ACHIEVING ELECTRONIC HEALTHCARE RECORD (EHR) INTEROPERABILITY ACROSS HEALTHCARE INFORMATION SYSTEMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÖZGÜR KILIÇ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING

JUNE 2008

Approval of the thesis

ACHIEVING ELECTRONIC HEALTHCARE RECORD (EHR) INTEROPERABILITY ACROSS HEALTHCARE INFORMATION SYSTEMS

submitted by **ÖZGÜR KILIÇ** in partial fullfillment of the requirements for the degree of **Doctor of Philosophy in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Volkan Atalay _____
Head of Department, **Computer Engineering**

Prof. Dr. Asuman Doğaç _____
Supervisor, **Computer Engineering Dept., METU**

Examining Committee Members:

Prof. Dr. İsmail Hakkı Toroslu _____
Computer Engineering Dept., METU

Prof. Dr. Asuman Doğaç _____
Computer Engineering Dept., METU

Prof. Dr. Mehmet Reşit Tolun _____
Computer Engineering Dept., Çankaya University

Prof. Dr. Özgür Ulusoy _____
Computer Engineering Dept., Bilkent University

Assoc. Prof. Dr. Nihan Kesim Çiçekli _____
Computer Engineering Dept., METU

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Özgür Kılıç

Signature :

ABSTRACT

ACHIEVING ELECTRONIC HEALTHCARE RECORD (EHR) INTEROPERABILITY ACROSS HEALTHCARE INFORMATION SYSTEMS

Kılıç, Özgür

Ph.D., Department of Computer Engineering

Supervisor: Prof. Dr. Asuman Doğaç

June 2008, 129 pages

Providing an interoperability infrastructure for Electronic Healthcare Records (EHRs) is on the agenda of many national and regional eHealth initiatives. Two important integration profiles have been specified for this purpose: the "IHE Cross-enterprise Document Sharing (XDS)" and the "IHE Cross Community Access (XCA)". XDS describes how to share EHRs in a community of healthcare enterprises and XCA describes how EHRs are shared across communities.

However, currently no solution addresses some of the important challenges of cross community exchange environments. The first challenge is scalability. If every community joining the network needs to connect to every other community, this solution will not scale. Furthermore, each community may use a different coding vocabulary for the same metadata attribute in which case the target community cannot interpret the query involving such an attribute. Another important challenge is that each community has a different patient identifier domain. Querying for the patient identifiers in another community using patient demographic data may create patient privacy concerns. Yet another challenge in cross community EHR access is the EHR interoperability since the communities may be using different EHR content standards.

In this thesis, we address each of these challenges and show how they can be handled in a super-peer based peer-to-peer architecture. Furthermore, we provide details of a solution

for the interoperability of EHR structure and content. We describe how two different EHR standards derived from the same Reference Information Model can be mapped to each other by using archetypes, Refined Message Information Model derivations and semantic tools.

Keywords: healthcare standards, semantic interoperability, peer-to-peer networks

ÖZ

SAĞLIK BİLGİ SİSTEMLERİ ARASINDA ELEKTRONİK SAĞLIK KAYITI BİRLİKTE İŞLERLİĞİNİN ELDE EDİLMESİ

Kılıç, Özgür

Doktora, Bilgisayar Mühendisliği Bölümü

Tez Yöneticisi: Prof. Dr. Asuman Doğaç

Haziran 2008, 129 sayfa

Elektronik sağlık kayıtları için birlikte işlerliğin sağlanması, birçok ulusal ve yöresel elektronik sağlık girişiminin gündemindedir. Bu amaçla iki önemli tümleştirme profili "IHE Cross-enterprise Document Sharing (XDS)" ve "IHE Cross Community Access (XCA)" tanımlanmıştır. XDS, elektronik sağlık kayıtlarının sağlık kurumlarından oluşan bir topluluğun içerisinde nasıl paylaşılacağını tanımlar ve XCA ise bu topluluklar arası paylaşımın nasıl yapılacağını tanımlar.

Bu çözümler, topluluklar arası veri alış verişi ortamındaki bazı önemli zorluklara hitap etmemektedir. Birinci zorluk ölçeklenebilirliktir. Eğer bir topluluğun ağa katılırken bütün diğer topluluklara bağlanması gerekirse, bu çözüm ölçeklenemez. Üstelik, her bir topluluk aynı yardımcı veri özelliği için farklı bir kodlama kelime dağarcığı kullanabilir ve bu durumda diğer topluluk böyle bir veri özelliğini yorumlayamaz. Başka bir önemli zorluk ise her topluluğun farklı bir hasta tanılayıcı tanım kümesine sahip olmasıdır. Hasta demografi bilgilerini kullanarak diğer topluluğun hasta tanılayıcısını sorgulamak hasta kişisel gizliliğiyle ilgili kaygılara neden olabilir. Farklı topluluklar, farklı elektronik sağlık kayıt içerik standardı kullanabilmesinden dolayı topluluklar arası erişimdeki başka bir zorluk da elektronik sağlık kayıtı birlikte işlerliğidir.

Bu tezde, bu zorlukların süper-görevdeş tabanlı görevdeş öğeler mimarisinde nasıl ele alınabileceğini gösteriyoruz. Ayrıca, elektronik sağlık kayıtlarının yapı ve içeriğinin birlikte işler-

liğinin çözümünün detaylarını sağlıyoruz. Aynı Reference Information Model'den türetilmiş iki farklı elektronik sağlık kayıt standardının Archetype, Refined Message Information Model türetmeleri ve anlamsal araçlarla birbirlerine nasıl eşlemlenebildiğini gösteriyoruz.

Anahtar Kelimeler: sağlık standartları, anlamsal birlikte işlerlik, görevdeş öğeler mimarisi

ACKNOWLEDGMENTS

First of all, I wish to express my deepest gratitude to my supervisor Prof. Dr. Asuman Dogaç for guiding and supporting me through the long Ph.D. process in a way that has sharpened my skills as a researcher. I really appreciate her for her ability to find an appropriate time slot for me in her busy schedule. Without her guidance, encouragement and motivation this work would never been possible.

I would also like to thank Assoc. Prof. Dr. Nihan Kesim Çiçekli and Prof. Dr. Mehmet Reşit Tolun for their valuable suggestions and comments throughout the steering meetings of this study.

I also acknowledge Gökçe Banu Laleci, Yıldıray Kabak and all other fellows at the Software Research and Development Center for the cooperation and support they have provided.

Finally, I would like to thank my dear wife, Burçin, who lived every moment of my Ph.D process with me. Her presence in my life has been my greatest strengths to complete this work. Without her continuous support and encouragement, I would have never had the strength to complete this work.

This thesis is dedicated to my dear wife Burçin and our lovely son Ozan Ege

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES

LIST OF TABLES

TABLES

CHAPTER 1

INTRODUCTION

Most of the health information systems today are proprietary and often only serve one specific department within a healthcare institute, and furthermore, a patient often receives medical treatment from different health professionals in various organizations, which do not interoperate [7]. It is important to share patient healthcare information through Electronic Healthcare Records (EHRs) to increase the quality of care and to decrease its cost. Therefore, providing an interoperability infrastructure for EHRs is on the agenda of many national and regional initiatives, such as Federal Health IT Initiatives in the United States [31], the Health Infoway project in Canada [14] and several national projects in the European Union as summarized in [78].

At the moment, EHR information is stored in all kinds of proprietary formats including relational database tables, structured document-based storage in various formats and unstructured document storage such as digitized hardcopies maintained in a classical document management system through various healthcare information systems existing on the market [28]. This results in severe interoperability problems.

To address the EHR interoperability problem, there are several EHR content standards currently under development which aim to provide standard interfaces to existing proprietary systems. The standardization efforts include the Health Level Seven (HL7) Clinical Document Architecture (CDA) [37], the European Committee for Standardization (CEN) EN 13606-1 (is referred to as EHRcom in the rest of this thesis) [16] and the openEHR [68]. Such standards define the structure and the markup of the clinical content to make EHR exchange interoperable. However, having more than one standard introduces the interoperability problem among institutes using different standards.

To address the interoperability problem in exchanging clinical documents across enterprises, an important initiative named "Integrating Healthcare Enterprise (IHE)" [50] has

specified a "community-based" integration profile, called "Cross-enterprise Document Sharing (XDS)" [52]. In the IHE XDS Integration Profile, healthcare enterprises that agree to work together for clinical document exchange decide on a common set of policies such as how the patients are identified, the EHR document formats to be used and the common set of coding terms (vocabularies) to represent the metadata of the documents. IHE XDS is document neutral, that is, enterprises in an XDS affinity domain themselves decide on which document format to use such as HL7 CDA, or EHRcom or Portable Document Format (PDF).

This community based approach addresses the needs of Regional Health Information Organizations (RHIOs) but larger scale integration efforts require an enhanced approach since it is not realistic to expect communities belonging to different administrative domains to agree on a single common set of policies. On the other hand, a single community based approach also has a scalability problem because of its centralized architecture.

There are efforts to federate XDS affinity domains [24], [49], that is, to allow systems from different administrative and technical domains to interoperate. A recent effort in this direction is the "IHE Cross Community Access (XCA)" [48] Integration Profile which defines a protocol to query and retrieve patient related healthcare data across communities. The protocol defines the transactions and actors between connected communities. However, the current version of the integration profile does not address the following challenges:

- Any solution providing interoperability among communities should address the network topology. For example, if every community joining the network needs to connect to every other community, that is, a pure peer-to-peer network, this solution will not scale. The challenge is that although any of the communities may contain the information another community needs, the topology should avoid a need for connecting every community to every other since this would increase the configuration complexity of the network. Otherwise, for each new community joining the network each existing community should update its configurations. Any solution to the network topology problem should instead minimize the configuration cost and should also prevent flooding by avoiding a search through all the communities, but it should still be possible to locate the information needed. IHE XCA mentions this problem stating that the Initiating Gateway must determine which Responding Gateways a request should be sent to, but how this will be achieved is left as a future work.

- Handling vocabulary differences across communities is an important issue which has

to be resolved in a cross community exchange environment. Each community may use different coding vocabulary for the same metadata attribute. If such attributes exist in a cross community query, the target community might not be able to understand the values of these attributes. Therefore, before execution of a query in the target community, the values of the attributes should be converted to corresponding attribute values used in the target community. If a mapping of the source coding vocabulary to the target coding vocabulary is provided, such a conversion is possible. However, if the number of communities in the network increases, building up and maintaining each mapping between community pairs becomes a challenge. Furthermore, before a new community joins the network, it would have to provide a vocabulary mapping for each community in the network. In other words, as the number of communities in the network increases, it becomes very difficult for a new community to join the network.

- Another important challenge in a cross community environment is the identification of the patients. The use of patient identifiers is the accepted practice within communities. However, each community may (and typically will) have a different patient identifier domain. Therefore, finding the corresponding patient identifier to be used in the target community is an important issue.

  IHE XCA Integration Profile describes two models for resolving the patient identity in a cross community exchange environment. In the first model, a master patient index (called "PIX Manager" in IHE terminology) cross-references the patient identifiers between communities. The second model relies on the existing IHE Patient Demographics Query (PDQ) Integration Profile. Prior to sending a cross community query, a Patient Demographics Query should be sent to the target community to obtain the patient identifier of the target community. This approach requires a significant number of policy decisions to be in place, coordinated with privacy consents in a cross community environment [48].

- IHE XDS is content format neutral, that is, enterprises in a community can decide which document formats to use such as the HL7 CDA, EHRcom or PDF. Therefore, another challenge in cross community EHR access is the EHR interoperability since the communities may be using different EHR standards.

This thesis addresses these challenges by proposing a scalable super-peer based peer-to-peer architecture to achieve interoperability among healthcare communities. The system

described in this thesis is realized within the scope of the RIDE Project [80] which addresses the interoperability of eHealth systems with special emphasis on semantic interoperability. The specific contributions of the work described in this thesis are as follows:

- *A scalable super-peer based peer-to-peer architecture to achieve interoperability among healthcare communities:* The IHE XCA Integration Profile has a decentralized architecture where each community provides a set of predefined services. However, as the number of communities increases, managing the configuration of community pairs becomes a challenge. In order to overcome the complexity introduced by one-to-one connections of communities, we propose a scalable peer-to-peer community network architecture based on the super-peer model described in [67].

- *Routing requests based on Privacy Preserving Control Numbers:* In order to route requests to a community which has the information for a given patient, the community should register its patients to its super-peer. When a super-peer uses patient-community relations to build its routing indices, it is possible to route requests only to the relevant communities. Since a patient has different patient identifiers in different communities, registering patients through their local patient identifiers is not suitable. On the other hand, using patient demographic data for identifying a patient creates concerns about privacy.

  The mechanism we propose for handling this problem is to use so-called control numbers instead of patient identifiers. Control numbers have previously been used for preserving patient privacy in other contexts [2], [4], [29]. Furthermore, control numbers enable fuzzy matching of patient data as demonstrated in the PID protocol [29]. The matching algorithm in super-peers not only compares the control number from the request against the control numbers existing in indices for equality, it also allows to identify the most "promising" matches from a larger set.

- *Vocabulary translation for community specific metadata attributes:* IHE XDS Integration Profile describes attributes used in defining metadata of documents. Some of these attributes are community specific that is in a IHE XCA Profile implementation each community may use different coding schemes to assign values to these attributes. Since the documents in an XDS community are described and queried through these attributes, a condition specifying a value for such an attribute needs to be modified in the case of cross community queries.

The proposed architecture achieves these modifications through performing vocabulary translations between the source and target communities for community specific metadata attributes. To handle vocabulary translations, we introduce an actor named Vocabulary Translator that stores vocabulary mappings between the local vocabulary of one community and the vocabulary of its super peer. In this way, when a community wishes to join a network, it provides only a single vocabulary mapping with its super-peers vocabulary.

- *Design and implementation of clinical statement interoperability between EHR standards:* Communities in a federation should be able to exchange information which implies that their EHR documents should be interoperable. Therefore this work proposes a solution to achieve semantic interoperability between healthcare information systems and this solution supports multiple EHR standards by providing transformations from one clinical model to another. An EHR includes clinical statements such as observations, laboratory tests, diagnostic imaging reports, treatments, therapies, drugs administered, and allergies. Formally, a clinical statement is an expression of a discrete item of clinically related information that is recorded because of its relevance to the care of a patient [46]. The thesis demonstrates the interoperability by showing how to transform the clinical statement instances of one EHR standard to another. In order to address this need, we tackle the interoperability of EHR standards which are derived from the HL7 Reference Information Model (RIM) [41]. Then we demonstrate how HL7 CDA and EHRcom clinical statement instances are mapped to each other by using archetypes and semantic tools and techniques. As a result it becomes possible to achieve interoperability through the federation of XDS Affinity Domains where each affinity domain may use different EHR standards for describing the content of clinical documents.

  A reference information model, like the HL7 RIM, defines a generic structure to express the concepts in a domain. This generic reference information model is then refined to subdomains and later to specific domain concepts. For example, HL7 RIM is used to derive the Domain Message Information Model (D-MIM) through a refinement process [42] where only the required classes, attributes, relationships for building the messages for a particular domain are included. The next step is to build the Refined Message Information Model (R-MIM) by including the necessary classes, attributes and associations used in a set of messages for a particular subdomain.

As an example, HL7 RIM can be specialized into "Clinical Document Architecture" for expressing clinical documents; "Clinical Genomics" for expressing clinical and personalized genomics data and "Claims and Reimbursement" for handling claims and reimbursements. Defining a generic RIM and specializing it to subdomains makes it possible for the RIM to stay static and stable, and the concepts derived in R-MIMs can be traced back to the RIM.

The approach taken in CEN recognizes the importance of EHR interoperability. Therefore, the possibility to represent the constructs of the CEN Reference Model as classes and attributes of the HL7 RIM is ensured [16]. For this purpose, CEN has produced a D-MIM correspondence of its reference model by deriving it from the HL7 RIM. The EHRcom R-MIM can be generated in a similar way as described in HL7 Development Framework (HDF) [39]. This is an ongoing work by the standard bodies. Therefore, we generate the EHRcom R-MIM used in this work by including the necessary classes, attributes and associations defined in the EHRcom D-MIM.

In the thesis, we show how to transform the clinical statement instances between EHR standards by using semantic mechanisms based on the R-MIM derivations and archetypes. We base our argument on the following facts:

- **Deriving R-MIMs**: The R-MIM of EHRcom is derived from the HL7 RIM by including the necessary classes, attributes and associations from the D-MIM. The derivation rules are described in Section 4.1.1. In other words, the R-MIMs of both of these EHR standards, namely, EHRcom and HL7 CDA, are obtained from the same RIM. It is important to note that an R-MIM is derived from the RIM through a set of well-defined rules [42].
- **Mapping through R-MIMs**: When R-MIMs of both EHRcom and HL7 CDA are expressed through Web Ontology Language (OWL), the class property mappings of two ontologies can be obtained by using the derivation rules of the R-MIMs. As a simple example, both the "Element" class in the R-MIM of EHRcom and the "ObservationMedia" class in the HL7 CDA R-MIM are derived from the "Observation" class of the HL7 RIM. The "Observation" class of the HL7 RIM has a "value" property. Based on the derivation rules, when mapping a class specializing the EHRcom "Element" to a class which specializes the HL7 CDA "ObservationMedia", the "value" property of the source class can be automatically mapped to the "value" property of the target class.

A RIM subdomain such as "Clinical Statement" is still too generic to express clinical concepts. Therefore, given a class in the source ontology, the corresponding class in the target ontology is not clear unless the content is known. For example, an instance of a class "Entry" conforming to EHRcom corresponds to one of the instances of Act, Encounter, Observation, ObservationMedia, Organizer, Procedure, RegionOfInterest, SubstanceAdministration or Supply classes in HL7 CDA depending on the coded value assigned to the instance of class "Entry".

Furthermore, if healthcare institutes structure their clinical documents differently, these documents may not be interoperable even when they conform to the same EHR standard. For example, both EHRcom and HL7 CDA have a class named "Section", and sections can have nested sections. When the sections of a clinical document are organized differently in a source instance, then generating the same hierarchy in the target instance would not create the meaning intended in the source document.

In order to express more specialized semantics, there is a need to use "archetypes". An archetype is a reusable, formal expression of a distinct, domain-level concept such as "blood pressure", "physical examination", or "laboratory result" expressed in the form of constraints on data whose instances conform to some reference information model [10]. In other words, an archetype specializes an information model concept. For example, when the archetype reference model is chosen to be HL7 CDA R-MIM, archetypes can be specified by placing constraints on the attributes of the HL7 CDA R-MIM "Observation" class, to define concepts like "Heart Rate" or "Penicillin Allergy".

In this work, we show how to transform the clinical statement instances between EHR standards by using semantic mechanisms based on the R-MIM derivations and archetypes. The methodology provides semi-automatic mapping of archetypes. Once the archetypes are mapped, clinical statement instances are transformed automatically given that they use a common terminology system.

- *Representing Archetypes through Web Ontology Language(OWL):* We believe that the archetype concept proves to be a powerful mechanism in mapping different EHR content semantics to each other for the following reason: Ontology mapping, in fact, involves introducing constraints when mapping the source ontology concepts to the target ontology concepts. For example, when we want to map a "Person" concept in a source ontology, to a "Female_person" concept in the target ontology, we place a constraint on the gender attribute. On the other hand, such constraints used in the se-

mantic mapping come prepackaged with the archetypes because archetypes themselves define constraints to express semantics. Hence, in this work, archetypes are used in order to map clinical statement concepts between two ontologies namely, HL7 CDA R-MIM and EHRcom R-MIM.

Since the archetypes are defined through a formal language, called Archetype Definition Language (ADL) [9], we need to describe how to map ADL into OWL. In [43] and [68], the OWL representations of some archetypes are given. However, we feel it necessary to give a formal description of how the ADL constructs can be mapped to OWL.

1.1 Outline

The thesis is organized as follows: Chapter 2 gives a brief overview of the enabling standards and technologies, namely, the peer-to-peer networks, healthcare standards and architectures including IHE XDS, IHE XCA, HL7, EHRcom, OpenEHR archetypes and OWL. In Chapter 3, we describe the details of the super-peer based peer-to-peer architecture to implement the IHE XCA Integration Profile. Chapter 4 proposes the architecture in which the clinical statements of HL7 CDA and CEN EHRcom can be transformed into each other's representation. Chapter 5 describes how archetype constraints are represented in OWL. Chapter 6 gives details about the implementation of the clinical statement interoperability architecture. Chapter 7 presents the related work. Finally, Chapter 8 concludes the thesis and presents the future work.

CHAPTER 2

BACKGROUND ON ENABLING TECHNOLOGIES AND STANDARDS

In this chapter, the standards and technologies used in the work explained in this thesis are briefly summarized.

2.1 Peer-to-Peer Systems

Peer-to-peer systems are becoming a prominent solution to overcome IT resource limitations caused by centralization. Decentralization can provide improvements in scalability since it eliminates the bottleneck caused by centralized operations. In general, each peer is both a provider and a consumer in the network and supports almost the same functionalities. Gnutella [34], which is a file sharing network, is an example of a decentralized peer-to-peer system.

In a fully decentralized peer-to-peer system, a peer can perform a search by sending the request to each node it is connected to. Each node receiving the request then forwards the request to all the nodes it is connected to. The request is propagated until a predetermined number of "hops" reached.

However, a fully decentralized peer-to-peer system introduces some challenges. A peer cannot easily gather the global view of all the peers and locate information in the network. Looking up the desired information requires many messages to be forwarded in the network which results in "network flooding". To improve the performance of peer-to-peer networks, hybrid architectures have been introduced. In these architectures, certain peers receive specific responsibilities to handle the routing of requests by providing directory services. For example, Napster [66] uses a single central server that stores the directory of peers and objects for discovery and routing. However, unlike Gnutella, Napster can be down by

shutting down single node since the entire network relies on a central server and this single node becomes a performance and scalability bottleneck.

A well established network topology in peer-to-peer networks preventing flooding is the super-peer model. In the super-peer model, special peers called super-peers provide directory services for their connected peers known as leaf nodes. The KaZaA network [56] is based on a simple version of a super-peer based architecture. In a super-peer network, a peer submit queries to its super-peer and receive results from it. Super-peers are connected to each other and responsible for forwarding requests and responses of their connected peers through the network. Figure 2.1, illustrates an example super-peer network where super-peers are labeled with "SP" and ordinary peers are labeled with "P".

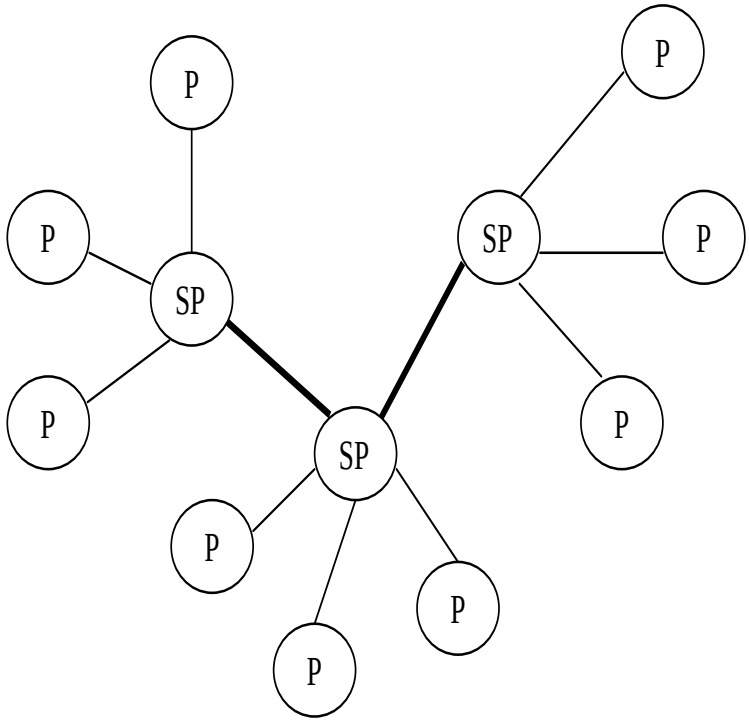


Figure 2.1: Super-Peer Network

In a super-peer network, a peer has to describe its resources during the registration of the peer to the network and maintain them when necessary to prevent flooding. This can be achieved using the routing indices located in super-peers. Super-peers perform routing by matching the conditions in the requests against these indices. Two levels of routing indices are described in [67]. The first level contains the super-peer/peer indices which are

used to route a request from a super-peer to its connected peers. The second level contains the super-peer/super-peer indices, which are extracted from the first level indices. Super-peer/super-peer indices refer to direct neighbor super-peers. They enable the routing of requests through the super-peer backbone.

Super-peer approach increases search performance with respect to flooding algorithms. Yang et al. [87] examine the general characteristics of super-peer networks, which combine the benefits of the efficiency of centralized search with the autonomy, load balancing and robustness to attacks provided by distributed search. In a super-peer network, a peer first looks up its super-peer during a search and super-peers handles query routing through routing indices. Different types of routing indices and their effect on improving the performance of peer-to-peer networks are discussed in [18].

The topology of a peer-to-peer network is also an important factor affecting the network performance. HyperCuP protocol [81] provides hypercube topology which ensures low network diameter. Super-peers of Edutella network [27] are organized according to the HyperCuP protocol. In HyperCuP protocol, nodes are organized into a graph structure based on hypercubes. A node can be thought of as the root of a specific spanning tree while sending messages from the node to the network. Assume there are N nodes in the network, then the topology allows for $log_2\ N$ path length and $log_2\ N$ number of neighbors. Figure 2.2 [81] shows how the nodes in a hypercube graph are serialized where the node labeled "0" chosen as the root node. This node sends messages to the network based on its serialized spanning tree.

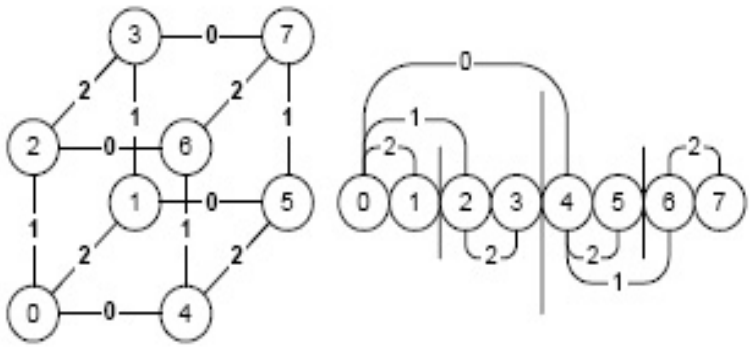


Figure 2.2: a. Hypercube graph b. Serialized notation (links incomplete) [81]

2.2 EHR Architectures

At the moment, EHR information is stored in all kinds of proprietary formats including relational database tables, structured document-based storage in various formats and unstructured document storage such as digitized hardcopies maintained in a classical document management system through various healthcare information systems existing on the market [28]. It is obvious that any approach dealing with the interoperability problem in healthcare has to take the existing healthcare information systems into account. To address the EHR interoperability problem, there are several EHR content standards currently under development which aim to provide standard interfaces to existing proprietary systems. These standards define the structure and the markup of the clinical content to make EHR exchange interoperable. Some of the important standards are:

- OpenEHR [68]
- CEN EN 13606 EHRcom [16]
- Health Level Seven (HL7) [36]

These standards have the following commonalities:

- Each models the health care domain
- Each defines an information model for communication purpose.
- Their information models are very generic in order to handle thousands of concepts.
- Each uses archetypes or templates for defining concepts by constraining the values of the attributes and hierarchies of the objects.

In the healthcare domain, there is also an important initiative named "Integrating Healthcare Enterprise (IHE)" which proposes workflow solutions for the healthcare systems based on the existing standards such as HL7 and DICOM [22]. IHE provides functional units which are called integration profiles for the healthcare information systems. To address the interoperability problem, two important integration profiles have been specified, namely, the "IHE Cross-enterprise Document Sharing (XDS)" and the "IHE Cross Community Access (XCA)". IHE XDS describes how to share EHRs in a community of healthcare enterprises and IHE XCA describes how EHRs are shared across communities.

In the following subsections these healthcare standards and architectures are briefly summarized.

2.2.1 Integrating the Healthcare Enterprise (IHE)

Integrating the Healthcare Enterprise (IHE) is an international initiative that proposes integration profiles for healthcare IT, based on existing standards such as HL7 and DICOM. Integration profiles are problem/solution scenarios such that they define a collection of real world functionality by involving the necessary actors and transactions to make it work. Actors represent a set of roles and responsibilities performed by a system. A real-world system may support several IHE Actors.

The IHE Process to achieve standards-based interoperability includes the following steps which repeat annually to promote steady improvements in integration [50]:

- *Identify Interoperability Problems:* Clinicians and IT experts work to identify common interoperability problems with information access, clinical workflow, administration and the underlying infrastructure.
- *Specify Integration Profiles:* Experienced healthcare IT professionals identify relevant standards and define how to apply them to adress the problems, documenting them in the form of IHE integration profiles.
- *Test Systems at the Connectathon:* Vendors implement IHE integration profiles in their products and test their systems for interoperability at the annual IHE Connectathon. This allows them to assess the maturity of their implementation and resolve issues of interoperability in a supervised testing environment.
- *Publish Integration Statements for use in Request for Proposals:* Vendors publish IHE integration statements to document the IHE integration profiles their products support. Users can reference the IHE integration profiles in requests for proposals, greatly simplifying the systems acquisition process.

The result of the technical work is published as the IHE Technical Framework [53] and revised annually. Currently, technical frameworks are available for the following domains:

- Anatomopathology
- Cardiology
- Eye Care
- IT Infrastructure
- Laboratory

- Patient Care Coordination
- Patient Care Devices
- Quality
- Radiation Oncology
- Radiology

Each technical framework covers integration profiles in its domain. For example, the IT Infrastructure Technical Framework covers use cases for inter-departmental or inter-institutional system integration by specifying the following integration profiles:

- Cross-Enterprise Document Sharing (XDS)
- Patient Identifier Cross-Referencing (PIX)
- Patient Demographics Query (PDQ)
- Audit trail and Node Authentication (ATNA)
- Consistent Time (CT)
- Enterprise User Authentication (EUA)
- Retrieve Information for Display (RID)
- Patient Administration Management (PAM)
- Patient Synchronized Applications (PSA)
- Personnel White Pages (PWP)

Among these integration profiles, Cross-Enterprise Document Sharing addresses how to share clinical documents across enterprises.

IHE Cross-enterprise Document Sharing (XDS) Profile

Cross-enterprise Document Sharing (XDS) is one of the IHE profiles described in the IT Infrastructure Technical Framework. XDS is designed by using ebXML Registry standards [26], SOAP, HTTP and SMTP. In the IHE XDS Integration Profile, healthcare enterprises that agree to work together for clinical document exchange decide on a common set of policies such as how the patients are identified, the EHR document formats to be used and

the common set of coding terms (vocabularies) to represent the metadata of the documents. Its aim is to provide a platform for sharing EHRs. In the IHE XDS Profile, a group of healthcare enterprises that agree to work together for clinical document sharing is called the "XDS Affinity Domain". A healthcare enterprise may belong to one or more XDS Affinity Domains. Healthcare enterprises belonging to an XDS affinity domain can work together in the care of patient by sharing patient's clinical records in the form of documents, which are called XDS Documents. Figure 2.3 [51] depicts the actors and the transactions between these actors for the XDS Integration Profile.

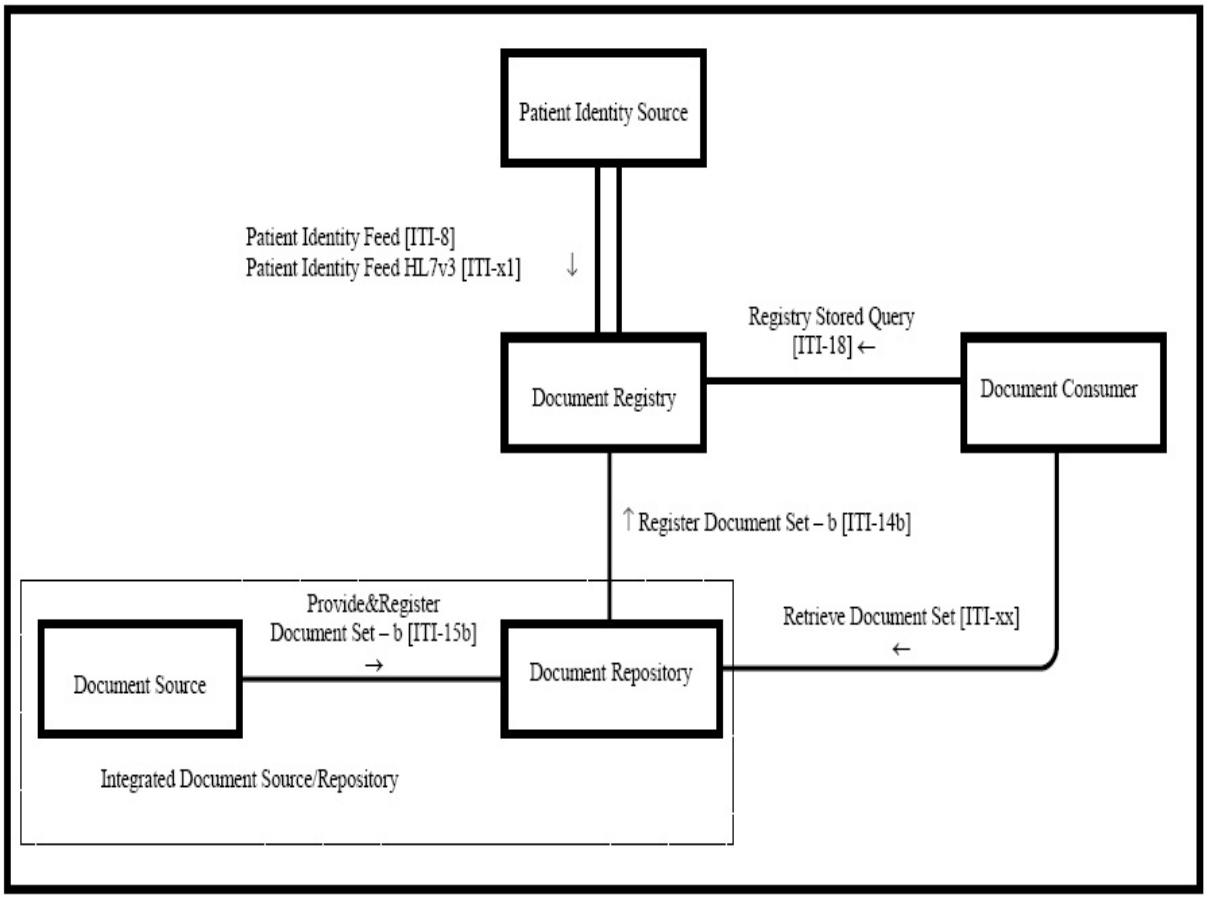


Figure 2.3: Cross-Enterprise Document Sharing Diagram [51]

XDS Integration Profile specifies the following actors:

- *Document Source* is the provider of documents. It is responsible for submitting documents with their metadata to the XDS Affinity Domain.

- *Document Repository* stores the received documents and registers them to the Document Registry by assigning a URI for each document and forwarding the metadata information and the URI information to the Document Registry.

- *Document Registry* stores the metadata and the URI information for each stored document in the Document Repository. The document registry is built on top of an ebXML registry.

- *Document Consumer* queries the Document Registry by defining criteria on the metadata and retrieves documents from the Document Repository using the URI which is obtained from the Document Registry in a previous transaction.

- *Patient Identity Source* is responsible for assigning and managing patient identifiers within the XDS affinity domain.

XDS Integration Profile also specifies the following transactions between the actors:

- *Provide and Register Document Set* is initiated by the Document Source Actor to submit an "XDS Submission Set" to a document repository. An XDS Submission Set consists of both the documents as an opaque octet stream and the corresponding metadata for each document.

- *Register Document Set* is initiated by a Document Repository Actor to register one or more documents with a Document Registry. Document Repository supplies metadata for each document which will be registered to the Document Registry.

- *Query Registry* is initiated by the Document Consumer Actor to search the registry with the specified query criteria. Document Registry returns a list of document entires containing metadata about the documents which satisfies the specified query criteria.

- *Retrieve Document* is initiated by the Document Source Actor to retrieve a document from the Document Registry.

- *Patient Identity Feed* is a transaction which populates the registry with patient identifiers that have been registered for the XDS Affinity Domain.

An XDS Document has a globally uniquely identifier which is location independent, that is, when an XDS Document moved from one XDS Document Repository to another one within an XDS Affinity Domain the identifier remains same, only the URI of the document changes.

An XDS Document is represented by an XDS Document Entry in the XDS Registry Data Model. XDS also introduces following two concepts to structure set of XDS documents as shown in Figure 2.4 [52]:

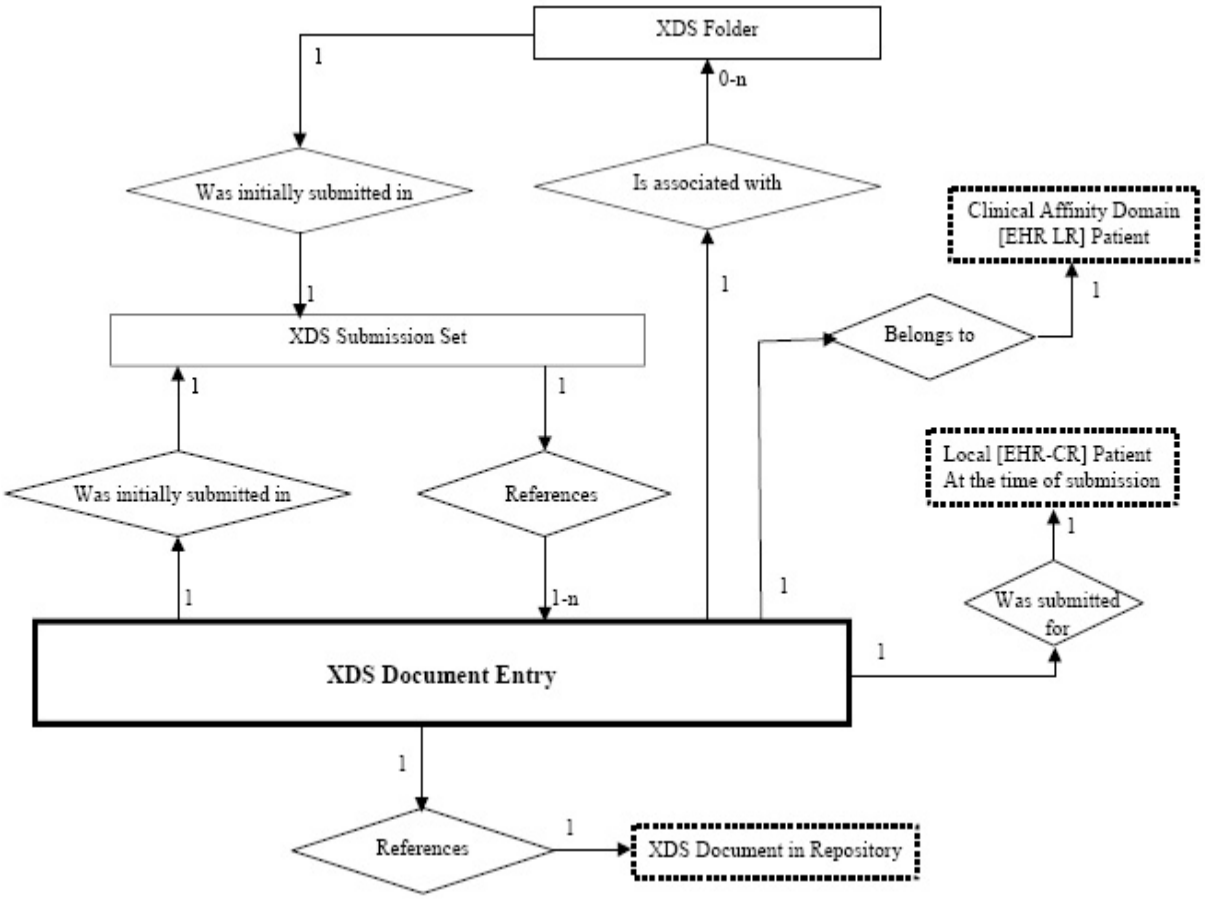


Figure 2.4: XDS Document Registry Data Model [52]

- *XDS Submission Set* groups XDS Documents submitted in a "Provide and Register Document Set" transaction. An already registered XDS Document included in a previous Submission Set may also be referenced by new XDS Submission Sets. In other words, a Submission Set groups XDS documents relevant to the same care events. Each Submission Set belongs to a single Document Source Actor.

- *XDS Folder* creates a logical group for XDS Documents belonging to the same patient. An XDS Document can be contained in zero or more folders.

XDS specifies metadata attributes which can be associated with XDS Documents, Submission Sets and Folders. These attributes can be used to search registered documents in the XDS Affinity Domain. These metadata are provided by Document Source Actors during the submission of documents to the XDS Repository and stored by the XDS Registry Actor to facilitate the discovery of XDS Documents by the Document Consumer Actors. Document Source Actor is responsible for the consistency between the metadata and the content of the associated XDS Document. Some of these metadata attributes are as follows:

- *author* represents the author of the document.
- *classCode* describes the class of the XDS Document from an XDS Affinity Domain specific coding scheme representing document classes such as Prescription, Discharge Summary, Report.
- *confidentialityCode* describes the level of confidentiality of the XDS Document from an XDS Affinity Domain specific coding scheme.
- *eventCodeList* includes list of codes representing the main clinical acts, such as a colonoscopy or an appendectomy involved by the XDS Document. The values of the list are chosen from an XDS Affinity Domain specific coding scheme.
- *formatCode* specifies the format of the XDS Document with a value from an XDS Affinity Domain specific coding scheme.
- *legalAuthenticator* represents the legally authenticating participants of the XDS Document.
- *patientId* represents the identifier of the patient within the XDS Affinity Domain.
- *uniqueId* represents the globally unique identifier of the XDS Document assigned by the Document Source.
- *URI* specifies the URI of the XDS Document to be used by Document Consumers for retrieval.

It should be noted that, IHE XDS is document neutral, that is, enterprises in a clinical affinity domain themselves decide on which document format to use such as HL7 CDA, EHRcom or PDF. IHE XDS solves the interoperability problem within the community but considering the regional and national integration efforts, it is not realistic to expect all the

healthcare institutes in a country or a region to be in the same community. As the number of communicating healthcare information systems increases it becomes unmanageable to interoperate them under a single centralized architecture. Therefore, it is not realistic to group all the existing healthcare information systems under a community based on an architecture like XDS.

A recent effort in this direction is the "IHE Cross Community Access" Integration Profile which defines a protocol to query and retrieve patient related healthcare data across communities. The protocol defines the transactions and actors between the connected communities.

IHE Cross Community Access (XCA)

Recently, IHE has proposed a new supplement called the IHE Cross Community Access (XCA) Integration Profile. The aim of XCA is to provide a mechanism to query and retrieve patient relevant medical data held by other communities. A community is defined as a coupling of facilities/enterprises that have agreed to work together using a common set of policies for the purpose of sharing clinical information via an established mechanism [48]. XDS Affinity Domain is an example of such a community. IHE XCA not only specifies the transactions/actors to execute incoming cross-community queries within the XDS Affinity Domain but also specifies the transactions/actors to forward queries and receive results across communities.

Figure 2.5 [48] shows the actors and the transactions between them which enable exchanging clinical information across XDS Affinity Domains. In order to support the IHE XCA Integration Profile, a community needs to implement the Initiating Gateway and Responding Gateway actors. The Initiating Gateway actor triggers outgoing inter-community communications and the Responding Gateway actor handles all the incoming inter-community communications.

It should be noted that "Registry Stored Query" and "Retrieve Document Set" transactions between the Document Consumer and Initiating Gateway actors are specified as XDS Affinity Domain option of the IHE XCA Integration Profile. It is expected that a non-XDS Affinity Domain supports only the following transactions:

- *Cross Gateway Query* is initiated by the Initiating Gateway Actor to perform search in other communities.
- *Cross Gateway Retrieve* is initiated by the Initiating Gateway actor to request the retrieval of a specific set of clinical documents from other communities.

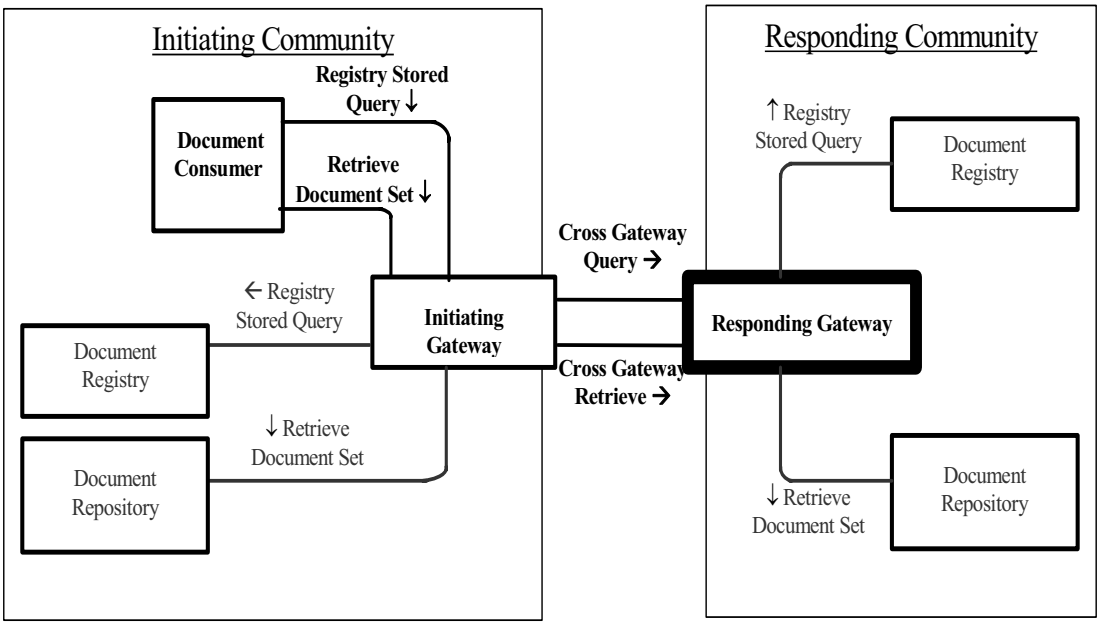


Figure 2.5: IHE Cross Community Access Integration Profile [48]

In the IHE XCA Integration Profile, each community is identified by a globally unique identifier called "homeCommunityId". Initiating Gateway Actor uses the "homeCommunityId" attribute to locate the the data held by the target community. In other words, if a request specifies the value for this attribute, it is forwarded only to the community represented by this attribute. However, a community can also issue a request without specifying a "homeCommunityId", for example, for gathering information about a specific patient from the other communities. In this case, the Initiating Gateway should identify which Responding Gateways the request should be sent to and furthermore it should find patient identifier for the target domain to use in the query.

However, the IHE XCA Integration Profile does not specify how these issues can be handled. But, it describes two possible alternatives for resolving the patient identity in a cross community exchange environment. In the first solution, the IHE XCA Integration Profile is combined with the Patient Identifier Cross-Referencing (PIX) Integration Profile. In this solution, a master patient index (called "PIX Manager" in IHE terminology) cross-references the patient identifiers between communities as shown in Figure 2.6 [48]:

Each community should feed the Topmost PIX Manager through their Patient Identity

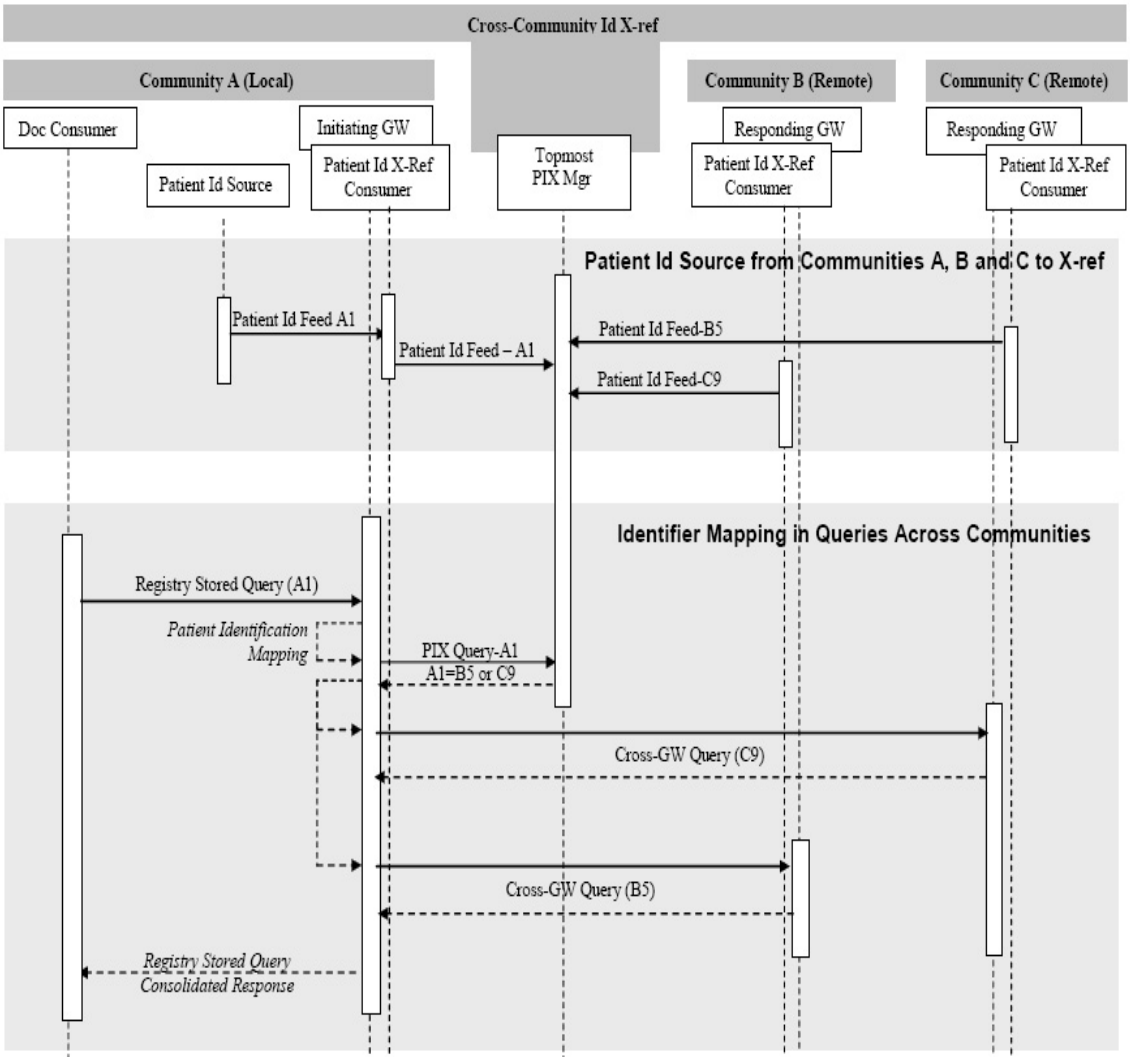


Figure 2.6: Patient Identification using PIX [48]

Source actors with their patient identifiers. Once the PIX Manager receives the Patient Identity Feed requests, it performs its internal logic to determine which, if any, patient identifiers can be "linked together" as being the same patient based on the corroborating information included in the Feed transactions it has received [52]. Before the Initiating Gateway initiates a "Cross Gateway Query" transaction, it requests the patient identifiers for the target communities from the Topmost PIX Manager as shown in Figure 2.6.

However, the maintenance of a master patient index becomes a real challenge when the number of communities in the network increases. The second model relies on the existing IHE Patient Demographics Query (PDQ) Integration Profile as shown in Figure 2.7 [48].

Prior to sending a cross community query, a Patient Demographics Query is sent to the target community to obtain the patient identifier of the target community. However,

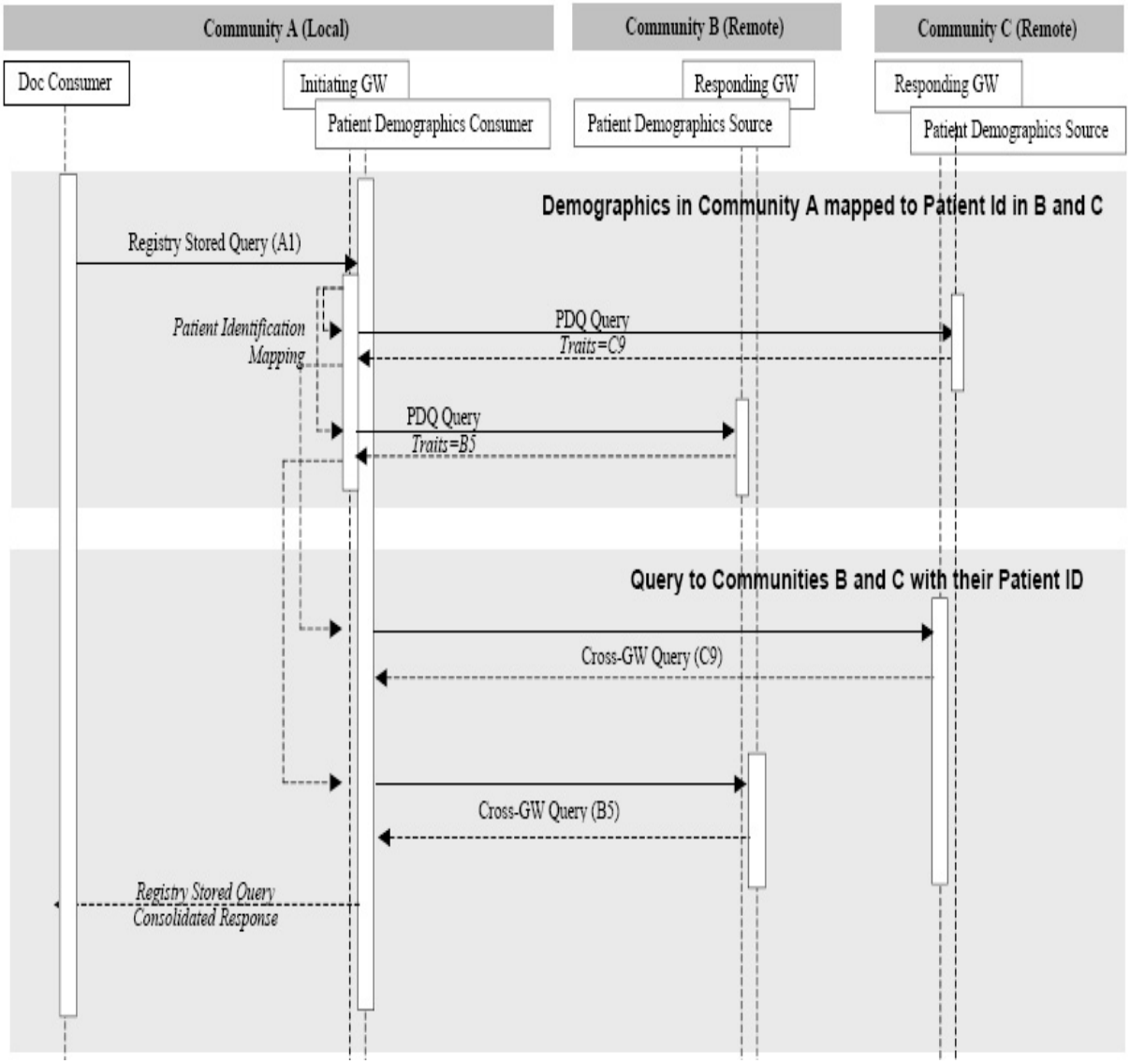


Figure 2.7: Patient Identification using PDQ [48]

resolving the target patient identifier through PDQ requires a significant number of policy decisions to be in place, coordinated with privacy consents in a cross community environment [48].

In addition to having no proper solution for identifying patients across communities, the XCA Integration Profile does not address how to determine the responding communities to send the requests and how to handle vocabulary differences across communities. Furthermore, the IHE XCA integration Profile does not address the EHR content interoperability problem which occurs when communities use different content standards.

2.2.2 Health Level Seven (HL7)

Health Level Seven (HL7) [36] is a not-for-profit ANSI [5] accredited Standards Developing Organization. Primary goal of HL7 is to provide standards for the exchange of clinical and administrative data among healthcare application systems. HL7 defines message structures and triggers events for this purpose. A trigger event causes the exchange of messages between these application systems, that is, when an event occurs in an HL7 compliant system, an HL7 message is sent to other HL7 compliant systems including the necessary data required by the receiving systems.

HL7 Version 3 is the latest ANSI approved standard which is the successor of HL7 Version 2 series. Although the HL7 Version 2 is the most widely implemented healthcare informatics standard in the world, being HL7 Version 2 compliant does not imply direct interoperability between healthcare systems since Version 2 allows many optional data fields. Having optionality provides great flexibility, however it results in hidden assumptions during message exchange process. HL7 has developed Version 3 in order to overcome problems resulting from the optionality of Version 2 series.

HL7 Reference Information Model (RIM)

HL7 Version 3 is based on an object-oriented data model which is called the Reference Information Model (RIM). The RIM is comprised of six "back-bone" classes as shown in Figure 2.8.

Every happening documented in the healthcare domain is represented by the Act class. The Participation class defines the context for an Act by defining the relationship between Act and Role classes. Physical things and beings that take part in healthcare are represented by the Entity class. The Role class establishes the roles that entities play as they participate in healthcare acts. The ActRelationship class defines the relationship between two instances of the Act class. Similarly, the RoleLink defines the relationship between two instances of the Role class.

The Act, Entity and Role classes are further specialized to subclasses. In the HL7 representation, a new subclass is added to the RIM only when new attributes or associations are needed which are not available in the super classes. For example, Account class which specializes the Act class introduces "balanceAmt", "currencyCode", "interestRateQuantity" and "allowedBalanceQuantity" attributes to represent a financial account.

A specialized concept which needs no further attributes or associations is represented

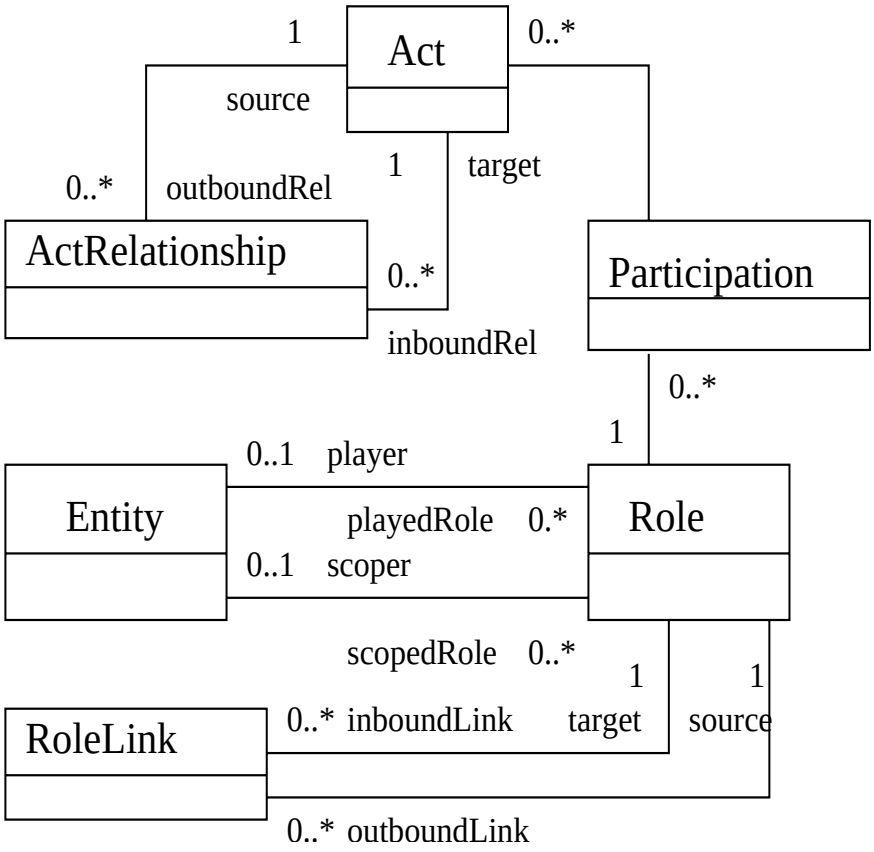


Figure 2.8: RIM back-bone classes

by assigning a unique code in the controlling vocabulary to specific attributes. Therefore, these three classes include the following coded attributes, which serve to further define the concept being modeled [41]:

- classCode (in Act, Entity and Role) represents the exact class or concept intended, whether or not that class is represented as a class in the RIM hierarchy. For example, the value "DOCCLIN" can be used to represent "Clinical Document" concept.
- moodCode (in Act) attribute distinguishes whether an Act is conceived of as a factual statement or in some other manner as a command, possibility, goal, etc. It is also used to describe the progression of a business activity from defined to planned to executed. For example, the value "EVN" for moodCode is used to describe a service that actually happens, may be an ongoing service or a documentation of a past service.
- determinerCode (in Entity) attribute distinguishes whether the class represents an instance or a kind of Entity.

- code (in Act, Entity and Role) provides for further classification within a particular classCode value, such as a particular type of observation within the Observation class. For example, the ellipse region can be derived by assigning the value "ELLIPSE" to the code attribute of the Observation class of the RIM. It should be noted that the value of the code should be consistent with the value of the classCode.

Similarly, the "typeCode" attribute in Participation, ActRelationship and RoleLink classes is used for representing variety of concepts in these categories. For example, the "author" concept which describes the party that originates the Act can be derived by assigning the value "AUT" to the typeCode of the Participation class of the RIM.

In Version 3, RIM is the source of all message contents. Figure 2.9 shows the steps of how message structures are defined based on the HL7 RIM.

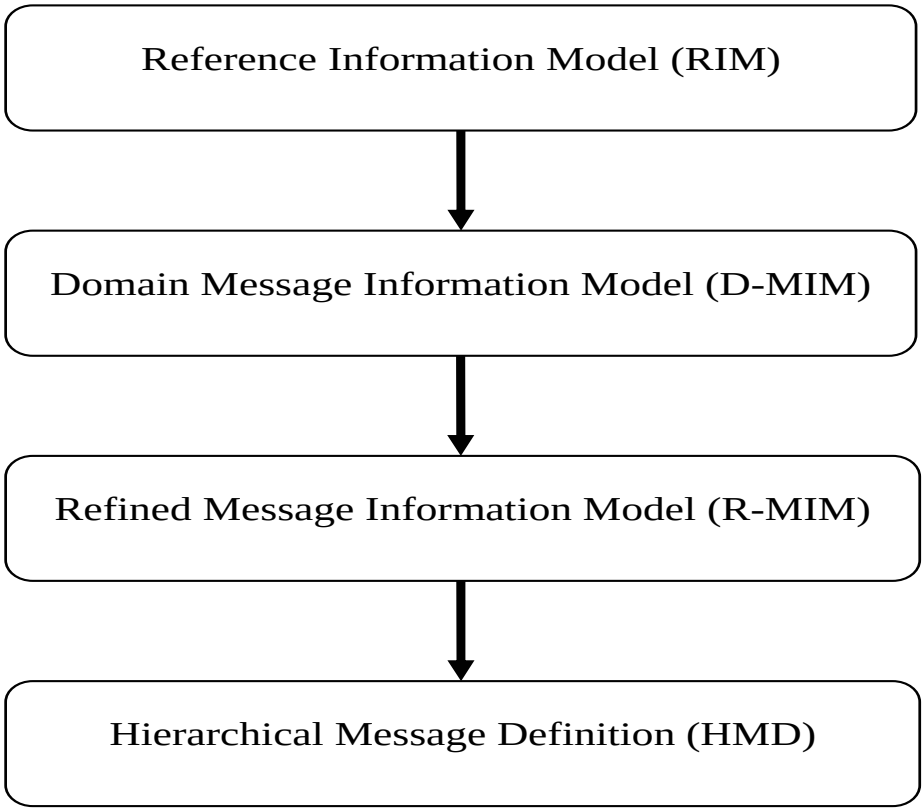


Figure 2.9: Message Structure Generation from the RIM in HL7 V3 Process

The first step to generate the message structures from the RIM is to derive the Domain Message Information Model (D-MIM) which is the subset of the RIM that includes a fully

expanded set of class clones, attributes and relationships that are used to create messages for any particular domain [45]. D-MIM is derived from RIM in such a way that only the required classes, attributes and relationships for building the messages for a particular domain are included. Furthermore multiple specializations of the same RIM class may appear in a diagram with different constraints or associations. Such classes are called as clones of the RIM class. For example, "ClinicalDocument" and "Section" are clones of the RIM "Act" class to represent the structure of a clinical document.

Next step is to build the Refined Message Information Model (R-MIM) which defines the information content for one or more Hierarchical Message Descriptions (HMDs). A R-MIM is derived from a D-MIM by including the necessary classes, attributes and associations used in set of messages derived from the HMDs. The result of the Version 3 process is the Hierarchical Message Definition (HMD) which is the tabular representation of the sequence of elements (i.e., classes, attributes and associations) represented in an R-MIM [45]. The primary goal of HMD is to define the message structure without reference to any implementation technology. HL7 also provides XML Implementable Technology Specification to express HMD in XML format.

HL7 defines several message structures in various domains including account and billing, bloodbank, clinical genomics, claims and reimbursement, laboratory etc. In addition, HL7 also specifies Clinical Document Architecture which describes the structure and semantics of clinical documents exchanged between care providers.

HL7 Clinical Document Architecture (CDA)

HL7 Clinical Document Architecture (CDA), previously called Patient Record Architecture (PRA), is a document markup standard that specifies the structure and semantics of a clinical document (such as a discharge summary or progress note) for the purpose of exchange [37]. A clinical document includes clinical observations and services about care events. A valid CDA document is encoded in Extensible Markup Language (XML) and conforms to CDA Schema which is derived from the CDA Hierarchical Description based on the XML Implementable Technology Specification. The CDA Hierarchical Description is derived from the CDA R-MIM through the process shown in Figure 2.9. In other words, HL7 RIM is the source of the structure and semantics of a CDA document.

So far, HL7 has released two versions of CDA. The CDA Release One (CDA R1) is the first specification derived from the HL7 RIM. It became an ANSI approved HL7 Standard in 2000. The CDA Release Two (CDA R2) became an ANSI-approved HL7 Standard in 2005

[25].

A CDA document has two main parts, the header and the body. In the CDA R1, only the header part is derived from the RIM. In the CDA R2, in addition to the header part, the clinical content in the document body is also derived from the RIM. Therefore the CDA R2 model enables the formal representation of clinical statements through CDA Entry classes. Only the CDA narrative blocks which are required fields in the document are not derived from the RIM.

A CDA header defines the context of the document by providing information on authentication, the encounter, the patient, and the involved providers whereas the CDA body includes the clinical report. The body part can be either an unstructured blob or a structured hierarchy which involves one or more section components. Within a section, narrative blocks and CDA entries are defined. Machine-processable clinical statements are represented by these CDA entries whereas the narrative blocks are human readable forms of these clinical statements. Figure 2.10 [25] depicts the major components of a CDA document.

```
<ClinicalDocument>
  ... CDA Header ...
  <structuredBody>
    <section>
      <text>...</text>
      <observation>...</observation>
      <substanceAdministration>
        <supply>...</supply>
      </substanceAdministration>
      <observation>
        <externalObservation>...
        </externalObservation>
      </observation>
    </section>
    <section>
        <section>...</section>
    </section>
  </structuredBody>
</ClinicalDocument>
```

Figure 2.10: Major components of a CDA document [25]

The "ClinicalDocument" is the root element of the document. The header is defined between the <ClinicalDocument> and the <StructuredBody> tags. Sections reside in the

"StructuredBody" element. Each section can contain a single narrative block located in the "text" element. A narrative block is the human readable portion of the section when rendered with an appropriate style sheet. Section also contains CDA entries which are used to represent structured content. CDA entires are machine-processable portion of the sections.

CDA entries are derived from the shared HL7 Clinical Statement Model [46] which provides a consistent representation of clinical statements across various V3 specifications. The model describes the clinical statements using the following entry classes which are derived from the classes of the HL7 RIM:

- *Act* is derived from the RIM Act class. It is a generic purpose class which is used when the other more specific classes of the model are not appropriate for defining the clinical information.
- *Observation* is derived from the RIM Observation class. It is used for representing clinical observations.
- *ObservationMedia* is derived from the RIM Observation class. It is used for representing multimedia which is logically part of the document.
- *SubstanceAdministration* is derived from the RIM SubstanceAdministration class. It is used for representing medication-related events.
- *Supply* is derived from the RIM Supply class. It is used for representing the provision of a material between entities.
- *Procedure* is derived from the RIM Procedure class. It is used for used for representing procedures.
- *RegionOfInterest* is derived from the RIM Observation class. It is used for referencing to specific regions in images.
- *Encounter* is derived from the RIM PatientEncounter class. It is used for representing the interaction between a patient and care provider.
- *Organizer* is derived from the RIM Act class. It is used for grouping clinical statements having a common context.

Figure 2.11 [37] gives a sample section describing a physical examination of the skin in which a rash is identified. The entry element within the section contains clinical statements

```
<section>
   <code code="8709-8" codeSystem="2.16.840.1.113883.6.1" codeSystemName="LOINC"/>
   <title>Skin Exam</title>
   <text>Erythematous rash, palmar surface, left index
    finger.<renderMultiMedia referencedObject="MM2"/>
   </text>
   <entry>
      <observation classCode="OBS" moodCode="EVN">
         <code code="271807003"
          codeSystem="2.16.840.1.113883.6.96" codeSystemName="SNOMED CT" displayName="Rash"/>
         <statusCode code="completed"/>
         <targetSiteCode code="48856004" codeSystem="2.16.840.1.113883.6.96"
          codeSystemName="SNOMED CT" displayName="Skin of palmer surface of index finger">
            <qualifier>
               <name code="78615007" codeSystem="2.16.840.1.113883.6.96"
                displayName="with laterality"/>
               <value code="7771000" codeSystem="2.16.840.1.113883.6.96" displayName="left"/>
            </qualifier>
         </targetSiteCode>
         <entryRelationship typeCode="SPRT">
            <regionOfInterest classCode="ROIOVL" moodCode="EVN" ID="MM2">
               <id root="2.16.840.1.113883.19.3.1"/>
               <code code="ELLIPSE"/>
               <value value="3"/>
               <value value="1"/>
               <value value="3"/>
               <value value="7"/>
               <value value="2"/>
               <value value="4"/>
               <value value="4"/>
               <value value="4"/>
               <entryRelationship typeCode="SUBJ">
                  <observationMedia classCode="OBS" moodCode="EVN">
                     <id root="2.16.840.1.113883.19.2.1"/>
                     <value mediaType="image/jpeg">
                        <reference value="lefthand.jpeg"/>
                     </value>
                  </observationMedia>
               </entryRelationship>
            </regionOfInterest>
         </entryRelationship>
      </observation>
   </entry>
</section>
```

Figure 2.11: A sample section describing a skin exam care event [37]

describing the physical examination of the skin. A region of interest is described using "RegionOfInterest" element which further references a file containing the hand image.

It should be noted that in Figure 2.11, "Section" element has been coded with Logical Observation Identifiers Names and Codes (LOINC) [62] and "Observation" element has been coded with SNOMED Clinical Terms [83]. The structure and the codes used in a CDA Document makes it machine processable.

HL7 Templates

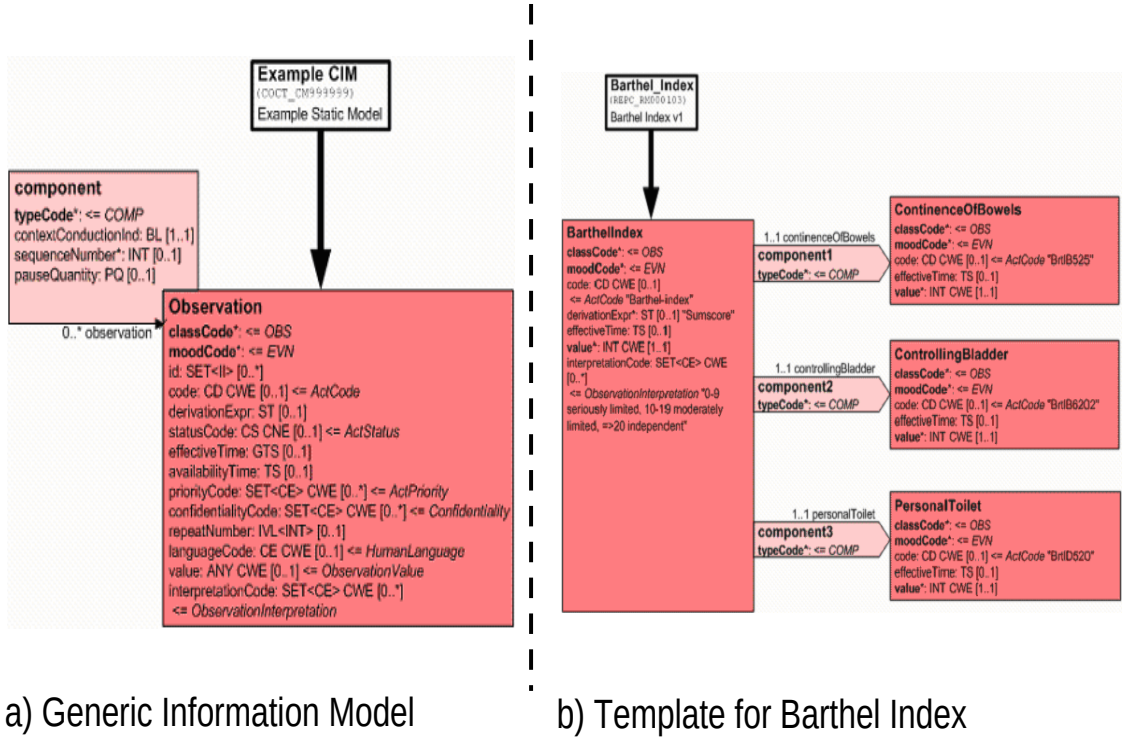


Figure 2.12: HL7 Template Example [44]

HL7 also works on a Template Specification [44] to describe further constraints on the models such as HL7 CDA. Static models such as R-MIMs are derived from the RIM by defining a set of constraints through the HL7 V3 process. However, these static models may become too generic to express real world concepts. Templates aid to constrain and define the structures of these more atomic concepts such as blood pressure observation, labrotary test, etc. HL7 templates have unique identifiers. When such an identifier is referenced by an instance, the instance should satisfy the constraints defined by the template. The templateId

attribute on the abstract class InfrastructureRoot which is superclass of all RIM classes is used to identify the templates.

In Figure 2.12 [44], a generic information model and a template constraining this model are given. The information model given in Figure 2.12 (a) is a part of the HL7 Clinical Statement Model. It describes clinical statements which may include an observation and the observation may have zero or more components which are also observations. The template given in Figure 2.12 (b) represents the part of Barthel Index concept which measures person daily activities. In order to describe Barthel Index concept the template introduces four classes namely BarthelIndex, continenceOfBowels, controllingBladder and personalToilet. The template conforms the information model since these classes are derived from the Observation class by constraining the cardinality of component relations, constraining their "code" attributes to relevant codes describing these classes and eliminating the unnecessary optional attributes in these classes.

```
<barthelIndex>
    <code codeSystem="2.16.840.1.113883.2.6.15.1"
         code="Barthel-index"/>
    <derivationExpr>Sumscore</derivationExpr>
    <effectiveTime value="200601191211"/>
    <value value="3" />
    <component1>
        <continenceOfBowels>
            <code
                codeSystem="2.16.840.1.113883.2.6.15.1.0"
                code="BrtlB525"/>
                 <value value="1"/>
         </continenceOfBowels>
    </component1>
    <component2>
        <controllingBladder>
            <code codeSystem="2.16.840.1.113883.2.6.15.1.0"
                  code="BrtlB6202"/>
            <value value="1"/>
        </controllingBladder>
    </component2>
    <component3>
        <personalToilet>
            <code codeSystem="2.16.840.1.113883.2.6.15.1.0"
                  code="BrtlD520"/>
            <value value="1"/>
         </personalToilet>
    </component3>
</barthelIndex>
```

a) Instance based on the Template Model

```
<observation>
    <templateId root="2.16.840.1.113883.10"
                extension=" REPC_RM000103"/>
    <code codeSystem="2.16.840.1.113883.2.6.15.1"
          code="Barthel-index"/>
    <derivationExpr>Sumscore</derivationExpr>
    <effectiveTime value="200601191211"/>
    <value value="3" />
    <component>
        <observation>
            <code codeSystem="2.16.840.1.113883.2.6.15.1.0"
                  code="BrtlB525"/>
            <value value="1"/>
        </observation>
    </component>
    <component>
        <observation>
            <code codeSystem="2.16.840.1.113883.2.6.15.1.0"
                  code="BrtlB6202"/>
            <value value="1"/>
        </observation>
    </component>
    <component>
        <observation>
            <code codeSystem="2.16.840.1.113883.2.6.15.1.0"
                  code="BrtlD520"/>
            <value value="1"/>
        </observation>
    </component>
</observation>
```

b) Instance based on the Reference Model

Figure 2.13: Instance conforming to a template [44]

An instance of Barthel Index concept is shown in Figure 2.13 [44]. In Figure 2.13 (a), the instance is constructed based on the template model. The equivalent instance based on the reference model is given in Figure 2.13 (b). It should be noted that the instance based on the reference model references the template of the Barthel Index concept by assigning the relevant template identifier to its "templateId" attribute.

2.2.3 The GEHR/openEHR Initiative

The Good Electronic Health Record (GEHR) has been developed from the Good European Health Record project which ran from 1992-1995 as an EU research project in the third framework programme. Currently, GEHR is maintained by the openEHR Foundation, a not for profit foundation, was created to enable the development of open specifications, software and knowledge resources for health information systems, in particular electronic health record (EHR) systems [68].

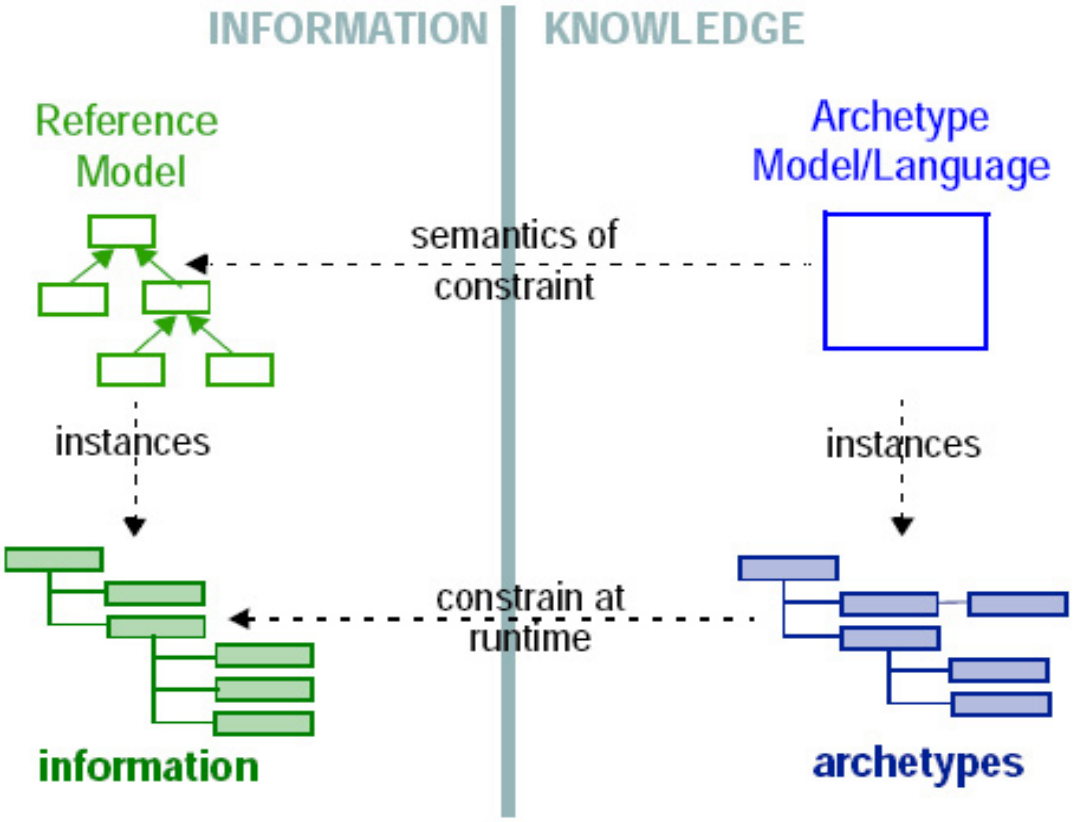


Figure 2.14: Archetype Model Meta-architecture [8]

The information models of GEHR/openEHR are developed using a "two-level modelling" approach as shown in Figure 2.14 [8]. In the information level, a generic reference model, which can be used to represent all possible instances, is developed. This model must be stable over time, therefore should include a few classes such as role, act, entity, participation. The knowledge level models the specific concepts of the domain as archetypes which define constraints on the reference model instances.

Archetypes

Archetypes are constraint-based models of domain entities and each archetype describes configurations of data instances whose classes conform to an information model [10]. Having a generic information model with just a few concepts makes it stable over time. However, the small number of generic concepts in the information model is not enough to describe the semantics of the domain specific concepts: these are described through archetypes. As an example, an information model class such as an observation can be restricted through archetypes to mean two different concepts such as "Blood Pressure" or "Body Temperature".

An archetype is composed of three parts: header section, definition section and ontology section. The header section contains a unique identifier for the archetype, a code identifying the clinical concept defined by the archetype. The header section also includes some descriptive information such as author, version and status. The definition section contains restrictions in a tree-like structure created from the information model. This structure constrains the cardinality and content of the information model instances complying with the archetype. Codes representing the meanings of nodes and constraints on text or terms, bindings to terminologies such as SNOMED or LOINC, are stated in the ontology section of an archetype. A formal language for expressing archetypes, namely, Archetype Definition Language (ADL) is described in [9].

ADL specializes the classes of the generic information model by constraining their attributes. The applicable constraints are as follows [9]:

- Constraints on the range of data-valued properties.
- Constraints on the range of object-valued properties.
- Constraints on the "existence" of a property indicating whether the property is optional or mandatory.
- Constraints on the "cardinality" of a property indicating whether the property refers

```
Observation[at0000] matches { -- Barthel Index
    code matches {
        CE matches {[ac0001]}
    }
    component cardinality matches {3} matches{
        Observation[at0001] occurrences matches {1} matches {  -- continence of bowels
            code matches {
                CE matches {[ac0002]}
            }
        }
        Observation[at0002] occurrences matches {1} matches {  -- controlling bladder
            code matches {
                CE matches {[ac0003]}
            }
        }
        Observation[at0003] occurrences matches {1} matches {  -- personal toilet
            code matches {
                CE matches {[ac0004]}
            }
        }
    }
}
```

Figure 2.15: A part of ADL definition of "Barthel Index" Archetype

to a container type, the number of member items it must have and their optionality, and whether it has a "list" or a "set" structure.

- Constraints on a property with "occurrences" indicating how many times in runtime data an instance of a given class conforming to a particular constraint can occur. It only has significance for objects, which are children of a container property.

As an example to an archetype definition in ADL, a part of a "Barthel Index" archetype definition is presented in Figure 2.15. The archetype uses the "Observation" class defined in the HL7 CDA R-MIM. The "Observation" class is restricted to create the "Barthel Index" archetype. It should be noted that the "Observation" class is bound to the local term "at0000". Such a term can be bound to a relevant terminology in the term_binding part of the archetype. It should also be noted that HL7 CDA Templates are very similar to openEHR Archetypes.

2.2.4 CEN EN 13606

CEN EN 13606 also known as Electronic Health Record Communication (EHRcom) is derived from the previous European standard ENV 13606 by adopting openEHR archetype methodology and taking into account the existing implementation experience on ENV 13606. CEN EN 13606 consists of five parts:

1. *Reference Model:* specifies the information architecture of the EHR data exchanged between systems and services.
2. *Archetype Reference Model:* specifies the Archetype Model and the language which are used to constrain the data.
3. *Term Lists:* specifies the code lists which are used with the standard.
4. *Security Requirements and Distribution Rules:* specifies the privileges and regulations necessary to access data.
5. *Implementation Guides:* gives implementation advice for vendors and implementers

Currently, only the reference model (EN 13606-1) has been published, whereas the other parts are still working drafts. The reference model involves four packages: Extract, Demographics, Access Control and Message. In Figure 2.16, core classes of the extract package is given. These classes define the structure of EHR content.

An EHR_Extract is the root class of part or all of the EHR of a single patient. It contains zero or more Folders while Folders may have nested Folders. A Composition resides in a Folder or directly in the EHR_Extract. Sections are contained in Compositions while Sections may have nested Sections. An Entry is either contained within a Composition or located under a Section. An Entry contains Element instances which are optionally contained within a Cluster hierarchy.

In CEN EN 13606-1, a clinical statement can be represented by an Entry hierarchy. An Entry cannot contain further Entry classes but it is possible to reference other Entry classes via links. Complex data structures such as a table, a tree or a time series are represented by Clusters. The leaf node of the EHR_Extract hierarchy is the Element class which holds a single data value.

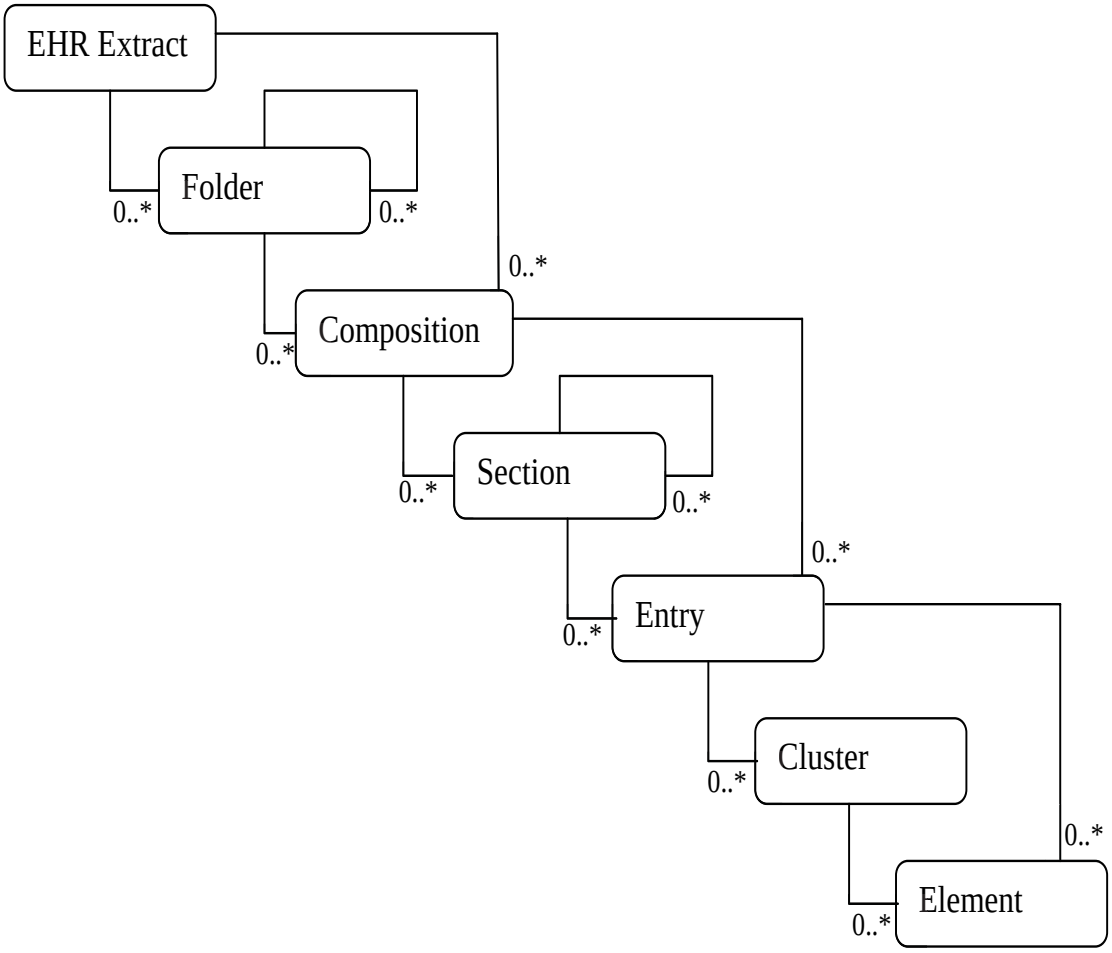


Figure 2.16: EHR Extract hierarchy

2.3 Web Ontology Language (OWL)

In [35], Gruber defines the ontology as a formal explicit specification of a shared conceptualization for a domain of interest. Therefore, an ontology for a given domain has to be defined with a formal language which allows representing the concepts and their relationships. Many ontology languages have been developed so far. These languages often enables reasoning over the represented domain. Some of these languages are as follows:

- Resource Description Framework (RDF) [74] is a general-purpose language for representing information in the Web. It represents a data model based on object, property, and value triples.
- RDF Schema [73] defines classes and properties that may be used to describe classes, properties, domain and range restrictions of properties and other resources.

- Ontology Inference Layer (OIL) [32] is an extension of RDF and RDF Schema based on the concepts developed in Description Logic (DL) and frame-based systems.
- DAML+OIL [17] combines the features of DAML [19] and OIL languages.
- Web Ontology Language (OWL) [85] is derived from the DAML+OIL and builds upon the RDF and RDF Schema.

World Wide Web Consortium (W3C) [86] has specified the OWL to extend the limited expressiveness of RDF Schema. OWL is a revision of the DAML+OIL web ontology language which is a union effort of American and European initiatives. OWL became a formal W3C recommendation on February 10, 2004. It provides the following three increasingly expressive sublanguages which are designed for use by specific communities of implementers and user [85]:

- OWL Lite supports a classification hierarchy and simple constraints. For example, it only allows cardinality values of 0 or 1.
- OWL DL provides the maximum expressiveness possible while retaining computational completeness, decidability, and the availability of practical reasoning algorithms.
- OWL Full has maximum expressiveness and the syntactic freedom of RDF with no computational guarantees. It is unlikely that any reasoning software will be able to support complete reasoning for OWL Full.

OWL uses classes and properties to describe ontologies. Classes provide an abstraction mechanism for grouping resources with similar characteristics and OWL distinguishes following types of class descriptions [20]:

- a class identifier (a URI reference)
- an exhaustive enumeration of individuals that together form the instances of a class
- a property restriction
- the intersection of two or more class descriptions
- the union of two or more class descriptions
- the complement of a class description

In OWL, properties are defined through their domains and ranges. The domain of a property limits the individuals to which the property can be applied and the range of a property limits the individuals that the property may have as its value [85].

OWL also supports axioms which make it possible to assert subsumption or equivalence with respect to classes or properties [47]. 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

CHAPTER 3

PEER-TO-PEER ARCHITECTURE FOR HEALTHCARE COMMUNITIES

In this section, we describe a scalable peer-to-peer architecture implementing the IHE XCA Integration Profile to address the following challenges:

- In order to provide a scalable solution, we propose to use a super-peer based peer-to-peer architecture with routing indices based on two parameters: the *homeCommunityId* attribute of the IHE XCA Integration Profile and so-called patient control numbers. The *homeCommunityId* attribute is used to uniquely identify a community in the network. The value of the *homeCommunityId* of a community peer is provided to its super-peer during the registration of the community peer to the network, and this value does not change in time. Routing indices use the *homeCommunityId* attribute to forward the request when the value of the *homeCommunityId* is known. However, most of the queries will be for gathering information about a specific patient from the other communities whose *homeCommunityId* attributes are not known in advance. These queries should identify the patient in such a way that other communities will be able to locate the right patient in their communities. In this case, since using patient demographic data for identifying a patient creates privacy concerns, we describe how patient control numbers can be used in this architecture.

- Queries and their results in a cross community exchange environment may involve values drawn from a community specific vocabulary. To handle vocabulary translations, we introduce an actor named Vocabulary Translator that is responsible for storing mappings between the local vocabulary and the vocabulary of the connected super peer. In this extension, an initiating gateway requests the transformed values of the

parameters from the Vocabulary Translator Actor before forwarding the request to a Responding Gateway.

Through this architecture, the scalability of the IHE XCA Integration Profile implementations will be greatly enhanced because:

- When a community wishes to join the network, it only needs to provide a single vocabulary mapping super-peer's vocabulary.
- Queries are efficiently routed through super-peers preventing a flooding of the network.

3.1 Community Network Architecture

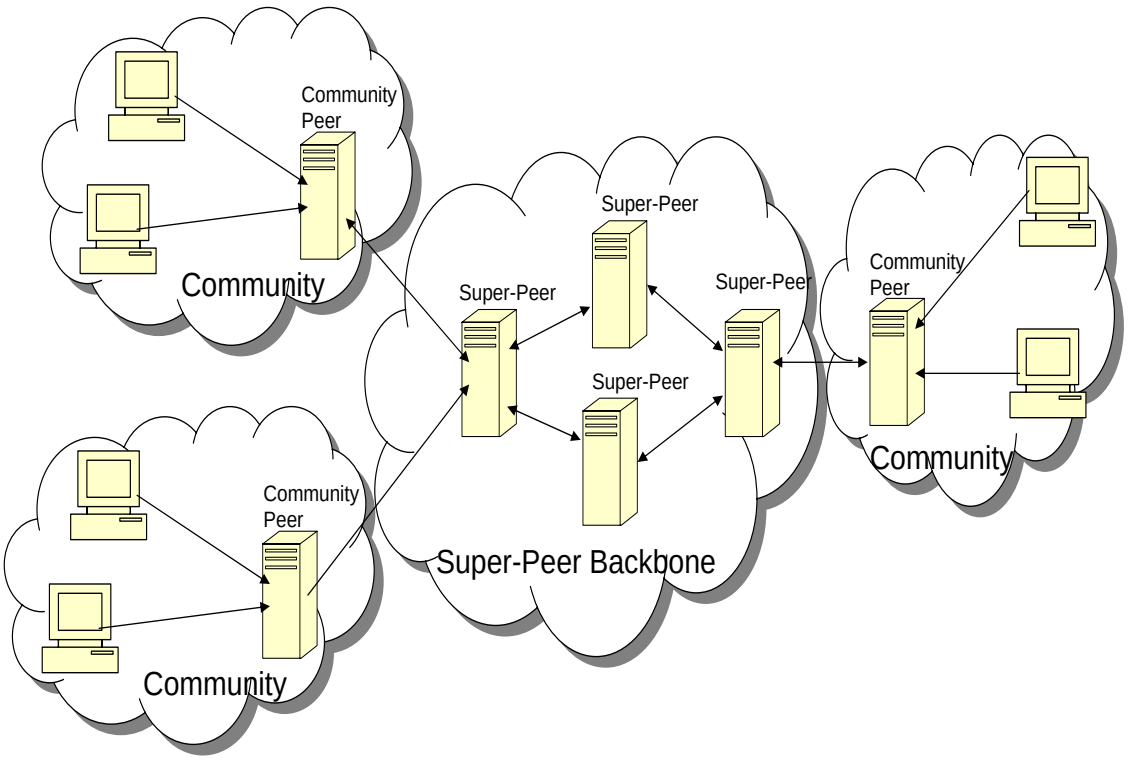


Figure 3.1: An Overview of the Federated Community Network Architecture

The IHE XCA Integration Profile has a decentralized architecture where each community provides a set of predefined services. However, as the number of communities increases, managing the configuration of community pairs becomes a challenge. In order to overcome the complexity introduced by one-to-one connections of communities, we propose a scalable

peer-to-peer community network architecture based on the super-peer model described in [67].

The overview of the federated community network architecture is shown in Figure 3.1 where each community is connected to the network through its community peer. Each community peer connects to a super-peer and forms local network with hub-and-spoke topology around a super-peer. In other words, community peers are leaf nodes in the network and super-peers are responsible for routing the requests in addition to vocabulary translations required during the routing process.

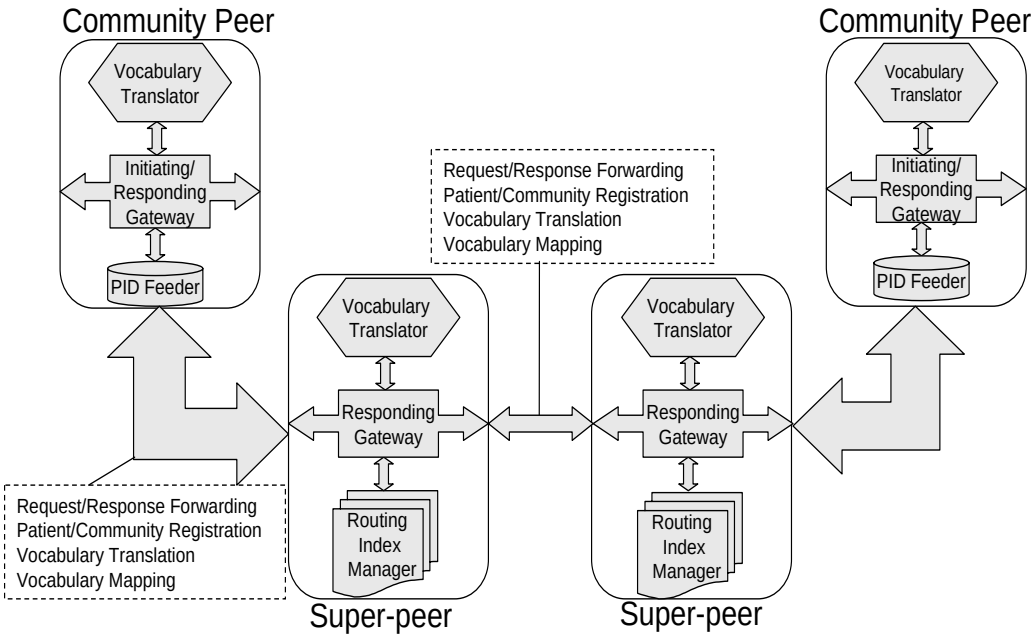


Figure 3.2: The Super-Peer Backbone Architecture of the Federated Community Network

A super-peer consists of three components as shown in Figure 3.2. The Responding Gateway is responsible for delegating incoming messages based on the values of *homeCommunityId* or patient control numbers in its indices. For this purpose, the Responding Gateway first queries the Routing Index Manager to find out the peers to which the message will be forwarded. The Routing Index Manager is responsible for maintaining the routing indices. It receives updates from the PID Feeder of its community peers as well as from the Routing Index Managers of the neighboring super-peers. Second step is to translate coding vocabulary values during the delegation of messages to neighboring super-peers. The Vocabulary

Translator actor exists in a super-peer is responsible for the vocabulary translation between neighboring super-peers, as described in Section 3.4.

A community peer also consists of three components as shown in Figure 3.2. The Initiating/Responding Gateway is responsible for sending requests to its super-peer and responding requests received from the super-peer. PID Feeder is responsible for registering control numbers of its patients to its super-peer. For example, in an XDS community the actor may be grouped with a Patient Identity Source actor and whenever a new patient is registered or patient data is updated, it receives the demographic data from the Patient Identity Source and generates control numbers for the patient and informs the super-peer by sending the generated control numbers. The Vocabulary Translator actor in a community-peer is responsible for transforming local vocabulary to the vocabulary of the super-peer. It should be noted that a super-peer builds and maintains a common vocabulary for its community-peers.

Assume that a community peer initiates a request containing a value for a *homeCommunityId*. As the super-peer of the community peer receives the request, it first checks the super-peer/peer indices to locate an appropriate community peer to forward the request. If the super-peer does not succeed in locating a community peer, it forwards the request based on the super-peer/super-peer indices.

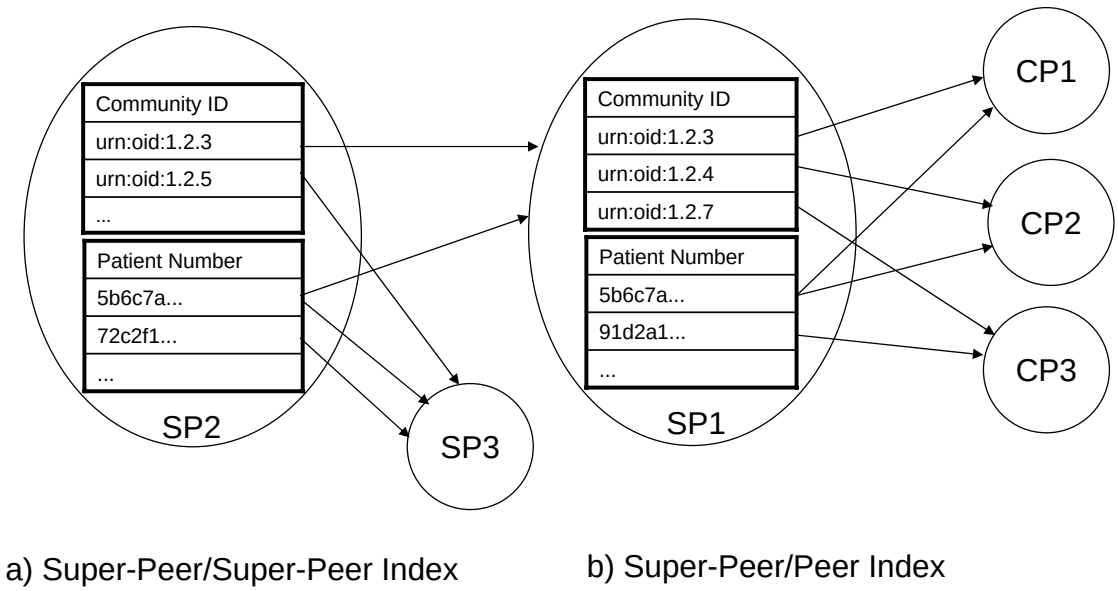


Figure 3.3: Index-based Routing in peer-to-peer network

As shown in Figure 3.3 (a), super-peer/super-peer indices of a super-peer have two parts.

The first part is responsible for routing requests based on *homeCommunityId* to locate the destination. In Figure 3.3, the value "urn:oid:1.2.3" points to super-peer "SP1" indicating that each request containing this value will be forwarded from "SP2" to "SP1".

However, a community can also issue a request without specifying a *homeCommunityId*, for example, for gathering information about a specific patient from the other communities. As already mentioned, broadcasting such requests in the network will be inefficient since all of the communities will have to process the request even when they do not have the information about the patient. Furthermore, broadcasting increases the network traffic.

In order to route requests to a community which has the information for a given patient, the community should register its patients to its super-peer. When a super-peer uses patient-community relations to build its routing indices, it is possible to route requests only to the relevant communities. Since a patient has different patient identifiers in different communities, registering patients through their local patient identifiers is not suitable. On the other hand, using patient demographic data for identifying a patient creates concerns about privacy.

The mechanism we propose for handling this problem is to use so-called control numbers instead of patient identifiers. Control numbers have previously been used for preserving patient privacy in other contexts [2], [4], [29]. Furthermore, control numbers enable fuzzy matching of patient data as demonstrated in the PID protocol [29]. The matching algorithm in super-peers not only compares the control number from the request against the control numbers existing in indices for equality, it also allows to identify the most "promising" matches from a larger set.

The second part of the super-peer/super-peer indices is responsible for routing requests based on the control number generated for a patient. In Figure 3.3 (a), the control number value "5b6c7a..." points to both super-peers "SP1" and "SP3" indicating that each request containing this value will be forwarded from "SP2" to "SP1" and "SP3". Figure 3.3 (b) depicts the super-peer/peer indices of the super-peer "SP1". The structure of super-peer/peer and super-peer/super-peer indices are the same. The difference is that super-peer/peer indices point to community peers instead of super-peers. The value "urn:oid:1.2.3" points to "CP1", that is, the community id of "CP1" is "urn:oid:1.2.3". Patient control numbers point to the community peers where patient information available for the given patient control number. In Section 3.2, we describe how to use the control numbers to provide patient privacy in our architecture.

3.2 Routing Requests based on Privacy Preserving Control Numbers

Control numbers are used to provide record linkage of data records by encrypting the characters of demographic values of personal data. For example, [84] uses control numbers to allow record linkage of anonymised records that describe cancer cases which are collected independently from multiple sources in the epidemiological cancer registries in Germany. Figure 3.4 [29] describes the processing steps for generating control numbers from demographic values.

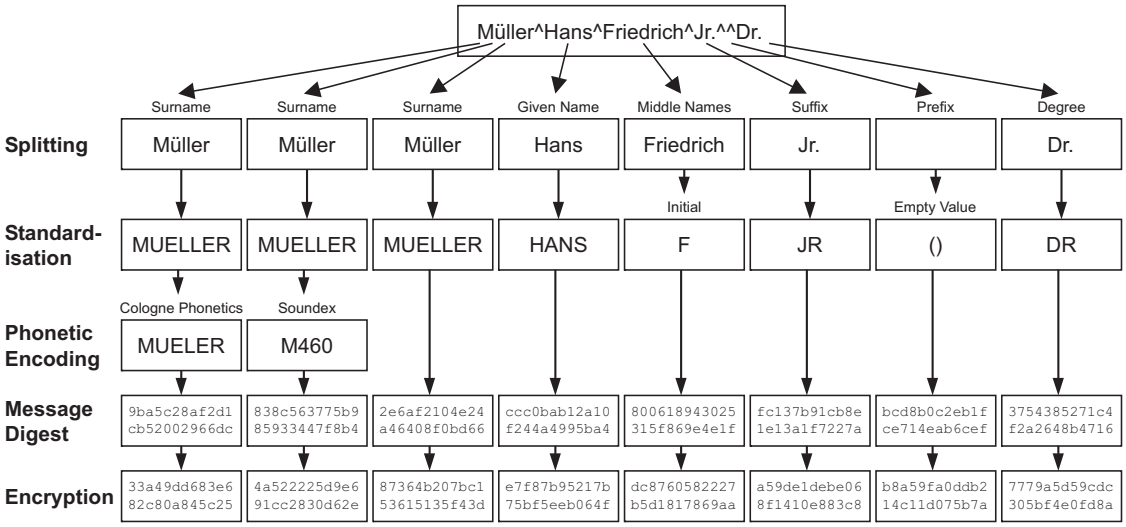


Figure 3.4: Steps From Plain Text Data to Control Numbers [29]

1. *Splitting* splits the demographic values into components which will be translated into different control numbers.

2. *Standardization* aligns the split values by converting them to uppercase and replacing language specific characters with the proper characters for cross-country use.

3. *Phonetic Encoding* is an optional step which encodes name values with phonetic encodings.

4. *Message Digest* performs an irreversible cryptographic algorithm on the values generated in the previous steps.

5. *Encryption* encrypts the message digest with a secret encryption key that needs to be known by all parties in the protocol.

The process prevents regenerating plain text data from control numbers, and the aim of the last step is to prevent control numbers from dictionary attacks. When a symmetric encryption with a secret key is used for the encryption process, it is necessary to change it regularly and keep it secure during distribution and in service [3]. Therefore it is essential to generate a new secret encryption key for each cross-community query. However, this solution is not suitable for our architecture because the super-peers should re-generate their indices each time a new key is used. This is inefficient when a super-peer owns a large amount of patient data. An alternate solution is to use asymmetric key algorithm during the encryption phase. In this case, a requester encrypts the message digest by using the public key of the super-peer, before sending the query to the super-peer. Since its private key is only accessed by the super-peer, the decryption of the control numbers by other parties is avoided. In this process forwarding a query through the super-peer back-bone requires an additional decryption/encryption phase as shown in Figure 3.5.

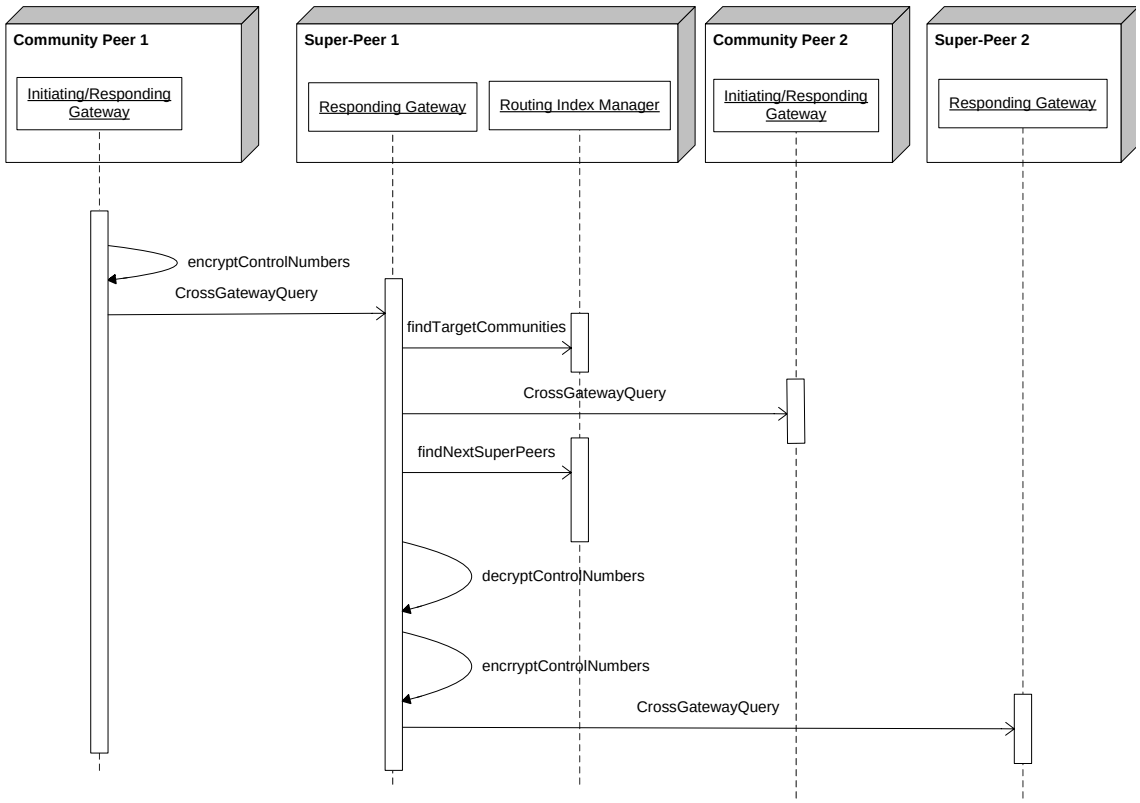


Figure 3.5: Cross Gateway Query using Control Numbers

In Figure 3.5, Cross Gateway Query transaction of IHE XCA Integration Profile is modified in such a way that it carries control numbers generated from the demographic information of the patient. A peer sends a request to its super-peer by encrypting the demographic information using the public key of its super-peer. Since the routing indices of the super-peer are generated using its public key, comparison of the control numbers in the request with the routing indices is possible. If the super peer finds out that a community peer such as "Community Peer 2" is holding information about the patient as a result of the comparison then it sends the request to the community peer using the Cross Gateway Query transaction. Before a super-peer forwards a request to another super-peer, the Responding Gateway first decrypts the patient control number it receives with its private key and it encrypts it with the public key of the target super-peer.

In our architecture, each community registers its patients to the network by generating control numbers for them. The registration process is performed between the "PID Feeder" Actor of the community peer and the "Routing Index Manager" Actor of its super-peer. For this purpose, following transactions between these actors are introduced :

1. *Register Patient Control Numbers* initiated by the PID Feeder to register a new patient by submitting its generated control numbers.

2. *Update Patient Control Numbers* initiated by the PID Feeder to update an existing patient by specifying its existing control numbers and new control numbers.

3. *Delete Patient Control Numbers* initiated by the PID Feeder to unregister an existing patient by specifying its control numbers.

When the "Routing Index Manager" of a super-peer is involved in one of these transactions, it updates its routing indices using the modified version of the method described in [67]. The control numbers involved in super-peer/super-peer indices should be updated in the super-peer backbone with an additional decryption/encryption step, since each super-peer has routing indices involving control numbers generated with its public key. When a query reaches the desired destination peer, next step is to find the local patient identity for the given control number. The PID protocol for finding local patient identifiers is described in [24].

3.3 Factors Affecting the Choice of a Super-peer

One of the important factors affecting the network performance is choosing the best fitting super-peer for a community peer. We assume that the community peers are organized around the super-peers by taking the jurisdictions and/or autonomous regions into account. Such a selection may decrease the number of routing index updates since it is likely that a patient mostly visits the healthcare service providers in his/her locality.

For example, let us assume that the communities A and B both connect to a super-peer S. If a patient P visits a healthcare institute in the community A for the first time, the super-peer/peer routing index of S is updated to indicate that the community A has the information about the patient P. Similarly, the super-peer/super-peer routing indices of the network are updated to indicate that the super-peer S has knowledge about the patient P. When the patient P visits the community B for the first time, the super-peer/peer routing index of S is updated to indicate that the community B also has information about the patient P. Fortunately, super-peer/super-peer indices do not require update since the network already knows that S has knowledge about patient P. Therefore, the network performance increases if the connected communities of a super-peer have overlapping patients.

3.4 Handling Different Vocabularies

Table 3.1 [51] describes some of the community specific attributes used in defining metadata of documents in the the IHE XDS Integration Profile. The documents in an XDS community are described and queried through these attributes. However, in a IHE XCA Profile implementation each community may use different coding schemes to assign values to these attributes. Therefore, a condition specifying a value for an attribute needs to be modified in the case of cross community queries. The proposed architecture achieves these modifications through performing vocabulary translations between the source and target communities.

To handle vocabulary translations, an actor named Vocabulary Translator Actor is introduced as shown in Figure 3.2. The Vocabulary Translator Actor of a community peer is responsible for storing mappings between the local vocabulary and the vocabulary of its super-peer. In this extension, an initiating gateway requests the transformed values of the parameters from the Vocabulary Translator Actor before forwarding the request to a Responding Gateway as shown in Figure 3.6. When a community peer receives a request from its super-peer, it transforms the values of the community specific metadata attributes to

Table 3.1: XDS Domain Specific Document Metadata Attributes

Attribute	**Definition**
classCode	The code specifying the particular kind of document (e.g. Prescription, Discharge Summary, Report).
confidentialityCode	The code specifying the level of confidentiality of the XDS Document.
eventCodeList	This list of codes represents the main clinical acts, such as a colonoscopy or an appendectomy, being documented. If one or more eventCodes are included, they shall not conflict with the values inherent in the classCode, practiceSettingCode or typeCode, as such a conflict would create an ambiguous situation. This short list of codes is provided to be used as key words for certain types of queries.
healthcareFacility TypeCode	This code represents the type of organizational setting of the clinical encounter during which the documented act occurred. The value shall not conflict with the value inherent in the typeCode, as such a conflict would create an ambiguous situation.
practiceSettingCode	The code specifying the clinical specialty where the act that resulted in the document was performed (e.g. Family Practice, Laboratory, Radiology).
typeCode	The code specifying the precise kind of document (e.g. Pulmonary History and Physical, Discharge Summary, Ultrasound Report).

local vocabulary.

Vocabulary mappings are first generated during the configuration of the Initiating Gateway Actor and the Responding Gateway Actor connections and maintained when necessary. When a community joins the network, it requests the vocabulary used by its super peer. The mapping is performed manually by the experts. The responsibilities of the Vocabulary Translator Actor of the community peer are as follows:

- Requesting vocabulary used in its super peer.
- Providing a vocabulary mapping interface for users who are responsible for mapping local vocabulary to vocabulary retrieved from the super peer.
- Storing the vocabulary mapping information.
- Providing a translation service to the Initiating Gateway Actor.

Note that each peer maps its vocabulary domain to its super-peer. Furthermore, a super

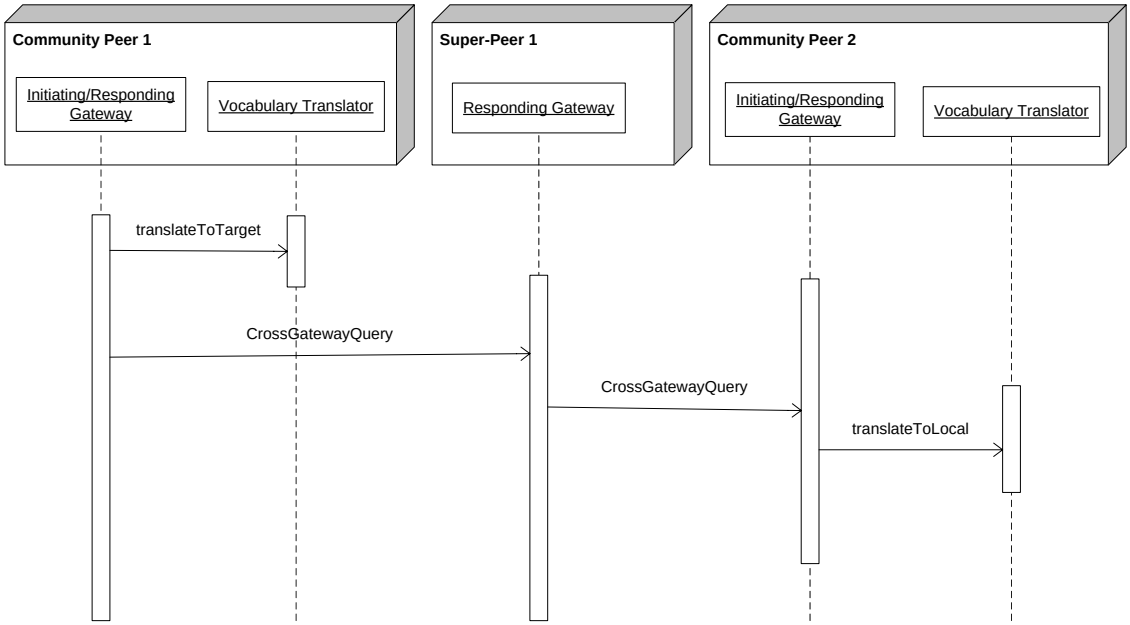


Figure 3.6: Vocabulary Translation between the Community Peer and Super-Peer

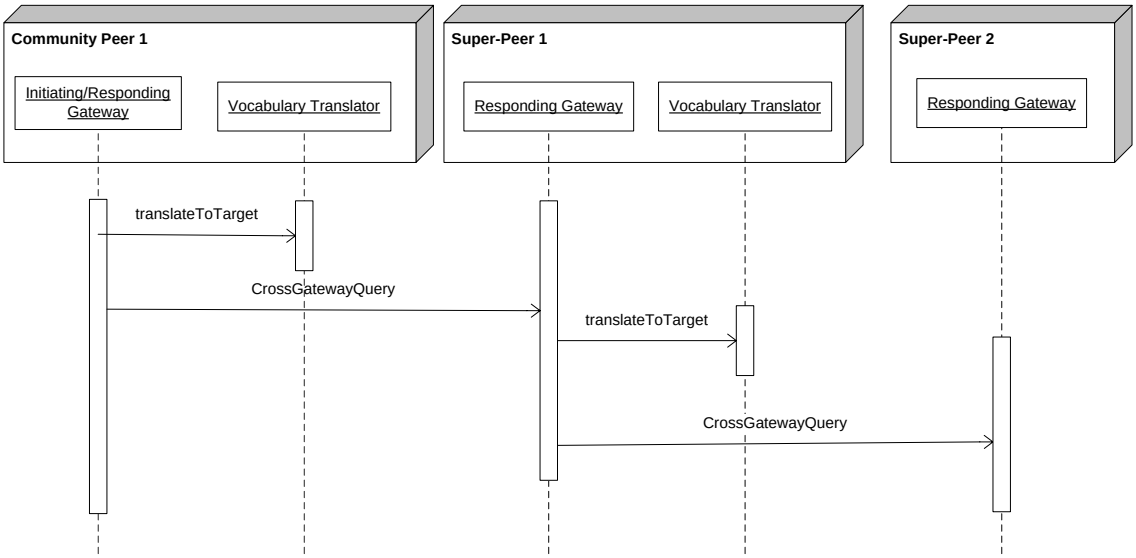


Figure 3.7: Vocabulary Translation between the Super-Peers

peer generates a mapping for its neighboring super-peers through its Vocabulary Translator Actor. Once the mappings are generated, community specific values of attributes are translated through the forwarding the process of messages as shown in Figure 3.7.

CHAPTER 4

ACHIEVING CLINICAL STATEMENT INTEROPERABILITY

Communities in a federation should be able to exchange information which implies that their EHR documents should be interoperable. In this chapter we propose a solution to achieve semantic interoperability between EHR standards by providing transformations from one clinical model to another. In order to address this need, we tackle the interoperability of EHR standards which are derived from the HL7 RIM. Then we demonstrate how HL7 CDA and CEN EN 13606-1 EHRcom clinical statement concepts are mapped to each other by using archetypes and semantic tools and techniques. Finally we show how clinical statement instance of one model transformed to clinical statement instance of another by using the mapping definitions defined in the previous step.

The R-MIM based mapping and transformation of two clinical statement instances that conform to different standards are accomplished in two phases as shown in Figure 4.1:

- In the first phase, the "Mapping Definitions" are produced between two archetypes which are based on different R-MIMs but express the same clinical concept. The classes of the source and the target archetypes are compared in order to discover the origins of the classes in the RIM to find out matching properties. Since this process involves reasoning, the OWL representations of the RIM, the R-MIMs (the source and the target) and the archetypes (the source and the target) are used. The mapping definitions produced in this phase are stored to be used later.

- In the second phase, the "Mapping Definitions" are used to transform one EHR instance to another. Since the source EHR instance is in XML format, first it is converted into an OWL instance. This process is called "Normalization". Given the archetype of the source EHR instance, the "Normalizer" tool implemented creates its instance in OWL.

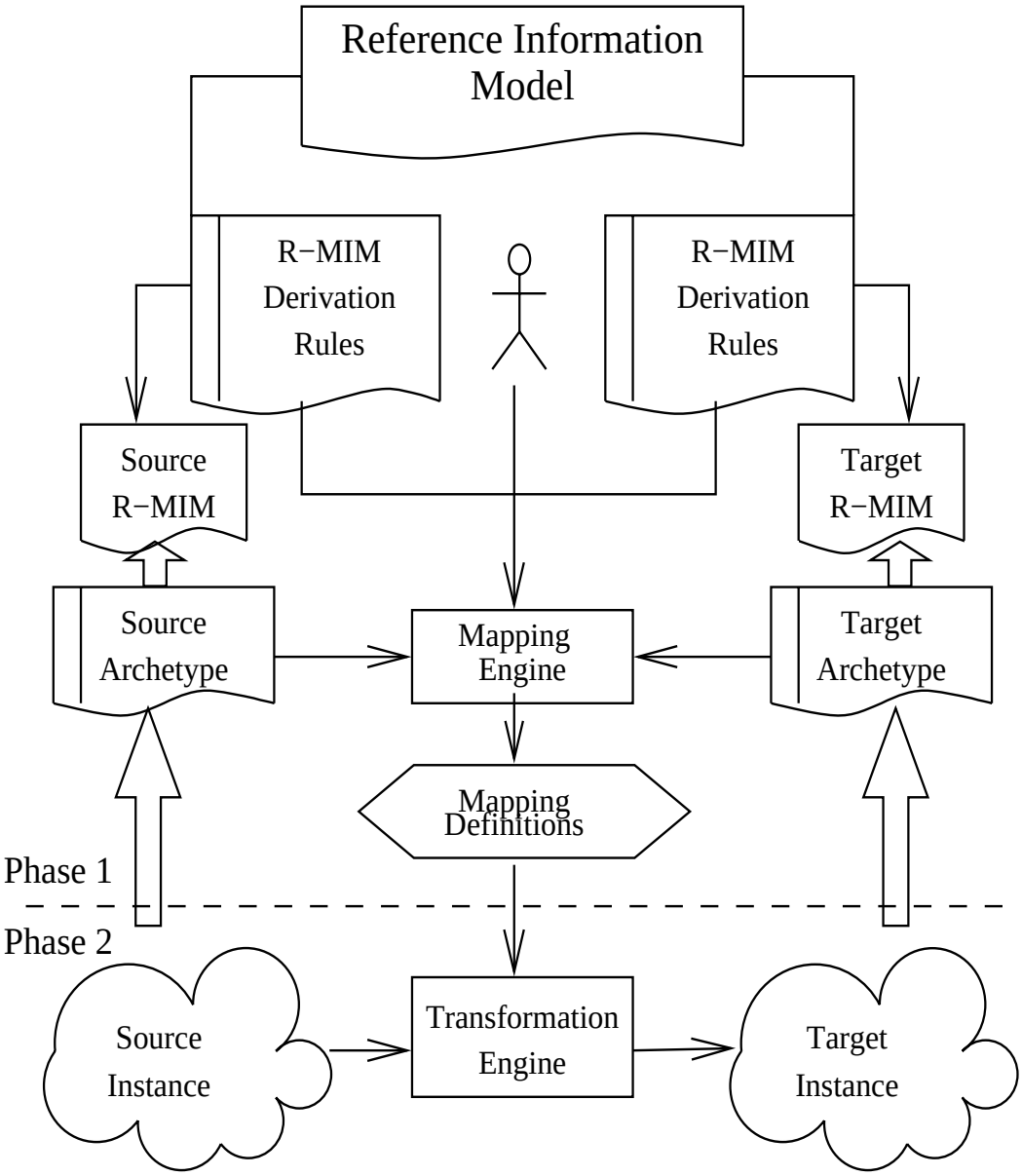


Figure 4.1: An Overview of the System Architecture

Then the "Transformation Engine" developed uses the "Mapping Definitions" to create the target EHR instance in OWL. The next step is de-normalizing the target EHR instance in OWL to XML format.

4.1 Deriving Archetype Property Mappings based on R-MIM Derivations

Property mappings between archetypes denoting the same clinical concept but constraining different R-MIMs are discovered based on the derivations of the R-MIMs from the RIM. Therefore, we first describe possible R-MIM derivations and then show how these derivations are used to discover property mappings between archetypes.

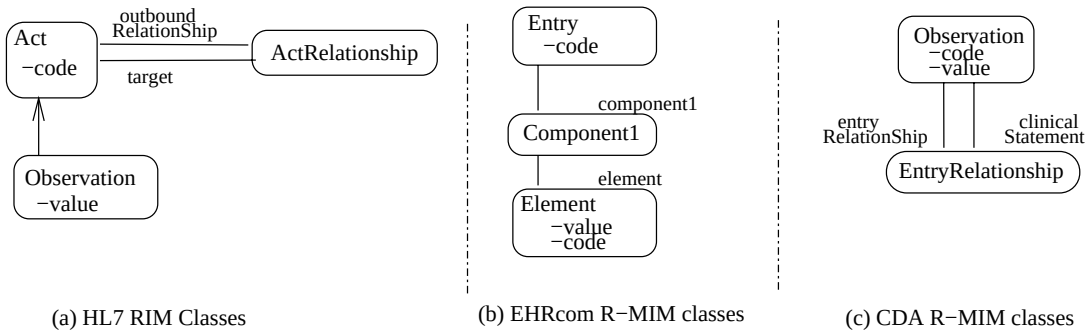
4.1.1 Deriving R-MIMs from the RIM

One of the building blocks used in deriving archetype property mappings is the R-MIM derivations from the RIM. An R-MIM is derived from the RIM by using well-defined set of rules. For example, assume B to be an R-MIM class which is derived from the class A of the RIM, then the following types of statements are applicable:

- Class B is derived from Class A of the RIM
- Class B does not inherit attribute x of Class A
- Class B inherits attribute x of Class A
  - The cardinality of attribute x is restricted
  - A constraint is defined on the type of attribute x
  - A constraint is defined on the value of attribute x
  - A constraint is defined on the range of attribute x
- Class B does not inherit association y of Class A
- Class B defines an association y_1 which is a specialization of association y of Class A
  - The Cardinality of association y_1 is restricted
  - A constraint is defined on the range of association y_1

As an example, consider a part of the HL7 RIM given in Figure 4.2 (a). A part of an EHRcom R-MIM derived from this RIM is given in Figure 4.2 (b). The EHRcom R-MIM is obtained from the RIM through the statements shown in Table 4.1.

Assume further that a part of an HL7 CDA R-MIM given in Figure 4.2 (c) is derived from the same RIM with the statements shown in Table 4.2.



(a) HL7 RIM Classes (b) EHRcom R−MIM classes (c) CDA R−MIM classes

Figure 4.2: A part of the HL7 RIM and the corresponding R-MIMs of EHRcom and CDA

Table 4.1: EHRcom R-MIM Derivation Statements

no.	Statement
1	The EHRcom "Entry" class is derived from the HL7 RIM "Act" class.
2	The EHRcom "Entry" class inherits the "code" attribute of the HL7 RIM "Act" class.
3	The EHRcom "Entry" class introduces an association, namely "component1" which is a specialization of the "outboundRelationship" association of the HL7 RIM "Act" class.
4	The range of the "component1" association in the EHRcom Entry class is restricted to the EHRcom "Component1" class.
5	The EHRcom "Component1" class is derived from the HL7 RIM "ActRelationship" class.
6	The EHRcom "Component1" class introduces an association, namely "element" which is a specialization of the "target" association of the HL7 RIM "ActRelationship" class.
7	The range of the "element" association in the EHRcom "Component1" class is restricted to the EHRcom "Element" class.
8	The EHRcom "Element" class is derived from the HL7 RIM "Observation" class.
9	The EHRcom "Element" class inherits the "code" attribute of the HL7 RIM "Act" class.
10	The EHRcom "Element" class inherits the "value" attribute of the HL7 RIM "Observation" class.
11	The maximum cardinality of the "value" attribute in the EHRcom "Element" is restricted to 1.
12	The minimum cardinality of the "value" attribute in the EHRcom "Element" is restricted to 1.

The R-MIM derivation statements identify the matching properties in the archetype-based mapping of the clinical statement instances. Since the process involves reasoning, the RIM, R-MIMs and the archetypes have to be represented in an ontology language. OWL is chosen as the ontology language, since the derivation statements can be represented in OWL. The EHRcom R-MIM and the HL7 CDA R-MIM are derived by specializing the classes and properties of the HL7 RIM. The OWL representations of the RIM (Figure 4.2 (a)) and the R-MIMs (Figure 4.2 (b) and (c)) are available in Appendix A. Archetypes are also represented in OWL. How to represent archetypes using OWL is described in Chapter 5.

Table 4.2: CDA R-MIM Derivation Statements

no.	Statement
1	The CDA “Observation" class is derived from the HL7 RIM “Observation" class.
2	The CDA “Observation" class inherits the “code" attribute of the HL7 RIM “Act" class.
3	The CDA “Observation" class inherits the “value" attribute of the HL7 RIM “Observation" class.
4	The CDA “Observation" class introduces an association called the “entryRelationship" which is a specialization of the “outboundRelationship" association of the HL7 RIM “Act" class.
5	The range of the “entryRelationship" association in the CDA “Observation" class is restricted to the CDA “EntryRelationship" class.
6	The CDA “EntryRelationship" class is derived from the HL7 RIM “ActRelationship" class.
7	The CDA “EntryRelationship" class introduces an association called the “clinicalStatement" which is a specialization of the “target" association of the HL7 RIM “ActRelationship" class.
8	The range of the “clinicalStatement" association in the CDA “EntryRelationship" class is restricted to the CDA “Observation" class.

4.1.2 Deriving Archetype Property Mappings

In order to transform the source clinical statement instances to target EHR instances, it is first necessary to map the archetype of the source instances to the archetype of target instances. When two archetypes are declared to be expressing the same clinical concept, it is necessary to compare classes of the source and target archetype in order to discover the origins of the classes in the RIM to find their matching properties. Figure 4.3 gives the algorithm realizing the property mappings. The notations used in the algorithms are shown in Table 4.3.

Note that there can always be mismatches between the source R-MIM and the target R-MIM. For example, the “targetSiteCode” attribute does not exist in EHRcom R-MIM; instead the “code” attribute is used. Therefore, in some cases modifications to the derived property mappings are necessary. In order to overcome such mismatches, following property modifications are allowed:

- If a source property does not have a corresponding target property then mark the source property to be discarded in the mapping process;
- If the target ontology and the source ontology use different properties to express the same information then allow a user-defined property mapping;
- If the value of a target property is constant and does not depend on the value of a source property then define it as a default value.

Table 4.3: Notations used in the Algorithms

Notation	**Description**
M_A	denotes an archetype mapping definition
M_P	denotes a property mapping definition
C_S	denotes a class in a source archetype
C_T	denotes a class in a target archetype
a_S	denotes a property in a source archetype
a_T	denotes a property in a target archetype
R_S	denotes a relation in a source archetype
R_T	denotes a relation in a target archetype
I_S	denotes an instance conforming to source archetype
I_T	denotes an instance conforming to target archetype
v_S	denotes a value of a property in a source instance
N	denotes a node in a path it can be a class, a relation or an attribute
$Path_N$	denotes a path to a node in an instance or class hierarchy
$omit(Path_N)$	indicates that the property mappings derived R-MIM derivations for the attribute specified by the path will be discarded
$Path_{a_S} \rightarrow Path_{a_T}$	denotes a property mapping definition from attribute a_S to attribute a_T in given paths
$defaultValue(Path_N)$	indicates that the attribute specified by the path has a default value
$userDefinedExists$ $(Path_{N_S} \rightarrow Path_{N_T})$	indicates a user defined property mapping exists from the given source path to the target path

During the mapping process, the property mappings are obtained from the R-MIM derivations by considering the defined property mapping modifications. The Property Mapping Algorithm (Figure 4.3) executes recursively and its initial parameters are as follows: the root class of source archetype C_S, the root class of target archetype C_T, the mapping definition of the archetypes M_A, the initial source path which involves only one node (C_S), the initial target path which involves only one node (C_T). The Property Mapping Algorithm is composed of two subroutines as shown in Figure 4.3. The "MapAttributes" routine is the main method of the algorithm. It traverses the source archetype hierarchy through the relations between classes starting from the root class of the archetype. For each class in the source archetype the "traverseTargetOnt" subroutine is called. The "traverseTargetOnt" subroutine traverses the target archetype hierarchy. For each class in the target hierarchy, it searches possible property mappings between its attributes and the attributes of the source class by taking the property mapping modifications into account.

The property mapping modifications are defined before executing the algorithm. The algorithm first considers the property mapping modifications for the source and the target property in question. If no such modification required for the source-target property pair,

```
1. mapAttributes(C_S,C_T, M_A, Path_{N_S}, Path_{N_T})
2. begin
3.   traverseTargetOnt(C_S,C_T, M_A, Path_{N_S}, Path_{N_T});
4.   forall R_S ∈ C_S do begin
5.     Path_{R_S} = appendNode(Path_{N_S},R_S);
6.     C_{R_S} = getClass(R_S);
7.     mapAttributes(C_{R_S},C_T, M_A, Path_{R_S}, Path_{N_T});
8.   end;
9. end;
```

```
1. traverseTargetOnt(C_S,C_T, M_A, Path_{N_S}, Path_{N_T})
2. begin
3.   for all a_S ∈ C_S do begin
4.     Path_{a_S} = appendNode(Path_{N_S}, a_S)
5.     //Ignore R-MIM derivations for a_S
6.     if omit(Path_{a_S}) do begin
7.       break;
8.     end;
9.     //Property mapping has been defined for a_S
10.    else if userDefinedExists(Path_{a_S} → Path_N ) do begin
11.      break;
12.    end;
13.    for all a_T ∈ C_T do begin
14.      Path_{a_T} = appendNode(Path_{N_T}, a_T)
15.      //Ignore R-MIM derivations for a_T
16.      if omit(Path_{a_T}) do begin
17.        break;
18.      end;
19.      //User defined mapping has been defined for a_T
20.      else if userDefinedExists(Path_N → Path_{a_T} ) do begin
21.        break;
22.      end;
23.      //Default value has been defined for a_S
24.      else if defaultValue(Path_{a_T} ) do begin
25.        break;
26.      end;
27.      //Derive mapping from R-MIM derivations
28.      else if correpondsTo(C_S.a_S,C_T.a_T ) do begin
29.        add(Path_{a_S} → Path_{a_T},M_A );
30.      end;
31.    end;
32.  end;
33.  forall R_T ∈ C_T do begin
34.    Path_{R_T} = appendNode(Path_{N_T},R_T);
35.    C_{R_T} = getClass(R_T);
36.    traverseTargetOnt(C_S,C_{R_T}, M_A, Path_{N_S}, Path_{R_T});
37.  end;
38. end;
```

Figure 4.3: Attribute Mapping Algorithm

then the algorithm checks whether these properties can be mapped through reasoning in OWL. The "correspondsTo" method checks whether the origins of the given properties are the same. If properties match then their mapping is stored in the "Mapping Definitions". After all the property mappings are discovered, the mappings are used in the EHR instance transformation.

The HL7 RIM defines two types of properties: attributes and associations. An association corresponds to a relation between two RIM classes and therefore it is represented as an object property in OWL. For example, the relation from the "Act" class to the "ActRelationship" class is denoted by the "outboundRelationship" association as shown in Figure 4.2 (a). On the other hand, an attribute can either hold a single value or it can have a complex structure. For example, the "code" attribute in Figure 4.2 (a) has a complex structure. Therefore attributes of the HL7 RIM are represented as object properties in OWL. Since the mapping is performed at attribute level in our algorithm, it is necessary to distinguish attributes and associations of a class. Thus two properties are added to the OWL representation of the RIM. All the object properties corresponding to the HL7 attributes are defined as subproperty of the "attribute" property and all the object properties corresponding to the HL7 associations are defined as the subproperty of the "relation" property.

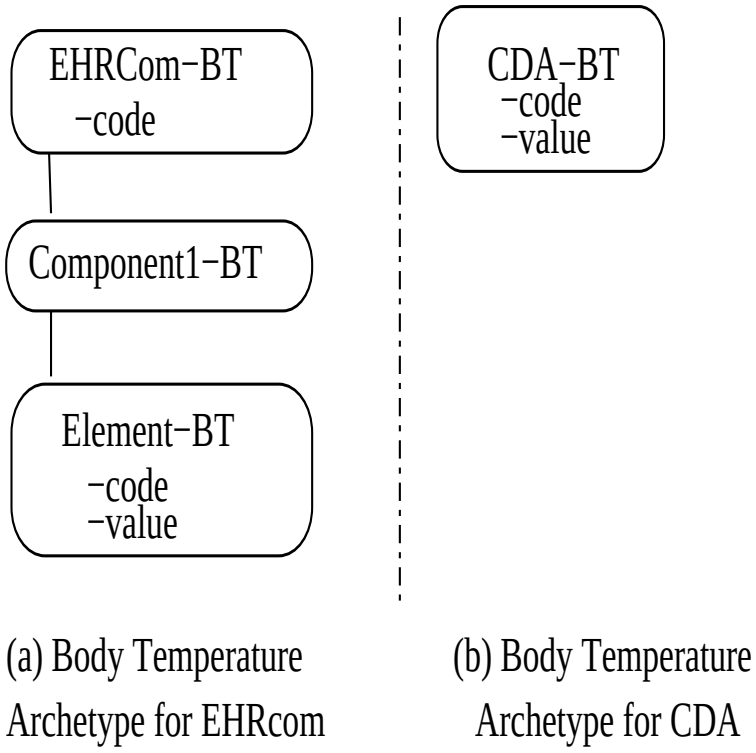


Figure 4.4: "Body Temperature" Archetypes based on EHRcom and HL7 CDA

The "attributes" and "relations" of a class are obtained through reasoning in OWL. The expression $a_S \in C_S$ states that the class C_S has an attribute named a_S. Such attributes are obtained by querying the knowledge base which contains the OWL representations of the

RIM, the R-MIMs and the archetypes. The expression $a_S \in C_S$ is true when the property a_S is a subproperty of the "attribute" property and the domain of the property is class C_S (or one of the superclasses of the class C_S). Furthermore the maximum cardinality of the a_S should not be restricted to "0". Restricting a property to have a "0" cardinality implies that the property is discarded from this class and all of its subclasses. Properties satisfying these conditions are obtained as the attributes of class C_S. The associations, in other words the relations of class C_S, are obtained from the properties which are subproperties of the "relation" property in a similar way.

For example, consider two archetypes given in Figure 4.4 (a) and (b) which define the same clinical concept, namely "Body Temperature", but constraining the EHRcom R-MIM and the HL7 CDA R-MIM respectively. When two archetypes are declared to express the same clinical concept, the algorithm starts by comparing the classes of the source and target archetypes to discover the origins of the properties in the classes in the RIM through reasoning in OWL. The OWL representations of the archetypes given in Figure 4.4 (a) and (b) are available in Appendix B. For the given archetypes, first the algorithm finds that the "EHRcom-BT" is a specialization of the "Entry" class of the EHRcom R-MIM and the "Entry" is a specialization of the "Act" class of the HL7 RIM. Therefore the "code" property whose domain is the "Act" class of the HL7 RIM is inherited by the "EHRcom-BT" class through specialization. Similarly, the "CDA-BT" class is found to be a specialization of the "Act" class of the HL7 RIM. For the same reason, the "code" property is inherited by the "CDA-BT" class. Since there is no restriction that limits the maximum cardinality of the "code" property to "0" in any of these classes and their super classes, the "code" attribute of the EHRcom-BT is discovered to match the "code" attribute of the "CDA-BT" by the Mapping Engine. The code attribute of "Element-BT" is also discovered to match the "code" attribute of the "CDA-BT". If the domain expert decides that one of the mappings is not suitable, the generation of the mapping can be prevented by marking the source property to be discarded.

In fact, during the property mapping process, a property is identified through a path involving all the relations from root class to the property. For example, the "value" attribute of the class "Element-BT" is denoted through the path "EHRcom-BT/component1/element/value". The property mappings are defined by matching the paths. For example, the "EHRcom-BT/component1/element/value" path matches the "CDA-BT/value" path since both "CDA-BT" and "EHRcom-BT" are eventually derived from the "Observation" class of the HL7 RIM.

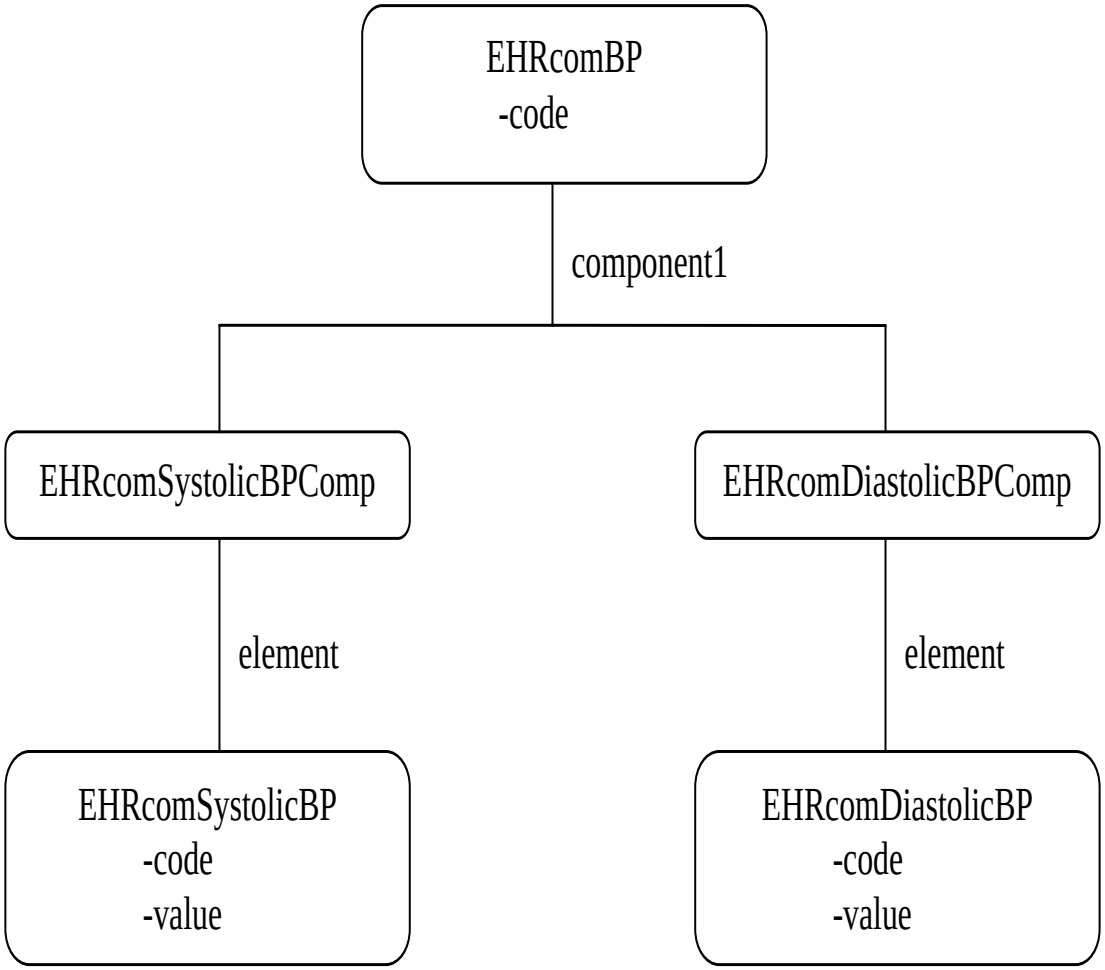


Figure 4.5: "Blood Pressure" Archetype based on EHRcom

In some cases, an archetype introduces multiple specializations of a class. In Figure 4.5, a class called "EHRcomBP" which specializes the "Entry" class of the EHRcom R-MIM in order to represent the "blood pressure" concept 's depicted. Furthermore, "EHRcomDiastolicBPComp" and "EHRcomSystolicBPComp" classes specialize the class "Component1" in order to denote relationships to relevant "Element" classes representing "Diastolic" and "Systolic" concepts. The path expression "EHRcomBP/component1" refers to both relationships. In order to differentiate the concepts, a notation is introduced in the path expressions such that the class denoting the concept is included in the brackets adjacent to the property node. For example, the "EHRcomBP/component1[EHRcomDiastolicBPComp]" path represents the relationship values which are instances of the "EHRcomDiastolicBPComp" class and the "EHRcomBP/component1[EHRcomSystolicBPComp]" path represents the relationship values which are instances of the "EHRcomSystolicBPComp" class.

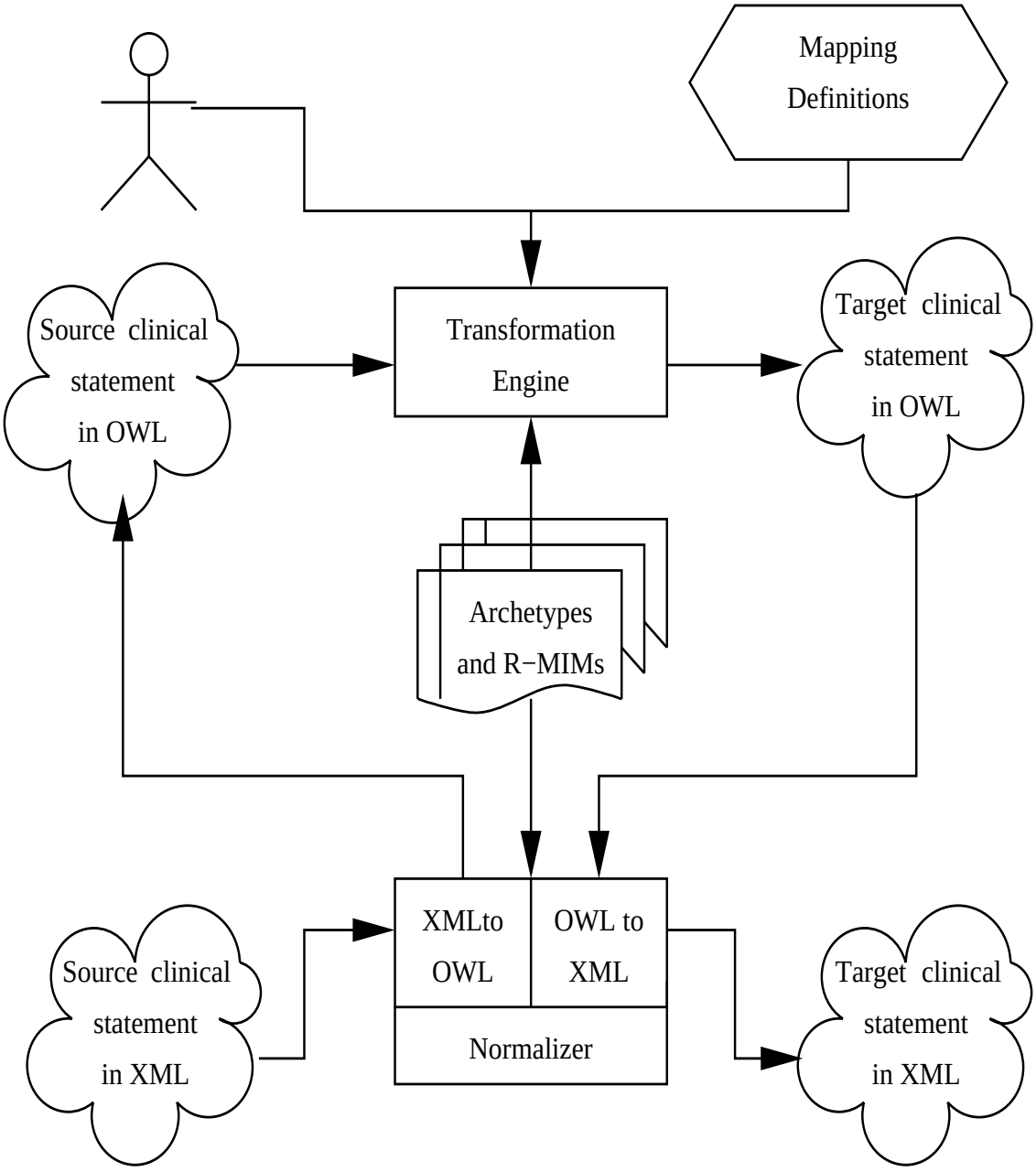


Figure 4.6: Instance Transformation Process

An overview of the instance transformation process is shown in Figure 4.6. The inputs of the process are the source clinical statement instance in XML format and the "Mapping Definitions". Since the clinical statement instance is in XML format, it is first converted

to OWL through the "XML to OWL Normalizer" component. In order to perform this conversion the "XML to OWL Normalizer" finds the archetype to which the source instance conforms to. This information is obtained from the instance which inherits the "templateId" attribute from "InfrastructureRoot" class in the HL7 RIM. Finding the archetype of the instance through more complex techniques is outside the scope of this work.

The OWL representation of a source instance is constructed from its archetype. The next step is finding the "Mapping Definitions" from the source archetype to the target archetype. The discovery is based on the semantics of the mapping definitions such as the archetypes involved, and the authorship. In [23], how ebXML Registry semantic constructs can be used for annotating, storing, discovering and retrieving archetypes based on their semantics is described. The same mechanisms are used for discovering the mapping definitions.

When such a mapping is found, the "Transformation Engine" starts creating the target EHR instance in OWL using the property mappings available in the "Mapping Definitions". After processing the property mappings, default values are assigned to the properties of the target instance. The next step is to transform the target OWL instance to XML format through the "OWL to XML Normalizer".

---

1. transformInstance(I_S,I_T, M_A, $Path_{N_S}$)
2. begin
3. forall $a_S \in I_S$ do begin
4. $Path_{a_S}$ = appendNode($Path_{N_S}$,a_S)
5. forall $M_P \in M_A$ do begin
6. if contains(M_P,$Path_{a_S}$) do begin
7. $Path_{a_T}$ = getTargetPath(M_P,$Path_{a_S}$);
8. transformAttribute(I_T,$Path_{a_T}$,a_S);
9. end;
10. end;
11. end;
12. forall $R_S \in I_S$ do begin
13. $Path_{R_S}$ = appendNode($Path_{N_S}$, R_S);
14. forall $v_S \in R_S$ do begin
15. transformInstance(v_S, I_T, M_A, $Path_{R_S}$);
16. end;
17. end;
18. end;

Figure 4.7: Instance Transformation Algorithm

A clinical statement instance has a hierarchical structure. Therefore the transformation starts from root object in the source instance hierarchy then the process recursively continues with the child objects in this hierarchy. Figure 4.7 depicts the instance transformation algorithm and in Figure 4.8, the subroutine which transforms values of a source property to the target property is given. Initial parameters of the algorithm shown in Figure 4.7 are the source instance I_S, a new instance of the root class of the target archetype I_T, the "Mapping Definitions" M_A and the path which involves only one node which is C_S.

1. transformAttribute(I_T,$Path_{a_T}$,a_S)
2. begin;
3. N_T = getLastNode($Path_{a_T}$);
4. while (a_S.cardinality > N_T.cardinality) do begin
5. N_T = getPreviousNode($Path_{a_T}$,N_T);
6. end;
7. $Path_{N_T}$ = getFirstPath($Path_{a_T}$,N_T);
8. forall $v_S \in a_S$ do begin
9. addValue(v_S,$Path_{N_T}$,$Path_{a_T}$)
10. end;
11. end;

Figure 4.8: Attribute Transformation Algorithm

```
<section>
  <code code="8716-3" codeSystem="2.16.840.1.113883.6.1" codeSystemName="LOINC"/>
  <title>Vital Signs</title>
  <text>Temperature is 36.9 C</text>
  <entry>
    <observation classCode="OBS" moodCode="EVN">
      <code code="386725007" codeSystem="2.16.840.1.113883.6.96"
            codeSystemName="SNOMED CT" displayName="Body temperature"/>
      <value xsi:type="PQ" value="36.9" unit="Cel"/>
    </observation>
  </entry>
</section>
```

Figure 4.9: An Example Clinical Statement in a HL7 CDA "section" [25]

```
<!DOCTYPE rdf:RDF [
        <!ENTITY rdf  'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
        <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
        <!ENTITY xsd  'http://www.w3.org/2001/XMLSchema#'>
        <!ENTITY owl  "http://www.w3.org/2002/07/owl#" >
        <!ENTITY rim "http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl#" >
        <!ENTITY cdabt "http://www.srdc.metu.edu.tr/ontologies/CDABodyTemperature.owl#" >

        ]>
<rdf:RDF xmlns="http://www.srdc.metu.edu.tr/instance.owl#"
 xmlns:base="http://www.srdc.metu.edu.tr/instance.owl#"
 xmlns:rdf="&rdf;" xmlns:rdfs="&rdfs;" xmlns:xsd="&xsd;" xmlns:owl="&owl;"
 xmlns:rim="&rim;" xmlns:cdabt="&cdabt;">
    <cdabt:CDA-BT rdf:ID="instance-CDABT">
        <rim:code rdf:resource="#code-BT" />
        <rim:value rdf:resource="#value-BT" />
        <rim:classCode rdf:resource="#classCode-OBS" />
        <rim:moodCode rdf:resource="#moodCode-EVN" />
    </cdabt:CDA-BT>
    <rim:CS rdf:ID="classCode-OBS">
        <rim:CD_code rdf:datatype="&xsd;string">OBS</rim:CD_code>
    </rim:CS>
    <rim:CS rdf:ID="moodCode-EVN">
        <rim:CD_code rdf:datatype="&xsd;string">EVN</rim:CD_code>
    </rim:CS>
    <rim:CD rdf:ID="code-BT">
        <rim:CD_code rdf:datatype="&xsd;string">386725007</rim:CD_code>
        <rim:CD_codeSystem rdf:datatype="&xsd;string">2.16.840.1.113883.6.96</rim:CD_codeSystem>
        <rim:CD_codeSystemName rdf:datatype="&xsd;string">SNOMED CT</rim:CD_codeSystemName>
    </rim:CD>
    <rim:PQ rdf:ID="value-BT">
        <rim:PQ_value rdf:datatype="&xsd;string">36.9</rim:PQ_value>
        <rim:PQ_unit rdf:datatype="&xsd;string">Cel</rim:PQ_unit>
    </rim:PQ>
</rdf:RDF>
```

Figure 4.10: OWL representation of the CDA Body Temperature Clinical Statement Instance

For each attribute of the source instance I_S, the "transformInstance" method given in Figure 4.7 checks whether an attribute mapping from the source attribute to an attribute of the target archetype I_T exists. If such a mapping is found, the "transformAttribute" subroutine given in Figure 4.8 is called to transform the value of the source attribute. The target attribute to which the source value assigned, is extracted from the attribute mapping definition. The "transformInstance" method also traverses the relations of the source instance I_S and recursively calls itself to transform the values of all the attributes of the source

hierarchy. Parameters of the "transformAttribute" subroutine shown in Figure 4.8 are the target instance I_T, path $Path_{a_T}$ identifying the target attribute and the source attribute a_S.

As an example of how the transformation process works, consider the CDA entry defined in Figure 4.9. This instance conforms to the CDA R-MIM archetype defined in Figure 4.4 (b). As a first step the clinical statement in XML format is converted to an OWL instance conforming its archetype ontology. OWL format of the CDA entry is given in Figure 4.10. Once the OWL instance created, it is loaded by the "Transformation Engine" for creating the target instance.

Second step is to retrieve the relevant archetype mappings from the "MappingEngine". Assume that the archetype mappings as described in Section 4.1 include the property mappings and the mapping modifications given in Figure 4.11.

1. CDA-BT/code ↔ EHRcom-BT/code
2. CDA-BT/code ↔ EHRcom-BT/component1/element/code
3. CDA-BT/value ↔ EHRcom-BT/component1/element/value
4. defaultValue(EHRcom-BT/classCode) = ACT.
5. defaultValue(EHRcom-BT/moodCode) = EVN.
6. defaultValue(EHRcom-BT/component1/typeCode) = COMP.
7. defaultValue(EHRcom-BT/component1/element/classCode) = OBS.
8. defaultValue(EHRcom-BT/component1/element/moodCode) = EVN.
9. defaultValue(CDA-BT/classCode) = OBS.
10. defaultValue(CDA-BT/moodCode) = EVN.

Figure 4.11: A Set of Example Property Mappings and Mapping Modifications

Once the archetype mappings are retrieved, the "Transformation Engine" starts creating the target OWL instance based on these mappings.The target instance creation steps are shown in Figure 4.12. Initially an empty instance of the root class of target archetype is created which is shown in Figure 4.12 (a). Then the algorithm traverses the attributes of the source instance looking for a property mapping for each attribute. As shown in Figure 4.12 (b), the "code" property of the "CDA-BT" instance is assigned to the "code" property of the "EHRcom-BT" instance as a consequence of the first property mapping. In

the second property mapping, an instance of the "Component1-BT" and an instance of the "Element-BT" are created and the "code" property of the "CDA-BT" class is assigned to the "code" property of the "Element-BT" instance as shown in Figure 4.12 (c). Finally, the third property mapping results in assignment of the "value" property of the "CDA-BT" instance to the "value" property of the "Element-BT" instance as shown in Figure 4.12 (d).

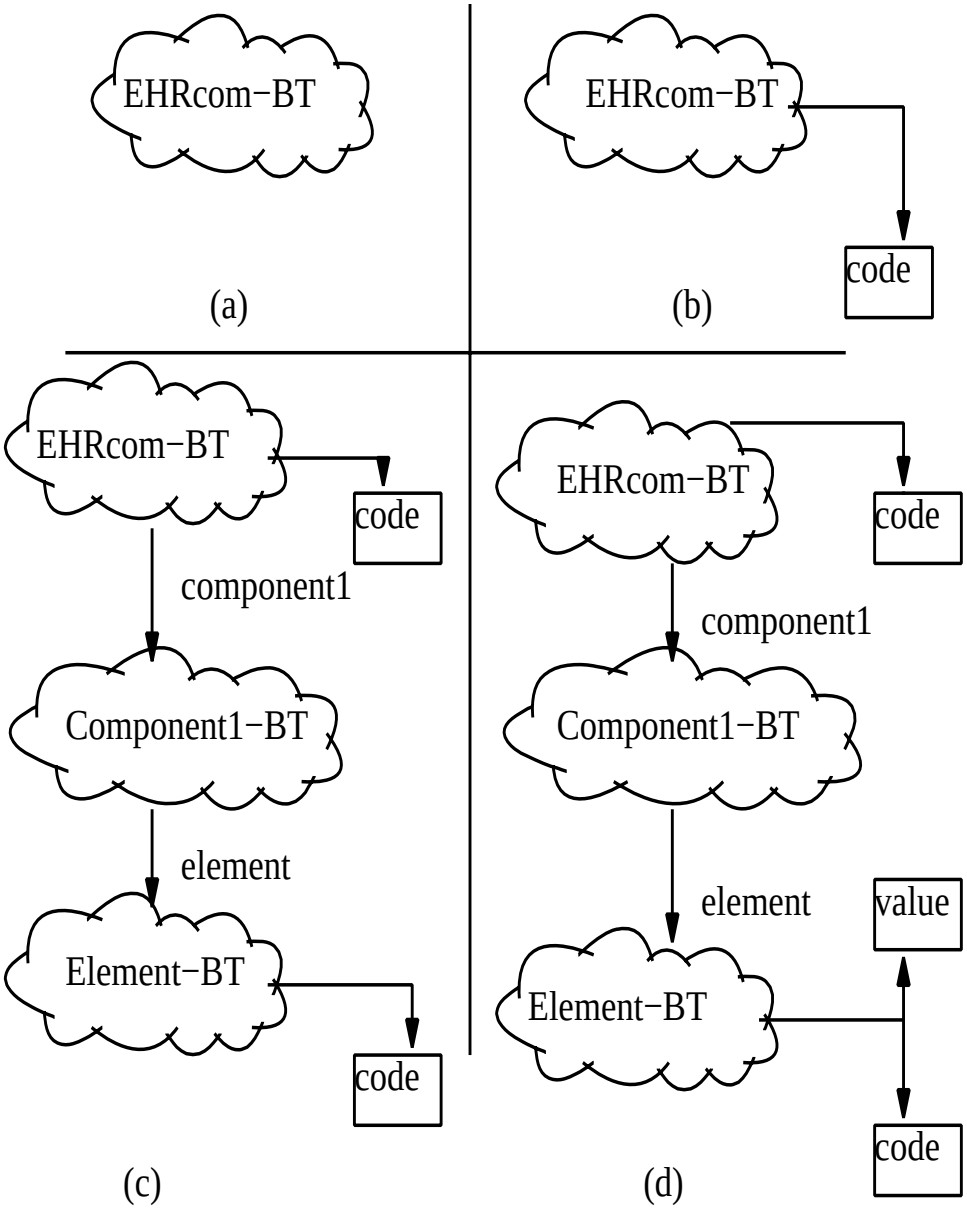


Figure 4.12: Target Instance Creation Steps

It should be noted that in some cases the cardinality of the property pairs do not match.

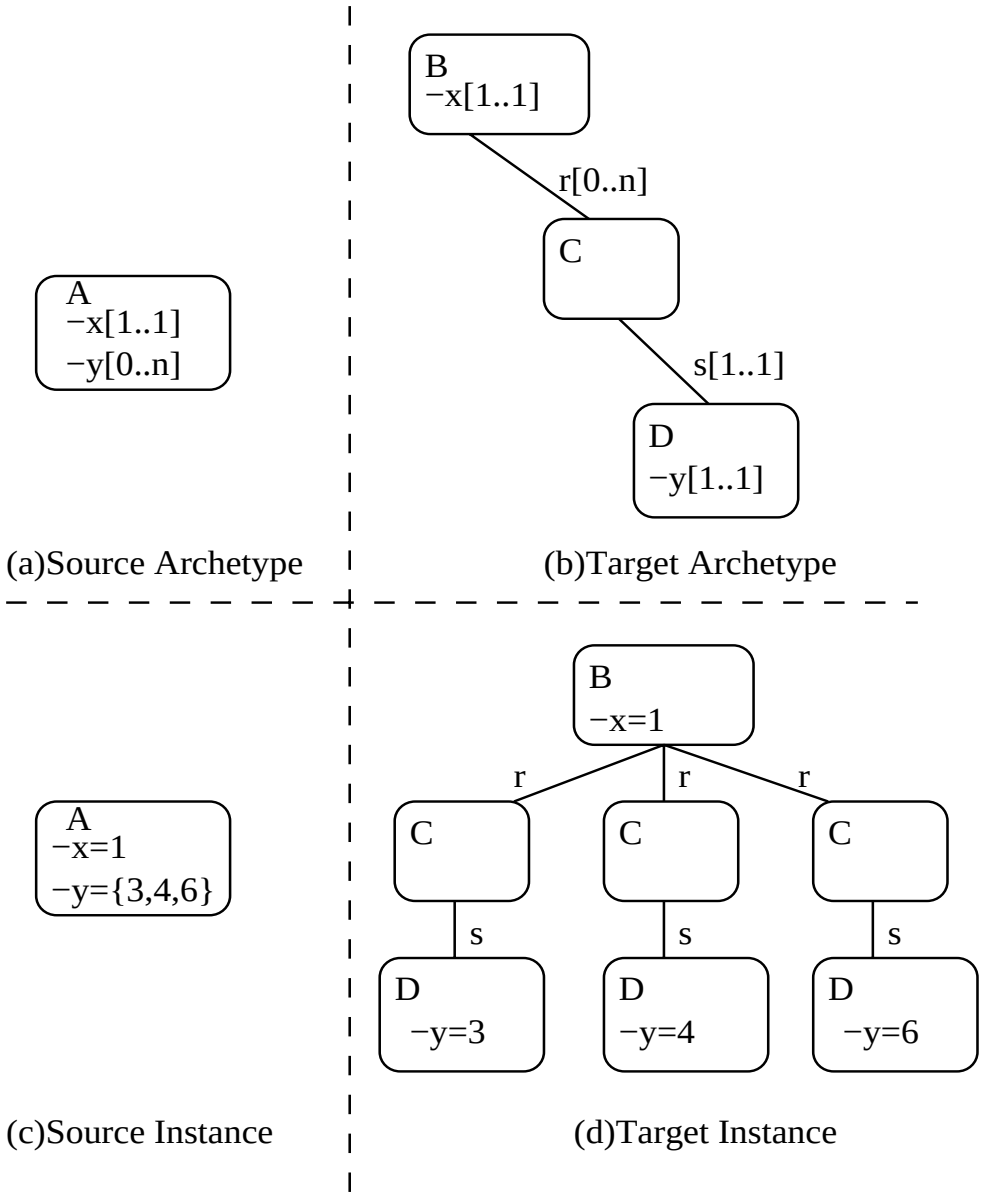


Figure 4.13: Cardinality Mismatch Example

For example, the "value" property of the "RegionOfInterest" class can have multiple values in the CDA R-MIM whereas the "value" property of the "Element" class is restricted to have only one value in the EHRcom R-MIM. To handle such cases, the algorithm shown in Figure 4.8 checks the cardinality of the relations taking part in the path of the target property. As an example, assume the path expression of the target property is "EHRcom-BT/component1[Component1-BT]/element[Element-BT]/value". The algorithm compares the cardinality of the source property with the cardinality of the properties in the target path from the leaf node (the "value" property) to the root class (the "EHRcom-BT" class).

First the algorithm compares the cardinality of the source property with the cardinality of the "value" property. If the cardinality of the "value" property does not match, then the cardinality of the "element" is compared. The process continues until a property which has a matching cardinality is found or the root class is reached.

1. defaultValue(EHRcom-BT/code.code) = 363816005.
2. defaultValue(EHRcom-BT/code.codeSystem) = 2.16.840.1.113883.6.96.
3. defaultValue(EHRcom-BT/code.codeSystemName) = SNOMED CT.
4. defaultValue(EHRcom-BT/code.displayName) = Body temperature measure.

Figure 4.14: Default values for the code property of EHRcom-BT class

Consider the source and target archetypes given in Figure 4.13 (a) and Figure 4.13 (b) respectively. Assume the source and target archetypes are representing the same concept for different R-MIMs and assume further that the property mappings "A/y→B/r[C]/s[D]/y" and "A/x→B/x" are obtained by the attribute mapping algorithm given in Figure 4.3. The cardinality of attribute x in class A is restricted to "1" and the attribute y does not have a cardinality restriction in class A. On the other hand the cardinality of the attribute y is restricted to "1" in class D of the target archetype. The property mapping defined by "A/y→B/r[C]/s[D]/y" expression has a cardinality mismatch since the cardinality of target property does not satisfy the cardinality of the source property. In order to overcome the cardinality mismatch problem, the attribute transformation algorithm seeks for an appropriate relation in the target path which satisfies the cardinality of the source property. The search process starts from the last node in the path and ends at the first node which is the root class of the archetype. If such a relation does not exist, then for each value in the source property, an instance of the root class is created and the source property value is assigned to the value in the target path for each instance. If a satisfying relation is found, such as relation "r" in the target archetype, then for each value in the source property a value for "r" is created and each source value is assigned to the target property reached from through instances of "r".

In Figure 4.13 (c) an instance conforming to the source archetype is depicted where attribute denoted by the path "A/y" has multiple values which are "3", "4" and "6". The

```
<rdf:RDF
    xmlns:arch="http://www.srdc.metu.edu.tr/ontologies/EHRcom-BT.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rmim="http://www.srdc.metu.edu.tr/ontologies/EHRcom.owl#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:rim="http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl#">
  <arch:EHRcom-BT rdf:ID="EHRcom-BT-instance">
    <rim:code>
      <rim:CD rdf:ID="code-BT">
        <rim:CD_code rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >363816005</rim:CD_code>
        <rim:CD_codeSystem rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >2.16.840.1.113883.6.96</rim:CD_codeSystem>
        <rim:CD_codeSystemName rdf:datatype="&xsd;string">SNOMED CT</rim:CD_codeSystemName>
    </rim:CD>
    <rmim:component1>
      <arch:EHRcom-BT-Comp rdf:ID="EHRcom-BT-Comp-instance-1">
        <rmim:element>
          <arch:EHRcom-BT-Element rdf:ID="EHRcom-BT-Element-instance-1">
            <rim:value>
              <rim:PQ rdf:ID="value-BT">
                <rim:PQ_value rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >36.9</rim:PQ_value>
                <rim:PQ_unit rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >Cel</rim:PQ_unit>
              </rim:PQ>
            </rim:value>
            <rim:code>
              <rim:CD rdf:ID="code-BT">
                <rim:CD_code rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >386725007</rim:CD_code>
                <rim:CD_codeSystem rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >2.16.840.1.113883.6.96</rim:CD_codeSystem>
        <rim:CD_codeSystemName rdf:datatype="&xsd;string">SNOMED CT</rim:CD_codeSystemName>
              </rim:CD>
            </rim:code>
          </arch:EHRcom-BT-Element>
        </rmim:element>
      </arch:EHRcom-BT-Comp>
    </rmim:component1>
    <rim:code rdf:resource="#code-BT"/>
  </arch:EHRcom-BT>
</rdf:RDF>
```

Figure 4.15: OWL representation of the EHRcom Body Temperature Clinical Statement

corresponding target instance is depicted in Figure 4.13 (d). Since the cardinality of attribute denoted by the path "B/r[C]/s[D]/y" is restricted to "1", the algorithm compares the cardinality of relation denoted by the path "B/r[C]/s[D]" with the cardinality of the source attribute. Unfortunately the comparison fails and the algorithm analyzes the previous relation in the path which is denoted by "B/r[C]". Since the the relation "B/r[C]" in target path satisfies the cardinality of the source attribute and the algorithm creates an instance of class "C" and an instance of class "D" for each value of the source attribute as shown in Figure 4.13 (d).

Next step of the instance transformation process is the assignment of default values which are specified during the archetype mapping process. These values are assigned to the attributes of the target instance based on the archetype mapping definition. The default values provided during the mapping phase of "Body Temperature" archetypes are shown in Figure 4.14. These values are assigned to relevant attributes automatically during the transformation phase of the clinical statement instance.

Final step of the instance transformation process is the denormalizing the OWL instance to XML format. The created EHRcom-BT instance created in OWL is shown in Figure 4.15. The converted XML fragment of the created EHRcom-BT instance is shown in Figure 4.16. The default values shown in Figure 4.14 are assigned to the "code" element of "cen_entry" element.

```
<cen_entry classCode="ACT" moodCode="EVN">
  <code code="363816005" codeSystem="2.16.840.1.113883.6.96"
            codeSystemName="SNOMED CT" displayName="Body temperature measure"/>
  <component1 typeCode="COMP">
    <element classCode="OBS" moodCode="EVN">
      <code code="386725007" codeSystem="2.16.840.1.113883.6.96"
            codeSystemName="SNOMED CT" displayName="Body temperature"/>
      <value xsi:type="PQ" value="36.9" unit="Cel"/>
    </element>
  </component1>
</cen_entry>
```

Figure 4.16: Body Temperature instance of EHRcom

4.2.1 EHRcom to CDA Clinical Statement Transformation Example

In this section, we demonstrate transforming clinical statements represented by EHRcom into its corresponding HL7 CDA R2 instance through an example EHRcom Entry which is shown in Figure 4.17.

The root object of the "Blood Pressure" clinical statement is the instance of the "Entry" class of the EHRcom R-MIM. It has an association which holds two instances of "Component1" class and each instance of "Component1" class has an association holding one instance of "Element"class of the R-MIM. It should be noted that the "Entry" and the "Element" classes are derivations of the "Act" and the "Observation" classes of the HL7 RIM

```
<cen_entry classCode="ACT" moodCode="EVN">
  <code code="251076008" codeSystem="2.16.840.1.113883.6.96"
        codeSystemName="SNOMED CT" displayName="Cuff blood pressure"/>
  <component1 typeCode="COMP">
    <element classCode="OBS" moodCode="EVN">
      <code code="271649006" codeSystem="2.16.840.1.113883.6.96"
            codeSystemName="SNOMED CT" displayName="Systolic BP"/>
      <value xsi:type="PQ" value="132" unit="mm[Hg]"/>
    </element>
  </component1>
  <component1 typeCode="COMP">
    <element classCode="OBS" moodCode="EVN">
      <code code="271650006" codeSystem="2.16.840.1.113883.6.96"
            codeSystemName="SNOMED CT" displayName="Diastolic BP"/>
      <value xsi:type="PQ" value="86" unit="mm[Hg]"/>
    </element>
  </component1>
</cen_entry>
```

Figure 4.17: Example EHRcom "Blood Pressure" Clinical Statement Instance

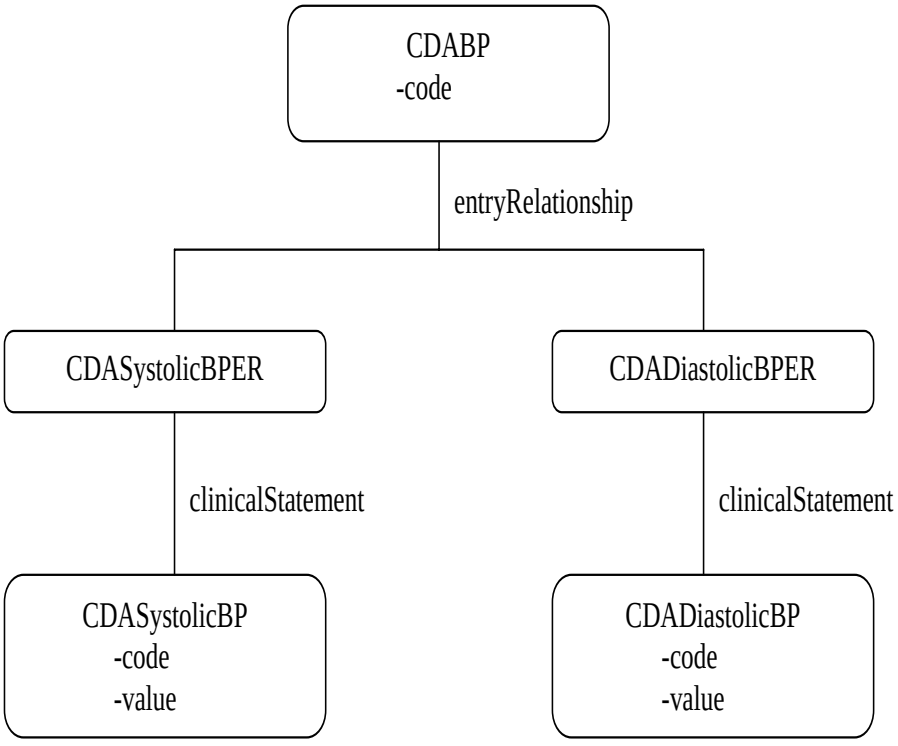


Figure 4.18: "Blood Pressure" Archetype based on EHRcom

respectively. The "Component1" class, which relates these two classes, is a derivation of the "ActRelationship" class of the HL7 RIM. The "Blood Pressure" clinical statement instance conforms to the archetype shown in Figure 4.5. In this archetype, the "EHRcomBP" class specializes the "Entry" class by restricting the range of the "component1" property to the union of "EHRcomSystolicBPComp" and "EHRcomDiastolicBPComp" classes. Both of these classes specialize the "Component1" class of the R-MIM by restricting the range of the target property to "EHRcomDiastolicBP" and "EHRcomSystolicBP" classes, respectively.

In Figure 4.5, the archetype which defines the blood pressure concept by specializing the EHRcom R-MIM classes is represented. An archetype specializes R-MIM classes by restricting the properties of the R-MIM. The "EHRcomBP" class specializes the "Entry" class of the EHRcom R-MIM by restricting the range of the "component1" property to the union of "EHRcomSystolicBPComp" and "EHRcomDiastolicBPComp" classes. Both of these classes specialize the "Component1" class of the R-MIM by restricting the range of the element property to "EHRcomDiastolicBP" and "EHRcomSystolicBP" classes, respectively. These specialized "Element" classes further restrict the range of the "code" properties. These restrictions can be observed in the OWL representation of the archetype which is available in Appendix B.

---

1. EHRcomBP/code ↔ CDABP/code
2. EHRcomBP/component1[EHRcomSystolicBPComp]/element[EHRcomSystolicBP]/code ↔ CDABP/entryRelationship[CDASystolicBPER]/clinicalStatement[CDASystolicBP]/code
3. EHRcomBP/component1[EHRcomSystolicBPComp]/element[EHRcomSystolicBP]/value ↔ CDABP/entryRelationship[CDASystolicBPER]/clinicalStatement[CDASystolicBP]/value
4. EHRcomBP/component1[EHRcomDiastolicBPComp]/element[EHRcomDiastolicBP]/code ↔ CDABP/entryRelationship[CDADiastolicBPER]/clinicalStatement[CDADiastolicBP]/code
5. EHRcomBP/component1[EHRcomDiastolicBPComp]/element[EHRcomDiastolicBP]/value ↔ CDABP/entryRelationship[CDADiastolicBPER]/clinicalStatement[CDADiastolicBP]/value

---

Figure 4.19: Mapping definitions of the blood pressure archetypes of EHRcom and CDA R-MIMs

Corresponding archetype of the blood pressure concept for HL7 CDA is given in Figure 4.18. The archetype introduces a class named "CDABP" which specializes the "Observa-

```
<rdf:RDF xmlns="http://www.srdc.metu.edu.tr/instance.owl#" xmlns:base="http://www.srdc.metu.edu.tr/instance.owl#"
    xmlns:rdf="&rdf;" xmlns:rdfs="&rdfs;" xmlns:xsd="&xsd;" xmlns:owl="&owl;"
    xmlns:rim="&rim;" xmlns:ehrcom="&ehrcom;" xmlns:ehrcombp="&ehrcombp;">
    <ehrcombp:EHRcom-BP rdf:ID="EHRcomBP-instance">
        <rim:code rdf:resource="#code-CuffBP" />
        <rim:classCode rdf:resource="#classCode-ACT" />
        <rim:moodCode rdf:resource="#moodCode-EVN" />
        <ehrcom:component1 rdf:resource="#EHRcomSystolicBPComp-instance" />
        <ehrcom:component1 rdf:resource="#EHRcomDiastolicBPComp-instance" />
    </ehrcombp:EHRcom-BP>
    <ehrcombp:EHRcomSystolicBPComp rdf:ID="EHRcomSystolicBPComp-instance">
    <ehrcom:element>
      <ehrcombp:EHRcom-Systolic-BP rdf:ID="EHRcom-Systolic-BP-instance">
        <rim:code>
          <rim:CD rdf:ID="Systolic-BP-code">
            <rim:CD_code rdf:datatype="http://www.w3.org/2001/XMLSchema#string">271649006</rim:CD_code>
            <rim:CD_codeSystem rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2.16.840.1.113883.6.96</rim:CD_codeSystem>
            <rim:CD_codeSystemName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">SNOMED CT</rim:CD_codeSystemName>
          </rim:CD>
        </rim:code>
        <rim:value>
          <owl:Thing rdf:ID="Systolic-BP-value">
            <rim:PQ_value rdf:datatype="http://www.w3.org/2001/XMLSchema#string">132</rim:PQ_value>
            <rim:PQ_unit rdf:datatype="http://www.w3.org/2001/XMLSchema#string">mm[Hg]</rim:PQ_unit>
          </owl:Thing>
        </rim:value>
      </ehrcombp:EHRcom-Systolic-BP>
    </ehrcom:element>
    </ehrcombp:EHRcomSystolicBPComp>
    <ehrcombp:EHRcomDiastolicBPComp rdf:ID="EHRcomDiastolicBPComp-instance">
    <ehrcom:element>
      <ehrcombp:EHRcom-Diastolic-BP rdf:ID="EHRcom-Diastolic-BP-instance">
        <rim:code>
          <rim:CD rdf:ID="Diastolic-BP-code">
            <rim:CD_code rdf:datatype="http://www.w3.org/2001/XMLSchema#string">271650006</rim:CD_code>
            <rim:CD_codeSystem rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2.16.840.1.113883.6.96</rim:CD_codeSystem>
            <rim:CD_codeSystemName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">SNOMED CT</rim:CD_codeSystemName>
          </rim:CD>
        </rim:code>
        <rim:value>
          <owl:Thing rdf:ID="Diastolic-BP-value">
            <rim:PQ_value rdf:datatype="http://www.w3.org/2001/XMLSchema#string">86</rim:PQ_value>
            <rim:PQ_unit rdf:datatype="http://www.w3.org/2001/XMLSchema#string">mm[Hg]</rim:PQ_unit>
          </owl:Thing>
        </rim:value>
      </ehrcombp:EHRcom-Diastolic-BP>
    </ehrcom:element>
    </ehrcombp:EHRcomDiastolicBPComp>
    <rim:CS rdf:ID="classCode-ACT">
        <rim:CD_code rdf:datatype="&xsd;string">ACT</rim:CD_code>
    </rim:CS>
    <rim:CS rdf:ID="moodCode-EVN">
        <rim:CD_code rdf:datatype="&xsd;string">EVN</rim:CD_code>
    </rim:CS>
    <rim:CD rdf:ID="code-CuffBP">
        <rim:CD_code rdf:datatype="&xsd;string">251076008</rim:CD_code>
        <rim:CD_codeSystem rdf:datatype="&xsd;string">2.16.840.1.113883.6.96</rim:CD_codeSystem>
        <rim:CD_codeSystemName rdf:datatype="&xsd;string">SNOMED CT</rim:CD_codeSystemName>
    </rim:CD>
</rdf:RDF>
```

Figure 4.20: OWL representation of the EHRcom “Blood Pressure” Clinical Statement

tion" class of the CDA R-MIM by restricting the range of the "entryRelationship" property to the union of "CDASystolicBPER" and "CDADiastolicBPER" classes.

The "CDASystolicBP" and "CDADiastolicBP" classes also specialize the "Observation" class of the CDA R-MIM by restricting the relevant inherited properties. Similarly, "CDASystolicBPER" and "CDADiastolicBPER" classes specialize the "EntryRelationship" class of the CDA R-MIM.

Once the blood pressure archetypes of both R-MIM are declared to represent the same concept through the "Mapping Engine". The "Mapping Engine" aids to generate property mappings for these archetypes. Assume the mapping definitions shown in Figure 4.19 has been generated for the blood pressure archetypes of both R-MIM.

According to the mapping definitions, "code" attribute of "EHRcomBP" matches the 'code" attribute of "CDABP". The "code" and "value" attributes of "EHRcomDiastolicBP" class matches the "code" and "value" attributes of "CDADiastolicBP', respectively. Similarly, the "code" and "value" attributes of "EHRcomSystolicBP" class matches the "code" and "value" attributes of "CDASystolicBP", respectively.

As mentioned, first step of the transformation is to normalize the clinical statement into OWL format. The "Normalizer" performs this process by using the archetype of the clinical statement. That is, clinical statement instance in OWL created by using the classes introduced in the archetype. For example, "cen_entry" element normalized as an instance of "EHRcomBP" class which specializes the "Entry" class of R-MIM. Similarly, "component1" and "element" tags are normalized using their corresponding classes introduced in the archetype. Normalized EHRcom "Blood Pressure" clinical statement instance in OWL is shown in Figure 4.20.

Once the normalized source instance generated, the "Transformation Engine" starts creating the target clinical statement instance using the mapping definitions shown in Figure 4.19. Initially an instance of "CDABP" class which is root class of the target archetype is created. The "code " attribute of this class gets its value from the "code" attribute of the "EHRcomBP" class as a consequence of the first property mapping. Second property mapping results in creation of "CDASystolicBP" and "CDASystolicBPER" instances and assigns the value of the "code" attribute of "EHRcomSystolicBP' to the "code" attribute of "CDASystolicBP". Third property mapping assigns the value of the "value" attribute of "EHRcomSystolicBP' to the "value" attribute of "CDASystolicBP". "CDADiastolicBP" and "CDADiastolicBPER" instances are created as a result of the fourth property mapping which assigns the value of

```
...
<rdf:RDF
    xmlns:arch="http://www.srdc.metu.edu.tr/ontologies/CDABloodPressure.owl#"
    xmlns:rmim="http://www.srdc.metu.edu.tr/ontologies/CDA.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:rim="http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl#">
  <arch:CDASystolicBP rdf:ID="CDASystolicBP-instance-1">
    <rim:code>
      <rim:CD rdf:ID="Systolic-BP-code">
        <rim:CD_code rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >271649006</rim:CD_code>
        <rim:CD_codeSystem rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2.16.840.1.113883.6.96</rim:CD_codeSystem>
        <rim:CD_codeSystemName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">SNOMED CT</rim:CD_codeSystemName>
      </rim:CD>
    </rim:code>
    <rim:value>
      <rim:PQ rdf:ID="Systolic-BP-value">
        <rim:PQ_value rdf:datatype="http://www.w3.org/2001/XMLSchema#string">132</rim:PQ_value>
        <rim:PQ_unit rdf:datatype="http://www.w3.org/2001/XMLSchema#string">mm[Hg]</rim:PQ_unit>
      </rim:PQ>
    </rim:value>
  </arch:CDASystolicBP>
  <arch:CDADiastolicBPER rdf:ID="CDADiastolicBPER-instance-1">
    <rmim:clinicalStatement>
      <arch:CDADiastolicBP rdf:ID="CDADiastolicBP-instance-1">
        <rim:value>
          <rim:PQ rdf:ID="Diastolic-BP-value">
            <rim:PQ_value rdf:datatype="http://www.w3.org/2001/XMLSchema#string">86</rim:PQ_value>
            <rim:PQ_unit rdf:datatype="http://www.w3.org/2001/XMLSchema#string">mm[Hg]</rim:PQ_unit>
          </rim:PQ>
        </rim:value>
        <rim:code>
          <rim:CD rdf:ID="Diastolic-BP-code">
            <rim:CD_code rdf:datatype="http://www.w3.org/2001/XMLSchema#string">271650006</rim:CD_code>
            <rim:CD_codeSystem rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2.16.840.1.113883.6.96</rim:CD_codeSystem>
            <rim:CD_codeSystemName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">SNOMED CT</rim:CD_codeSystemName>
          </rim:CD>
        </rim:code>
      </arch:CDADiastolicBP>
    </rmim:clinicalStatement>
  </arch:CDADiastolicBPER>
  <rim:CD rdf:ID="code-CuffBP">
    <rim:CD_code rdf:datatype="http://www.w3.org/2001/XMLSchema#string">251076008</rim:CD_code>
    <rim:CD_codeSystem rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2.16.840.1.113883.6.96</rim:CD_codeSystem>
    <rim:CD_codeSystemName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">SNOMED CT</rim:CD_codeSystemName>
  </rim:CD>
  <arch:CDA-BP rdf:ID="CDA-BP-instance">
    <rmim:entryRelationship rdf:resource="#CDADiastolicBPER-instance-1"/>
    <rmim:entryRelationship>
      <arch:CDASystolicBPER rdf:ID="CDASystolicBPER-instance-1">
        <rmim:clinicalStatement rdf:resource="#CDASystolicBP-instance-1"/>
      </arch:CDASystolicBPER>
    </rmim:entryRelationship>
    <rim:code rdf:resource="#code-CuffBP"/>
  </arch:CDA-BP>
</rdf:RDF>
```

Figure 4.21: OWL representation of the CDA "Blood Pressure" Clinical Statement

the "code" attribute of "EHRcomDiastolicBP' to the "code" attribute of "CDADiastolicBP". Finally the last property mapping assigns the value of the "value" attribute of "EHRcomDiastolicBP' to the "value" attribute of "CDADiastolicBP".

The "Transformation Engine" generates the target instance in OWL. Generated CDA "Blood Pressure" clinical statement instance in OWL is shown in Figure 4.21. Final step of the transformation process is to denormalize the target instance in OWL to XML format. The result of the transformation process based is given in Figure 4.22.

```
<observation classCode="OBS" moodCode="EVN">
  <code code="251076008" codeSystem="2.16.840.1.113883.6.96"
      codeSystemName="SNOMED CT" displayName="Cuff blood pressure"/>
  <entryRelationship typeCode="COMP">
    <observation classCode="OBS" moodCode="EVN">
      <code code="271649006" codeSystem="2.16.840.1.113883.6.96"
            codeSystemName="SNOMED CT" displayName="Systolic BP"/>
      <value xsi:type="PQ" value="132" unit="mm[Hg]"/>
    </observation>
  </entryRelationship>
  <entryRelationship typeCode="COMP">
    <observation classCode="OBS" moodCode="EVN">
      <code code="271650006" codeSystem="2.16.840.1.113883.6.96"
            codeSystemName="SNOMED CT" displayName="Diastolic BP"/>
      <value xsi:type="PQ" value="86" unit="mm[Hg]"/>
    </observation>
  </entryRelationship>
</observation>
```

Figure 4.22: CDA construct of CEN Entry

CHAPTER 5

REPRESENTING ARCHETYPES IN OWL

In this section, we will present how archetype constraints are represented in OWL by mapping ADL construct of each archetype constraint to its OWL counterpart. In [43] and [68], the OWL representations of some archetypes are given. However, we feel it necessary to give a formal description of how the ADL constructs can be mapped to OWL. As mentioned in Section 2.2.3, archetypes specialize the classes of the generic information model by constraining their attributes. The applicable constraints are as follows [9]:

- Constraints on the range of data-valued properties
- Constraints on the range of object-valued properties
- Constraints on the "existence" of a property indicating whether the property is optional or mandatory .
- Constraints on the "cardinality" of a property indicating whether the property refers to a container type, the number of member items it must have and their optionality, and whether it has a "list" or a "set" structure.
- Constraints on a property with "occurrences" indicating how many times in runtime data an instance of a given class conforming to a particular constraint can occur. It only has significance for objects, which are children of a container property.

It is also possible to reuse previously defined archetypes and archetype fragments. There are two constructs for this purpose: The first one is the "use_node" construct, which is used to reference an archetype fragment by a path expression. The "use_node" references an archetype fragment within the archetype. The second one is the "allow_archetype" construct, which is used to reference other archetypes by defining a criteria for allowable archetypes.

As described in [9], the first step in representing archetypes in OWL is to construct the reference information model of the domain in OWL. A simple algorithm for object model to OWL mapping is given in [43].

- Represent each class in the reference information model as an OWL class.
- Represent each relationship in the reference information model as an ObjectProperty in OWL.
- Represent each data-valued property in the reference information model as a Datatype-Property in OWL.
- Represent cardinalities of relationships and properties in the reference information model as cardinality restrictions in OWL.

Once the reference information model is represented in OWL, it is possible to represent archetype constraints in OWL based on the reference information model. In the following subsections, we describe how each ADL construct can be represented in OWL.

5.1 Specializing the Root Class

Basically, an archetype restricts the instances of the domain classes. It has an hierarchical structure and restriction starts with a root class. The root class of the archetype is the class of the instances which are validated against the archetype. In fact, the root class of the archetype is a specialization of the corresponding class of the reference information model. Consider the ADL fragment shown in Figure 5.1, it constrains the "PERSON" class of the reference information model by defining restictions on the attributes and relations of the class. The "[at0000]" denotes the "Doctor" concept as defined in the ontology part of the ADL, therefore the root class specializes the "PERSON" class of the reference information model to "Doctor" concept.

As stated in [43], each ADL object node generates an OWL Class declaration. Similarly, specialization of the root node can be represented by defining a subclass of the corresponding class in the reference information model. In Figure 5.2, a class named "Doctor" is introduced to represent the root class of the ADL fragment shown in Figure 5.1. The name of the class can be extracted from the ontology part of the ADL. The "Doctor" is declared as a subclass of "Person" class of the reference information model. The "Doctor" class is also specializes the "Archetype" class to indicate the doctor class is the root class of archetype. We introduce "Archetype" as a marker class to easily find root classes of archetypes in OWL documents.

```
PERSON[at0000] matches {
  name matches{
    PERSON_NAME[at0001] matches{
      ...
    }
  }
}
ontology
  primary_language = <"en">
  term_definitions("en") = <
    items("at0000") = <
      text = <"Doctor">
      description = <" Doctor of the patient ">
    >
    items("at0001") = <
      text = <"DoctorName">
      description = <"Name of the doctor">
    >
...
```

Figure 5.1: An ADL fragment representing "Doctor" concept

```
<owl:Class rdf:ID="Doctor">
  <rdfs:subClassOf rdf:resource=
  "http://www.example.org/Domain.owl#Person"/>
  <rdfs:subClassOf rdf:resource=
  "http://www.example.org/Archetype.owl#Archetype"/>
...
</owl:Class>
```

Figure 5.2: Representing the Root Class of Archetype in OWL

The "Doctor" class inherits the properties of its superclass and applies further restrictions on these properties. In other words, properties of the class of the reference information model have local restrictions inside the introduced subclass and these local restrictions are derived from the archetype defined in the ADL document. How these local restrictions are handled in OWL is discussed in the following subsections. Note that, archetypes do not introduce new properties in addition to the properties of the reference information model. They introduce new classes which define local restrictions on the properties of the imported reference information model.

5.2 Constraints on the range of Object-valued Properties

In an archetype hierarchy, nodes which refer to the properties of a class in the reference information model, may have object values. These nodes refer to classes in the reference information model.

As an example, in Figure 5.1, "name" property of the "Person" is an object property. In the reference information model, the range of the "name" property is defined as "Person_Name" class. However, the ADL fragment in Figure 5.1, restricts the property by defining further restrictions on its attributes and states that the property represents the "DoctorName" concept.

In OWL, "DoctorName" class which is a subclass of "Person_Name" class is introduced as shown in Figure 5.3 Furthermore, the constraints defined on the "name" property in the archetype are defined as further restrictions in this newly introduced subclass, namely, "DoctorName".

```
<owl:Class rdf:ID="Doctor">
  <rdfs:subClassOf rdf:resource=
  "http://www.example.org/Domain.owl#Person"/>
  <rdfs:subClassOf rdf:resource=
  "http://www.example.org/Archetype.owl#Archetype"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource="#DoctorName"/>
      <owl:onProperty rdf:resource=
        "http://www.example.org/Domain.owl#name"/>
    </owl:Restriction>
  </rdfs:subClassOf>
...
</owl:Class>

<owl:Class rdf:ID="DoctorName ">
  <rdfs:subClassOf rdf:resource=
  "http://www.example.org/Domain.owl#PersonName"/>
...
</owl:Class>
```

Figure 5.3: Representing Object-valued Properties in OWL

As a result, each constraint on an object-valued property introduces a new class by

specializing the class which is the range of the property in the reference information model. The range of such a property is defined to be the newly introduced subclass. In order to avoid class name conflicts, newly introduced classes to represent concepts should be named properly. The name of the bound concept, for the above example it is the name of the concept defined by "at0001", can be used for giving meaningful names to the introduced subclasses.

It should be noted that, in our approach, a constraint on an object-valued property generates only a new class in OWL unlike the approach in [43] which also introduces a new ObjectProperty. Introducing a new ObjectProperty may make it difficult to relate the object properties in the archetype with the object properties in the reference model.

5.3 Constraints on the range of Data-valued Properties

As already mentioned, archetypes also constrain data-valued properties. However, certain desired constraints on primitive types can not be directly expressed in OWL. In OWL, it is possible to compare the value of a data-valued property by "hasValue" construct and declare the type of such a property by "rdf:datatype". However, complex constraints such as defining allowable characters and sequences for a string-valued property or defining a maximum and a minimum value for an integer-valued property are not possible.

Yet, OWL allows XML Schema to be used as a data type. Therefore, our solution for defining constraints on data-valued properties is to use "User-derived datatypes" in XML Schema. "User-derived datatypes" are those datatypes that are defined by individual schema designers [13].

Therefore, for each constraint on a data-valued property in an archetype, the corresponding user-derived datatype is declared in an XMLSchema. Data-valued properties of the reference information model are constrained by setting their types to user-derived types in the corresponding OWL class. If the constraint on the data-valued property is nothing but a value equality comparison, then "hasValue" can be used instead of creating a new type in XMLSchema. If the data-valued property is restricted to have a value from a set of values, then "owl:oneOf" construct is used to describe the possible values of the property.

5.4 Representing Existence Constraints

In ADL, an existence constraint on a property shows the optionality of the property. In Figure 5.4, an example usage of existence constraint is presented within an ADL fragment. The "specializedOn" attribute of the reference information model is restricted to be mandatory in the scope of the archetype represented by the ADL fragment.

```
PERSON[at0000] matches {
  specializedOn existence matches{1..1} matches{
   ...
  }
  ...
}
...
```

Figure 5.4: An Example Existence constraint in ADL

Optionality of a property in OWL can be specified by using minCardinality, maxCardinality and cardinality constructs. If a property has "minCardinality = 0" and "maxCardinality =1" then the property is optional. If a property has "cardinality =1" then the property is required. In Figure 5.5, the OWL fragment corresponding to the existence constraint defined in Figure 5.4 is presented. A local restriction is introduced in the "Doctor" class by defining the cardinality of the "specializedOn" property as "1".

```
<owl:Class rdf:ID="Doctor">
   <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource=
           "http://www.example.org/Domain.owl#specializedOn"/>
        <owl:cardinality rdf:datatype=
         "http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
      </owl:Restriction>
   </rdfs:subClassOf>
...
</owl:Class>
```

Figure 5.5: Defining Existence Constraints in OWL

5.5 Representing Cardinality Constraints

A cardinality constraint indicates that a property has a container type in ADL. Cardinality defines the number of valid elements by giving an upper and a lower bound to the element count for the container class. In Figure 5.6, "events" attribute defined as having 'minCardinality = 1' and 'maxCardinality' unbounded. It is also stated that elements in events can be an instance of one of the two subclasses of "Event" class.

```
...
data matches {
  History[at0003] matches {
    events cardinality matches {1..*} matches {
      Event[at0004] matches {...}
      Event[at0005] matches {...}
    }
  }
}
...
```

Figure 5.6: Cardinality Constraints in ADL

Cardinality constraints of ADL can be represented by using minCardinality, maxCardinality and cardinality constructs in OWL. Assume that "at0003", "at0004" and "at0005" concepts mean *OneMonthHistory*, *WeightGain*, *HeightGain* respectively. Then ADL fragment in Figure 5.6 can be represented in OWL as shown in Figure 5.7. A class named "OneMonthHistory" is introduced to represent *OneMonthHistory* concept defined in the ADL fragment. The minimum cardinality of "events" attribute is restricted to "1" in the scope of "OneMonthHistory" class. The OWL representation also involves classes "WeightGain" event or "HeightGain" to represent the concepts *WeightGain* and *HeightGain* respectively. The range of the "events" attribute is restricted to the union of "WeightGain" event or "HeightGain" classes.

5.6 Representing Occurrence Constraints

In ADL, occurrence constraints are defined on the elements of a container class and they constrain the number of instances for each type of element in the container. In Figure 5.8, gives an example for the usage of occurrence constraint in ADL. The ADL fragment declares

```
<owl:Class rdf:ID="OneMonthHistory">
  <rdfs:subClassOf rdf:resource=
  "http://www.example.org/Domain.owl#History"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom>
        <owl:unionOf  rdf:parseType="Collection">
          <owl:Class rdf:about="#WeightGain "/>
          <owl:Class rdf:about="#HeightGain"/>
        </owl:unionOf>
      </owl:allValuesFrom>
      <owl:onProperty rdf:resource=
           "http://www.example.org/Domain.owl#events"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource=
           "http://www.example.org/Domain.owl#events"/>
      <owl:minCardinality rdf:datatype=
      "http://www.w3.org/2001/XMLSchema#int">1</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
...
</owl:Class>

<owl:Class rdf:ID="WeightGain">
  <rdfs:subClassOf rdf:resource=
  "http://www.example.org/Domain.owl#Event"/>
  ...
</owl:Class>
<owl:Class rdf:ID="HeightGain">
  <rdfs:subClassOf rdf:resource=
  "http://www.example.org/Domain.owl#Event"/>
  ...
</owl:Class>
```

Figure 5.7: Representing ADL's cardinality constraint in OWL

that "events" should contain exactly one instance of "at0001" concept, at most one instance of "at0002" concept, and exactly three instances of "at0003" concept .

Occurrence constraints in ADL correspond to the qualified number restrictions of Description Logics. DAML+OIL [17] from which the OWL is derived supports the usage of qualified number restrictions with the following language elements: "daml:cardinalityQ",

“daml:hasClassQ” ,“daml:maxCardinalityQ”, “daml:minCardinalityQ”.

```
events cardinality matches {*} matches {
  EVENT[at0001] occurrences matches {1} matches {...}
  EVENT[at0002] occurrences matches {0..1} matches {...}
  EVENT[at0003] occurrences matches {3} matches {...}
}
```

Figure 5.8: Occurrence constraint in ADL

OWL is developed as a vocabulary extension of RDF (the Resource Description Framework) and is derived from the DAML+OIL Web Ontology Language [20]. However, OWL does not inherit qualified number restrictions from DAML+OIL. Therefore, it is not possible to represent occurrence constraints in OWL.

5.7 Representing Invariant Constraints

An invariant constraint is an expression which should be satisfied by all instances of an archetype. Figure 5.9 gives an invariant example in ADL, stating that the “value” property should exist when the “code” property exists.

```
Observation[at0000] matches {
    classCode cardinality matches {1} matches {...}
    moodCode cardinality matches {1} matches {...}
    id matches {...}
    code matches {...}
    confidentialityCode matches {...}
    uncertaintyCode matches {..}
    value matches  {...}

invariant:
  basic_validity: exists code implies exists value }
```

Figure 5.9: Invariant in ADL

Since representing invariants is not directly addressed in OWL, we propose to represent them outside OWL. There are several ways of representing invariants in XML documents

[30]. One of the solutions is to use XSLT/XPath stylesheets. We propose defining invariants through XSLT/XPath stylesheets and executing these invariant constraints over OWL instances using XSLT processors. Figure 5.10 depicts how the invariant constraint of Figure 5.9 can be expressed through XSLT.

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        version="1.0">
  <xsl:output method="text"/>
  <xsl:template match="/">
    <xsl:if test= "boolean(/Observation/code)">
      <xsl:if test= "not(boolean(/Observation/value))">
        <xsl:text>Error! code exists without value</xsl:text>
      </xsl:if>
    </xsl:if>
    <xsl:if test= "boolean(/Observation/code)">
      <xsl:if test= "boolean(/Observation/value)">
        <xsl:text>Instance document is valid</xsl:text>
      </xsl:if>
  </xsl:template>
</xsl:stylesheet>
```

Figure 5.10: Representing Invariant using XSLT

5.8 Representing Internal References

The aim of internal references in ADL is not to repeat a previously defined constraint in an archetype. Internal references are indicated by path expressions in ADL. Such references can be enforced by reusing the classes or data types in OWL.

5.9 Representing Archetype Reuse

In ADL "allow_archetype" construct is used to reference other archetypes by defining a criteria for allowable archetypes.

An example of "allow_archetype" is presented in Figure 5.11. The elements of the "items" list described in Figure 5.11 are instances of "ENTRY" class of the reference information model and they are constrained by archetype classes whose id matches the indicated representation.

```
SECTION[at2000] occurrences matches {0..1} matches {
  items cardinality matches {0..*} matches {
    allow_archetype ENTRY occurrences matches  {0..1} matches {
      include
        id matches {/.*\.iso-ehr\.entry\..*\..*/}
    }
  }
}
```

Figure 5.11: Archetype reuse in ADL

In OWL the range of the "items" property is restricted in such a way that values of "items" are instances of "ENTRY" and also instances of an "Archetype" whose id has a pattern restriction. Such a range restriction can be defined by introducing a new class, and setting the range of the "items" to this class. This new class is subclass of "ENTRY" and restricts the "archetypeID" by setting its range to a user-derived type which defines the pattern restriction.

OWL class defined in Figure 5.12 can be used for this purpose to implement the "allow_archetype" constraint in OWL for the example in Figure 5.11.

```
<owl:Class rdf:ID="ReusedArchetype1">
  <rdfs:subClassOf rdf:resource=
     "http://www.example.org/Domain.owl#Entry"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource=
        "http://www.example.org/ArchetypeSchema#entryPattern"/>
      <owl:onProperty rdf:resource=
        "http://www.example.org/Archetype.owl#archtypeID"\>
    </owl:Restriction>
  </rdfs:subClassOf>
...
</owl:Class>
```

Figure 5.12: Implementing archetype reuse in OWL

CHAPTER 6

IMPLEMENTATION

In order to realize the architecture described in Chapter 4, a prototype implementation of the the clinical statement interoperability is developed. The software which we called Clinical Statement Mapping and Transformation Tool aims to transform Health Level Seven clinical statement instances to EHRcom instances and vice versa by using semantic mechanisms based on R-MIM derivations and archetypes.

The Clinical Statement Mapping and Transformation Tool has been developed using Java 2 Platform Standard Edition 5.0. In addition, following tools and libraries are used:

- *Jena, A Semantic Web Framework for Java:* Jena [54] provides RDF API, OWL API, SPARQL query engine and reading and writing RDF in RDF/XML. As shown in Figure 6.1 [54], Jena also provides a mechanism for attaching external reasoners to Jena models through DIG (DL Implementors Group) Interface [21] which is a specification for a simple API to DL reasoners.

- *Pellet version 1.4:* Pellet [71] is an open source, OWL DL reasoner completely written in Java. Pellet enables validating ontology species, checking consistency of ontologies, classifying the taxonomy, checking entailments and answer a subset of RDQL queries. Pellet also implements the DIG Interface, therefore it can be integrated with Jena.

- *Xerces:* Xerces [6] is a library which implements a number of standard APIs for XML parsing, including DOM, SAX and SAX2.

- *Log4j:* Log4j [61] is a logging utility which is used primarily as a debugging tool.

- *JUnit:* JUnit [55] is a framework to perform unit tests for the Java programming language.

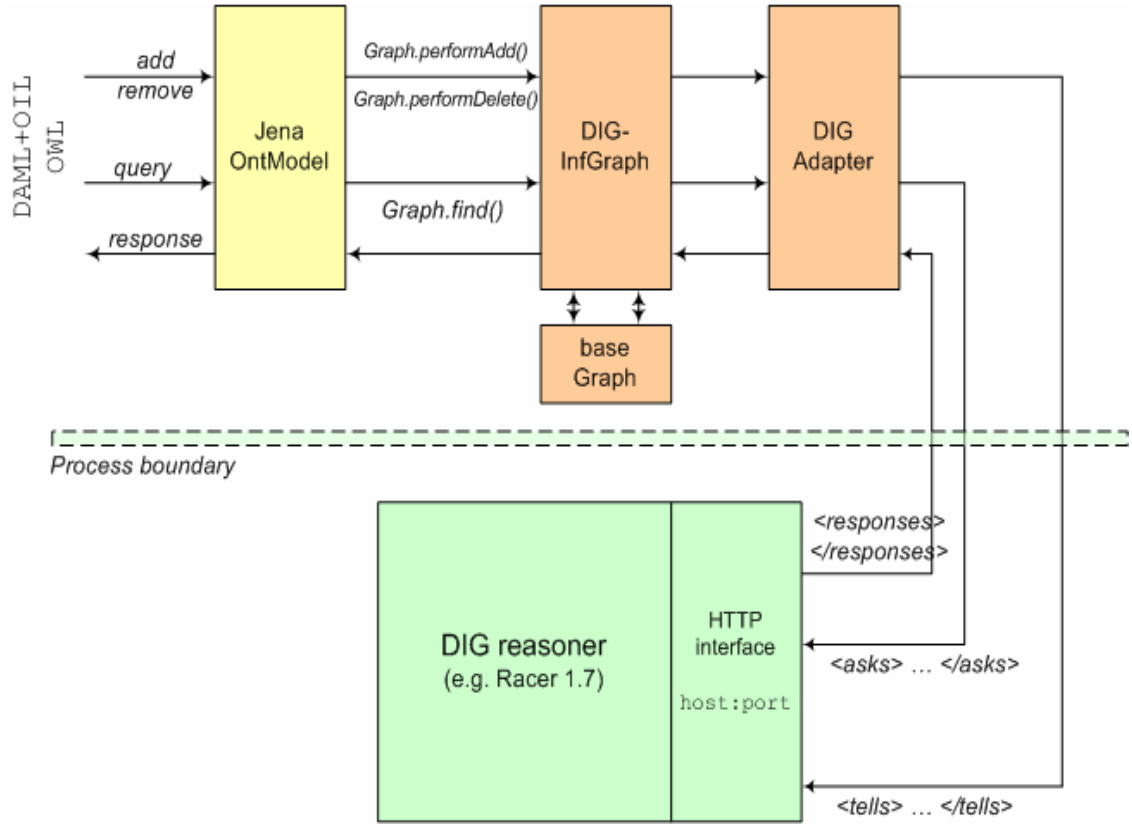


Figure 6.1: Schematic of dataflows between Jena model and DIG reasoner [54]

The Clinical Statement Mapping and Transformation Tool has been developed based on the Jena framework conjunction with an external DIG Reasoner which is Pellet in our implementation. Before executing the tool, the Pellet DIG server should be started. The server listens the port "http://localhost:8081" as shown in Figure 6.2

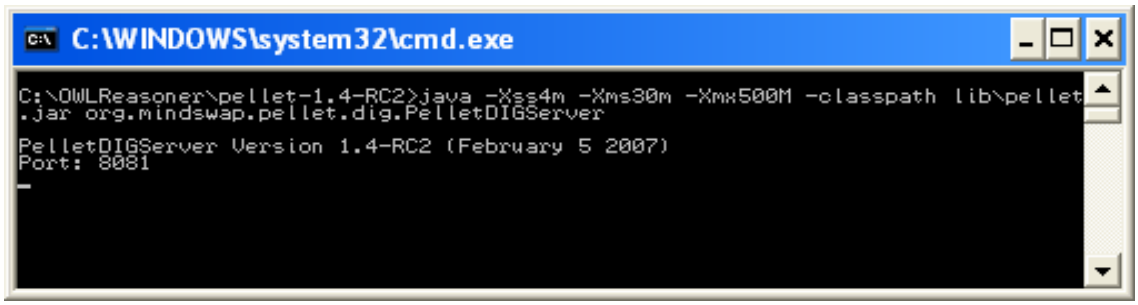


Figure 6.2: Pellet DIG server listening the port "http://localhost:8081"

Clinical Statement Mapping and Transformation Tool has two modules: Mapping Engine and Transformation Engine. Mapping Engine module provides a mapping tool which is used to produce the "Mapping Definitions" between two archetypes which are based on different RMIMs but express the same clinical concept. Transformation Engine uses these "Mapping Definitions" to transform one EHR instance to another.

6.1 Mapping Engine Module

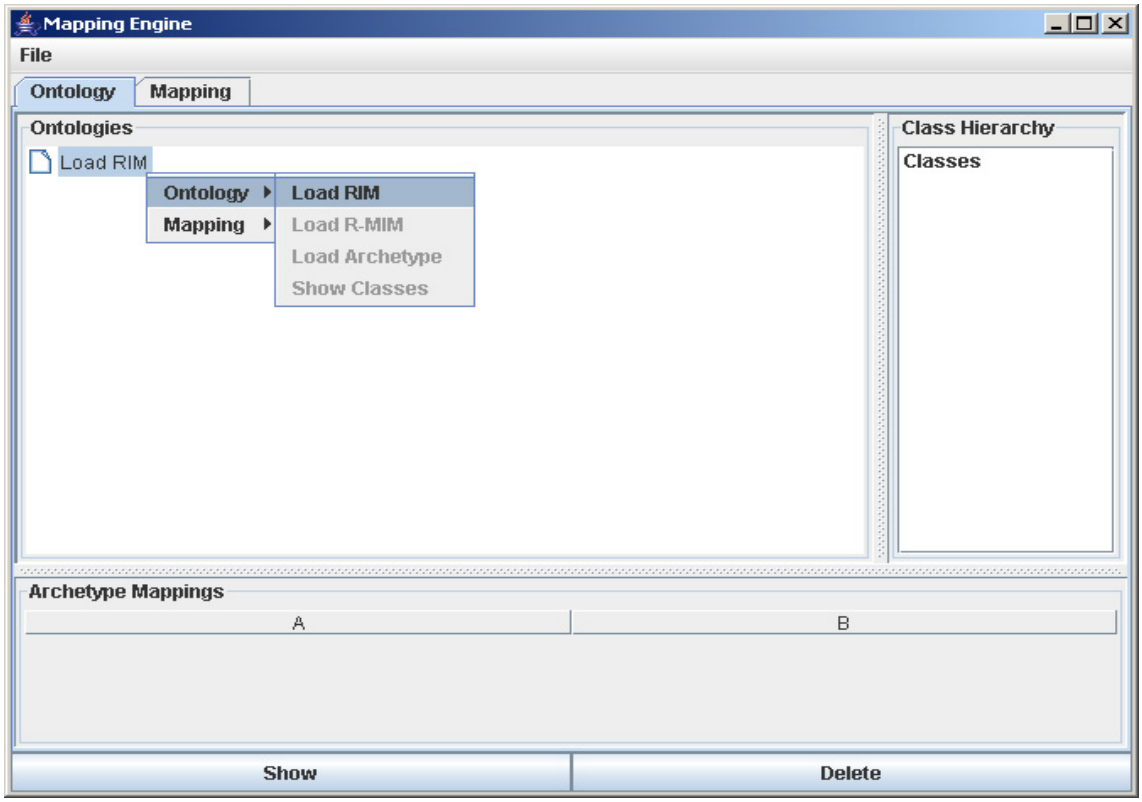


Figure 6.3: Loading RIM Ontology to Mapping Engine Module

Mapping Engine is the module for mapping archetypes of different R-MIMs based on the same RIM. Initially, it is installed with an empty knowledge base. Prior to mapping archetypes, ontologies of the RIM, R-MIMs and archetypes should be loaded to knowledge base of the Mapping Engine module. Initial screen of the Mapping Engine module is shown in Figure 6.3. It involves a tabbed pane which has two parts. "Ontology" pane enables user to load and navigate ontologies. "Mapping" pane enables user to map archetypes of

different R-MIMs. Initially, "Ontology" pane involves a node named "Load RIM" as shown in Figure 6.3. Whe the RIM ontology loaded this node is replaced with node representing the RIM ontology.

Figure 6.4: Selecting the location of the RIM Ontology

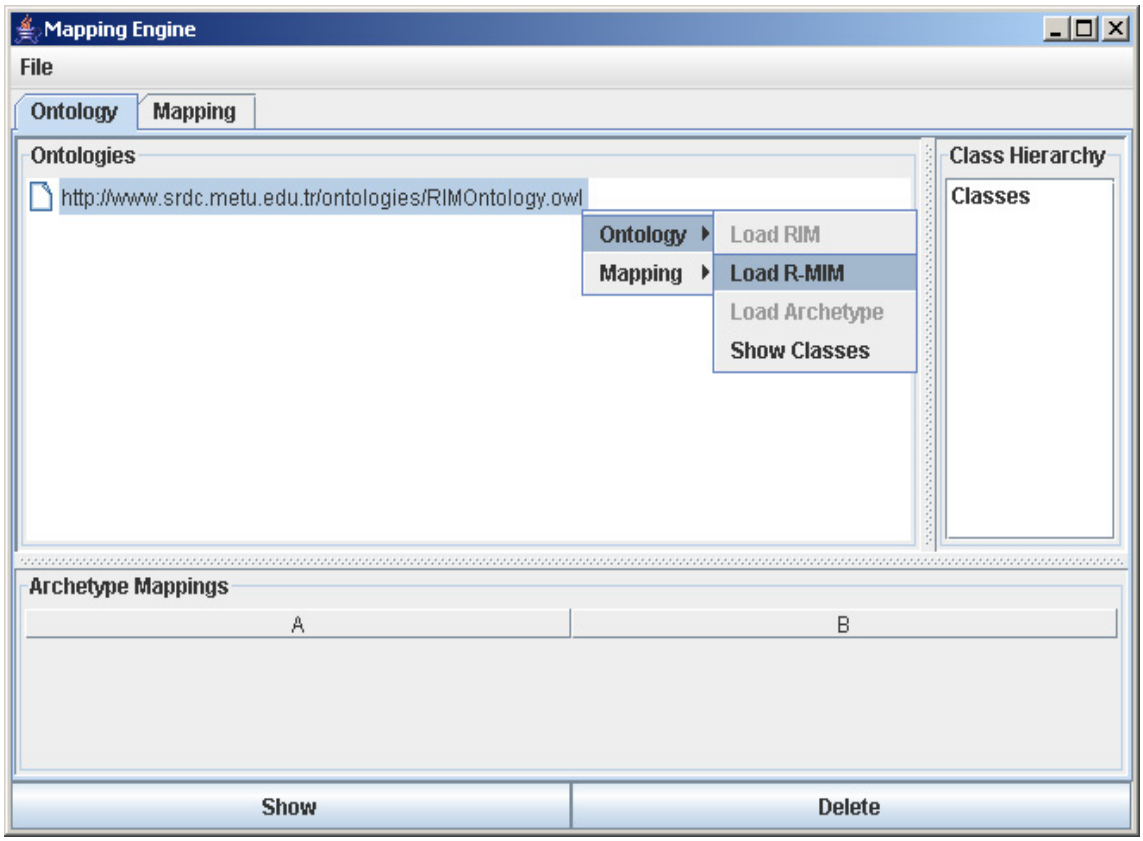


Figure 6.5: Loading R-MIM Ontology to Mapping Engine module

"Ontology" pane has a popup menu which enables user to select the RIM ontology through the dialog shown in Figure 6.4. The dialog requests the URL of the RIM ontology from the

user. The user is also able to select a local file if the URL is unreachable. Once the user confirms the location of the RIM ontology. Mapping Engine module loads the ontology from the given location to its knowledge base. A node having the URL of the RIM as its label is replaced with the current node named "Load RIM' to indicate that the RIM ontology is loaded by the Mapping Engine module. In our case the location of the RIM is "http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl".

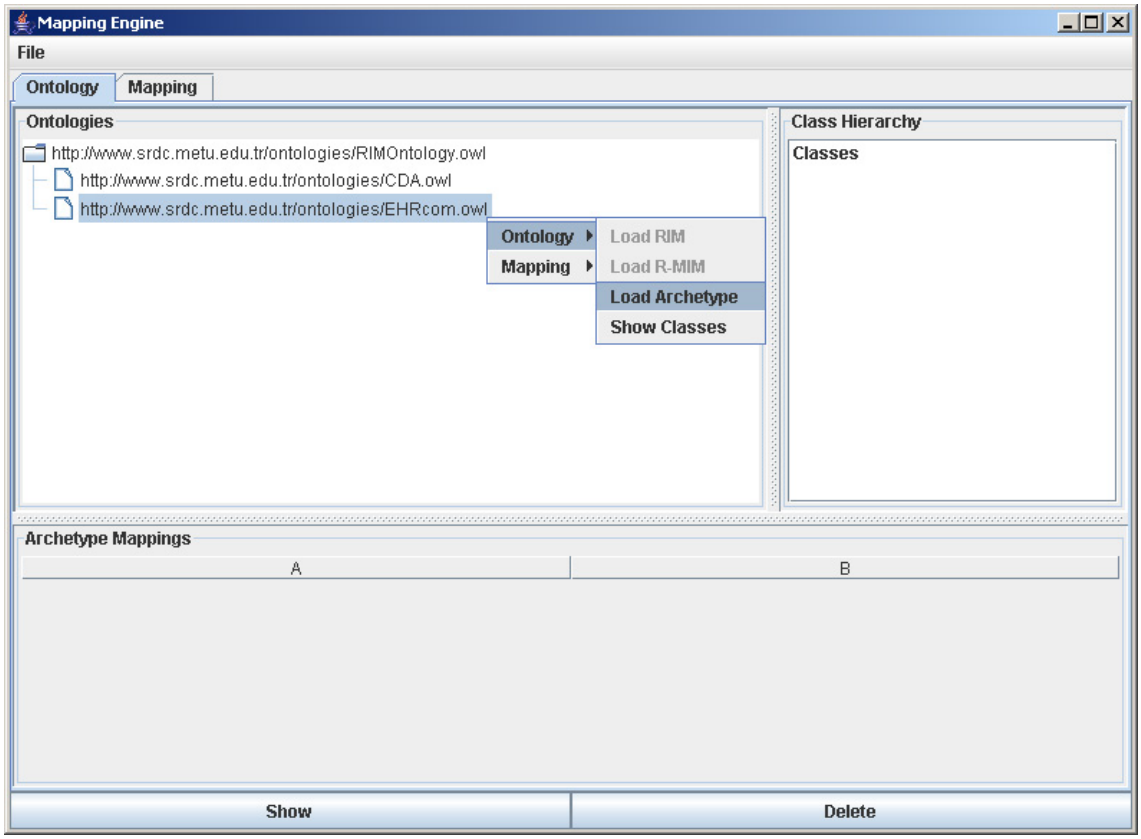


Figure 6.6: Loading Archetype to Mapping Engine Module

Secondly, Mapping Engine module requires R-MIM ontologies which are derived from the same RIM that is loaded to the knowledge base in the previous step. The popup menu enables user to load an R-MIM as shown in Figure 6.5. Mapping Engine module requests the location of the R-MIM through a similar dialog shown in Figure 6.4 and loads the R-MIM ontology to its knowledge base when the user confirms the location. A child node appended to the node identifying the RIM ontology for each loaded R-MIM ontology in the "Ontology" pane. Each node is labeled with location of the represented ontology. In our case the location of the

CDA R-MIM and EHRcom R-MIM are "http://www.srdc.metu.edu.tr/ontologies/CDA.owl" and "http://www.srdc.metu.edu.tr/ontologies/EHRcom.owl", respectively.

In fact, "Ontology " pane holds an hierarchy of ontologies where the RIM ontology is the root ontology. R-MIM ontologies are the children of the RIM ontology and finally archetype ontologies are the leaf ontologies. An archetype ontology can be loaded to knowledge base by using the popup menu as shown in Figure 6.6. Mapping Engine module requests the location of the archetype through a similar dialog shown in Figure 6.4 and loads the archetype ontology to its knowledge base when the user confirms the location. When an archetype is loaded, a node labeled with the location of the archetype is appended as a child to the node which denotes the R-MIM of the archetype.

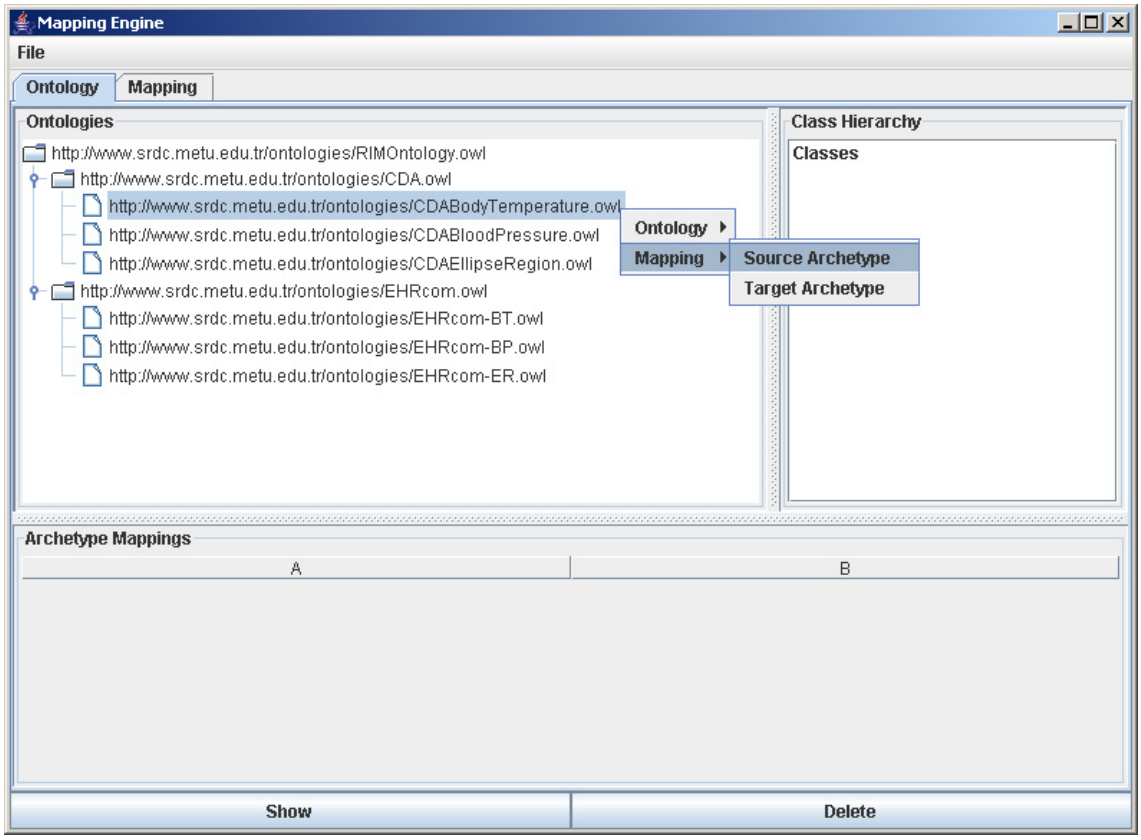


Figure 6.7: Selecting Source and Target Archetype for Mapping

It should be noted that these ontologies should be defined in OWL. OWL representaions of RIM, R-MIMs and some of the archetypes are available in Appendix A and Apppendix B.

Mapping process can be initiated when archetypes representing the same concept are defined for both R-MIMs. The popup menu in Figure 6.7 enables user to select source and target archetype for the mapping process. Selected source and target archetypes are loaded in the "Mapping" pane.

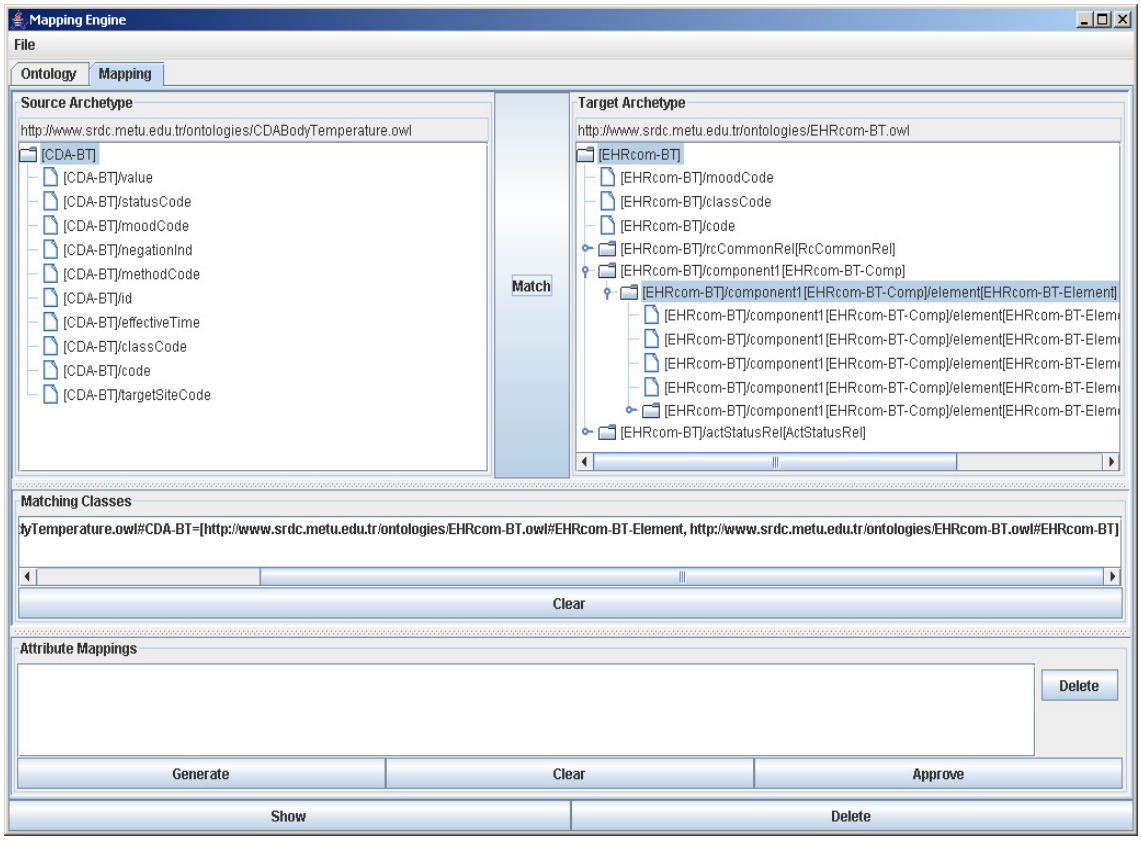


Figure 6.8: Manually Matching Classes of Source and Target Archetypes

Mapping panel displays both the source and target archetype in a hierarchical form as shown in Figure 6.8. The tree on the left hand side represents the source archetype hierarchy and the the tree on the right hand side represents the target archetype hierarchy. The root node of the hierarchy is the root class of the archetype. The rest of the tree is constructed by adding the attributes and the relations as child nodes. Child nodes of a relation is extracted from the class of the relation shown in brackets. It should be noted that attributes are the leaf nodes and relations are the intermediate nodes of the hierarchy. Each node is labeled with the path from the root class to the attribute/relation represented by the node as described in Chapter 4.

The mapping process has two phases. In the first phase user-defined mappings are provided. Class matching is an optional phase of manual operations. When the source and target archetype specialize classes of their R-MIM for different purposes, there is no need to match the attributes of these specialized classes. To prevent generation of unnecessary attribute mappings user can match relevant classes of source and target archetype. Furthermore, matching classes increases the performance of the mapping process since the algorithm performs the attribute mappings only for the matched classes. Class matching can be performed by selecting the relevant source and target classes from the source and target archetype hierarchies as shown in Figure 6.8. Then pressing the "Match" button in the middle of the hierarchy trees generates the class matching. The result of the class matching is shown in the list named "Matching Classes" just below the trees representing archetype hierarchies. In Figure 6.8, user matches CDA-BT class in the source archetype with the EHRcom-BT and EHRcom-BT-Element classes in target archetype.

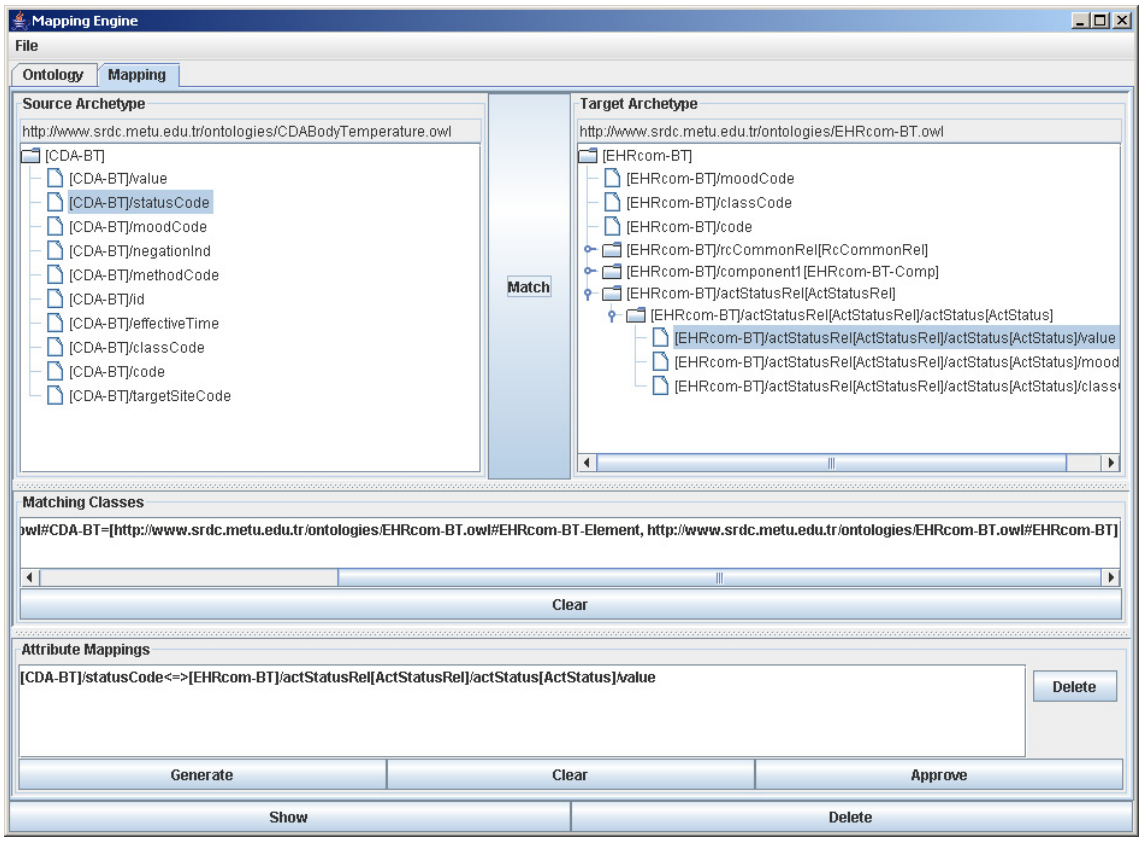


Figure 6.9: Manually Matching Attributes of Source and Target Archetypes

In some cases the target ontology and the source ontology use different properties to express the same information. Mapping such attributes is not possible through reasoning. Therefore Mapping Engine module allows describing user-defined property mappings. Mapping attributes can be performed manually by selecting the relevant source and target attributes from the source and target archetype hierarchies as shown in Figure 6.9. Then, pressing the "Match" button in the middle of the hierarchy trees generates the attribute mapping between the selected source and target attributes. The resulting attribute mapping is added to the list named "Attribute Mappings" which is in the bottom of the "Mapping" pane. The source attribute denoted by the path "CDABT/statuscode" is mapped to the attribute denoted by the path "[EHRcom-BT]/actStatusRel[ActStatusRel]/actStatus[ActStatus]/value", as shown in Figure 6.9.

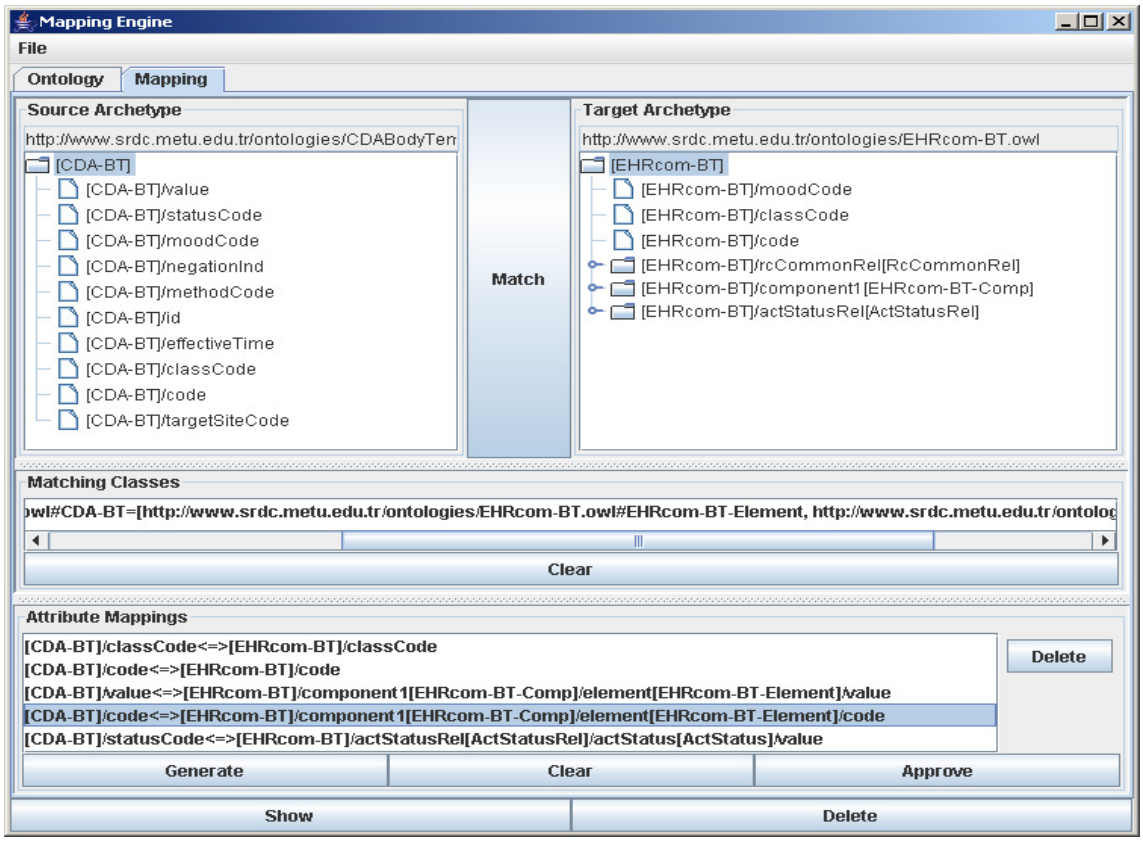


Figure 6.10: Generated Attributes of Source and Target Archetypes

When manual mappings are completed, Mapping Engine module becomes ready for the generation of attribute mappings automatically. The "Generate" button triggers the auto-

matic mapping process. The Mapping Engine module finds the matching attributes using the derivation rules of these R-MIMs and archetypes through reasoning in OWL. Finding root class of an archetype, finding the attributes/relations of class by filtering the attributes having zero cardinalities, finding the value restrictions of an attribute are examples of functions requires reasoning in OWL. The Mapping Engine module implements the mapping algorithms described in Chapter 4. The generated attribute mappings are appended to the "Attribute Mappings" list. When the user observes undesired mappings, he or she can delete a mapping from the list using the "Delete" button next to the list. In Figure 6.10, generated attribute mappings for the given archetypes are displayed. It should be noted that the list not only involves the generated attribute mappings but also lists the manually mapped attributes.

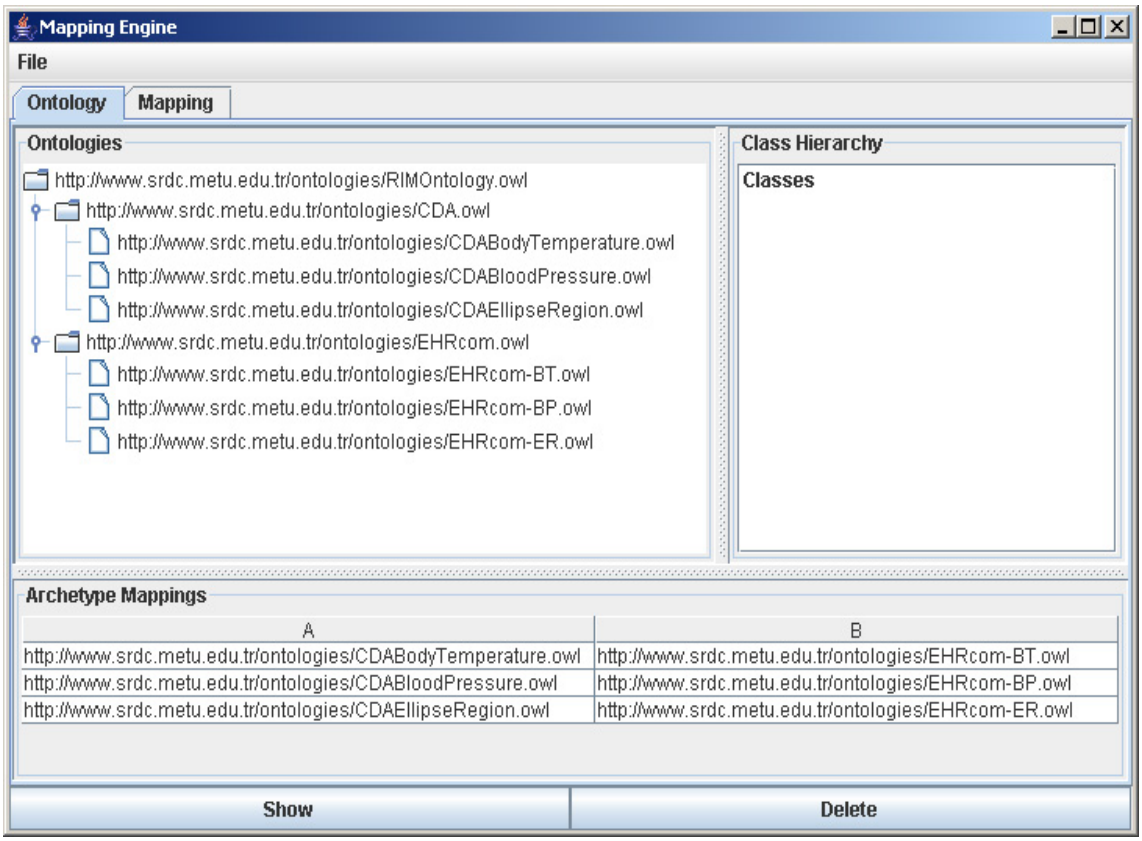


Figure 6.11: Mapping Engine Module User Interface after Archetypes are Mapped

When the attribute mappings are finalized, user can approve the archetype mapping using the "Approve" button located just below the "Attribute Mappings" list. Once the archetype

mapping is approved, it is listed in the "Archetype Mappings" table in the "Ontology" pane as shown in Figure 6.11. User can delete an archetype mapping using the "Delete" button just below the "Archetype Mappings" table. Furthermore, user can view the attribute mappings of an archetype mapping by selecting an archetype mapping from the "Archetype Mappings" table and clicking the "Show" button. Attribute mappings of the archetypes are listed in the "Attribute Mappings" list of the "Mapping" pane.

As mentioned, the aim of the Mapping Engine module is to enable user to generate archetype mappings which will be used later in the clinical instance transformation process performed by the Transformation Engine module. In order to make the archetype mappings available to Transformation Engine, Mapping Engine module persists the ontology hierarchy and the archetype mappings to an XML file. The "Save" action in the "File" menu of the Mapping Engine module can be used to persist the current state of the ontology hierarchy and the generated archetype mappings.

6.2 Transformation Engine Module

The aim of the Transformation Engine module is to transform the clinical statement instances of the source R-MIM to instances conforming to the target R-MIM. In order to perform the transformation, the module requires archetype mappings generated by the Mapping Engine module. The Transformation Engine parses the definition file generated by the Mapping Engine module and retrieves the archetype mappings.

Current version of the Transformation Engine module act as a command line tool. It has two parameters one of which is the location of a file containing a clinical statement instance of source R-MIM serialized in RDF. Second parameter is the URL location of the archetype to which the source clinical statement instance conforms.

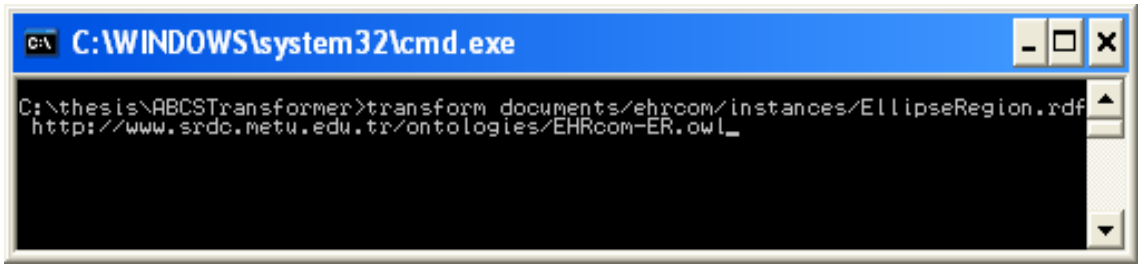


Figure 6.12: Running Transformation Engine through Command line

How to execute the Transformation Engine module is shown in Figure 6.12. The executable "transform" is called with two parameters: "ehrcom/instances/EllipseRegion.rdf" (relative path of the clinical statement instance file in RDF), "http://www.srdc.metu.edu.tr/-ontologies/EHRcom-ER.owl" (archetype to which the clinical statement instance conforms).

```
private void transformInstance(Model sourceInstanceModel, String sourceArchetypeURI)throws Exception{
    //load the source archetype ontology
    OntModel sourceArchetype = readArchetype(sourceArchetypeURI);
    //build an inferred model for source instance by attaching the source RDF model to the DIG Reasoner
    InfModel infmodel = ModelFactory.createInfModel(InferenceEngineFactory.getInstance().getReasoner(),
        sourceArchetype, sourceInstanceModel);
    OWLInstance sourceInstance = new OWLInstance();
    sourceInstance.setInfModel(infmodel);
    //retrieve the archetype mapping
    ArchetypeMapping aMapping = findArchetypeMapping(sourceArchetypeURI);
    //create an empty model for target instance
    Model targetInstanceModel = ModelFactory.createOntologyModel();
    //get the target archetype from the mapping
    OntModel targetArchetype = readArchetype(aMapping.getTargetArchetypeURL());
    //build an inferred model for target instance by attaching empty RDF model to the DIG Reasoner
    infmodel = ModelFactory.createInfModel(InferenceEngineFactory.getInstance().getReasoner(),
        targetArchetype, targetInstanceModel);
    OWLInstance targetInstance = new OWLInstance();
    targetInstance.setInfModel(infmodel);
    targetInstance.setDataModel(targetInstanceModel);
    setTargetInstance(targetInstance);
    //generate the target instance
    targetInstance = transformInstance(sourceInstance,aMapping,targetInstance);
    //serialize the target instance to a file
    File out = new File("targetInstance.rdf");
    FileWriter writer = new FileWriter(out);
    targetInstance.write(writer,"RDF/XML-ABBREV","http://www.srdc.metu.edu.tr/targetInstance.owl#");
}
```

Figure 6.13: High level algorithm for the transformation process

The Transformation Engine implements the transformation algorithms described in Chapter 4. Figure 6.13 gives the high level algorithm for the transformation process. The Transformation Engine first loads the source archetype based on its URL. Then it builds an inferred model for the source instance by attaching the source instance model to the DIG reasoner.

Next, it retrieves the archetype mapping of the source archetype from the XML file generated by the Mapping Engine. The Transformation Engine, also builds an inferred model for the empty target instance by attaching the it to the DIG reasoner. It builds the target instance using the inferred models of source and target instances. Finally the generated target instance is serialized to a file.

CHAPTER 7

RELATED WORK

Currently, providing the interoperability of heterogeneous information systems through ontology mediation is a very active research area.

XeOML [69] proposes an extensible language called "XML-based extensible Ontology Mapping Language (XeOML)" for describing mappings between domain ontologies. The mapping language is defined by an XML schema, called AbstractMapping, which is used for describing mappings between ontologies, detailing the structure of a mapping document and defining the set of elements that populate an ontology. However, as stated in [70], using OWL instead of XML for describing abstract XML schema would bring stronger semantics to the generic mapping primitives of XeOML.

MAFRA [63] presents a framework for ontology mediation which supports semantic mapping definition and the reconciliation engine. It describes a meta-ontology called Semantic Bridge Ontology to relate the source and target ontology. Semantic Bridge Ontology is represented in DAML+OIL and Semantic Bridges in this meta-ontology are used to define complex structural mappings such as specialization, abstraction, composition and alternatives. However, supported formats are limited to XSD, RDFS and KAON.

Chatty Web [1] discusses that establishing local agreements is a less challenging task than establishing global agreements by means of globally agreed schemas or shared ontologies. Therefore, it uses peer-to-peer paradigm to improve semantic interoperability by addressing how global semantic agreements can be achieved based on the local agreements.

In [59], an ontology based information integration method is described for palliative care. A methodology is described for designing an ontology and it is used for facilitating information sharing in the palliative care systems.

Artemis project [7], proposes a semantic web service-based peer-to-peer infrastructure for the interoperability of medical information systems. It uses prominent healthcare standards

to semantically annotate Web services. Two types of ontologies namely, Service Functionality Ontology and Service Message Ontology are introduced. Service Functionality Ontology is used to classify coarse-grained Web services in healthcare domain and Service Message Ontology is used to specify the semantics of Web service messages and is developed through prominent healthcare standards.

In the healthcare domain, [12] describes how to mediate between HL7 Version 2 and HL7 Version 3 messages based on semantics. In [11], archetype-based semantic interoperability of Web Service messages exchanged in the health care domain is described. This approach is specific to Web Service messages where each Web service is annotated with the OWL representation of the archetypes.

An important aspect of semantic interoperability in the healthcare domain is terminologies. Medicine has a long tradition in structuring its domain knowledge through disease taxonomies, medical procedures, anatomical terms in a wide variety of medical terminologies, thesauri and classification systems [77]. In order to provide interoperability, all of these classifications and terminologies need translations from one to another [79]. The Unified Medical Language System [60] provides a framework to retrieve and integrate disparate terminology systems. Another important framework is GALEN Project [76] which uses a descriptive logic based language called GRAIL [75] to represent conceptual medical knowledge. GALEN common reference model is developed using GRAIL which forms the underlying structural foundation for the services provided by the GALEN Terminology Servers.

There are also several studies on the usage of archetypes in the health care domain. In [33], the feasibility and usefulness of expressing clinical data sets as openEHR archetypes are analyzed by transforming the clinical data sets into archetypes, and outlining typical problems with clinical data sets.

In [72], the semantic issues which arose when generating lexical and semantic matches of terms from the archetype model to relevant SNOMED codes are described. Local terms can be used to define concepts in archetypes and these terms can later be bound or mapped to external terminology codes. In [72], the task of matching local terms of the archetype model to SNOMED terminology codes is discussed, however the study is specific to the openEHR reference model.

In [64] and [65], an archetype modeling framework and LinkEHR-ED for transforming existing electronic healthcare data into structure conforming to an EHR content standard are presented. LinkEHR-ED is a tool for developing archetypes for clinical data integration. Representing archetypes differs from our approach in such a way that LinkEHR-ED

represents reference models of EN13606 and OpenEHR using XML Schema and it specifies a constrained multiplicity list (CML) language to represent the structure of archetypes.

In [82], an archetype editor providing manual or semi-automatic creation of bindings between archetypes and terminology systems is presented.

There is an ISO/CEN/HL7 coordination group which is addressing harmonization of the HL7, ISO and CEN data types, the HL7 Common Message Element Types [38] and EHRcom General Purpose Information Components [15]. An implementation guide [40] of EHRcom in HL7 RIM formalism is currently being prepared jointly by CEN and HL7 experts.

Finally, a survey and analysis of EHR standards are presented in [28].

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

Many countries have reached a critical stage in the application of Information and Communication Technologies to health where most healthcare providers, such as hospitals, have already computerised at the departmental level and at least partially integrated across the enterprise. The focus now is the regional and/or national integration among healthcare organisations [79]. The main aim of all these efforts is to make EHRs of a patient accessible any where at any time to all authorized users. In this way, the countries aim effective and efficient patient care by facilitating the retrieval and processing of clinical information about a patient from different sites, speeding up healthcare delivery and reducing the cost by eliminating duplicate testing and prescribing.

It should be noted that, different health professionals in different organisations provide healthcare services to a patient over his/her lifetime. Therefore, it is of significant importance that the health professionals involved in the care delivery of a patient communicate and exchange the EHRs of a patient. Given that there are more than one EHR standards, the most prominent ones being EHRcom and HL7 CDA, eventually there will be a need for transforming the content of an EHR instance from one standard to other since it will not be realistic to expect a single standard to prevail.

In this thesis, we describe a super-peer based peer-to-peer architecture which we present in [58] for enabling the exchange of EHRs across communities based on IHE XCA Integration Profile. The proposed solutions provide the following advantages:

- The use of Super-peer/Super-peer and Super-peer/peer indices based on the *homeCommunityId* or the control numbers makes it possible to forward queries only to the relevant target communities.

  This is important because otherwise each healthcare enterprise in the network is forced to process each request broadcasted to the network. Forwarding requests only to

relevant peers also reduces the network traffic by avoiding flooding.

- To handle vocabulary translations, we introduce an actor named Vocabulary Translator that stores vocabulary mappings between the local vocabulary of one community and the vocabulary of its super peer. In this way, when a community wishes to join a network, it provides only a single vocabulary mapping with its super-peer's vocabulary.

- Using patient demographic queries across communities to obtain patient identifiers may raise patient privacy issues. We describe a routing mechanism for this super-peer based peer-to-peer network based on control numbers that preserves the privacy of the patients.

- A final issue addressed in this thesis is the following: IHE XDS is content format neutral, that is, enterprises in a community can decide which document formats to use such as the Health Level Seven (HL7) Clinical Document Architecture (CDA) [37], or the European Committee for Standardization (CEN) EN 13606-1 [16]. Therefore, another challenge in cross community EHR access is the EHR interoperability since the communities may be using different EHR standards. It will help medical practitioners in one community to easily evaluate documents of other communities if automated translation of different document standards is possible. As a first step in this direction, in [57], we show the interoperability of EHR structure and content and described how the clinical statements of two different EHR standards derived from the same RIM can be mapped to each other by using archetypes, R-MIM derivations and semantic tools.

  Since an R-MIM still contains a small number of classes for some domains like clinical statement, more specialized semantics is expressed through the archetypes by constraining the R-MIM classes. This, in return gives the opportunity to use these prepackaged semantic constraints in mapping a source concept into a target concept. It should be noted that archetypes and their mappings should be derived by the experts who have sufficient domain knowledge.

  Once archetypes and their mappings are defined, we demonstrate that the instances of different EHR standards can be transformed to each other. The semantic tools developed for this purpose include the "XML to OWL" and "OWL to XML" normalizers to transform the EHR XML instances to OWL instances and back; the R-MIM based mapping rules which are expressed in OWL and the OWL representation of the RIM, R-MIMs and archetypes. Furthermore attribute mappings are discovered

through reasoning in OWL.

If this approach shows to be viable, it would be possible to implement reliable Web services translating clinical statements from one format to another as a future work. This would allow for Document Consumer actors (i.e., the recipients of documents) to be enhanced in such a way that they can call document conversion services and convert a document into a structure suitable for local storage (e.g. inclusion into a local EHR) or processing. However there are legal issues concerning document transformation from one format to another. Therefore, the practitioners should be able to access the original document as well.

Although the framework provided by the RIM is a critical component of semantic interoperability, it is not sufficient since the content interoperability also implies the interoperability of coded terms. Furthermore, each terminology overlaps with the RIM in different ways which leads to express a clinical statement in many ways, often with no ability for a computer to determine the equivalences [25]. The future work involves integrating the work described in this thesis with terminology systems.

REFERENCES

[1] K. Aberer, P. C. Mauroux, and M. Hauswirth. The Chatty Web: Emergent Semantics Through Gossiping. In *Proceedings of the Twelfth International World Wide Web Conference*, May 2003.

[2] T. Aden and M. Eichelberg. 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, last visited on May 2008.

[3] T. Aden and M. Eichelberg. ARTEMIS Deliverable D5.2.1: Patient Identification Protocol and IHE Retrieve Information for Display Integration into ARTEMIS. http://www.srdc.metu.edu.tr/webpage/projects/artemis/calendar.php, last visited on May 2008.

[4] T. Aden, M. Eichelberg, and W. Thoben. A fault-tolerant cryptographic protocol for patient record requests. In *Proceedings of EuroPACS-MIR 2004 in the Enlarged Europe*, pages 103–106. EuroPACS, 2004. ISBN 88-8303-150-4.

[5] American National Standards Institute (ANSI). http://www.ansi.org/, last visited on May 2008.

[6] Apache Xerces Parser. http://xerces.apache.org/, last visited on May 2008.

[7] Artemis Project. http://www.srdc.metu.edu.tr/webpage/projects/artemis, last visited on May 2008.

[8] T. Beale. Archetypes: Constraint-based domain models for future-proof information systems, In: OOPSLA-2002 Workshop on behavioural semantics, 2002.

[9] T. Beale and S. Heard. Archetype Definition Language Version 2. http://svn.openehr.org/specification/TRUNK/publishing/architecture/am/adl2.pdf, last visited on May 2008.

[10] T. Beale and S. Heard. Archetype definitions and principles (Revision 0.6). http://svn.openehr.org/specification/TRUNK/publishing/architecture/am-/archetype_principles.pdf, last visited on May 2008.

[11] V. Bicer, O. Kilic, A. Dogac, and G. Laleci. Archetype-based Semantic Interoperability of Web Service Messages in the Healthcare Domain. *International Journal on Semantic Web and Information Systems*, 1(4):1–22, 2005.

[12] V. Bicer, G. Laleci, A. Dogac, and Y. Kabak. Artemis message exchange framework: semantic interoperability of exchanged messages in the healthcare domain. *ACM Sigmod Record*, 34(3):71–76, 2005.

[13] P. V. Biron and A. Malhotra. XML Schema Part 2: Datatypes, eds. W3C Recommendation 02 May 2001. http://www.w3.org/TR/xmlschema-2/, last visited on May 2008.

[14] Canada Health Infoway. http://www.infoway-inforoute.ca/, last visited on May 2008.

[15] CEN prEN 14822-1. General purpose information components. Draft European Standard prEN 14822-1, European Committee for Standardization, 2003.

[16] CEN/TC 251, EN 13606-1, Health Informatics - Electronic Health Record Communication - Part 1: Reference Model. http://www.centc251.org/, last visited on May 2008.

[17] D. Connolly, F. Harmelen, I Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. DAML+OIL Reference Description. http://www.w3.org/TR/daml+oil-reference, last visited on May 2008.

[18] A. Crespo and H. Garcia-Molina. Routing Indices For Peer-to-Peer Systems. In *ICDCS '02: Proceedings of the 22 nd International Conference on Distributed Computing Systems (ICDCS'02)*, page 23, Washington, DC, USA, 2002. IEEE Computer Society.

[19] DARPA Agent Markup Language (DAML). http://www.daml.org/, last visited on May 2008.

[20] M. Dean and G. Schreiber. Web Ontology Language OWL Reference. http://www.w3.org/TR/2004/REC-owl-ref-20040210/, last visited on May 2008.

[21] DIG (DL Implementors Group) Interface. http://dl.kr.org/dig/, last visited on May 2008.

[22] Digital Imaging and Communications in Medicine DICOM. http://medical.nema.org/, last visited on May 2008.

[23] A. Dogac, G. Laleci, Y. Kabak, S. Unal, S. Heard, T. Beale, P. L. Elkin, F. Najmi, C. Mattocks, D. Webber, and M. Kernberg. Exploiting ebXML registry semantic constructs for handling archetype metadata in healthcare informatics. *International Journal of Metadata, Semantics and Ontologies (IJMSO)*, 1(1):21–36, 2006.

[24] A. Dogac, G. B. Laleci, T. Aden, and M. Eichelberg. Enhancing IHE XDS for Federated Clinical Affinity Domain Support. *IEEE Transactions on Information Technology in Biomedicine*, 11(2):213–221, 2007. ISSN 1089-7771.

[25] R. H. Dolin, L. Alschuler, S. Boyer, C. Beebe, F. M. Behlen, P. V. Biron, and A. Shabo. HL7 Clinical Document Architecture, Release 2. *Journal of the American Medical Informatics Association*, 13(1):30–39, 2006.

[26] ebXML. http://www.ebxml.org/, last visited on May 2008.

[27] Edutella website. http://edutella.dev.java.net, last visited on May 2008.

[28] M. Eichelberg, T. Aden, J. Riesmeier, A. Dogac, and G. Laleci. A survey and analysis of Electronic Healthcare Record standards. *ACM Computing Surveys*, 37(4):277–315, 2005.

[29] M. Eichelberg, T. Aden, and W. Thoben. A Distributed Patient Identification Protocol based on Control Numbers with Semantic Annotation. *International Journal on Semantic Web and Information Systems*, 1(4):24 – 43, 2005. ISSN 1552-6283.

[30] Extending XML Schemas. http://www.xfront.com/ExtendingSchemas.pdf, last visited on May 2008.

[31] Federal Health IT Initiatives. http://www.hhs.gov/healthit/, last visited on May 2008.

[32] D. Fensel, F. Harmelen, I. Horrocks, D. L. McGuinness, and P. F. Patel-Schneider. OIL: An Ontology Infrastructure for the Semantic Web. *IEEE Intelligent Systems*, 16(2):38–45, 2001.

[33] S. Garde, E. Hovenga, J. Buck, and P. Knaup. Expressing Clinical Data Sets with openEHR Archetypes: A Solid Basis for Ubiquitous Computing. *International Journal of Medical Informatics*, 76:334–341, 2007.

[34] Gnutella website. http://www.gnutella.com, last visited on May 2008.

[35] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[36] Health Level Seven (HL7). http://www.hl7.org/, last visited on May 2008.

[37] HL7 Clinical Document Architecture, Release 2. http://www.hl7.org/v3ballot/html/-infrastructure/cda/cda.htm, last visited on May 2008.

[38] HL7 Common Message Element Types. http://www.hl7.org/v3ballot/html/domains-/uvct/uvct.htm, last visited on May 2008.

[39] HL7 Development Framework (HDF). http://www.hl7.org/v3ballot/html/help/hdf/-hdf.htm, last visited on May 2008.

[40] HL7 Implementation Guide for 13606. http://www.hl7.org.uk/repository/uploads-/402/1/HL7%20Implementation%20Guide%20for%2013606%20scopev0r6.doc, last visited on May 2008.

[41] HL7 Reference Information Model. http://www.hl7.org/v3ballot/html/infrastructure/rim/rim.htm, last visited on May 2008.

[42] HL7 Refinement, Constraint and Localization, Release 2. http://www.hl7.org/v3ballot/html/infrastructure/conformance/conformance.htm, last visited on May 2008.

[43] HL7 Template and Archetype Architecture Version 3.0. http://www.hl7.org/library/committees/template/HL7_Atlanta_10_20_04.doc, last visited on May 2008.

[44] HL7 Template Specification. http://www.hl7.org/Library/Committees/template/-Templates%20Specification.doc, last visited on May 2008.

[45] HL7 V3 Guide. http://www.hl7.org/v3ballot/html/help/v3guide/v3guide.htm, last visited on May 2008.

[46] HL7 Version 3 Standard: Clinical Statement Pattern, Release 1. http://www.hl7.org/v3ballot/html/domains/uvcs/uvcs.htm, last visited on May 2008.

[47] I. Horrocks. DAML+OIL: a Description Logic for the Semantic Web. *IEEE Data Engineering Bulletin*, 25(1):4–9, 2002.

[48] IHE Cross Community Access XCA. http://www.ihe.net/Technical_Framework/-upload/IHE_ITI_TF_Supplement_XCA_TI_2007_08_15.pdf, last visited on May 2008.

[49] IHE Cross Community Information Exchange including Federation of XDS Affinity Domains. http://www.ihe.net/Technical_Framework/upload/IHE_ITI_TF_White-_Paper_Cross_Community_PC_2006_08_15.pdf, last visited on May 2008.

[50] IHE IT Infrastructure Integration Profiles. http://www.ihe.net/, last visited on May 2008.

[51] IHE IT Infrastructure Technical Framework, Cross-Enterprise Document Sharing-b (XDS.b). http://www.ihe.net/Technical_Framework/upload/-IHE_ITI_TF_Supplement_XDS-2.pdf, last visited on May 2008.

[52] IHE IT Infrastructure Technical Framework, Volume 1 Integration Profiles, Revision 4.0 Final Text, by the IHE ITI Technical Committee. http://www.ihe.net/-Technical_Framework/upload/IHE_ITI_TF_4_0_Vol1_FT_2007_08_22.pdf, last visited on May 2008.

[53] IHE Technical Frameworks. http://www.ihe.net/Technical_Framework/, last visited on May 2008.

[54] Jena A Semantic Web Framework for Java. http://jena.sourceforge.net/, last visited on May 2008.

[55] JUnit: a unit testing framework for the Java programming language. http://junit.org/, last visited on May 2008.

[56] KaZaA, Web Site . http://www.kazaa.com/, last visited on May 2008.

[57] O. Kilic and A. Dogac. Achieving Clinical Statement Interoperability using R-MIM and Archetype-based Semantic Transformations. *IEEE Transactions on Information Technology in Biomedicine*. to be published.

[58] O. Kilic, A. Dogac, and M. Eichelberg. Providing Interoperability of eHealth Communities through Peer-to-Peer Networks. *IEEE Transactions on Information Technology in Biomedicine*. submitted.

[59] C. E. Kuziemsky, F. Lau, I. Bilykh, J. H. Jahnke, G. McCallum, C. Obry, A. Onabajo, and G. M. Downing. Ontology-Based Information Integration in Health Care: A Focus on Palliative Care. In *STEP '03: Proceedings of the Eleventh Annual International Workshop on Software Technology and Engineering Practice*, pages 164–172, Washington, DC, USA, 2003. IEEE Computer Society.

[60] D. Lindberg and B. L. Humphreys. The unified medical language system (UMLS) and computer-based patient records. In Ball MJ, Collen MF, eds. Aspects of the computer-based patient record. New York: Springer-Verlag. pages 165–175, 1992.

[61] Log4j : a Java-based logging utility. http://logging.apache.org/log4j/, last visited on May 2008.

[62] Logical Observation Identifiers Names and Codes (LOINC). http://www.loinc.org, last visited on May 2008.

[63] A. Maedche, B. Motik, N. Silva, and R. Volz. MAFRA - A MApping FRAmework for Distributed Ontologies. In *EKAW '02: Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, pages 235–250, London, UK, 2002. Springer-Verlag.

[64] J.A. Maldonado, D. Moner, D. Tomás, C. Ángulo, Robles. M., and J.T. Fernández. Framework for Clinical Data Standardization Based on Archetypes. *Proceedings of MedInfo, K. Kuhn et al. (Eds), IOS publishing*, pages 454–458, 2007.

[65] D. Moner, J.A. Maldonado, D. Bosca, J.T. Fernandez, C. Angulo, P. Crespo, P.J. Vivancos, and M. Robles. Archetype-Based Semantic Integration and Standardization of Clinical Data. *In: Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE*, August.

[66] Napster website. http://www.napster.com, last visited on May 2008.

[67] W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. T. Schlosser, I. Brunkhorst, and A. Löser. Super-peer-based routing strategies for RDF-based peer-to-peer networks. *Journal of Web Semantics*, 1(2):177–186, 2004.

[68] openEHR Community . http://www.openehr.org/, last visited on May 2008.

[69] M. T. Pazienza, A. Stellato, M. Vindigni, and F. M. Zanzotto. XeOML: An XML-based extensible Ontology Mapping Language. *Workshop on Meaning Coordination and Negotiation, held in conjunction with 3rd International Semantic Web Conference (ISWC-2004)*.

[70] M. T. Pazienza, A. Stellato, F. M. Zanzotto, L. Henriksen, and P. Paggio. Ontology Mapping to Support Ontology-Based Question Answering. In *Proceedings of the Meaning 05 Workshop, Trento, Italy*, pages 67–73, February 2005.

[71] Pellet: OWL DL reasoner. http://pellet.owldl.com/, last visited on May 2008.

[72] R. Qamar and A. Rector. Semantic Issues in Integrating Data from Different Models to Achieve Data Interoperability. *Proceedings of MedInfo, K. Kuhn et al. (Eds), IOS publishing*, pages 674–678, 2007.

[73] RDF Vocabulary Description Language 1.0: RDF Schema. http://www.w3.org/TR-/rdf-schema/, last visited on May 2008.

[74] RDF/XML Syntax Specification. http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/, last visited on May 2008.

[75] A. L. Rector, S. Bechhofer, C. A. Goble, I. Horrocks, W. A. Nowlan, and W. D. Solomon. The GRAIL concept modeling language for medical terminology. *Artificial Intelligence in Medicine*, 9:139–171, 1997.

[76] A. L. Rector, W. A. Nowlan, and A. Glowinski. Goals for concept representation in the GALEN project. *Proceedings of the Seventeenth Annual Symposium on Computer Applications in Medical Care*, pages 414–418, 1993.

[77] RIDE D.4.4.2 - Implementation Guidelines For Member States. http://www.srdc.metu.edu.tr/webpage/projects/ride/deliverables/RIDE-D4.4.2v05.doc, last visited on May 2008.

[78] RIDE Deliverable 2.1.1 European Best practices. http://www.srdc.metu.edu.tr/webpage/projects/ride/modules.php?name=Calendar, last visited on May 2008.

[79] RIDE Deliverable 2.3.1 Requirement Analysis for the RIDE Roadmap. http://www.srdc.metu.edu.tr/webpage/projects/ride/modules.php?name=Calendar, last visited on May 2008.

[80] RIDE Project. http://www.srdc.metu.edu.tr/webpage/projects/ride/, last visited on May 2008.

[81] M. Schlosser, Decker S. Sintek, M. and, and W. Nejdl. HyperCuP – Hypercubes, Ontologies and Efficient Search on P2P Networks, International Workshop on Agents and P2P Computing, 2002.

[82] E. Sundvall, R. Qamar, M. Nyström, M. Forss, H. Petersson, H. Åhlfeldt, and A. Rector. Integration of Tools for Binding Archetypes to SNOMED CT. In *Proceedings of Semantic Mining Conference on SNOMED CT, Copenhagen, Denmark*, pages 64–68, October 2006.

[83] Systematized Nomenclature of Medicine (SNOMED) Clinical Terms. http://-www.snomed.org/snomedct/index.html, last visited on May 2008.

[84] W. Thoben, H.J. Appelrath, and S. Sauer. Record Linkage of Anonymous Data by Control Numbers. In W. Gaul, D. Pfeifer (eds.), From Data to Knowledge: Theoretical and Practical Aspects of Classification, Data Analysis and Knowledge Organisation, Oldenburg, pages 412-419. Springer-Verlag, Berlin, 1995 .

[85] Web Ontology Language OWL. http://www.w3.org/TR/owl-features/, last visited on May 2008.

[86] World Wide Web Consortium (W3C). http://www.w3.org/, last visited on May 2008.

[87] B. Yang and H. Garcia-Molina. Designing a Super-Peer Network. *In 19th International Conference on. Data Engineering (ICDE)*, pages 49–, 2003.

APPENDIX A

OWL REPRESENTATIONS OF RIM AND R-MIMS

This appendix provides the OWL representations of the RIM and the R-MIMs presented in Chapter 4. The OWL representation of the RIM (Figure 4.2 (a)) is given in Figure A.1. The OWL representations of and the R-MIMs (Figure 4.2 (b) and (c)) are shown in Figure A.2 and Figure A.3, respectively.

[43] describes a simple algorithm for object model to OWL mapping. As a first step, each class in the reference information model is represented as an OWL class. In Figure A.1, "Act", "Observation" and "ActRelationship" are those classes. [43] suggests to represent each relationship as an ObjectProperty and each data-valued property as a DatatypeProperty in OWL. The HL7 RIM defines two types of properties: attributes and associations. An association corresponds to a relation between two RIM classes and therefore it is represented as an object property in OWL. For example, the relation from the "Act" class to the "ActRelationship" class is denoted by the "outboundRelationship" association and represented as an object property as shown in Figure A.1. On the other hand, an attribute can either hold a single value or it can have a complex structure. For example, the "code" attribute in Figure 4.2 (a) has a complex structure. Therefore, we represent attributes of the HL7 RIM as object properties in OWL. In order to distinguish attributes and associations of a class, "attribute" property and "relation" property are added to the OWL representation of the RIM. All the object properties corresponding to the HL7 attributes are defined as subproperty of the "attribute" property and all the object properties corresponding to the HL7 associations are defined as the subproperty of the "relation" property. In Figure A.1, "outboundRelationship" is defined as the subproperty of the "relation" property and the "code" attribute is defined as subproperty of the "attribute" property.

```
<rdf:RDF xmlns="http://www.srdc.metu.edu.tr/webpage/links/rim.owl#"
     xml:base="http://www.srdc.metu.edu.tr/webpage/links/rim.owl"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#">
    <owl:Ontology rdf:about=""/>
    <owl:Class rdf:ID="Act"/>
    <owl:Class rdf:ID="ActRelationship"/>
    <owl:Class rdf:ID="ANY"/>
    <owl:ObjectProperty rdf:ID="attribute"/>
    <owl:Class rdf:ID="CD">
        <rdfs:subClassOf rdf:resource="#ANY"/>
    </owl:Class>
    <owl:DatatypeProperty rdf:ID="CD_code">
        <rdfs:domain rdf:resource="#CD"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="CD_codeSystem">
        <rdfs:domain rdf:resource="#CD"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="CD_codeSystemName">
        <rdfs:domain rdf:resource="#CD"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="CD_displayname">
        <rdfs:domain rdf:resource="#CD"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
    <owl:ObjectProperty rdf:ID="code">
        <rdfs:domain rdf:resource="#Act"/>
        <rdfs:range rdf:resource="#CD"/>
        <rdfs:subPropertyOf rdf:resource="#attribute"/>
    </owl:ObjectProperty>
    <owl:Class rdf:ID="Observation">
        <rdfs:subClassOf rdf:resource="#Act"/>
    </owl:Class>
    <owl:ObjectProperty rdf:ID="outboundRelationship">
        <rdfs:domain rdf:resource="#Act"/>
        <rdfs:range rdf:resource="#ActRelationship"/>
        <rdfs:subPropertyOf rdf:resource="#relation"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:ID="relation"/>
    <owl:ObjectProperty rdf:ID="target">
        <rdfs:domain rdf:resource="#ActRelationship"/>
        <rdfs:range rdf:resource="#Act"/>
        <rdfs:subPropertyOf rdf:resource="#relation"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:ID="value">
        <rdfs:domain rdf:resource="#Observation"/>
        <rdfs:range rdf:resource="#ANY"/>
        <rdfs:subPropertyOf rdf:resource="#attribute"/>
    </owl:ObjectProperty>
</rdf:RDF>
```

Figure A.1: The OWL representation of the RIM (Figure 4.2 (a))

```
<rdf:RDF xmlns="http://www.srdc.metu.edu.tr/webpage/links/ehrcom.owl#"
     xml:base="http://www.srdc.metu.edu.tr/webpage/links/ehrcom.owl"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rim="http://www.srdc.metu.edu.tr/webpage/links/rim.owl#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#">
    <owl:Ontology rdf:about="">
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/webpage/links/rim.owl"/>
    </owl:Ontology>
    <owl:Class rdf:ID="Component1">
        <rdfs:subClassOf rdf:resource="&rim;ActRelationship"/>
    </owl:Class>
    <owl:ObjectProperty rdf:ID="component1">
        <rdfs:domain rdf:resource="#Entry"/>
        <rdfs:range rdf:resource="#Component1"/>
        <rdfs:subPropertyOf rdf:resource="&rim;outboundRelationship"/>
    </owl:ObjectProperty>
    <owl:Class rdf:ID="Element">
        <rdfs:subClassOf rdf:resource="&rim;Observation"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;value"/>
                <owl:cardinality rdf:datatype="&xsd;int">1</owl:cardinality>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:ObjectProperty rdf:ID="element">
        <rdfs:domain rdf:resource="#Component1"/>
        <rdfs:range rdf:resource="#Element"/>
        <rdfs:subPropertyOf rdf:resource="&rim;target"/>
    </owl:ObjectProperty>
    <owl:Class rdf:ID="Entry">
        <rdfs:subClassOf rdf:resource="&rim;Act"/>
    </owl:Class>
</rdf:RDF>
```

Figure A.2: The OWL representation of the EHRcom R-MIM (Figure 4.2 (b))

```
<rdf:RDF xmlns="http://www.srdc.metu.edu.tr/webpage/links/cda.owl#"
     xml:base="http://www.srdc.metu.edu.tr/webpage/links/cda.owl"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rim="http://www.srdc.metu.edu.tr/webpage/links/rim.owl#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#">
    <owl:Ontology rdf:about="">
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/webpage/links/rim.owl"/>
    </owl:Ontology>
    <owl:ObjectProperty rdf:ID="clinicalStatement">
        <rdfs:domain rdf:resource="#EntryRelationship"/>
        <rdfs:range rdf:resource="#Observation"/>
        <rdfs:subPropertyOf rdf:resource="&rim;target"/>
    </owl:ObjectProperty>
    <owl:Class rdf:ID="EntryRelationship">
        <rdfs:subClassOf rdf:resource="&rim;ActRelationship"/>
    </owl:Class>
    <owl:ObjectProperty rdf:ID="entryRelationship">
        <rdfs:domain rdf:resource="#Observation"/>
        <rdfs:range rdf:resource="#EntryRelationship"/>
        <rdfs:subPropertyOf rdf:resource="&rim;outboundRelationship"/>
    </owl:ObjectProperty>
    <owl:Class rdf:ID="Observation">
        <rdfs:subClassOf rdf:resource="&rim;Observation"/>
    </owl:Class>
    <rdf:Description rdf:about="&rim;ActRelationship"/>
    <rdf:Description rdf:about="&rim;Observation"/>
</rdf:RDF>
```

Figure A.3: The OWL representation of the CDA R-MIM (Figure 4.2 (c))

APPENDIX B

OWL REPRESENTATIONS OF ARCHETYPES

This appendix provides the OWL representations of the archetypes described in Chapter 4. In Figure B.1, "Archetype" class which is used to mark the root classes of the archetypes is shown.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://www.srdc.metu.edu.tr/ontologies/Archetype.owl#"
  xml:base="http://www.srdc.metu.edu.tr/ontologies/Archetype.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Archetype"/>
</rdf:RDF>
```

Figure B.1: Archetype Class

```
<rdf:RDF xmlns="http://www.srdc.metu.edu.tr/ontologies/CDABodyTemperature.owl#"
         xml:base="http://www.srdc.metu.edu.tr/ontologies/CDABodyTemperature.owl"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
         xmlns:cda="http://www.srdc.metu.edu.tr/ontologies/CDA.owl#"
         xmlns:rim="http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:owl="http://www.w3.org/2002/07/owl#">
    <owl:Ontology rdf:about="">
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl"/>
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/CDA.owl"/>
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/Archetype.owl"/>
    </owl:Ontology>
    <owl:Class rdf:ID="CDA-BT">
        <rdfs:subClassOf rdf:resource="&cda;Observation"/>
        <rdfs:subClassOf rdf:resource="&arc;Archetype"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;classCode"/>
                <owl:hasValue rdf:resource="#classCode-OBS"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;moodCode"/>
                <owl:hasValue rdf:resource="#moodCode-EVN"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;code"/>
                <owl:allValuesFrom rdf:resource="#CDA-BT-CD"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&cda;entryRelationship"/>
                <owl:maxCardinality rdf:datatype="&xsd;int">0</owl:maxCardinality>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
...
```

Figure B.2: OWL representation of the CDA Body Temperature Archetype (Part1)

```
...
    <owl:Class rdf:ID="CDA-BT-CD">
        <rdfs:subClassOf rdf:resource="&rim;CD"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;CD_code"/>
                <owl:hasValue rdf:datatype="&xsd;string">386725007</owl:hasValue>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;CD_codeSystem"/>
                <owl:hasValue rdf:datatype="&xsd;string">2.16.840.1.113883.6.96</owl:hasValue>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <rim:CS rdf:ID="classCode-OBS">
     <rim:CD_code rdf:datatype="&xsd;string">OBS</rim:CD_code>
    </rim:CS>
    <rim:CS rdf:ID="moodCode-EVN">
     <rim:CD_code rdf:datatype="&xsd;string">EVN</rim:CD_code>
    </rim:CS>
</rdf:RDF>
```

Figure B.3: OWL representation of the CDA Body Temperature Archetype (Part 2)

```
<rdf:RDF xmlns="http://www.srdc.metu.edu.tr/ontologies/EHRcom-BT.owl#"
     xml:base="http://www.srdc.metu.edu.tr/ontologies/EHRcom-BT.owl"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:ehrcom="http://www.srdc.metu.edu.tr/ontologies/EHRcom.owl"
     xmlns:rim="http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#">
    <owl:Ontology rdf:about="">
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl"/>
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/EHRcom.owl"/>
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/Archetype.owl"/>
    </owl:Ontology>

<owl:Class rdf:ID="EHRcom-BT">
  <rdfs:subClassOf rdf:resource="&ehrcom;Entry"/>
  <rdfs:subClassOf rdf:resource="&arc;Archetype"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom>
          <owl:Class rdf:about="#EHRcom-BT-Comp"/>
      </owl:allValuesFrom>
      <owl:onProperty rdf:resource="&ehrcom;component1"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="EHRcom-BT-Comp">
  <rdfs:subClassOf rdf:resource="&ehrcom;Component1"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource="#EHRcom-BT-Element"/>
      <owl:onProperty rdf:resource="&ehrcom;element"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="EHRcom-BT-Element">
  <rdfs:subClassOf rdf:resource="&ehrcom;Element"/>
</owl:Class>
</rdf:RDF>
```

Figure B.4: OWL representation of the EHRcom Body Temperature Archetype

```
<rdf:RDF xmlns="http://www.srdc.metu.edu.tr/ontologies/EHRcom-BP.owl#"
         xml:base="http://www.srdc.metu.edu.tr/ontologies/EHRcom-BP.owl"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
         xmlns:ehrcom="http://www.srdc.metu.edu.tr/ontologies/EHRcom.owl"
         xmlns:rim="http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:owl="http://www.w3.org/2002/07/owl#">
    <owl:Ontology rdf:about="">
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl"/>
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/EHRcom.owl"/>
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/Archetype.owl"/>
    </owl:Ontology>
    <owl:Class rdf:ID="EHRcom-BP">
        <rdfs:subClassOf rdf:resource="&ehrcom;Entry"/>
        <rdfs:subClassOf rdf:resource="&arc;Archetype"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:allValuesFrom>
                    <owl:Restriction>
                        <owl:unionOf  rdf:parseType="Collection">
                            <owl:Class rdf:about="#EHRcomSystolicBPComp"/>
                            <owl:Class rdf:about="#EHRcomDiastolicBPComp"/>
                        </owl:unionOf>
                    </owl:Restriction>
                </owl:allValuesFrom>
                <owl:onProperty rdf:resource="&ehrcom;component1"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;code"/>
                <owl:allValuesFrom rdf:resource="#EHRcom-CuffBloodPressure-CD"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>

    <owl:Class rdf:ID="EHRcomSystolicBPComp">
        <rdfs:subClassOf rdf:resource="&ehrcom;Component1"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:allValuesFrom rdf:resource="#EHRcom-Systolic-BP"/>
                <owl:onProperty rdf:resource="&ehrcom;element"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
...
```

Figure B.5: CEN Blood Pressure Archetype OWL representation (part 1)

```
...
    <owl:Class rdf:ID="EHRcomDiastolicBPComp">
        <rdfs:subClassOf rdf:resource="&ehrcom;Component1"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:allValuesFrom rdf:resource="#EHRcom-Diastolic-BP"/>
                <owl:onProperty rdf:resource="&ehrcom;element"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="EHRcom-Systolic-BP">
        <rdfs:subClassOf rdf:resource="&ehrcom;Element"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;code"/>
                <owl:allValuesFrom rdf:resource="#EHRcom-SystolicBP-CD"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="EHRcom-Diastolic-BP">
        <rdfs:subClassOf rdf:resource="&ehrcom;Element"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;code"/>
                <owl:allValuesFrom rdf:resource="#EHRcom-DiastolicBP-CD"/>
            </owl:Restriction>
        </rdfs:subClassOf>

    </owl:Class>

    <owl:Class rdf:ID="EHRcom-CuffBloodPressure-CD">
        <rdfs:subClassOf rdf:resource="&rim;CD"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;CD_code"/>
                <owl:hasValue rdf:datatype="&xsd;string">251076008</owl:hasValue>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;CD_codeSystem"/>
                <owl:hasValue rdf:datatype="&xsd;string"
                        >2.16.840.1.113883.6.96</owl:hasValue>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
...
```

Figure B.6: CEN Blood Pressure Archetype OWL representation (part 2)

```
...
    <owl:Class rdf:ID="EHRcom-SystolicBP-CD">
        <rdfs:subClassOf rdf:resource="&rim;CD"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;CD_code"/>
                <owl:hasValue rdf:datatype="&xsd;string">271649006</owl:hasValue>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;CD_codeSystem"/>
                <owl:hasValue rdf:datatype="&xsd;string"
                        >2.16.840.1.113883.6.96</owl:hasValue>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>

    <owl:Class rdf:ID="EHRcom-DiastolicBP-CD">
        <rdfs:subClassOf rdf:resource="&rim;CD"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;CD_code"/>
                <owl:hasValue rdf:datatype="&xsd;string">271649006</owl:hasValue>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;CD_codeSystem"/>
                <owl:hasValue rdf:datatype="&xsd;string"
                        >2.16.840.1.113883.6.96</owl:hasValue>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>

</rdf:RDF>
```

Figure B.7: CEN Blood Pressure Archetype OWL representation (part 3)

```
<rdf:RDF xmlns="http://www.srdc.metu.edu.tr/ontologies/CDABloodPressure.owl#"
         xml:base="http://www.srdc.metu.edu.tr/ontologies/CDABloodPressure.owl"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
         xmlns:cda="http://www.srdc.metu.edu.tr/ontologies/CDA.owl#"
         xmlns:rim="http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:owl="http://www.w3.org/2002/07/owl#">
    <owl:Ontology rdf:about="">
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/RIMOntology.owl"/>
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/CDA.owl"/>
        <owl:imports rdf:resource="http://www.srdc.metu.edu.tr/ontologies/Archetype.owl"/>
    </owl:Ontology>
    <owl:Class rdf:ID="CDA-BP">
        <rdfs:subClassOf rdf:resource="&cda;Observation"/>
        <rdfs:subClassOf rdf:resource="&arc;Archetype"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;value"/>
                <owl:maxCardinality rdf:datatype="&xsd;int">0</owl:maxCardinality>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:allValuesFrom>
                    <owl:Restriction>
                        <owl:unionOf  rdf:parseType="Collection">
                            <owl:Class rdf:about="#CDASystolicBPER"/>
                            <owl:Class rdf:about="#CDADiastolicBPER"/>
                        </owl:unionOf>
                    </owl:Restriction>
                </owl:allValuesFrom>
                <owl:onProperty rdf:resource="&cda;entryRelationship"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;code"/>
                <owl:allValuesFrom rdf:resource="#CDA-CuffBloodPressure-CD"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
...
```

Figure B.8: CDA Blood Pressure Archetype OWL representation (Part 1)

```
...
    <owl:Class rdf:ID="CDASystolicBPER">
        <rdfs:subClassOf rdf:resource="&cda;EntryRelationship"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:allValuesFrom rdf:resource="#CDASystolicBP"/>
                <owl:onProperty rdf:resource="&cda;clinicalStatement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>

    <owl:Class rdf:ID="CDADiastolicBPER">
        <rdfs:subClassOf rdf:resource="&cda;EntryRelationship"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:allValuesFrom rdf:resource="#CDADiastolicBP"/>
                <owl:onProperty rdf:resource="&cda;clinicalStatement"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>

    <owl:Class rdf:ID="CDASystolicBP">
        <rdfs:subClassOf rdf:resource="&cda;Observation"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;classCode"/>
                <owl:hasValue rdf:resource="#classCode-OBS"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;moodCode"/>
                <owl:hasValue rdf:resource="#moodCode-EVN"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&cda;entryRelationship"/>
                <owl:maxCardinality rdf:datatype="&xsd;int">0</owl:maxCardinality>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
...
```

Figure B.9: CDA Blood Pressure Archetype OWL representation (Part 2)

```
...
    <owl:Class rdf:ID="CDADiastolicBP">
        <rdfs:subClassOf rdf:resource="&cda;Observation"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;classCode"/>
                <owl:hasValue rdf:resource="#classCode-OBS"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;moodCode"/>
                <owl:hasValue rdf:resource="#moodCode-EVN"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&cda;entryRelationship"/>
                <owl:maxCardinality rdf:datatype="&xsd;int">0</owl:maxCardinality>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>

    <owl:Class rdf:ID="CDA-CuffBloodPressure-CD">
        <rdfs:subClassOf rdf:resource="&rim;CD"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;CD_code"/>
                <owl:hasValue rdf:datatype="&xsd;string">251076008</owl:hasValue>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&rim;CD_codeSystem"/>
                <owl:hasValue rdf:datatype="&xsd;string"
                        >2.16.840.1.113883.6.96</owl:hasValue>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <rim:CS rdf:ID="classCode-OBS">
     <rim:CD_code rdf:datatype="&xsd;string">OBS</rim:CD_code>
    </rim:CS>
    <rim:CS rdf:ID="moodCode-EVN">
     <rim:CD_code rdf:datatype="&xsd;string">EVN</rim:CD_code>
    </rim:CS>
</rdf:RDF>
```

Figure B.10: CDA Blood Pressure Archetype OWL representation (Part 3)

VITA

PERSONAL INFORMATION

Surname, Name:	Kılıç, Özgür
Nationality:	Turkish (TC)
Date and Place of Birth:	23 October 1975, Eskişehir
Marital Status:	Married
Phone:	+90 312 495 2868
e-Mail:	kilic_ozgur@hotmail.com

EDUCATION

Degree	**Institution**	**Year of Graduation**
MS	METU-Computer Engineering	2001
BS	Bilkent University-Computer Engineering	1998
High School	Eskişehir Science High School	1993

WORK EXPERIENCE

Year	**Place**	**Enrollment**
2004-present	Havelsan A.Ş.	Software Engineer
1999-2004	Information Systems Framework Ltd. Şti	Software Engineer
1998-1999	cyberSoft	Software Engineer

FOREIGN LANGUAGES

Advanced English

PUBLICATIONS

1. Kilic O., Dogac A., Eichelberg M. "Providing Interoperability of eHealth Communities through Peer-to-Peer Networks," *IEEE Transactions on Information Technology in Biomedicine*, submitted.

2. Kilic O., Dogac A., "Achieving Clinical Statement Interoperability using R-MIM and Archetype-based Semantic Transformations," *IEEE Transactions on Information Technology in Biomedicine*, to be published.

3. Bicer V., Kilic O., Dogac A., Laleci G., "Archetype-based Semantic Interoperability of Web Service Messages in the Healthcare Domain", *Int'l Journal on Semantic Web Information Systems*, Vol. 1, No.4, October 2005, pp. 1-22.

4. Dogac A., Kabak Y., Gulderen O., Namli T., Okcan A., Kilic O., Gurcan Y., Orhan U., Laleci G., ebBP Profile for Integrating Healthcare Enterprise (IHE) Submitted to OASIS ebXML Business Process Technical Committee.

5. Ozen B., Kilic O., Altinel M., Dogac A. "Highly Personalized Information Delivery to Mobile Clients" *Wireless Network: The Journal of Mobile Communication, Computation and Information* , Volume 10, No. 6, November 2004, pp. 665-683.