Integrating the Healthcare Enterprise

IHE IT Infrastructure (ITI)

White Paper

Cross-Community Dynamic Data

Public Comment

Date: September 28, 2009

Author: Karen Witting

Email: witting@us.ibm.com, ihe@himss.org

CONTENTS

1 Introduction

This white paper explores the requirements and solutions for sharing health information across communities, or Health Information Exchanges (HIEs), that go beyond the capabilities currently available in the Cross-Community Access (XCA) profile. XCA addresses the sharing of documents created as a consequence of healthcare activity:persistent documents attested to by a clinical provider or healthcare professional. This white paper extends that work to allow for sharing of health information content that is compiled dynamically in response to a request from an external community: machine generated content built specifically because of a request for healthcare data.

2 Requirements

Sharing healthcare information across communities involves a requesting community asking for healthcare information and a responding community providing constrained data.

2.1 Assumptions

The assumptions of this white paper are:

- **Patient Identification:** The request for healthcare information is patient specific. The identification of the patient is outside the scope of this paper and resulted in a unique patient identifier known by the responder and included in the request

2.2 Use Cases

The use cases for this white paper are the same as the XCA and XCPD use cases, except that they enable communication of healthcare content that is more dynamic. The following is a list of a subset of the current XCA and XCPD use cases:

- Multiple Primary Residences: This use case describes the situation where a patient maintains more than one principal residence where the principal residences are not geographically close so the medical data generated while in each residence would be managed by different communities.
- Border Towns: This use case involves a patient who lives on the border between two communities or works and lives at some distance.
- Patient Move: A patient moves from one region to a different, remote region.
- Vacationer: A patient is traveling and goes to the hospital.

2.3 Community architectural models

When sharing across communities, the sharing infrastructure must consider the types of architectural models used to share healthcare data within communities. Three models have been considered for this paper.

2.3.1 Sharing of Stable, Source Attested Documents

In this model, health data is published by a well-identified and responsible source (clinician, care delivery organization, individual consumer, etc.). The source determines when a meaningful collection of data should be published based on clinical events or other activities understood by the source and potential remote entities. The source publishes stable collections of data, or documents, for potential use by remote entities. Remote entities access the documents by querying for a list of documents that have been published and retrieving those that are of interest. Documents are source attested; consistency and wholeness is the responsibility of the source, which provides explicit context for each document (legal record keeping requirements). The documents are stored in a federated set of repositories within the community. If errors need to be corrected or updates are needed they are the responsibility of the source. Entities accessing these records are offered trust guaranties equivalent or superior to paper records shared today.

A common approach to implementing this architecture is by building a Cross-Enterprise Document Sharing (XDS) Affinity Domain, which is a well-known model to the community authoring this paper. It is defined by an XDS Document Registry, which collects entries for documents available within the community. Each entry identifies a stable, persisted document which can be retrieved. XCA is designed to work seamlessly with communities supporting this model.

2.3.2 Single Organization HIE

In this model a single organization assumes the responsibility for the overall consistency of health data it holds on a specific patient, potentially across multiple sites. This data is generally managed in one or more tightly integrated community databases which could be populated through various means, such as HL7 interfaces to/from EHR systems, or all members of the community using a single EHR system in a Software-as-a-Service or hosted (ASP) model. The database is a relational database, which, while it may have fields to identify the encounter that a piece of data came from, is not primarily encounter-based, but rather provides a longitudinal view of the patient's records

By using the community database(s) the organization is able to dynamically collect information based on the constraint of a particular request to share it outside of the community. Collected information may be shared as a set of documents, most of them of static nature (whether they preexist or are created on the fly may vary).

2.3.3 Distributed Source Databases

This model makes use of a community portal which interacts with a distributed set of source databases, each belonging to a distinct healthcare delivery organization, to dynamically collect information based on the constraints of a particular request. The community portal is an application which is specifically built to collect information from disparate sources and present it in an aggregated view. The content collected by the community portal is presented to the requestor and not reused, so that every request results in retrieval of all relevant data from the distributed databases and no data accessible within the community portal. The community portal is dependent on the distributed databases for all content and has limited control over changes to that content.

2.4 Life-Cycle Categories of Healthcare Data

When responding to a request for healthcare data there are several categories of life-cycle for content sharing that may be desirable, both from a requesting and responding side. In some cases content reflects rather stable health information and other cases the content will reflect rather dynamic health information.

In the following, the term document is used to reflect a particular collection of healthcare data, forming an extract of a patient health record.

2.4.1 Stable Documents

As part of an extract of a health record, one or more stable documents may be generated. The typical properties of "stable" documents are persistence, stewardship, potential for authentication and wholeness. Examples are hospital discharge, referral requests, emergency department referrals, results of a laboratory order, diagnostics imaging report, etc. Stable documents may be generated and saved or may exist in virtual form within a healthcare database. In both cases the document content is quite stable once created, unless an error is identified or the document needs to be deprecated. These documents are a record of what is known at the time they are authored, including the healthcare process context. The clinical content of these documents is generally attested to by a healthcare professional and/or a health organization (or consumer in case of PHR created data). Because of this, stable documents come with a robust level of context description (e.g. CDA header) and offer a strong level of legal and clinical trustworthiness.

To obtain up-to-date information, a set of stable documents is identified, retrieved and processed by the consuming system. It is the task of the edge consuming system to aggregate and present to the clinician well organized and easy to navigate information extracted from the retrieved set of documents.

XCA is designed to support access to this type of document. It assumes the document has been created prior to the response to the query. One requirement presented in this paper is to allow for deferral of creation of the document to when it is actually retrieved. This requirement stems from cases where the document exists in virtual form within a database and can be created if needed. Once created it is persisted so that any number of consuming entities will access the same uniquely identified document instance. If later unexpected updates or errors are discovered, the attesting source would simply replace the document, deprecate the erroneous one and offer a link to the replacement. This supports a simple but effective means for remote systems that have accessed prior information to issue an XCA query based on document ID to assess if the stable document, from which information may have been extracted and used for care, is still current.

2.4.2 Dynamic Documents

As part of an extract of a health record, some documents, containing specific types of content, may be generated with non-stable or dynamic content. These documents are "dynamic" in that each retrieval of the document may result in different content. A dynamic document exists only at the moment of retrieval and has no stable properties like persistence or stewardship. The wholeness of a dynamic document will not be based on any clinician attestation and may require careful clinical interpretation depending on the content and the span of aggregation performed by the document authoring system. If the document authoring system originates from a single care delivery organization, context and wholeness may be quite good. If the aggregation was performed across multiple health delivery organizations, the aggregator may have difficulties assuming legal and clinical responsibility for the aggregated content. Examples of dynamic documents may be a summary that collects information related to multiple healthcare events or on-going healthcare events.

Deciding what types of healthcare data to be included in any particular dynamic document may be determined by the requestor or the responder, depending on the agreements between them. In

many cases a national standards organization will define specific document formats, which will detail which types of healthcare data should be included for any particular format and content template. A common adoption process would be that the requestor and responder agree to use a nationally recognized document format/template for exchanging dynamic documents. For stable documents this issue is not as significant since the document source chooses the format/template which best fits its needs as a publisher of documents and the requestor determines if the format/template is acceptable prior to retrieval.

In addition to constraints on the types of healthcare data to be included, another constraint is the date range of the data. This is often starting at some time in the past and continuing to the moment of retrieval.

It is important to understand that there is some interesting overlap between the two types of documents. For example, in responding to a request for lab results for the past six months, the responding community may respond with:

1. A set of stable documents, where each document includes the results associated with one lab order for which the results are final. The set of orders selected would fall within the six month period. There may be some partial lab results that are not communicated because the source decided to wait until they are final.
2. One dynamic document that aggregates all lab results across all orders for the last six months for which there are final results. There may be some partial lab results that may or may not be communicated.

There may be value in encouraging the use of the first approach, which more easily allows the system consuming the lab results to trace the imported lab results back to a specific document instance and enable the checking that these reports are still current (i.e. not deprecated).

2.4.2.1 Security Considerations impact from dynamic documents

The use of dynamic documents requires a new risk analysis regarding the clinical and legal responsibilities assumed by potential aggregators and receivers of aggregated information. When content is aggregated across several source independent systems and shared as the aggregation the receiver of that aggregation may need to take specific action to avoid risks inherent in such an environment. <TBD: need help here>

3 Current solutions

Below we review two existing approaches to the requirements.

3.1 IHE Query for Existing Data (QED)

The Query for Existing Data Profile (QED) supports dynamic queries for clinical data.

The QED profile classifies information into six different categories for the purpose of determining where it might be found.

- Vital Signs
- Problems and Allergies
- Diagnostic Results
- Medications
- Immunizations
- Professional Services

QED also supports the ability to request more than one type of data in a query, resulting in a combination of multiple data types being returned.

QED is intended to be a query between a clinical system and a clinical repository, such as an EMR or EHR. Because of the following concerns this model was not selected for use in the cross-community environment.

- **Incompatibility with XCA**: QED provides data in a somewhat different format from XCA. Most XCA retrieved documents are coded using CDA. QED uses HL7 V3 messaging to encode the content. HL7 V3 messages, while they carry the same type of content, are coded just different enough to require effort on the receiver to convert from HL7 V3 to CDA or vice versa. Adoption of a QED like interface will require all implementers to be able to receive similar content in two different encodings, or pick one to support and not support the other, a very undesirable result.
- **Limited expandability**: QED supports only limited types of data to be exchanged, namely the list above. To support any other content, lab results for example, significant work effort is required to define a query for that content, and enable implementations to support that specific query. By comparison, Document Content exchange is generic enough that a change in type of content does not require a new service be implemented on the receiving side.
- **Complicated Configuration**: Because each QED query type requires specific capabilities on the responding side the requestor must have a mechanism to learn the capabilities of the responder before sending a query. When exchanging document content the capabilities of the responder are expressed inherently in the query response by listing the types of documents that are supported. No specific configuration is needed.
- **Auditing**: It is not clear how to audit QED exchanges in a cross-community environment

3.2 U.S. NHIN "dynamic document" support

The United Status NHIN project relaxed the constraints of the XCA specification to allow support for "dynamic document". In this solution the response to an XCA Cross Gateway Query may contain entries which have a -1 value for hash and size. This indicates that the document does not yet exist, but would be created if requested. At the time of receipt of a XCA Cross Gateway Retrieve referencing an entry for a "dynamic" document the responding community will create the document and return it to the requestor. Once created the document is persisted.

4 Enhancing XCA Query and Retrieve to support dynamic content

XCA assumes that documents are created as a consequence of care delivery and those stable documents are re-used for the purposes of cross-community healthcare information sharing. This interaction pattern assumes that the requesting community will retrieve documents of interest and compile locally the relevant data into a presentation useful for the care provider.

The new requirement emerging from cross-community work is to support documents where the compilation of relevant data is provided by the responding community based on the requirements in the request. When this is used in a query/retrieve model the interaction looks as follows:

1. Query request with optional constraints on document types, time frames, etc.
2. Query response including a list of documents that could be created matching the constraints
3. Retrieve request for one or more of the documents listed (can be repeated to get all)
4. Creation and return of the documents identified in the request

In this model the documents are created as a consequence of a request rather than as a consequence of patient care. This is a key distinction between this requirement and the requirements XCA is designed to address.

The model above may apply in situations where a) a community does not create stable documents, i.e. data is stored mainly in discrete database fields and/or b) where a requestor prefers a dynamic document rather than a collection of stable documents. This dual nature of the model suggests that allowing for both stable and dynamic documents in one transaction is the most flexible approach. This is the solution presented in this paper.

It is important to realize that the aggregation of the content of multiple dynamic documents by the requesting community will still be needed in order to appropriately present the content since information will generally originate from more than a single HIE. So dynamic documents should not be seen as a way to move complexity from the requesting community to the responding community.

Change Proposal 401, submitted to the IHE ITI Technical committee in January 2009, requests the following:

> The XCA profile doesn't assume that a community will have an XDS infrastructure. At the same time the response to an XCA query requires two pieces of metadata, which assume that a document exists within the community – hash, and size. If, indeed, XCA is meant to work with communities without an XDS infrastructure, there should be an option to accommodate workflows where the responding community can provide the requested document (e.g. a Discharge Summary), but will create the document only when an actual document request is made. In such a case the hash and size cannot be supplied to the response of an XCA query.

This White Paper is the first step in responding to this request, by clarifying the requirements and outlining a proposed solution.

4.1 Overview

The current XCA Cross Gateway query returns a list of available documents; each document is represented by a Document Entry which contains metadata about the document.

This proposal allows for two new types of entries to be returned as part of a XCA Cross Gateway query.

- **Deferred Creation**: a "deferred creation" entry reflects a document that will be created, and persisted, only when retrieved. Retrievals resulting from "deferred creation" entries look identical to retrievals of normal documents, except for an additional side-effect.
- **Dynamic**: a dynamic entry reflects a potential document that will collect the latest, most up-to-date information at the time of retrieval. Retrievals resulting from dynamic entries are different from normal documents because the content returned contains a different unique identifier than that which is used in the request.

We will refer to the three types of entries returned in the XCA Cross Gateway Query as:

- Stable Document Entry
- Deferred Document Entry
- Dynamic Entry

The sections specific to each of these types of entries refer to identifiers of various types. How these identifiers are used and relate to each other is central to understanding the mechanisms used to describe and retrieve content.

- DocumentEntry uniqueID – is the identifier found in the uniqueID metadata field of a DocumentEntry. It is a unique identifier for the entry and also uniquely defines the document associated with the entry. It is used in a retrieve request to identify which specific document should be retrieved.
- CDA Header id – Every HL7 CDA R2 document header includes a value for the ClinicalDocument/id field which uniquely identifiers the document. This value is always the same as the DocumentEntry uniqueID for the DocumentEntry which contains metadata about this document. Although we address CDA specifically in the explanation, the concepts also apply to any type of document content where a unique identifier exists in a code field within the document. Some document content does not include a unique identifier so these explanations do not apply to that situation.
- Template uniqueID – is the identifier found in the uniqueID metadata field of a Dynamic Entry. It is a unique identifier for the entry but does not identifier any document.

4.1.1 Stable DocumentEntry

A stable document entry is defined in detail within the XDS profile. A few details are included here as background for discussion of the new types.

A stable document entry contains metadata about an existing document available for retrieval. The XDS specification defines the following metadata (some are required, some are optional):

- Patient Identifier

- Coded values describing the document: class code, practice setting code, healthcare facility type code, event codes, confidentiality code, format code
- Timeframe of the healthcare service that was performed which resulted in creation of the document (service start/stop time)
- Time of creation of the document
- Document author and legal authenticator
- Unique identifier for use in retrieval (DocumentEntry uniqueId, repositoryID and homeCommunityId)
- Size and hash of the document
- AvailabilityStatus = "Approved" or "Deprecated" indicating whether the entry is currently active or has been deprecated in favor of a more up-to-date document.

If the document returned on a retrieve request is CDA, it will have in the `ClinicalDocument/id` in the HL7 CDA R2 header the same value of the DocumentEntry uniqueId. An XCA query against this document entry will allow the verification that the document content is still valid per the stewardship of its source.

To summarize, Stable Document Entries:

- Object type = DocumentEntry
- Status = Approved or Deprecated
- DocumentEntry uniqueId = CDA header id
- Content created prior to query response

4.1.2 Deferred Document Entry

A deferred document entry describes a stable document which has not yet been created. The content of the document has been identified and exists in some form, probably as database fields, but the document creation has been deferred until the document is retrieved. The goal of this deferral is to avoid the overhead of creating and persisting documents that are never requested. A Deferred Document Entry is transitioned into a Stable Document Entry with the same DocumentEntry uniqueID at the time of retrieval. If the document returned on a retrieve request is CDA, it will have in the `ClinicalDocument/id` in the HL7 CDA R2 header the value of the DocumentEntry uniqueId metadata attribute.

A Deferred Document Entry is identical to a Stable Document Entry except:

- availabilityStatus = "DeferredCreation"
- size, hash, and creationTime are not specified

Every Deferred Document Entry with the same DocumentEntry uniqueID will refer to the exact same document content. Once the first retrieval of that DocumentEntry uniqueId has occurred the document becomes stable, i.e. it is created, the Document Entry becomes a normal stable document entry and all future retrievals for that DocumentEntry uniqueId (whether from the same initiator or a different initiator) must have the identical content. An XCA query against this document entry will allow the verification that the document content is still valid per the stewardship of its source.

To summarize, Deferred Document Entries:

- Object type = DocumentEntry
- Status = DeferredCreation
- Size, hash and creationTime are not specified
- DocumentEntry uniqueId = CDA header id
- Content created at time of retrieve
- Becomes a Stable DocumentEntry at time of retrieve

4.1.3 Dynamic Entry

A dynamic entry describes the ability to generate a collection of dynamic content which is the latest, greatest information at the time of the request. Dynamic Entries never reflect actual document content, but rather the potential for a document with the characteristics described in the metadata of the dynamic entry. A dynamic entry is never transitioned into another type of entry.

The uniqueID associated with a Dynamic Entry, referred to as a template uniqueID, represents a template for a potential document but will never represent an actual document. A retrieve request specifying a template uniqueID will return content identified by a uniqueID different than the template uniqueID.

A Dynamic Entry contains the same set of metadata as a Stable Document Entry except:

- objectType = DynamicEntry
- size, hash, and creationTime are not specified
- service start/stop time not specified
- template uniqueID refers to a template not a document

Every Dynamic Entry with the same template uniqueID will refer to the same potential content. Actual content depends on the time of retrieval. Dynamic Entries are never transitioned to Document Entries. A dynamic entry defines a “handle” or template to the most up-to-date clinical data. That “handle” is good forever, so the holder of the handle can re-use it in a retrieve request to get the latest information, without the need for an additional query. If the underlying content has not changed, the next retrieve request for a template uniqueID may have the same returned uniqueID. If the returned uniqueID is the same as the prior request for that template uniqueID, then the requestor can be sure that the underlying content has not changed. If the returned uniqueID is not the same as the one from the prior request for that template uniqueID then the requester cannot know whether the content is the same.

To summarize, Dynamic Entries:

- Object type = DynamicEntry
- Status = Approved or Deprecated
- Size, hash and creationTime are not specified
- Template uniqueId not equal CDA header id

- Content created at time of retrieve
- Might create a new Stable DocumentEntry at time of retrieve

4.1.3.1 Dynamic with persistence

When a retrieve request is received specifying a template uniqueID the responder may choose to persist the document generated as a result and allow requestor access to the metadata via query or ability to retrieve the generated document via a Stable Document Entry uniqueID. When this type of persistence is supported by the responder the entry is refered to as "Dynamic with persistence". The persistence refers not only to the saving of the content for re-use, but more specifically to the ability of the requester to use retrieve to access that exact, possibly now historic, content and use a query to get metadata about the content.

4.1.3.2 Dynamic without persistence

When a retrieve request is received specifying a template uniqueID the responder may choose not to persist the document or allow subsequent retrieval of the exact content. In this case the entry is refered to as "Dynamic without persistence".

4.1.4 Summary of Query Response Entry Types

Entry Type	ObjectType & status	Result of retrieve	CDA Header id	Creation of content	Adjustment to metadata	Omitted metadata
Stable	Document Entry / Approved	DocumentEntry uniqueId	DocumentEntry uniqueId	Prior to query response	None	None
Deferred	Document Entry / Deferred	DocumentEntry uniqueId	DocumentEntry uniqueId	As part of retrieve	Transition to Stable	Hash, size, creationTime
Dynamic with persistence	Document Template / Approved	New Stable DocumentEntry uniqueId	New Stable DocumentEntry uniqueId	As part of retrieve	Creation of Stable	Hash, size, creationTime, service start and stop time
Dynamic without persistence	Document Template / Approved	New Transient uniqueId	New Transient uniqueId	As part of retrieve	None	Hash, size, creationTime, service start and stop time

Figure 4.1-1 Summary of Query Response Entry Types

4.2 Process flows using deferred and dynamic entries

4.2.1 Stable Document Content – deferred creation

Figure 4.2-1 shows an example workflow where a deferred document entry is used to reflect a document with stable content. That document is created only when it is first retrieved, and re-used subsequently on receipt of additional retrieve requests for the document. The uniqueID of the document is 5 and the CDA ClinicalDocument id also contains 5.

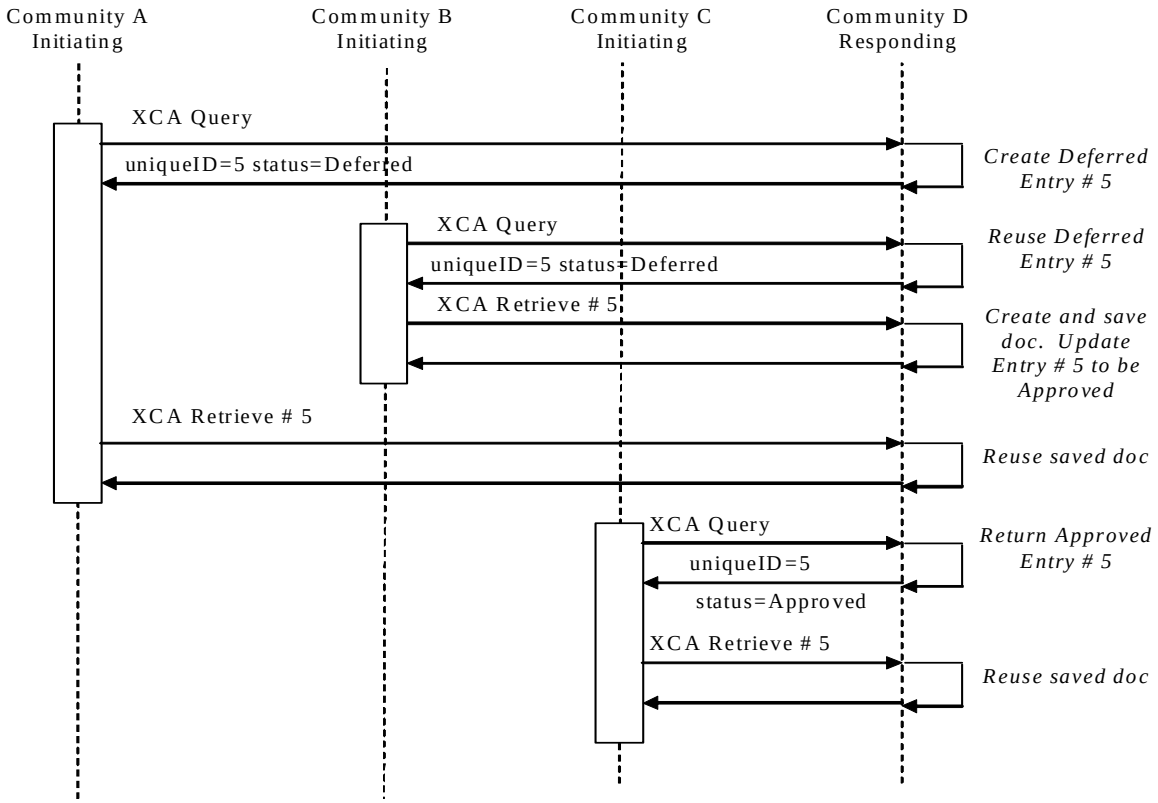

Figure 4.2-1: Stable Document Content – deferred creation

4.2.2 Dynamic Document Content – deferred creation

Figure 4.2-2 shows an example workflow which uses deferred creation to provide dynamic document content. Since the content may change between retrievals a different uniqueID must be used for each query request. Community A is given uniqueId=5 and Community B is given uniqueId=6. If each community retrieves the content, two documents are created and saved, each referenced by the corresponding, now stable, Document Entry. The two documents will contain the exact same content if the underlying healthcare data has not changed. After the documents are created as a consequence of retrieve they must also be made available in response to a new query request. Thus when Community C sends its query, it may need to choose

between the two previously created documents, or a new deferred document. This workflow is attempting to show that using the deferred path for delivering dynamic content results in unnecessary overhead and, although feasible, it is preferable to use a dynamic document entry for this type of content, see the next section.

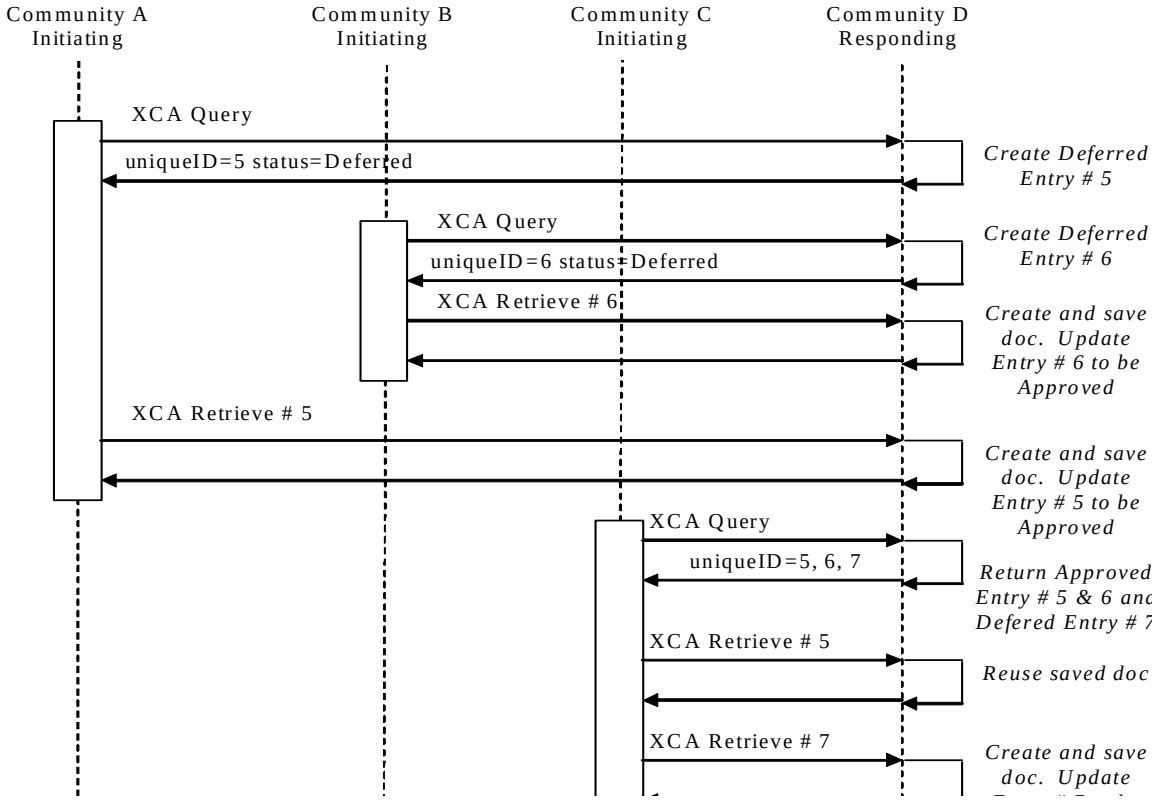

Figure 4.2-2: Dynamic Document Content – deferred creation

4.2.3 Dynamic Document Content – dynamic creation with persistence

Figure 4.2-3 shows the use of a dynamic document entry to allow access to a most up-to-date set of data from a responding community. This workflow includes persisting each document returned as a result of a retrieve and creating a stable document entry to reflect it. The semantics of the response to a retrieve of a dynamic document are different than the semantics of the response to a retrieve of stable or deferred documents. The response to a retrieve of a dynamic document contains a new uniqueId, different than the uniqueId specified in the request, which is the same as the value within the ClinicalDocument id of the document returned. Use of an appropriate query against that new uniqueId will allow the requestor access to updated metadata about the document, including size, hash, etc. This workflow assumes that some change to the underlying data occurs between the time community A retrieves dynamic document #5 and community C retrieves dynamic document #5. That is why #6 is not re-used in response to community C's request.

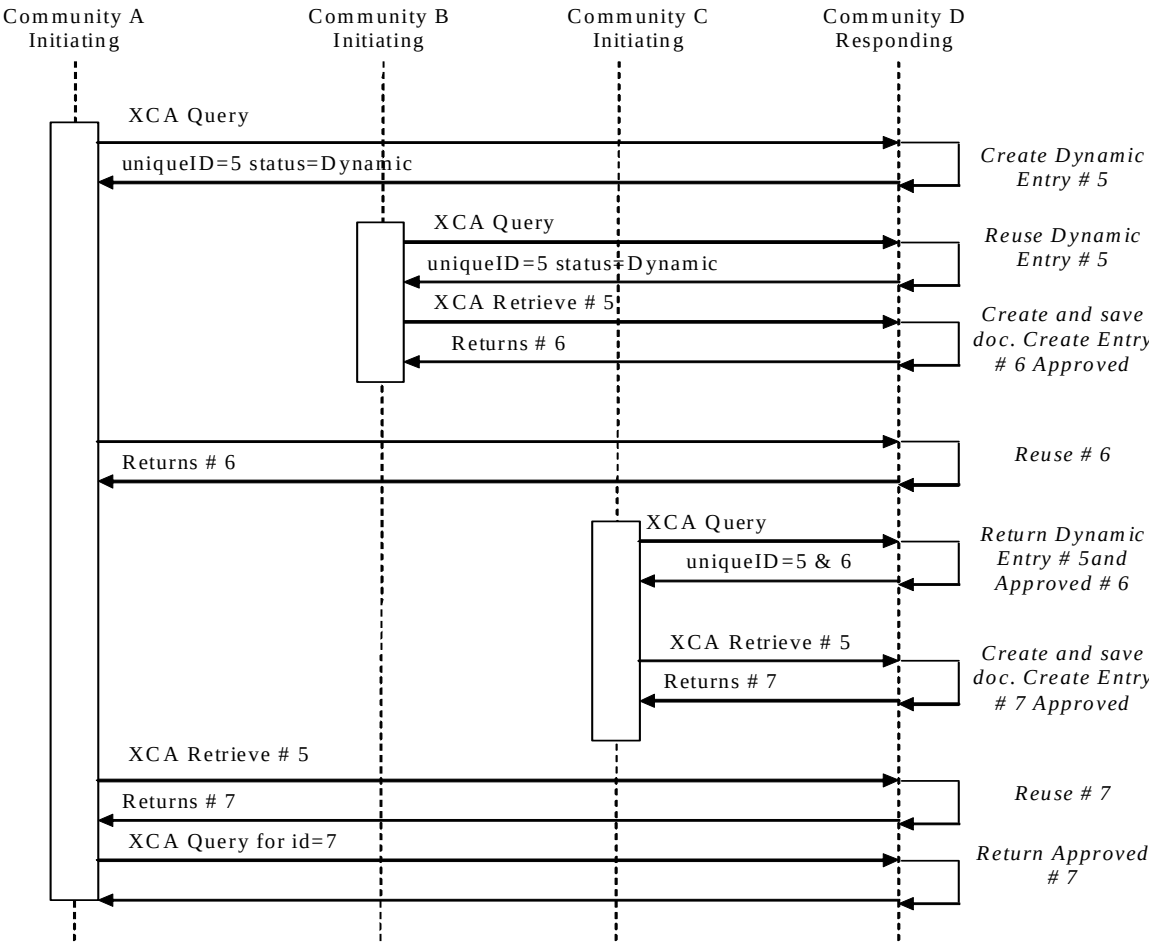

Figure 4.2-3: Dynamic creation with persistence

4.2.4 Dynamic Document Content – dynamic creation without persistence

Figure 4.2-4 shows the use of a dynamic document entry to allow access to a most up-to-date set of data. This workflow does not include persisting each document returned as a result of a retrieve. The semantics of a retrieve of a dynamic document change without persistence in that the response contains the same uniqueId as the request which is **not** the same as the value within the ClinicalDocument id. It is not possible for the requestor to access updated metadata about the document like size, hash, etc.

s

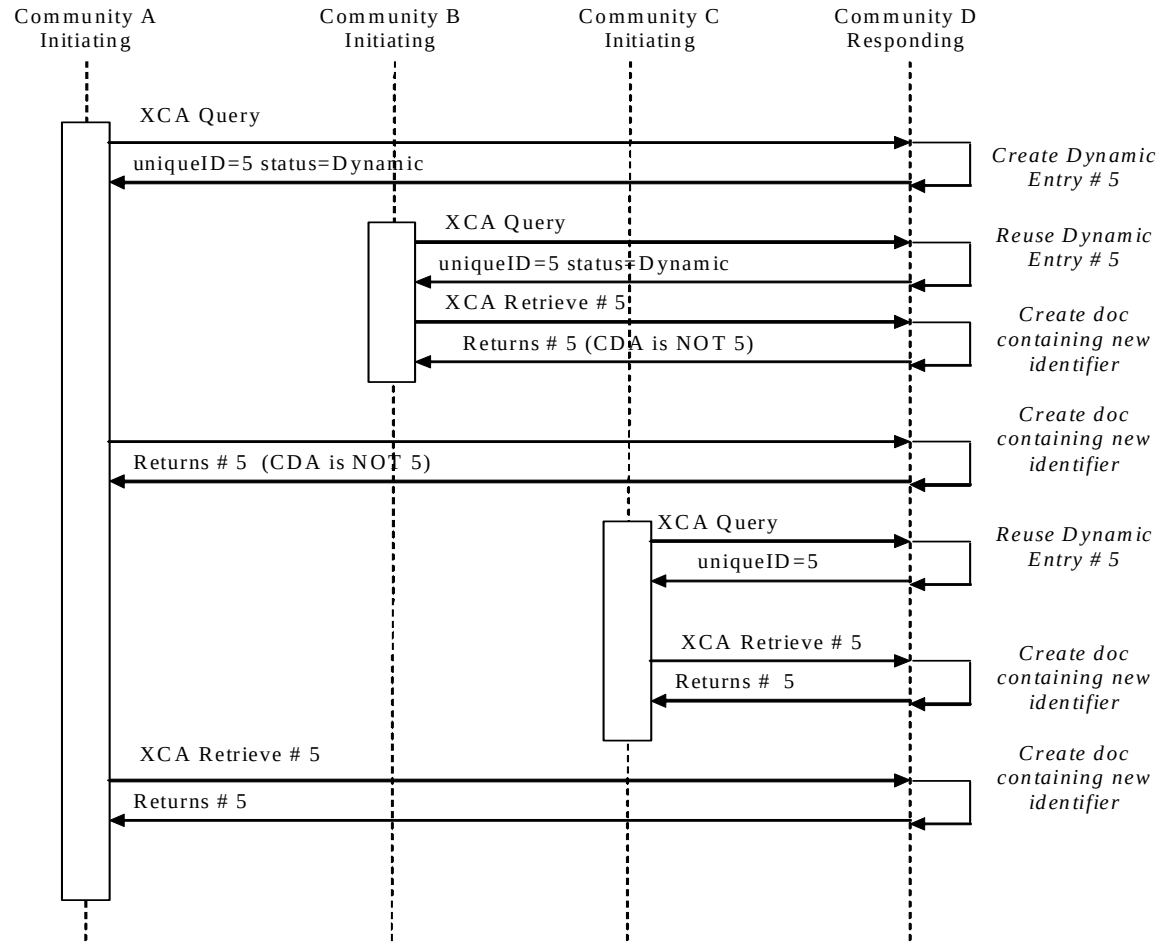

Figure 4.2-4: Dynamic creation without persistence

4.3 Actors and Transactions

This white paper suggests the following changes to the XCA profile:

- A new option for XCA Initiating Gateway and XCA Responding Gateway for support of Deferred and Dynamic. When support for this new options is declared:
    - Responding Gateways shall be capability of generating Deferred and Dynamic entries and shall conform to the requirements of these Entries during retrieve.
    - Initiating Gateways shall be able to request Deferred and Dynamic entries as part of a query and able to retrieve and deliver the entries as appropriate for their implementation.
- An update to the Cross Gateway Query Transaction allowing for the new query parameter
- An update to the Cross Gateway Retrieve Transaction allowing for the return of additional information for retrieval of Dynamic Entries

Interoperability between gateways declaring this option and gateways not declaring this option is enabled via the following mechanisms:

- An Initiating Gateway which does not support the new option will never specify the new status or query parameter in the query, so a Responding Gateway will never return that content.
- A Responding Gateway which does not support the new options will not ignore the new query parameter. Concern remains regarding its handling of the new status value.

4.4 Implementation Details

4.4.1 Deferred Document Processing Details

4.4.1.1 Responding Gateway

A Responding Gateway shall, at its discretion, return entries to a XCA Cross Gateway Query request with:

- availabilityStatus = “DeferredCreation”
- size, hash, creationTime unspecified
- a uniqueId attribute uniquely assigned to the potential document described by the metadata

Upon receipt of a XCA Cross Gateway Retrieve request for a DocumentEntry uniqueId associated with a deferred document the Responding Gateway shall:

- create and save the document
- update the document entry associated with the DocumentEntry uniqueId:
    - to include values for size and hash based on created document
    - creationTime equal to current time
    - availabilityStatus = “Approved”

Upon receipt of a subsequent XCA Cross Gateway Query where the parameter constraints include this document entry, the Responding Gateway shall include the entry updated during retrieve.

Upon receipt of a subsequent XCA Cross Gateway Retrieve request for this DocumentEntry uniqueId the Responding Gateway shall return a bit-identical version of the previously created document.

4.4.1.2 Initiating Gateway

The Initiating Gateway shall accept a Document Entry without size, hash, creationTime values when availabilityStatus=”DeferredCreation”. The Initiating Gateway may retrieve the referenced document via a normal XCA Cross Gateway Retrieve. If the Initiating Gateway has need of the size and hash for the document it shall perform both of the following steps in this precise order:

1. Issue a XCA Cross Gateway Retrieve for the referenced document. The document is generated and saved by the responding community as a result of this request.

2. Issue a XCA Cross Gateway Query specifying the document uniqueId in a GetDocuments stored query. The response to the Cross Gateway Query GetDocuments request shall include availabilityStatus=”Approved” and size and hash value set.

4.4.2 Dynamic Document Processing Details

4.4.2.1 Responding Gateway

A Responding Gateway shall, at its discretion, return entries to a XCA Cross Gateway Query request with:

- objectType = DynamicEntry
- availabilityStatus = “Approved” or “Deprecated”
- size, hash, creationTime unspecified
- service start/stop time not specified
- a template uniqueId attribute uniquely assigned to one potential collection of data

Upon receipt of a XCA Cross Gateway Retrieve request for a template uniqueId associated with a dynamic document the Responding Gateway shall:

- create content reflecting the most up-to-date collection of data represented by the template uniqueID
- Assign a unique identifier to this collection of data, called new uniqueID
- If a CDA document was created, set `ClinicalDocument/id` in the HL7 CDA R2 header equal to the new uniqueID. For other document formats, use new uniqueID as appropriate.
- Return new uniqueID in the new Retrieve Response field labeled “NewUniqueID”.
- If the Responding Gateway chooses to respond to subsequent retrieve and query requests specifying the new uniquieID it shall:
    - Created a Stable Document Entry representing the content just created, using the new uniqueId for this new Stable Document Entry
    - Specify in the retrieve response that a stable document was created.
    - Upon receipt of a subsequent XCA Cross Gateway Query where the parameter constraints include this new stable document entry, the Responding Gateway shall include the entry in the response.
- If the Responding Gateway chooses not to support the new content in subsequent retrieve and query requests, it shall specify in the retrieve response that a stable document was not created.
- Upon receipt of another XCA Cross Gateway Retrieve request for the same dynamic entry the Responding Gateway may return the same document, if the underlying data has not changed, or may create a new document.

4.4.2.1.1 Dynamic Entry retrieve response - Stable document created

```
<DocumentResponse>
  <HomeCommunityId> urn:oid:1.3.6.1.4.941</HomeCommunityId >
```



```
  <RepositoryUniqueId>1.3.6.1.4...1000</RepositoryUniqueId>
  <DocumentUniqueId>1.3.6.1.4...2300</DocumentUniqueId>
  <NewDocumentUniqueId>1.3.6.1.4...2897</NewDocumentUniqueId>
  <StableEntryCreated>true</StableEntryCreated>
  <mimeType>text/xml</mimeType>
  <Document>UjBsR09EbGhjZ0dTQUxNQUFBUUNBRU1tQ1p0dU1GUXhEUzhi</Document>
</DocumentResponse>
```

4.4.2.1.2 Dynamic Entry retrieve response - Stable document not created

```
<DocumentResponse>
  <HomeCommunityId> urn:oid:1.3.6.1.4.941</HomeCommunityId >
  <RepositoryUniqueId>1.3.6.1.4...1000</RepositoryUniqueId>
  <DocumentUniqueId>1.3.6.1.4...2300</DocumentUniqueId>
  <NewDocumentUniqueId>1.3.6.1.4...2897</NewDocumentUniqueId>
  <StableEntryCreated>false</StableEntryCreated>
  <mimeType>text/xml</mimeType>
  <Document>UjBsR09EbGhjZ0dTQUxNQUFBUUNBRU1tQ1p0dU1GUXhEUzhi</Document>
</DocumentResponse>
```

4.4.2.2 Initiating Gateway

The Initiating Gateway shall accept a Dynamic Entry without size, hash, creationTime, service start/stop values. The Initiating Gateway may retrieve a version of the dynamic document via a normal XCA Cross Gateway Retrieve. If the a stable document was created during the retrieve the Initiating Gateway may access the additional metadata for the document by sending a XCA Cross Gateway Query using the new uniqueID specified in the response.

4.4.3 Query Parameters

4.4.3.1 Application of existing query parameters

All current query parameters apply correspondingly to the new types of entries except:

- **Dynamic Entries**: creationTime and service start and stop times are not considered when deciding which Dynamic Entries to return from a query request.
- **Deferred Document Entries**: The creationTime query parameter does not make sense in respect to Deferred Document Entries because the creationTime has not been set yet. So creationTime is not considered when deciding which Deferred Document Entries to return from a query request.

4.4.3.2 New query parameter

In order to enable the ability for the initiating side to select which types of entries to be returned a new query parameter is needed. The new query parameter, called ObjectTypes for now, would be added to the following stored queries:

- FindDocuments
- GetAll

The ObjectTypes parameter would contain a list of ObjectTypes to be included in the respond to the query. Valid values for the ObjectTypes parameter:

- urn:uuid:7edca82f-054d-47f2-a032-9b2a5b5186c1 – indicates stable document entries are desired
- urn:uuid:34268e47-fdf5-41a6-ba33-82133c465248 – proposed urn to indicate dynamic document entries are desired

The default for FindDocuments would be stable Document Entries only. For GetAll would be all except DynamicEntry. (Open Issue: review other types of entries, i.e. Folder, Submission Set, and consider how they should be addressed via this new parameter and the other stored queries.)

Review of other stored queries and the effect of this change:

- FindSubmissionSets – no effect
- FindFolders – no effect
- GetDocuments – would this allow for retrieving a dynamic entry by template uniqueID?
- GetFolders – no effect
- GetAssociations – no effect
- GetDocumentsAndAssociations - would this allow for retrieving a dynamic entry by template uniqueID?
- GetSubmissionSets - would this allow for retrieving submission sets by dynamic entry template uniqueID?
- GetSubmissionSetAndContents - would this allow for retrieving submission sets by dynamic entry template uniqueID?
- GetFolderSetAndContents - would this allow for retrieving folders by dynamic entry template uniqueID?
- GetFoldersForDocument - would this allow for retrieving folders by dynamic entry template uniqueID?
- GetRelatedDocuments – how does it apply to dynamic?

4.5 Open Issues and Questions

- X003: Should the approach also be applied to XDS. See section 4.5.1 for discussion of this idea.
- X010: Need to check what the consequences of specifying a deferredcreation status on a query sent to a responder which does not support the option. This may result in an error, how should this be handled?
- X011: Should there be one option for both Deferred and Dynamic, or separate options for each?
- X012: How does the new types of entries affect query, see section 4.4.3.2 for details.

4.5.1 Discussion of application to XDS

The following are comments from Bill Majurski to be used as we consider this question:

I believe this general approach should be available in XDS. It is possible to map out the same high level functionality for the operation of an Integrated Source/Repository (ISR). I have not yet attempted to map it to individual Source & Repository so I cannot yet say whether this is possible.

To make this work for deferred documents, the registry transaction would include no size or hash. It would contain a creationTime which would be a time after the creation of the raw contents and before the register transaction. Later, on first retrieve, the ISR would be required to use a feature out of Metadata Versioning to update size and hash.

The use of dynamic documents would likely be restricted to the 'register if retrieved' flavor given the nature of XDS. The register transaction would contain no size, hash, creationTime, or serviceTimes as in XCA. Assuming 'register if retrieved' then a normal register transaction could be generated to record uniqueId, size, hash, creationTime, and potentially serviceTime. Dynamic documents do not require the use of metadata versioning.

4.6 Closed Issues

- X001: Should the XDS Cross Gateway request include an indicator that the Initiating Gateway is able to process deferred creation and dynamic documents? Lack of this indicator would restrict the Responding Gateway to return only stable documents. Alternatively, support for this feature could be required of all Initiating Gateways. **Answer:** Support for this function will be an option on both sides. The Gateway will indicate its support for the function as part of its query. If it supports deferred documents it will include deferredcreation as a requested status value. For dynamic we will need to extend the query to add a new parameter indicating which objecttypes to return. Default will be to return document entry object types. Responding gateways which don't support this new parameter will ignore it, per the current query specification.
- X002: How is an XCA Initiating Gateway expected to interact with an XDS Document Consumer in support of these new types of entries? Should the XDS Document Consumer be enabled to understand the new types of entries defined here? Will the XCA Initiating Gateway be expected to strip out some of the response, or convert it? **Answer:** Consumer will interact with the initiating gateway in the same way that the initiating gateway interacts with responding. Consumer will have the ability to declare an option to support these new types and will populate any queries with details about its request.
- X004: Is availabilityStatus the right place to designate the different types of document entries. **Answer:** This field is appropriate for designating deferred entries, but dynamic entries are so different that another approach is desirable. Agreed to use a new objectType to designate this type of entry. For now call it DynamicEntry.
- X005: A retrieve cannot request more than one dynamic document in a single request. To do so would make it impossible to identify which dynamic document (by generated uniqueId) correlated to which dynamic document definition/template as identified by the requested template uniqueId. We would need to determine if this potential confusion is important. Consider extending the schema for retrieve to account for this new use. Potentially the retrieve response could include a new element. **Answer:** Agreed to extend

the schema so that all the request data is returned as is done for a normal retrieve, and additional elements carry information specific to the dynamic case.

- X006: Need to analyze the impact on the initiating community systems using retrieved dynamic content which has been aggregated by a responding community across several source independent sources. For example, where are the clinical and legal responsibilities? **Answer:** Agreed that a section is needed to address this issue.
- X007: What about DSUB? Should it be enabled to allow subscriptions to the new type of entries described in this paper? **Answer:** Declared out of scope for this paper.
- X008: Describe life cycles of deferred and dynamic documents **Answer:** Described as part of details section.
- X009: Describe how query parameters affect the choice of deferred/dynamic documents to be returned. **Answer:** for dynamic documents creationTime and service start and stop times are ignored, do not apply. For deferred docs use of creationTime might suggest that deferred documents, which aren't created yet, should not be included. If a start creationTime but no stop creationTime was specified then of course deferred documents would be included if applicable.

