Enhancing IHE XDS for Federated Clinical Affinity Domain Support

Asuman Dogac, *Member, IEEE,* Gokce B. Laleci, Thomas Aden, Marco Eichelberg

***Abstract*— One of the key problems in healthcare informatics is the inability to share patient records across enterprises. To address this problem, an important industry initiative called "Integrating the Healthcare Enterprise (IHE)" specified the "Cross Enterprise Document Sharing (XDS)" Profile.**

In the IHE XDS, healthcare enterprises that agree to work together form a "Clinical Affinity Domain" and store healthcare documents in an ebXML registry/repository architecture to facilitate their sharing. The affinity domains also agree on a common set of policies such as coding lists to be used to annotate clinical documents in the registry/repository and the common schemes for patient identification. However, since patients expect their records to follow them as they move from one clinical affinity domain to another, there is a need for affinity domains to be federated to enable information exchange.

In this paper we describe how IHE XDS can be enhanced to support federated clinical affinity domains. We demonstrate that federation of affinity domains are facilitated when ontologies, rather than coding term lists, are used to annotate clinical documents. Furthermore we describe a patient identification protocol that eliminates the need to keep a master patient index file for the federation.

***Index Terms*— eHealth, Semantic Interoperability, Integrating Healthcare Enterprise, Clinical Document Sharing, Patient Identification**

I. Introduction

One of the key problems in healthcare informatics is the inability to share patient records across enterprises. There are several standardization efforts to digitally represent clinical data such as HL7 CDA [12], CEN EN 13606 EHRcom [18] and openEHR [17]. These standards, which are currently under development, aim to structure and markup the clinical content for the purpose of exchange. However since there are more than one standard, it is still difficult to achieve interoperability. To address this document sharing problem in the healthcare domain in a practical manner, an industry initiative called "Integrating the Healthcare Enterprise (IHE)" [13] specified the "Cross Enterprise Document Sharing (XDS)" Profile [14]. The basic idea of IHE XDS is to store healthcare documents in an ebXML registry/repository architecture to facilitate their sharing. IHE XDS is not concerned with document content; it only specifies metadata to facilitate the discovery of documents.

Asuman Dogac and Gokce B. Laleci are with the Middle East Technical University, Ankara, Turkey and Thomas Aden and Marco Eichelberg are with OFFIS – Oldenburger Forschungs- und Entwicklungsinstitut für Informatik-Werkzeuge und -Systeme, Oldenburg, Germany

This work is supported by the European Commission through IST-1-002103-STP Artemis project and in part by the Scientifi c and Technical Research Council of Turkey (TÜBİTAK), Project No: EEEAG 104E013

In the IHE XDS Profile, healthcare enterprises that agree to work together for clinical document sharing is called the "Clinical Affinity Domain". Such institutes agree on a common set of policies such as how the patients are identified, the access is controlled, and the common set of coding terms to represent the metadata of the documents.

However since patients expect their records to follow them as they move from one clinical affinity domain to another, there is a need for the clinical affinity domains to be federated, that is, they should be able to exchange information. IHE XDS plans to address this issue in the future.

Federation of XDS-based affinity domains requires a number of problems to be addressed. In this paper, we describe how IHE XDS can be enhanced to support the federated affinity domains by addressing the following problems:

- In an XDS clinical affinity domain, most of the metadata are defined through domain specific coding lists. Hence, given the metadata in one affinity domain, it is very difficult to locate a document in another affinity domain by using this metadata. To alleviate this problem, we propose to use metadata ontologies rather than coding lists. It should be noted that an ontology describes *consensual knowledge*. More importantly, an ontology is machine processable since it is defined through a formal ontology language. We show that when metadata ontologies are defined through formal ontology languages like Web Ontology Language (OWL) [22], it becomes possible both to map the metadata concepts one into other through ontology mapping and to use rules to further enhance the semantics. In this way, accessing documents across clinical affinity domains are greatly facilitated.
- In IHE XDS, each affinity domain uses a common mechanism for patient identification. Individual institutions can still use their own patient IDs, but a mapping between the local patient ID and the affinity-domain-global patient ID must be provided.
  When clinical affinity domains are federated, in order to identify a patient in another affinity domain, patient identification across affinity domains must be addressed. A master patient index that spans multiple affinity domains is not feasible to create and maintain manually. Therefore we introduce a secure and fault-tolerant patient ID look-up mechanism based on patient demographics such as name, birthdate, place of birth etc. to be used in federated IHE XDS.

The paper is organized as follows: In Section II, we briefly describe the ebXML architecture emphasizing the ebXML

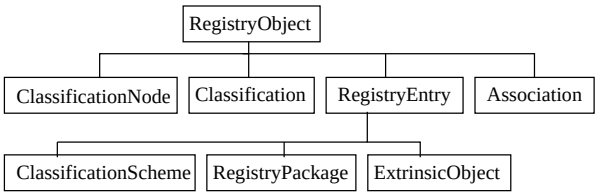

Fig. 1. A Part of the ebXML RIM Class Hierarchy

registry semantic mechanisms. Section III introduces the IHE "Cross Enterprise Document Sharing (XDS)" Profile. Since Web Ontology Language (OWL) is used in this work in defining semantics, Section IV briefly summarizes OWL. Section V is devoted to the semantic annotation of clinical documents in IHE XDS. Section VI describes how to make use of metadata ontologies in XDS. In Section VII, we introduce a patient identification protocol that can be combined with XDS to provide a look-up of patient IDs based on demographics in a secure and fault-tolerant manner. Finally, Section VIII concludes the paper and presents the future work.

II. EBXML ARCHITECTURE

The ebXML specification aims to facilitate electronic business as follows:

- in order for enterprises to conduct electronic business with each other, they must first discover each other and the products and services they have to offer. ebXML provides a registry/repository architecture specification where such information can be published and discovered. A repository is a location (or a set of distributed locations) where a document pointed at by the registry resides and can be retrieved by conventional means (e.g., http or ftp). The repository is capable of storing any type of electronic content, while the registry is capable of storing metadata that describes content. The content within the repository is referred to as "repository items" while the metadata within the registry is referred to as "registry objects". The term object is used to refer to either registry object or repository item. An ebXML registry mechanism allows business documents and relevant metadata to be registered and retrieved as a result of a query.
- An enterprise needs to determine which business processes and documents are necessary to communicate with a potential partner. Registry metadata can be used for searching relevant documents and business processes.

A. ebXML Registry Information Model

The Registry Information Model (RIM) [11] defines what types of objects are stored in the registry and how the stored objects are organized.

RIM allows to define semantics basically through two mechanisms: first, it allows properties of registry objects to be defined through "slots" and, secondly, metadata can be stored in the registry through a "classification" mechanism. This information can then be used to classify (annotate) the registry objects and later to use this classification (annotation) to discover the registry objects through the ebXML query mechanisms.

Figure 1 presents the part of the ebXML RIM [11] related with metadata information. The top level class in RIM is the "RegistryObject". This is an abstract base class used by most classes in the model. It provides minimal metadata for registry objects. "Slot" instances provide a dynamic way to add arbitrary attributes to "RegistryObject" instances. "Association" instances are used to define many-to-many associations between objects in the information model. An Association instance represents an association between a source RegistryObject and a target RegistryObject. Each Association must have an "associationType" attribute that identifies the type of that association. There are a number of predefined *Association Types* that a registry must support to be ebXML compliant [11] as shown in Table I and ebXML allows this list to be expanded.

"ClassificationScheme" instances describe a structured way to classify or categorize RegistryObject instances. A ClassificationScheme defines a tree structure made up of "ClassificationNode"s. RegistryPackage instances, on the other hand, group logically related RegistryObject instances together.

There is another ebXML registry object called *Classification*. Classification instances are used to classify other RegistryObject instances. A Classification instance identifies a ClassificationScheme instance and taxonomy value defined within the classification scheme.

Classification schemes could be internal or external depending on whether the referenced classification scheme is stored internal to the registry or it is external. An internal classification always references the node directly by its id while an external classification references the node indirectly by specifying a representation of its value that is unique within the external classification scheme.

An important observation is that ebXML registry semantic structures can adequately represent not only taxonomies but also ontologies. A detailed comparison of taxonomies vs. ontologies is given in [6] and how Web Ontology Language (OWL) constructs can be stored in ebXML registries is described in [7], [8].

III. IHE XDS

In IHE XDS, the ebXML repository is used for storing the clinical documents in a persistent manner and the metadata stored at the ebXML registry is used to facilitate the discovery of the documents.

An XDS document, stored in the repository, is represented as an *ebXML ExtrinsicObject* in the registry. ExtrinsicObjects in ebXML are used to provide metadata that describes submitted content whose type is not intrinsically known to the registry. In order to group the related documents, the XDS documents are organized into folders (e.g. a period of care, a problem, immunizations, etc.). In this way a document consumer can find all the entries placed in the same folder. XDS document folders are constructed in the ebXML registry by using *ebXML RegistryPackage*. As previously mentioned ebXML RegistryPackage is used to group logically related RegistryObject instances together.

TABLE I

PREDEFINED ASSOCIATION TYPES IN EBXML REGISTRIES

Name	*Description*
RelatedTo	Defines that the source RegistryObject is related to the target RegistryObject.
HasMember	Defines that the source RegistryPackage object has the target RegistryObject object as a member. Reserved for use in Packaging of RegistryEntries.
ExternallyLinks	Defines that the source ExternalLink object externally links the target RegistryObject object. Reserved for use in associating ExternalLinks with RegistryEntries.
Contains	Defines that the source RegistryObject contains the target RegistryObject.
EquivalentTo	Defines that the source RegistryObject is equivalent to the target RegistryObject.
Extends	Defines that the source RegistryObject inherits from or specializes the target RegistryObject.
Implements	Defines that the source RegistryObject implements the functionality defined by the target RegistryObject.
InstanceOf	Defines that the source RegistryObject is an Instance of the target RegistryObject.
Supersedes	Defines that the source RegistryObject supersedes the target RegistryObject.
Uses	Defines that the source RegistryObject uses the target RegistryObject in some manner.
Replaces	Defines that the source RegistryObject replaces the target RegistryObject in some manner.
SubmitterOf	Defines that the source Organization is the submitter of the target RegistryObject.
ResponsibleFor	Defines that the source Organization is responsible for the ongoing maintainence of the target RegistryObject.
OffersService	Defines that the source Organization object offers the target Service object as a service. Reserved for use in indicating that an Organization offers a Service.

TABLE II

SOME EXAMPLE DOCUMENT METADATA ATTRIBUTE DEFINITIONS

XDS Document Attribute	*Definition*	*ebRIM Attribute Type*
authorDepartment	Represents a specific department within a healthcare facility under which the human and/or machines authored the document.	Slot
classCode	The code specifying the particular kind of document (e.g. Prescription, Discharge Summary, Report). It is suggested that the XDS Affinity Domain draws these values from a coding scheme providing a coarse level of granularity	External Classification
classCodeDisplayName	The name to be displayed for communicating to a human the meaning of the classCode	Name attribute on Classification object, coding scheme name saved as codingScheme slot on Classification
parentDocument Relationship	The type of relationship that the document has with the parentDocument (e.g. Replace, addendum, or transformation).	Association Type
eventCodeList	This list of codes represents the main clinical acts, such as a colonoscopy or an appendectomy, being documented	External Classification
practiceSettingCode	The code specifying the clinical specialty where the act that resulted in the document was performed	External Classification
typeCode	The code specifying the precise kind of document (e.g. Pulmonary History and Physical, Discharge Summary, Ultrasound Report)	External Classification

In order to support atomic submission of documents to the registry and to make a permanent record in the registry of objects, XDS documents are submitted as "XDS SubmissionSet"s. The submission sets are also represented through *ebXML RegistryPackage* constructs and submitted by using *ebXML SubmitObjectsRequest*. An "XSD Submission Request" includes information on the metadata of the documents as well as the folders that the documents belong and/or new folders to be created.

A. IHE XDS Metadata

XDS specifies a set of predefined metadata elements to be associated with XDS documents, submission sets and folders to facilitate their discovery. Some of the metadata elements are straightforward document properties such as "authorDepartment" or "creationTime". Healthcare domain specific semantics is provided only through "classCode"s. A "classCode" is represented as an *External Classification* in ebXML registry. There are a number of native *External Classifications* in the ebXML registry implementation (OMAR [10]) such as North American Industrial Classification Scheme (NAICS) codes [16] and Universal Standard Products and Services Classification (UNSPSC) [21]. Other types of external classifications have to be introduced to the registry explicitly. As an example, Figure 2 shows how the Logical Observation Identifiers Names and Codes (LOINC) External Classification [15] can be introduced to the ebXML registry. After introducing such an External Classification to the registry, in order to use it, its UUID assigned by the registry is necessary. The query shown in Figure 3 is used to retrieve the UUID.

```
<SubmitObjectsRequest>
 <rim:RegistryObjectList>
  <rim:ClassificationScheme id="loinc" isInternal="false"
   nodeType="urn:uuid:bd0c092a-cb38-4578-8520-81274054f678">
    <rim:Name>
     <rim:LocalizedString charset="UTF-8" value="LOINC"/>
    </rim:Name>
    <rim:Description>
     <rim:LocalizedString charset="UTF-8" value="LOINC coding"/>
    </rim:Description>
  </rim:ClassificationScheme>
 </rim:RegistryObjectList>
</SubmitObjectsRequest>
```

Fig. 2. A SubmitObjectsRequest introducing the LOINC Term List to the ebXML registry

Table II gives some example metadata attributes for XDS Documents such as "authorDepartment", "classCode"

"classCodeDisplayName" and "parentDocument Relationship". Such metadata is defined in the registry through ebXML RIM semantic constructs. For example, "authorDepartment" is defined as a slot of the ExtrinsicObject representing an XDS Document in the registry. "parentDocument Relationship", on the other hand is defined as an "Association Type" which extends the predefined Association types of ebXML registry (Table I) with three new values, namely, "append" (APND), "replace" (RPLC) and "confirm" (XFRM).

```
<AdhocQueryRequest >
 <query:ResponseOption returnComposedObjects="true"
  returnType="LeafClassWithRepositoryItem"/>
 <rim:AdhocQuery id="tempId">
  <rim:QueryExpression queryLanguage=
   "urn:uuid:c26215e8-7732-4c7f-8b04-bd8115c325e9">
    SELECT * FROM CLASSSCHEME WHERE ID IN
      (SELECT PARENT FROM NAME WHERE VALUE LIKE 'LOINC')
  </rim:QueryExpression>
 </rim:AdhocQuery>
</AdhocQueryRequest>
```

Fig. 3. A Query to find the UUID of a External Classification submitted in Figure 2

Similarly, the metadata for submission sets, such as "authorDepartment" is defined as the slot of the "ebXML RegistryPackage" object; "contentTypeCode" whose values are drawn from a vocabulary defined by the affinity domain use External Classification. Here, a submission set is related with a node in the External Classification by using "ebXML Classification". As an example, "MyXDSSubmissionSet" is related with "LOINC's Hospital Discharge Summary Note" as shown in Figure 4 by classifying "MyXDSSubmissionSet" through a Classification node whose nodeRepresentation is "34105-7" (corresponding to LOINC's [15] Hospital Discharge Summary Note code) and whose classificationScheme value is "urn:uuid:c82d4b0b-d042-4d49-a5d1-8ee5153aea9f" which is the UUID obtained as a result of the query shown in Figure 3.

```
<SubmitObjectsRequest >
 <rim:RegistryObjectList>
  <rim:RegistryPackage id = 'MyXDSSubmissionSet'>
   <rim:Name> <rim:LocalizedString value = 'Example '/>
   </rim:Name> </rim:RegistryPackage>
  <rim:Association id = 'hasMember' associationType =
  'urn:uuid:2d03bffb-f426-4830-8413-bab8537a995b'
  sourceObject = 'MyXDSSubmissionSet'
  targetObject = 'LabResuts'/>
  <rim:ExtrinsicObject id="LabResults" mimeType="text/xml">
    <rim:Name> <rim:LocalizedString value = 'ExampleLR'/>
    </rim:Name>
  </rim:ExtrinsicObject>
  <rim:Classification nodeRepresentation="34105-7"
    classifiedObject="MyXDSSubmissionSet"
    id="LOINC's Hospital Discharge Summary Note"
    classificationScheme=
    "urn:uuid:c82d4b0b-d042-4d49-a5d1-8ee5153aea9f" />
 </rim:RegistryObjectList>
</SubmitObjectsRequest>
```

Fig. 4. An Example to classifying a Submission Set with an External Classification

Folder metadata include attributes such as "codeList", and "codeDisplayName List" which are handled as in submission sets.

These metadata sets are basic but they are not sufficient to address interoperability of clinical documents in federated clinical affinity domains.

B. XDS Registry Adaptor

There are certain functionality required by the IHE XDS specification that is not supported by ebXML registry specification. For example, XDS requires the submitted metadata to be validated. As another example, the submission to the registry must be an atomic operation. Such functionality that is not available in the ebXML registry standard, is provided by the XDS Registry Adaptor.

C. Querying the Registry

The metadata defined for XDS documents, folders and submission sets can be used to retrieve the desired documents by querying the registry through SQL. Example queries include retrieving specific types of documents of a patient for a time interval, or by author person, or all documents in a folder or a submission set.

All queries return either metadata for one or more registry objects, or object references for one or more registry objects (registry UUIDs). To query the registry, the user has to know the underlying relational schemas of the ebXML registry specification.

Consider for example the query given in Figure 5 which retrieves all approved Hospital Discharge Summary Notes of a patient whose patient id is '123'. The schemas of the relations accessed in this query are given in Table III. In this query, the LOINC External Classification [15] is used which has been introduced to the registry as shown in Figure 2. When the Classification instance represents such an external classification, then the nodeRepresentation attribute is required. The "classificationScheme" value is "urn:uuid:c82d4b0b-d042-4d49-a5d1-8ee5153aea9f" which is the UUID obtained as a result of the query shown in Figure 3. The value of "nodeRepresentation" attribute is 34105-7 from LOINC which denotes "Hospital Discharge Summary Note".

```
SELECT * FROM ExtrinsicObject doc
WHERE
doc.id in (SELECT doc.id FROM ExtrinsicObject
doc, ExternalIdentifier ei
WHERE
 doc.objectType=XDSDocumentEntry AND
 ei.identificationScheme=XDSPatientId AND
 ei.registryObject=doc.id AND
 ei.value='123') AND
 doc.id in
 (SELECT classifiedObject FROM Classification
  WHERE classificationScheme=
  'urn:uuid:c82d4b0b-d042-4d49-a5d1-8ee5153aea9f' AND
  nodeRepresentation='34105-7') AND
  doc.Status='Approved'
```

Fig. 5. An Example XDS SQL Query to retrieve all discharge summaries for a patient

IV. WEB ONTOLOGY LANGUAGE (OWL)

Web Ontology Language (OWL) [22] is a semantic markup language developed by the World Wide Web consortium for publishing and sharing ontologies. OWL is based on the Resource Description Framework (RDF) [19].

OWL describes the *structure* of a domain in terms of *classes* and *properties*. Classes can be names (URIs) or *expressions* and the following set of *constructors* are provided for

TABLE III
EBXML RELATIONAL SCHEMAS

ExtrinsicObject(id, home, objectType, status, expiration, majorVersion, minorVersion, stability, userVersion, isOpaque, mimeType)
ExternalIdentifier(id, home, objectType, status, registryObject, identificationScheme, value)
Classification(id, home, objectType, status, classificationNode, classificationScheme, classifiedObject, nodeRepresentation)
ClassificationNode(accessControlPolicy, id, objectType, code, parent, path)
Association(accessControlPolicy, id, objectType, associationType, sourceObject, targetObject, isConfirmedBySourceOwner, isConfirmedByTargetOwner)
Name_(charset, lang, value, parent)

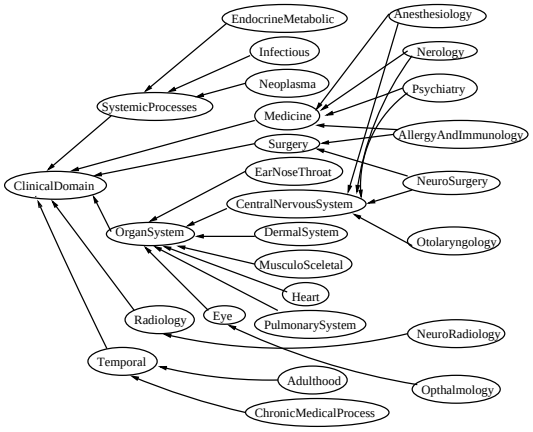

Fig. 6. An Example Clinical Domain Ontology

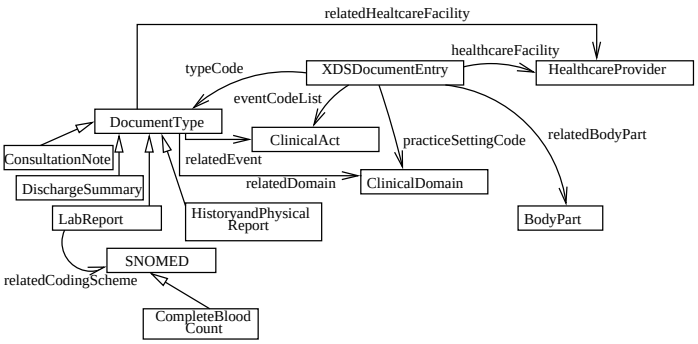

Fig. 7. An Example Document Entry Metadata Ontology

building class expressions: *owl:intersectionOf, owl:unionOf, owl:complementOf, owl:oneOf, owl:allValuesFrom, owl:someValuesFrom, owl:hasValue*.

In OWL, properties can have multiple domains and multiple ranges. Multiple domain (range) expressions restrict the domain (range) of a property to the intersection of the class expressions.

Another aspect of the language is the axioms supported. These axioms make it possible to assert subsumption or equivalence with respect to classes or properties [4]. The following are the set of OWL axioms: *rdfs:subClassOf, owl:sameClassAs, rdfs:subPropertyOf, owl:samePropertyAs, owl:disjointWith, owl:sameIndividualAs, owl:differentIndividualFrom, owl:inverseOf, owl:transitiveProperty, owl:functionalProperty, owl:inverseFunctionalProperty*.

V. SEMANTIC ANNOTATION OF CLINICAL DOCUMENTS IN IHE XDS

As described in Section III, in IHE XDS, the basic metadata elements are defined in affinity domain specific coded terms. Therefore, given the coded terms in one domain, it is very difficult to discover documents in another domain. To overcome this problem we propose to use ontologies to describe the semantics of clinical documents. An ontology is machine processable since it conforms to a formal, well-defined syntax. In this way, even when different XDS domains use different ontologies, it is possible to formally translate the concepts.

Additionally, when coding terms are used to define metadata elements as proposed by IHE XDS, some dependencies and overlappings between the ranges of the different metadata elements become difficult to handle:

- There are a number of inheritance relationships between the values of metadata elements. For example, "eventCodeList" attribute of XDSDocumentEntry can be a further specialization of the attribute "typeCode". Hence, for example, if the "typeCode" of a document is set to be "History and Physical Report", then this document should be related with the "History and Physical" ClinicalAct through the "eventCodeList" attribute.
- There are constraints between the possible values of the metadata elements. For example, if one or more "eventCodes" are set for a document, they should not conflict with the values inherent in the "classCode", "practiceSettingCode" or "typeCode", since such a conflict would create an ambiguous situation.

These problems stem from the fact that, in IHE XDS, the range of the metadata elements are defined to be affinity domain specific coding terms which are taxonomies. Through taxonomies, only very limited amount of semantics can be provided: there is no way to describe the properties of nodes; no way to relate nodes with one another (except for the hierarchy) and no way to relate the properties of nodes. Therefore, when coding lists are used as metadata elements document coding becomes error prone.

On the other hand, all such useful information can be provided by ontologies. When metadata is defined through ontologies, it becomes possible to assist the user in an automated way to handle the constraints and dependencies. Additionally, even if different XDS domains use different ontologies, there is a possibility of a formal translation of concepts.

In Figure 7, we present a part of an example ontology that can be used for annotating IHE XDS DocumentEntries. It should be noted that our purpose is not to propose a clinical document ontology but rather to show how it can be exploited once it is specified by standard bodies. In this ontology, the

```
DocumentEntry(?x1) & DocumentType (?x2) & ClinicalAct(?x3) &
typeCode(?x1,?x2) & relatedEvent(?x2,?x3) =>
eventCodeList(?x1,?x3)
```

Fig. 8. An example rule

ranges of the metadata attributes defined in IHE XDS are defined as root classes of other ontologies, instead of plain taxonomies.

In this Document Metadata Ontology, the range of the "practiceSettingCode" property is defined to be the "ClinicalDomain" class. The "ClinicalDomain" class in Figure 6 is defined to have subclasses such as "OrganSystems", "SystemicProcesses", "Temporal", "Medicine","Surgery" and "Radiology". Leaf level clinical domains are defined as subclasses of one or more of these classes. As an example, "Neurosurgery" domain is defined as a subclass of both "CentralNervousSystem" (which is a subclass of "OrganSystems" class), and "Medicine" classes.

In the example Metadata Ontology presented in Figure 7, the range of "typeCode" property is defined to be the "DocumentType" class. The "DocumentType" class has subclasses such as "ConsultationNote" and "HistoryandPhysicalReport". These subclasses can be further specialized by relating them to specific clinical domains, clinical acts and healthcare facilities. We use OWL to define these ontologies.

In OWL, relationships between classes are formally defined through "objectProperties". For example, we can define a "relatedEvent" object property to associate "ClinicalActs" class with the "DocumentType" class. Through this object property, the "HistoryandPhysicalReport" DocumentType can be related with the "History and Physical" ClinicalAct class. When such relationships are formally defined through ontology languages, further constraints and dependencies can be enforced through rules. For example, the rule presented in Figure 8 states that if a DocumentEntry is related with a DocumentType through a "typeCode" property, and if this DocumentType is related with a ClinicalAct through a "relatedEvent" property, then this DocumentType should be automatically related with this specific ClinicalAct through the "eventCodeList" objectProperty. When such a rule is defined, and a DocumentEntry is annotated with, say, the "HistoryandPhysicalReport" DocumentType which is related with the "History and Physical" ClinicalAct, then the processing of this rule will automatically reveal that this DocumentEntry is related with the "History and Physical" ClinicalAct.

VI. Exploiting Metadata Ontologies in IHE XDS

In IHE XDS, the metadata is expressed through basic ebXML registry semantic mechanisms such as slots and association types as described in Section III-A and queried as described in Section III-C. It is possible to enhance this mechanism by storing clinical document metadata ontologies to the ebXML registry. In the following we describe how this enhanced semantics can be represented and accessed in ebXML registries:

- The ebXML registry allows classification hierarchies to be defined in the registry as explained in Section II-A. Therefore, when an OWL ontology needs to be stored in an ebXML registry, it must be expressed in terms of registry constructs. In [7], [8], we describe how OWL ontologies can be stored to ebXML registries. Basically, OWL classes are represented as "Registry Information Model (RIM) Classification Nodes" and OWL properties are represented as "RIM Associations". After the OWL constructs are parsed and converted into ebXML registry semantic constructs, a "SubmitObjectsQuery" is created and sent to the registry. An example classification hierarchy which is constructed as a result of such a query is presented in Figure 9 (a).
- As previously noted, clinical documents are represented in the Registry as a "RIM Extrinsic Object" as shown in Figure 9 (b). Note that "Extrinsic Objects" point to the Repository items where their contents are stored.
- In order to establish the relationship with clinical document "Extrinsic Objects" and the Document Metadata ontology stored in the ebXML registry, ebXML "Classification" objects are used. Once the Document Metadata Ontology is stored in the ebXML registry, and the rules are defined to represent the constraints as presented in Figure 8, the user can be guided with a graphical interface to annotate an XDS document entry as depicted in 9 (c). We have developed a graphical interface for guiding the user to create the metadata of the DocumentEntries before submitting them to the ebXML registry. This graphical tool, shown in Figure 10, presents the XDS Document Metadata Ontology stored in the ebXML Registry to the user, and guides her to annotate the XDS DocumentEntries with the Classification Nodes. While the user is annotating a document, the rules defined over the schema of the Metadata ontology are processed, and the user is guided accordingly. As presented in Figure 10, when

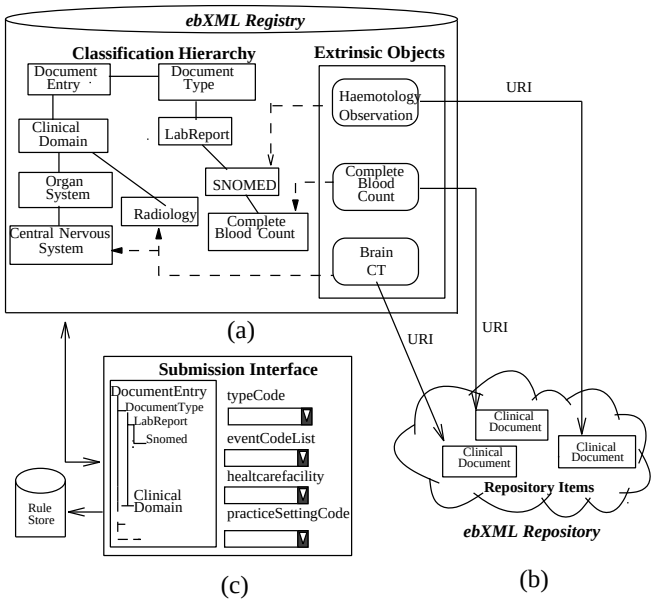

Fig. 9. Storing and Querying Clinical Document Semantics in ebXML Registry

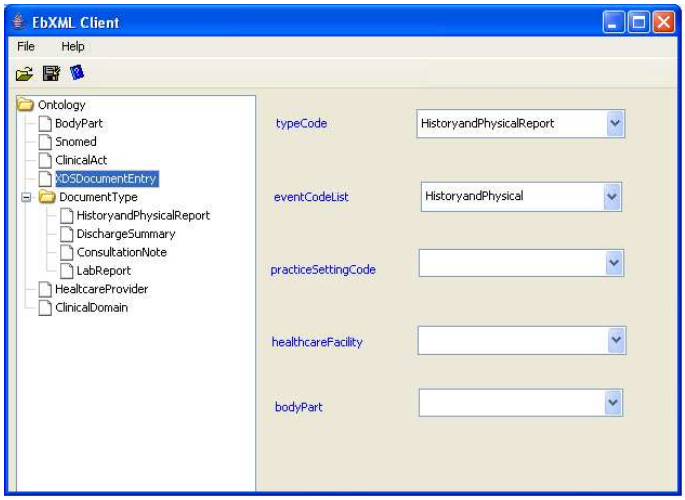

Fig. 10. The GUI for annotating submission sets

the user selects the "HistoryandPhysicalReport" as the typeCode which has been related with the "HistoryandPhysical" ClinicalAct, the "HistoryandPhysical" ClinicalAct is automatically added to the eventCodeList of the DocumentEntry through processing the rule presented in Figure 8.

Additionally as specified by IHE XDS, the inherited metadata should not conflict with the ones the user selects manually. For example, consider the case where a user selects the "ProcedureReport" DocumentType as a value of the typeCode, and relates this DocumentType with the "Neurology" ClinicalDomain with the "relatedDomain". Then, s/he may choose the "DermalSystem" ClinicalDomain as the value of the practiceSettingCode metadata of the XDS document element. In this case, the user is warned that the inherited and the selected ClinicalDomains conflict with each other. This is enabled since the graphical tool checks the class-subclass relationships between the ClinicalDomain ontology classes presented in Figure 6. Since there is no direct class-subclass relationship between "Neurology" and "DermalSystem", the user is warned. It should be noted that more complex conflict relationships between classes can be defined through objectProperties and rules.

The ebXML registry implementations store registry data in a relational database. While storing the ontology in the ebXML registry, we do not change the original relational schema in the specification. Therefore all this enhanced semantic can be accessed from the registry through SQL queries.

VII. Distributed Patient Identification for Federated IHE XDS

The underlying assumption of the XDS query model is that a document consumer already knows the patient ID of the patient for which documents are looked up. The precise way of how the XDS document consumer determines the patient ID is undefined – it is assumed that a master patient index is maintained in some way not defined in the scope of the XDS profile. The maintenance of a master patient index becomes a real challenge when it comes to federating XDS affinity domains. A manual creation and maintenance of an index that covers possibly millions of entries is hardly feasible. In this situation, only a probabilistic matching of patient records based on commonly available patient demographics is possible. This is a scenario that is not well supported by the IHE Patient Identifier Cross-referencing (PIX) integration profile that is usually combined with XDS when patient identifiers from multiple domains have to be resolved. Instead of using PIX, we propose the use of the Patient Identification Protocol (PID protocol) [1], [2], [9] that has been developed in the framework of the Artemis project [3]. The PID protocol resembles the IHE PIX integration profile in the sense that there is an actor called *PID consumer* that needs to determine the patient ID of a specific patient for some XDS affinity domain and there is an actor called *PID query manager* that maintains the necessary database to perform such a look-up – the corresponding actors in IHE PIX are called *PIX consumer* and *PIX manager*. The PID consumer actor would typically be grouped with an XDS document consumer. The main difference is that for the PID protocol the query is not based on a patient ID along with an identifier of the patient identification domain, but is based on patient demographics commonly available, including the patient's name, date and place of birth, etc. The protocol also introduces a few additional actors that support data protection requirements. It should be noted that the set of demographics available may vary, and that spelling errors in medical record archives are not uncommon and need to be accounted for. The PID protocol uses so-called control numbers along with semantic annotation and a probabilistic record linkage as a way of addressing the "fuzziness" of demographic data and at the same time preventing a premature communication of personal data.

A. Control Numbers, Record Linkage and Blocking Variables

Control numbers are a concept used in the epidemiological cancer registries in Germany to allow record linkage of anonymised records describing cancer cases [20]. The PID protocol uses this concept in a modified form. Control numbers are generated from a set of demographic values through the following series of processing steps:

- *Splitting:* The available demographics are split into fields that are later converted into different control numbers. As an example, the date of birth would typically be separated into its components for year, month and day.
- *Standardisation:* This step addresses character set issues. Standardisation would typically involve conversion of names to uppercase ASCII characters, zero-padding of numbers such as day and month of the birth date and the initialisation of unknown and empty fields with well-known constants.
- *Phonetic encoding:* Optionally, name components may be encoded with a language specific phonetic encoding such as the Soundex coding system, the Metaphone algorithm or for example the "Cologne phonetics" for the German language.

- *Message digest:* Each standardised and possibly phonetically encoded field in the set of demographics is subjected to an irreversible cryptographic message digest algorithm. However, dictionary based attacks are still possible, which explains the fifth and the final step.
- *Encryption:* Each message digest is encrypted with a secret encryption key that needs to be known by all parties in the protocol that generate control numbers to be compared with one another. Each encrypted message digest is called a control number.

Given a set of control numbers describing a query and a number of sets of control numbers describing the patients in an affinity domain, "matches" can be identified using *record linkage*. The PID protocol uses a probabilistic record linkage algorithm as described by [5]. This class of algorithms not only compares each pair of control numbers for equality, it also considers the significance of each control number. Basically, such an algorithm allows to identify the most "promising" matches from a larger set. Since it is not practical to create a set of control numbers for each patient for each incoming request, the protocol uses so-called blocking variables to reduce the number of patients for which control numbers have to be computed. Blocking variables are demographics which are available to the PID query manager in plaintext form. Different XDS affinity domains will typically use different, though certainly overlapping, sets of control numbers accounting for country or region specific aspects such as phonetic encoding or national unique patient identifiers. Since control numbers can only be compared for binary equality, it is of prime importance that all actors have a common understanding about the meaning of each control number and blocking variable. The use of ontology based semantic annotation allows to introduce an amount of flexibility into the protocol that is needed to make it work in a setting where different sets of control numbers are supported by different affinity domains. The use of control numbers in the PID query makes sure that a PID consumer cannot abuse the protocol to acquire information about a patient before access to the XDS registry has been granted. To prevent dictionary attacks against the control numbers, actors possessing the session key must not be granted access to control numbers generated by any other actor. For this reason the protocol introduces two additional actors between the PID consumer and the PID query manager. The actors and their transactions in slightly simplified form are shown in Figure 11.

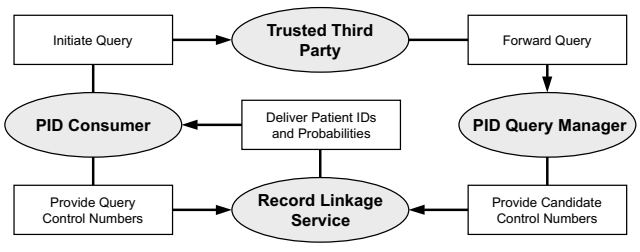

Fig. 11. Actors and Transactions in the PID Protocol (simplified)

Figure 11 shows the actors involved in the protocol as ovals and the transactions (i. e. message exchange between them) as boxes with an arrow indicating the direction of the information flow:

- *Record Linkage Service (RLS)*: an actor that receives a set of control numbers from the PID consumer and a possibly large number of sets of control numbers generated by the PID query managers. The RLS performs the probabilistic record linkage, identifies probable matches and reports them to the PID consumer. Since the RLS does not have access to the session key, it cannot re-identify the patient demographics.
- *Trusted Third Party (TTP)*: an actor that enables distribution of the query containing the session key and blocking variables among the possibly multiple PID query managers known to the federated XDS affinity domain.

B. Integrating XDS with PID

The integration of a federated XDS with the PID protocol would work in a very similar way as a combination of XDS with PIX. Each XDS document consumer requiring access to a "remote" XDS registry would be grouped with a PID consumer. Before issuing a query to the XDS registry, a matching patient ID for the corresponding affinity domain would be queried and selected based on the probability of the responses generated by the PID RLS actor. Each affinity domain would have to operate one PID query manager which could either receive all demographic information needed from the patient identity feed that is also provided to the XDS registry, or it could be directly grouped with the central patient identity source for the affinity domain. The TTP and RLS actors would typically be standalone actors provided centrally for each affinity domain or even for the federation of XDS affinity domains. In summary, access across affinity domain boundaries would be possible without the need for a manual creation and maintenance of a master patient index cross-referencing the patient identification domains of the multiple XDS affinity domains integrated in a federated XDS.

VIII. Conclusions and Future Work

In this paper, we describe how one of the major industry initiatives, namely, Integrating Healthcare Enterprise (IHE), Cross Enterprise Document Sharing (XDS) can be enhanced to support federated clinical affinity domains.

The contributions of the paper are as follows:

- IHE XDS has defined metadata element sets to be used in document registries to facilitate document discovery. Most of metadata elements are based on coded terms selected by the affinity domains. This makes it difficult to express the dependencies among the metadata elements and more importantly it becomes very difficult to locate documents across affinity domains.
  We show that when ontologies are used rather than the plain coding lists, the dependencies among the metadata elements can be handled in an automated way by defining the related rules. Furthermore, document discovery across affinity domains are faclitated through ontology mapping when metadata is defined through ontologies.

- When clinical affinity domains are federated, in order to identify a patient in another affinity domain, patient identification across affinity domains must be addressed. Since a master patient index that spans multiple affinity domains would be quite difficult to create and maintain, a look-up of patient IDs based on patient demographics such as name, birthdate, place of birth etc. is needed. For this purpose, we introduce a patient identification protocol that can be combined with XDS to provide a look-up of patient IDs based on demographics in a secure and fault-tolerant manner.

The work described in this paper is realized within the scope of the Artemis project [3]. As a future work, we intend to provide business process support across affinity domains by supporting IHE workflows through ebXML Business Process specification.

REFERENCES

[1] Aden, T, Eichelberg, M., ARTEMIS Deliverable D5.1.1: Relevant Electronic Healthcare Record Standards and protocols for accessing medical information, http://www.srdc.metu.edu.tr/webpage/projects/artemis/calendar-.php.

[2] Aden, T., Eichelberg, M., Thoben, W., "A fault-tolerant cryptographic protocol for patient record requests",in the Proceedings of EuroPACS-MIR 2004 in the enlarged Europe, Trieste(Italy), September 2004.

[3] The Artemis Project, http://www.srdc.metu.edu.tr/webpage/projects-/artemis

[4] Baader, F. , Horrocks, I., Sattler, U., "Description Logics", Handbook on Ontologies, 2004.

[5] Blakely, T., Salmond, C., "Probabilistic record linkage and a method to calculate the positive predictive value", in International Journal of Epidemiology Vol. 31 (2002), pp. 1246-1252, Oxford University Press.

[6] Dogac, A., Laleci, G. B., Kabak, Y., Cingil, I., "Exploiting Web Service Semantics: Taxonomies vs. Ontologies", IEEE Data Engineering Bulletin, Vol. 25, No. 4, December 2002.

[7] Dogac, A., Kabak, Y., Laleci, G., "Enriching ebXML Registries with OWL Ontologies for Efficient Service Discovery", in Proc. of RIDE'04, Boston, March 2004.

[8] Dogac, A., Kabak, Y., Laleci, G., Mattocks, C., Najmi, F., Pollock, J., "Enhancing ebXML Registries to Make them OWL Aware",Distributed and Parallel Databases Journal, Springer, to appear. http://www.srdc.metu.edu.tr/webpage/publications/2004/_DAPD_ebXML-OWL.pdf.

[9] Eichelberg, M., Aden, T., and Thoben, W., "A Distributed Patient Identification Protocol based on Control Numbers with Semantic Annotation", submitted for publication.

[10] freebXML (OMAR) implementation, http://ebxmlrr.sourceforge.net/-3.0/index.html

[11] ebXML Registry Information Model v2.5, http://www.oasis-open.org/-committees/regrep/documents/2.5/specs/ebRIM.pdf

[12] HL7 Clinical Document Architecture, Release 2.0. PDF format. Version: August 30, 2004, http:/xml.coverpages.org/CDA-20040830v3.pdf

[13] IHE IT Infrastructure Integration Profiles, http://www.rsna.org/IHE/tf/-ihe_tf_index.shtml

[14] IHE IT Infrastructure Technical Framework, Supplement 2004-2005: Cross-Enterprise Clinical Documents Sharing (XDS),IHE ITI Technical Committee, Trial Implementation Version, August 15, 2004.

[15] Logical Observation Identifiers Names and Codes (LOINC), http://-www.loinc.org/

[16] North American Industrial Classification Scheme (NAICS) codes http://www.naics. com.

[17] openEHR Community, http://www.openehr.org/

[18] prEN 13606-1, Health informatics - Electronic health record communication - Part 1: Reference model, Draft for CEN Enquiry, CEN/TC 251 Health Informatics, European Committee for Standardization, Brussels, Belgium, July 2004.

[19] RDF Semantics, http://www.w3.org/TR/rdf-mt/

[20] Thoben, W., Appelrath, H.-J., Sauer, S., "Record linkage of anonymous data by control numbers", in From Data to Knowledge: Theoretical and Practical Aspects of Classification, Data Analysis and Knowledge Organisation, Oldenburg (1994), pp. 412-419, Springer.

[21] Universal Standard Products and Services Classification (UNSPSC), http://eccma.org/unspsc

[22] Web Ontology Language OWL, http://www.w3.org/TR/owl-features/