document Repository**					X	X
logicalID	**A globally unique identifier used to identify the logical entry.**		X			X	X
version	**Version number of a DocumentEntry.**		X			X	X

Editor: Add new Section 4.1.5.

Reviewers: *This text originated in the XDS Metadata Update Supplement-Rev. 1.9. It has been modified to remove content not used in this Profile and other minor editorial changes. These differences will be resolved as part of the Public Comment process prior to Trial Implementation.*

4.1.5 Metadata Object Versioning Semantics

One part of metadata updating is the management of metadata object versioning as specified in ebRIM 3.0. ebRIM 3.0 version control introduces the following concepts to support versioning:

Metadata Object Instance – a single metadata object representing a single version of an object

Logical Metadata Object – the collection of metadata object instances that are the versions of a single object. Each metadata object instance is a different version of the logical metadata object.

Before the introduction of metadata update, a logical metadata object was always represented by a single instance so differentiating logical and instance was not important.

A logical DocumentEntry metadata object represents a single document in a Document Repository. The logical Document Entry encompasses all the versions (Document Entry instances) that have historically represented the repository document.

An association, through its sourceObject and targetObject attributes, references metadata object instances (particular versions of the objects).

Metadata versions are identified/managed using two metadata attributes: logicalID and version:

logicalID

- Each object instance is assigned a logicalID (@lid) along with its entryUUID (@id)
- The first version of an object has entryUUID equal to logicalID
- Each metadata object instance has a unique value for the entryUUID attribute
- Each logical object has a unique value for the logicalID attribute
- Each logical object is represented by one or more object instances.
- All objects with the same logicalID shall be of the same object type. The logicalID shall identify a group of DocumentEntry objects with the same objectType attribute.
- The rules for interpreting logicalID are:
    1. All object instances with the same logicalID are versions of the same logical object
    2. Each object instance has a unique entryUUID
    3. The first version of a logical object has logicalID equals the entryUUID.
    4. The second and later versions of a logical object have logicalID shall not entryUUID
    5. If an object instance is submitted with no logicalID attribute, the value for logicalID defaults to the value of the entryUUID for that object instance becoming the first version of the logical metadata object.

version

- Instances of a metadata object are assigned a version through the version attribute as described in ITI TF-3: Tables 4.2.3.2-1 and 4.2.3.4-1 (currently in the XDS Metadata Update Trial Implementation Supplement).
- The highest numbered version of an object instance shall have availabilityStatus of Approved. All older versions shall have availabilityStatus of Deprecated.
- When updates are submitted, they reference the version being updated
- Changes shall only be accepted for the most recent version.

When an updated version of a DocumentEntry object is submitted, certain HasMember and Relationship Associations will be replicated from the existing to the submitted new version of a DocumentEntry object unless overridden by the requester. The receiving actor is responsible for performing these actions. This responsibility is called Association Propagation.

The uniqueId, logicalID, and objectType attributes of a logical DocumentEntry object shall not be altered through versioning. They are required to be consistent across all object instances within a logical object. Document Replacement shall be used when any of these attributes are required to be changed.

4.2.3 Metadata Attributes

…

4.2.3.2 Document Metadata Attribute Definition

Editor: Update ITI TF-3: Table 4.2.3.2-1 Document Metadata Attribute Definition as shown below

***Reviewers**: These are the changes shown in the XDS Metadata Update Supplement-Rev. 1.9.*

Table 4.2.3.2-1: DocumentEntry Metadata Attribute Definition (previously Table 4.1-5)

DocumentEntry Metadata Attribute	Description	Data Type	Coding (Section 4.2.3.1.8)	Detail (See Section)
…				
creationTime	The time the author created the document. Shall have a single value.	DTM	ebRIM Slot	4.2.3.2.6
documentAvailability	**The status of the Document in the Document Repository**	**Predefined URN**	**ebRIM Slot**	**4.2.3.2.30**
entryUUID	A globally unique identifier used to identify the entry.	UUID	XML attribute	4.2.3.2.7
limitedMetadata	Indicates whether the Document Entry was created using the less rigorous requirements of metadata as defined for the Metadata-Limited Document Source.	4.2.3.2.29	ebRIM Classification	4.2.3.2.29
logicalID	**A globally unique identifier used to identify the logical entry.**	**UUID**	**XML attribute**	**4.2.3.2.31**
mimeType	MIME type of the document.	MIME type	XML attribute	4.2.3.2.15
…				
URI	The URI for the document.	URI	ebRIM Slot	4.2.3.2.27
version	**Version number of a DocumentEntry.**	**Integer**	**ebRIM VersionInfo**	**4.2.3.2.32**

Editor: Add the following new sub-sections to Section 4.2.3.2 as shown

***Reviewers:** These are the same changes in the XDS Metadata Update Supplement-Rev. 1.9 with one exception highlighted in yellow. This will be resolved as part of Public Comment process prior to Trial Implementation.*

4.2.3.2.30 DocumentEntry.documentAvailability

Description:

The status of the Document in the Document Repository. Online indicates the Document in the Document Repository is available to be retrieved. Offline indicates the Document in the Document Repository is not available to be retrieved.

Coding:

This slot, if present, shall have a single value.

Takes one of two values: urn:ihe:iti:2010:DocumentAvailability:Online or urn:ihe:iti:2010:DocumentAvailability:Offline. If the attribute is not present in metadata its value defaults to Online.

The following example indicates the document associated with the DocumentEntry is not available to be retrieved.

```
<rim:Slot name=”documentAvailability”>
  <rim:ValueList>
    <rim:Value>urn:ihe:iti:2010:DocumentAvailability:Offline</rim:Value>
  </rim:ValueList>
</rim:Slot>
```

4.2.3.2.31 DocumentEntry.logicalID

Description:

A globally unique identifier used to identify the logical entry. All versions of a DocumentEntry carry the same logicalID but different and unique entryUUIDs. If not present in a submission, logicalID defaults to the value of the entryUUID attribute. The first version of a DocumentEntry has logicalID equal to entryUUID. Other versions have logicalID not equal to entryUUID (also see Section 4.1.5).

Coding:

Max length is unbounded. The format of the logicalID value is UUID. LogicalID shall never be submitted in symbolic form.

The value of the logicalID is coded in the lid XML attribute on the ExtrinsicObject representing the DocumentEntry. In the example below, the entryUUID is urn:uuid:a6e06ca8-0c75-4064-9e5c-88b9045a96f6

The initial version of a DocumentEntry could have no LogicalID present:

```
<rim:ExtrinsicObject
    id=”urn:uuid:3cce0135-cedb-4a26-ba00-8698ee8dde04”>
…
</rim:ExtrinsicObject>
```

Or could have logicalID equal to entryUUID as shown:

```
<rim:ExtrinsicObject
    id=”urn:uuid:3cce0135-cedb-4a26-ba00-8698ee8dde04”
    lid=”urn:uuid:3cce0135-cedb-4a26-ba00-8698ee8dde04”>…
</rim:ExtrinsicObject>
```

One of these two forms shall be submitted in the Registry Document Set transaction.

The following form, with entryUUID (id) different from logicalID (lid), shall only be submitted in the Update Document Set [ITI-57] transaction.

```
<rim:ExtrinsicObject
    id=”urn:uuid:3cce0135-cedb-4a26-ba00-8698ee8dde04”
    lid=”urn:uuid:e0985823-dc50-45a5-a6c8-a11a829893bd”>
…
</rim:ExtrinsicObject>
```

4.2.3.2.32 DocumentEntry.version

Description:

Version number of a DocumentEntry. This value is assigned by the Document Registry and shall be ignored if present in a submission. The first version of a DocumentEntry shall have a value of 1. Subsequent versions get values of 2, 3, etc. This attribute shall be returned in query responses.

Coding:

Max length is unbounded.

The value of the version attribute is coded in XML as the "versionName" attribute of the VersionInfo element.

```
<VersionInfo versionName=”2”/>
```

4.2.4 Success and Error Reporting

…

4.2.4.1 RegistryError Element

Editor: Update ITI TF-3: Table 4.2.4.1-2 Error Codes as shown. Note the update to Note 1and 2 below the table.

***Reviewers**: The changes shown are based on Rev. 14 Final Text. Three of the error codes shown are from XDS Metadata Update Supplement-Rev. 1.9.*

Table 4.2.4.1-2: Error Codes (previously Table 4.1-11)

Error Code	Discussion	Transaction (See Note 1)
…		
XDSMissingHomeCommunityId	A value for the homeCommunityId is required and has not been specified	SQ, XGQ, RS, XGR**, RU**
XDSUnknownCommunity	A value for the homeCommunityId is not recognized	SQ, XGQ, RS, XGR**, RU**
XDSUnavailableCommunity	A community which would have been contacted was not available **or capable of processing the request.** See Note 2.	SQ, RS**, RU**
InvalidDocumentContent	The recipient has rejected this submission because it detected that one of the documents does not match the metadata (e.g., formatCode) or has failed other requirements for the document content. When the RegistryError element contains this error code, the @codeContext shall contain the DocumentUniqueID of the document in error. If multiple documents are in error, there shall be a separate RegistryError element for each document in error.	P**, RU**
UnresolvedReferenceException	The recipient cannot resolve an entryUUID reference in the transaction.	P, R, RU
…		
XDSMetadataUpdateError	**General metadata update error. Use only when more specific error code is not available or appropriate.**	**RU**
XDSPatientIDReconciliationError	**Update encountered an error where patient identifiers did not match**	**RU**

Error Code	Discussion	Transaction (See Note 1)
XDSMetadataVersionError	**The version number included in the update request did not match the existing object. One cause of this is multiple simultaneous update attempts.**	**RU**
…		
XDSObjectTypeError	**The receiving actor cannot store the request as the objectType is either not supported or does not match the value of the existing object.**	**RU**
XDSMetadataIdentifierError	**The receiving actor cannot store the request because the identifier is in conflict with an existing known value.**	**RU**
XDSInvalidRequestException	**The receiving actor detected that an initial version of a metadata object instance was received in an update transaction.**	**RU**
UnmodifiableMetadataError	**An update for a metadata object includes changes to values for attributes which are prohibited for the transaction.**	**RU**
LocalPolicyRestrictionError	**The receiving actor has detected that the request is not permitted because of local policy restrictions or violation of a previously reached agreement.**	**RU**
XDSMetadataUpdateAnnotationError	**The receiving actor has detected an error with a SS-HM annotation.**	**RU**

Note 1:

P = Provide and Register-b

R = Register-b

SQ = Stored Query

RS = Retrieve Document Set

XGQ = Cross Gateway Query

XGR = Cross Gateway Retrieve

RU = Restricted Update Document Set

Note 2:

Two examples of the use of error code XDSUnavailableCommunity are:

A Cross Gateway Query or Cross Gateway Retrieve fails because the community identified by a homeCommunityId could not be contacted.

A Cross Gateway Query based on Patient ID could not contact some known communities to relay the query.

The error would be generated by the Initiating Gateway and returned in the Registry Stored Query or Retrieve Document Set. This would only apply when XDS Affinity Domain Option was used.

An Update Initiator fails because the community's Update Responder, identified by a homeCommunityId, could not be contacted or was not capable of processing the request.

4.3 Additional Document Sharing Requirements

…

Editor:

Update Table 4.3.1-1 as shown: Adding actor RMU Update Initiator.

Update Table 4.3.1-2 as shown: Adding code I – "Immutable".

Update Table 4.3.1-3 as shown: add the new attributes in alphabetical order.

Update Table 4.3.1-3 as shown: add new column for RMU UI.

Update Table 4.3.2.1-3 as shown: add the new attributes in alphabetical order.

Reviewers:

The changes shown are based on Rev. 14 Final Text. Changes for the new attributes in Table 4.3.1-3 are unchanged from XDS Metadata Update Supplement-Rev. 1.9.

4.3.1 Submission Metadata Attribute Optionality

This section lists which metadata attributes an actor shall provide when initiating a Submission Type Transaction.

The Actor/Transaction pairs addressed by this section are as follows:

Table 4.3.1-1: Sending Actor/Transaction Pairs

Actor	Transaction	Shortname
XDS Document Source	ITI-41 Provide and Register Document Set-b	XDS DS
XDS Document Repository	ITI-42 Register Document Set-b	XDS DR
XDM Portable Media Creator	ITI-32 Distribute Document Set on Media	XDM MC
XDR Document Source	ITI-41 Provide and Register Document Set-b	XDR DS
XDR Metadata-Limited Document Source	ITI-41 Provide and Register Document Set-b	XDR MS
XDS On-Demand Document Source	ITI-61 Register On-Demand Document Entry	XDS OD
RMU Update Initiator	**ITI-X1 Restricted Update Document Set**	**RMU UI**

For each actor/transaction pair across the top of Table 4.3.1-3, and each metadata attribute row, the cell indicates the requirement for that actor when creating a submission request using the paired transaction. The requirements are expressed through these codes:

Table 4.3.1-2: Sending Actor Metadata Attribute Optionality Code Definitions

Code	Meaning
R	Required – A value for the attribute shall be supplied by the sending actor when sending the submission
R2	Required if Known – A value for the attribute shall be supplied by the sending actor when sending the submission unless the actor does not have any value for the attribute
O	Optional – The sending actor may or may not supply a value for this attribute
X	Prohibited – when sending a submission, a value for the attribute shall not be supplied by the sending actor.
I	**Immutable – when sending a submission, a value for the attribute may not be changed from its original value**

Table 4.3.1-3: Sending Actor Metadata Attribute Optionality

Metadata Element	Metadata Attribute	XDS DS	XDS DR	XDM MC	XDR DS	XDR MS	XDS OD	**RMU UI**
DocumentEntry	author	R2	R2	R2	R2	R2	R2	**R2**
DocumentEntry	availabilityStatus	O	O	O	O	O	O	**I**
DocumentEntry	classCode	R	R	R2	R	R2	R	**R**
DocumentEntry	comments	O	O	O	O	O	O	**O**
DocumentEntry	confidentialityCode	R	R	R2	R	R2	R	**R**
DocumentEntry	creationTime	R	R	R2	R	R2	X	**R**
DocumentEntry	**documentAvailability**	**O**	**O**	**O**	**O**	**O**	**O**	**I**
DocumentEntry	entryUUID	R	R	R	R	R	R	**R**
DocumentEntry	eventCodeList	O	O	O	O	O	O	**O**
DocumentEntry	formatCode	R	R	R2	R	R2	R	**R**
DocumentEntry	hash	O	R	R	O	O	X	**I**
DocumentEntry	healthcareFacility TypeCode	R	R	R2	R	R2	R	**R**
DocumentEntry	homeCommunityId	O	O	O	O	O	O	**I**
DocumentEntry	languageCode	R	R	R2	R	R2	R	**R**
DocumentEntry	legalAuthenticator	O	O	O	O	O	O	**O**
DocumentEntry	limitedMetadata	X	X	O	X	R	X	**X**
DocumentEntry	**logicalID**	**O**	**O**	**O**	**O**	**O**	**O**	**I**
DocumentEntry	mimeType	R	R	R	R	R	R	**R**
DocumentEntry	objectType	R	R	R	R	R	R	**I**
DocumentEntry	patientId	R	R	R2	R	R2	R	**I**
DocumentEntry	practiceSettingCode	R	R	R2	R	R2	R	**R**
DocumentEntry	referenceIdList	O	O	O	O	O	O	**O**
DocumentEntry	repositoryUniqueId	O	R	O	O	O	R	**I**
DocumentEntry	serviceStartTime	R2	R2	R2	R2	R2	O	**O**
DocumentEntry	serviceStopTime	R2	R2	R2	R2	R2	O	**O**

Metadata Element	Metadata Attribute	XDS DS	XDS DR	XDM MC	XDR DS	XDR MS	XDS OD	RMU UI
DocumentEntry	size	O	R	R	O	O	X	**I**
DocumentEntry	sourcePatientId	R	R	R2	R	R2	R	**I**
DocumentEntry	sourcePatientInfo	O	O	R2	O	R2	O	**O**
DocumentEntry	title	O	O	O	O	O	O	**O**
DocumentEntry	typeCode	R	R	R2	R	R2	R	**R**
DocumentEntry	uniqueId	R	R	R	R	R	R	**I**
DocumentEntry	URI	O	O	R	O	O	O	**O**
DocumentEntry	**version**	**O**	**O**	**O**	**O**	**O**	**O**	**I**
SubmissionSet	author	R2	R2	R2	R2	R2	R2	**R2**
SubmissionSet	availabilityStatus	O	O	O	O	O	O	**I**
SubmissionSet	comments	O	O	O	O	O	O	**O**
SubmissionSet	contentTypeCode	R	R	R2	R	R2	R	**R**
SubmissionSet	entryUUID	R	R	R	R	R	R	**R**
SubmissionSet	homeCommunityId	O	O	O	O	O	O	**O**
SubmissionSet	intendedRecipient	O	O	R2	R2	R2	O	**O**
SubmissionSet	limitedMetadata	X	X	O	X	R	X	**X**
SubmissionSet	patientId	R	R	R2	R	R2	R	**R**
SubmissionSet	sourceId	R	R	R	R	R	R	**R**
SubmissionSet	submissionTime	R	R	R	R	R	R	**R**
SubmissionSet	title	O	O	O	O	O	O	**O**
SubmissionSet	uniqueId	R	R	R	R	R	R	**R**
Folder	availabilityStatus	O	O	O	O	O	O	**X**
Folder	codeList	R	R	R2	R	R2	R	**X**
Folder	comments	O	O	O	O	O	O	**X**
Folder	entryUUID	R	R	R	R	R	R	**X**
Folder	homeCommunityId	O	O	O	O	O	O	**X**
Folder	lastUpdateTime	O	O	O	O	O	O	**X**
Folder	limitedMetadata	X	X	O	X	R	X	**X**
Folder	patientId	R	R	R2	R	R2	R	**X**
Folder	title	R	R	O	R	O	R	**X**
Folder	uniqueId	R	R	R	R	R	R	**X**

4.3.2 Requirements on Query Type Transactions

4.3.2.1 Query Type Metadata Attribute Optionality

This section lists which attributes shall contain content in a response to a query transaction. The query transactions covered in Table 4.3.2.1-1. The content of the query request can be found within the referenced transaction.

Table 4.3.2.1-1: Responding Actor/Transaction Pairs

Actor	Transaction	Shortname
XDS Document Registry	ITI-18 Registry Stored Query	XDS DR
XCA Initiating Gateway	ITI-18 Registry Stored Query	XCA IG
XCA Responding Gateway	ITI-38 Cross Gateway Query	XCA RG
MPQ Document Registry	ITI-51 Multi-Patient Stored Query	MPQ DR

For each actor/transaction pair across the top of the table, and each metadata attribute row, the cells indicates the requirement for that actor when creating a query response to the paired transaction. The requirements are expressed through these codes:

Table 4.3.2.1-2: Responding Actor Metadata Attribute Optionality Code Definitions

Code	Meaning
R	Required – A value for the attribute shall be supplied by the responding actor when responding to a query
R2	Required if Known – A value for the attribute shall be supplied by the responding actor when responding to the query if a value is available to the actor. For the Document Registry it must supply the value specified in the submission request.
O	Optional – The responding actor may or may not supply a value for this attribute. For the Document Registry it must supply the value specified in the submission request.
X	Prohibited – When responding to a query, a value for the attribute shall not be supplied by the responding actor.
R3	Required for Stable DocumentEntries and not allowed for On-Demand DocumentEntries.

Table 4.3.2.1-3: Responding Actor Metadata Attribute Optionality

Metadata Element	Metadata Attribute	XDS DR	XCA IG	XCA RG	MPQ DR
DocumentEntry	author	R2	R2	R2	R2
DocumentEntry	availabilityStatus	R	R	R	R
DocumentEntry	classCode	R	R	R	R
DocumentEntry	comments	O	O	O	O
DocumentEntry	confidentialityCode	R	R	R	R
DocumentEntry	creationTime	R3	R3	R3	R3
DocumentEntry	**documentAvailability**	**O**	**O**	**O**	**O**
DocumentEntry	entryUUID	R	R	R	R

Metadata Element	Metadata Attribute	XDS DR	XCA IG	XCA RG	MPQ DR
DocumentEntry	eventCodeList	O	O	O	O
DocumentEntry	formatCode	R	R	R	R
DocumentEntry	hash	R3	R3	R3	R3
DocumentEntry	healthcareFacilityTypeCode	R	R	R	R
DocumentEntry	homeCommunityId	O	R	R	O
DocumentEntry	languageCode	R	R	R	R
DocumentEntry	legalAuthenticator	O	O	O	O
DocumentEntry	limitedMetadata	X	X	X	X
DocumentEntry	**logicalID**	**O**	**O**	**O**	**O**
DocumentEntry	mimeType	R	R	R	R
DocumentEntry	objectType	R	R	R	R
DocumentEntry	patientId	R	R	R	R
DocumentEntry	practiceSettingCode	R	R	R	R
DocumentEntry	referenceIdList	O	O	O	O
DocumentEntry	repositoryUniqueId	R	R	R	R
DocumentEntry	serviceStartTime	R2	R2	R2	R2
DocumentEntry	serviceStopTime	R2	R2	R2	R2
DocumentEntry	size	R3	R3	R3	R3
DocumentEntry	sourcePatientId	R	R	R	R
DocumentEntry	sourcePatientInfo	O	O	O	O
DocumentEntry	title	O	O	O	O
DocumentEntry	typeCode	R	R	R	R
DocumentEntry	uniqueId	R	R	R	R
DocumentEntry	URI	O	O	O	O
DocumentEntry	**version**	**O**	**O**	**O**	**O**
SubmissionSet	author	R2	R2	R2	R2
SubmissionSet	availabilityStatus	R	R	R	R
SubmissionSet	comments	O	O	O	O
SubmissionSet	contentTypeCode	R	R	R	R
SubmissionSet	entryUUID	R	R	R	R
SubmissionSet	homeCommunityId	O	R	R	O
SubmissionSet	intendedRecipient	O	O	O	O
SubmissionSet	limitedMetadata	X	X	X	X
SubmissionSet	patientId	R	R	R	R
SubmissionSet	sourceId	R	R	R	R
SubmissionSet	submissionTime	R	R	R	R
SubmissionSet	title	O	O	O	O
SubmissionSet	uniqueId	R	R	R	R

Metadata Element	Metadata Attribute	XDS DR	XCA IG	XCA RG	MPQ DR
Folder	availabilityStatus	R	R	R	R
Folder	codeList	R	R	R	R
Folder	comments	O	O	O	O
Folder	entryUUID	R	R	R	R
Folder	homeCommunityId	O	R	R	O
Folder	lastUpdateTime	O	O	O	O
Folder	limitedMetadata	X	X	X	X
Folder	patientId	R	R	R	R
Folder	title	R	R	R	R
Folder	uniqueId	R	R	R	R