Engineering Knowledge Exchange for Translational Research Informatics

by

F. Mason-Blakley
B.Sc., University of Victoria, 2003

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Engineering Knowledge Exchange for Translational Research Informatics

by

F. Mason-Blakley
B.Sc., University of Victoria, 2003

Supervisory Committee

Dr. J.-H. Weber, Supervisor
(Department of Computer Science)

Dr. M. Tory, Departmental Member
(Department of Computer Science)

Supervisory Committee

Dr. J.-H. Weber, Supervisor
(Department of Computer Science)

Dr. M. Tory, Departmental Member
(Department of Computer Science)

ABSTRACT

Engineering effective knowledge exchange pathways between scientists and clinicians will accelerate the development and improvement of clinical treatments extracted from lab bench experiments. Many standards development organizations in the field of translational research informatics have attempted to prescribe mechanisms which would provide these knowledge exchange pathways; however, concrete implementations of these standards and the software structures which support them are still lacking. We have explored key technologies and techniques which may facilitate knowledge exchange through clinical coding, a domain specific version of the more general technique of semantic annotation. During the development process we identified and provided potential solutions to four primary problem areas in engineering software enabled knowledge exchange pathways for translational research: architecture, terminology, validation and interface design. We provide both a technical and practical evaluation of a multicomponent architecture which was conceived as a mechanism for producing knowledge exchange pathways between researchers in the field of cancer informatics; however, the principles and process which we apply to cancer informatics could easily be applied to other areas of clinical informatics.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

Cliff McCollum, and GenoLogics Life Sciences, for the support and resources they provided.

Dr. Jens Weber-Jahnke, for mentoring, support, encouragement, and patience.

Mitacs Accelerate BC, and NSERC for enabling my funding.

DEDICATION

For my family who motivated me to keep at it

Chapter 1

Introduction

1.1 Overview

The lack of informatics support in translational research is impeding the evolution of basic science research discoveries into bedside clinical treatments. It is hoped that facilitating the exchange of knowledge between the experts in the plethora of translational research subdomains will accelerate these evolutions. The language barriers which exist between these experts are principal impediments to this exchange of knowledge. Supplementing this difficulty are the variety of formats in which these actors exchange the data they collect. A further difficulty still is the mosaic of heterogeneous informatics architectures which have been created to support the various member institutions and researchers in the field.

In this thesis we propose a conceptual architecture which can be adapted to meet the specific knowledge exchange needs of a given expert within the translational research domain. The architecture incorporates a translation engine which addresses the terminological barriers between the actors in the field. As part of this translation engine, we incorporate a secondary contribution: a terminology visualization. This visualization is designed to facilitate the understanding of diverse terminologies and also to facilitate semantic annotation of scientific and medical results. The architecture also incorporates a modularized persistence engine which is designed to facilitate the integration of data between disparate informatics systems, and finally, it incorporates an export, transform and load engine which utilizes electronic medical record standards to verify system output. We have validated our conceptual architecture with feedback from external industrial collaborators at GenoLogics Life Sciences Inc.,

and by basing our design decisions on design theory and best engineering practices, as derived from a literature review.

To place these contributions in context, we will now provide an introduction to translational research. We will use this introduction to focus the reader's attention on the specific subdomain of the field which we have targeted with our contributions.

1.2 Translational Research

Communication and knowledge exchange are central themes in the field of translational research, a domain which encompasses our target subdomain: clinical research informatics. According to the National Institute of Health (NIH)[64], translational research is the study of medicine from bench to bedside, but also from bedside to bench. Discoveries are sometimes first made in basic research on the bench and subsequently percolate through clinical trials and then into clinical practice at the bedside. This communication also occurs in the opposite direction. Clinical researchers sometimes make discoveries about the nature and progression of disease that stimulate basic scientific investigations. In each of these facets of translational research, knowledge exchange is required to further our understanding of human health.

Embi[21] emphasizes the theme of communication and knowledge exchange with a diagram which has been replicated in fig. 1.1. This emphasis is common in the literature; it is repeated, for example, by both Beaulah[6] and Woolf[86]. In his diagram, Embi illustrates his classification of the informatics domains which support translational research. Collectively, these fields are referred to in general as translational research informatics (TRI).

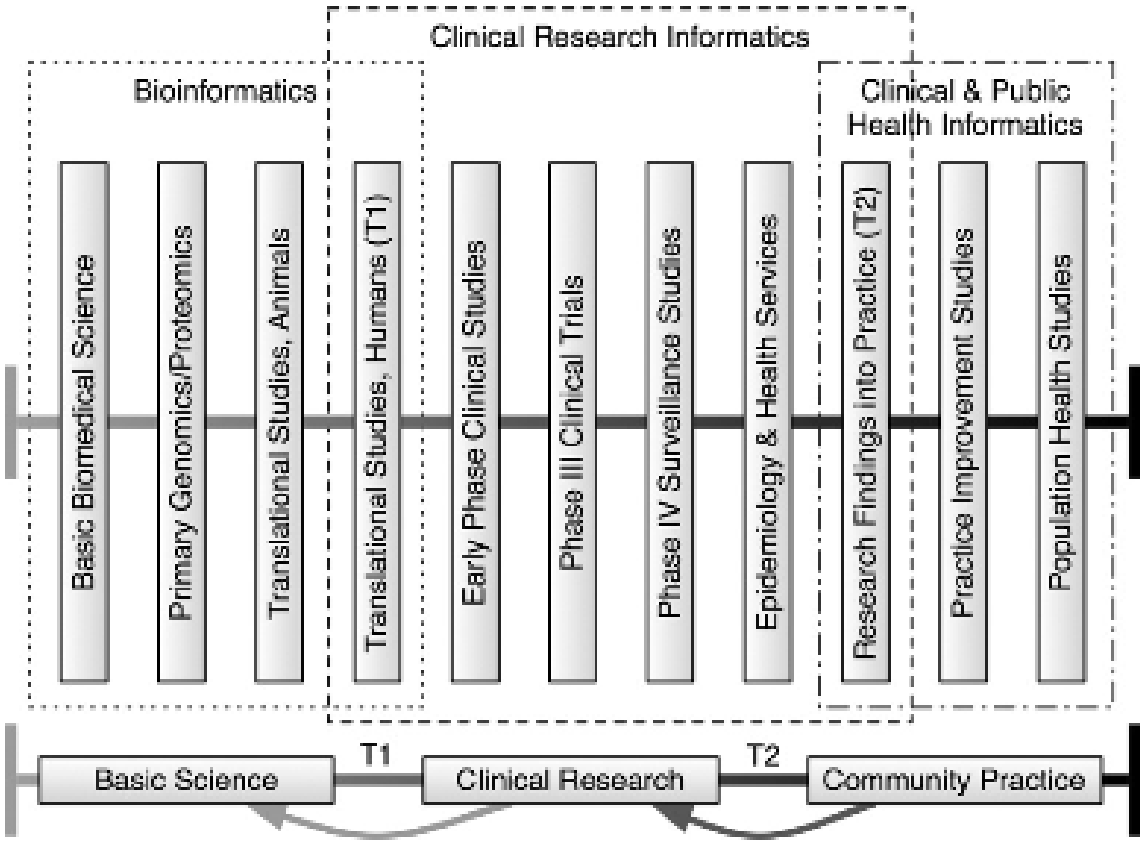


Figure 1.1: Embi's overview of translational research[21]

As shown in fig 1.1, the informatics support required by the field of translational research can be divided into three principal categories: bioinformatics, clinical research informatics and clinical and public health informatics. Bioinformatics fulfills the computational and electronic communication needs of the basic sciences. Clinical and public health informatics fulfills the same needs in clinical practice. Finally, clinical research informatics (CRI) fills the needs in between these fields, but with some overlap. The purpose of each of these subdomains of translational research is to provide mechanisms with which researchers can communicate and exchange their knowledge.

This thesis will focus primarily on the subdomain of CRI. The architecture which we propose in this thesis has a narrower focus still. It provides a framework for systems which are intended to provide interoperable informatics support for the subsection of

the CRI subdomain which is commonly referred to as the T2 block, as illustrated in fig 1.1. The focus of this block is on the incorporation of clinical findings into clinical practice, in other words the exchange of knowledge between clinical researchers and clinical practitioners.

Having now focused the reader's attention on CRI, and specifically the T2 subdomain of CRI, we will now motivate our choice of subdomain and our approach to the problem of knowledge exchange in this subdomain by discussing the work of Liebman[52]. We will also use Liebman's work to clarify what it is that we mean when we discuss knowledge exchange.

1.3 Driving Innovation with a Top Down Approach

The architecture we provide as our primary contribution is designed to facilitate the transfer of knowledge from medical specialists to their associates. Liebman[52] suggests that increasing knowledge exchange between medical specialists and researchers in the basic sciences is a key way in which innovation in medical treatments and personalized medicine can be driven. We facilitate these advancements by providing a mechanism for knowledge exchange between experts in clinical practice and clinical research that can be adapted to satisfy the needs of stakeholders in the basic sciences.

As we discussed in the previous paragraph, Liebman advocates for increasing the rate of transfer of knowledge from clinicians down the chain of researchers to those in the basic sciences. But what is knowledge? Liebman makes a series of statements which provide implied definitions of data, information, and knowledge when he discusses how biomedical data matures through a number of stages and is eventually incorporated into clinical practice.

> [D]ata are converted into information when redundancies are removed and [they are] cleaned to remove spurious results; information becomes knowledge when its interpretation leads to new discoveries, e.g., biomarkers, pathways, gene correlations; and knowledge evolves to clinical utility when it finally is incorporated into the actual practice of medicine, e.g., biomarkers become diagnostics and hopefully, causal diagnostics.[52] [1]

[1] If the reader is challenged by the language in this quotation, appendix 1 provides a sample scenario of knowledge exchange in oncology and a subsequent analysis which may alleviate some of this difficulty

By providing these implied definitions, Liebman gives guidance on what types of content might be exported from a knowledge source which had the potential to drive revolutionary biomedical research. We attempt to incorporate his guidance into the recommendations we make in our proposed architecture.

Having now motivated our selection of subdomain, we provide additional motivation for our research by discussing the financial cost of the shortage of informatics support in translational research.

1.4 The Financial Cost

The difficulty in communication in translational research translates to a real financial cost not only as a result of the time taken to integrate legacy systems, but also through the cost of opportunity. Frank[24] discusses the cost of research and development in the pharmaceutical industry, and specifically in the process of galenical drug formulation. He states, circa 1995, that the development of a typical drug from chemical synthesis to market takes on the order of twelve years and costs four hundred million dollars. He then goes on to inform us that drug patents typically only last twenty years leaving only eight years in which pharmaceuticals can leverage their exclusive rights. Beaulah[6] implies in his 2008 publication that these great financial pressures on pharmaceutical companies remain fifteen years later. It is therefore still of exceptional importance to the pharmaceutical companies, from a financial perspective, to reduce their research and development costs and to shorten the drug development process. When we combine this information with the statements made by Embi[21], Beaulah and Woolf[86] which each indicate that a primary impediment to the progression of research in the field is knowledge exchange, we begin to understand how a contribution which facilitates this exchange is in fact significant. In order to reduce the time taken to develop basic science discoveries into clinical treatments we must address the communications challenges faced by the experts in this field. These challenges are discussed in the next section.

1.5 The Challenge of Communication

An increasing number of clinical research infromatics (CRI) communications now occur using electronic medical records (EMR). EMRs have been designed with the intent of providing a mechanism for knowledge exchange; however, Chen[15] and

Maldonado[56] describe how there is great difficulty in agreeing on a reference model which should be used as the basis for EMRs. Reference models are one of many mechanisms used to facilitate knowledge exchange. There are a variety of standards development organizations (SDOs) that are attempting to prescribe the content and format of these EMRs, including Health Level Seven (HL7), Open Electronic Health Records (Open EHR) and Integrating the Healthcare Enterprise (IHE), but it is challenging for these SDOs to satisfy all of their stakeholders.

Both Embi[21] and Beaulah[6] also advocate for further discussion on the mechanisms which should be used to exchange knowledge in CRI. Embi states that many *previous studies have indicated that a lack of information technology tools has been an impediment to the "expedient, effective and resource-efficient conduct of clinical research activities"*. He states that the availability of clinical care data for secondary purposes has become a paramount concern in the CRI field, and describes how "clinical researchers are faced with significant and increasingly complex workflow and information management challenges". Beaulah on the other hand takes a less philosophical and more pragmatic approach by advocating for the development and exploration of translational research focused workflow software.

Through his study, Embi discovered that major players in the CRI field perceive a variety of significant problems in their domain all of which pertain in one way or another to knowledge exchange. Embi and his study subjects identify workflow, standards, education and data access, integration and analysis as areas of primary need in CRI. Beaulah supports Embi's claim that the lack of workflow support in the field is a significant problem in the field. Woolf's[86] discussion on the lack of coalescence about the definition and scope of the T2 block of translational research speaks clearly to the lack of understanding of the domain, even by experts, thus indicating the need for education. Ruttenberg[71] discusses how semantic web technologies and standards are not yet sufficiently mature to support automated semantic translational research knowledge exchange. From the clinical perspective, we can read about Dolin's[19] efforts with the Health Level Seven Version 3 standard to learn about some of the perceived shortcomings of current clinical informatics standards. To support Embi's claims about the challenges in data access, integration and analysis we can refer the reader to work by Beaulah, Ruttenberg or Patterson[68] respectively.

On the point of workflow, Embi informs us that many CRI players complain that the available informatics platforms in the domain do not support existing or optimized clinical research workflows and that this failure is acting as a barrier to

adoption. Workflow in many cases could be described as the process of data collection or analysis. Embi writes, as does Beaulah, that there is a general call for validation of workflow and informatics intervention strategies in the CRI domain. Embi states that this is particularly problematic at the interface between clinical research and clinical care, in other words, the T2 block of translational research which we have targeted with our architecture. Beaulah emphasizes on the other hand that most translational research informatics systems display shortcomings in their ability to support portions of the researcher's work flows which require inter-institutional communication.

Embi claims that there is a "need for CRI data standards [and] models", that there is a need to "[a]pply clinical standards to research", that CRI experts "[n]eed ways to span biological to clinical ontologies", and that there is a "[n]eed to standardize nontechnical institutional and sponsor requirements. The general need to improve standards is reflected in Dolin's work on the evolution of the Health Level 7 family of standards. It is also reflected in the work of Kussaibi[51] who was involved with Daniel[18] and others in the authoring of the Integrating the Healthcare Enterprise Anatomic Pathology Structured Report Profile. We address this issue by employing standards which are developed through large scale consensus mechanisms and standards which undergo continual practical evaluations.

On the point of education, Embi states that a lack of education about CRI theory and practice is impeding progress in the field. In short, without education of all stakeholders, from the scientist at the bench, to the clinician at the bedside, all the players in between and also all of their taskmasters, it is difficult to describe the scope and difficulty of the problem and consequently it is difficult to encourage the attribution of the necessary resources.

On the point of data access, integration and analysis, Embi identifies policy, organizational and practical issues which are limiting the availability of CRI data for secondary clinical uses. This spectrum of issues is also mentioned in passing by Beaulah.

As Embi has described, the primary difficulties in the CRI domain revolve principally around communication, and as Liebman described, the transmission of messages is only a first step in communication. Developing a mechanism to decode those messages in such a way as to exchange knowledge is an important next step.

Given that we now recognize the need for knowledge exchange, we might now create or select mechanisms with which to provide this service. The primary way in which we address this issue in our architecture is through a process called semantic

annotation. In medicine, this process is also called clinical coding. In the next section we provide a brief introduction to the concept of semantics and to the technique of semantic annotation.

1.6 Knowledge, Semantics and Annotation

How do we provide mechanisms for knowledge exchange? In order to derive understanding from clinical or experimental results it is critical that we understand what those results mean. The study of meaning is sometimes referred to as semantics. This term however, has taken on a more broad definition in computing where it is often used to refer simply to the meaning of data. Weng declares that "[s]emantic interoperability is one of the great challenges in biomedical informatics."[84] He also claims that "[m]ethods such as ontology alignment or use of metadata neither scale nor fundamentally alleviate semantic heterogeneity among information sources."[84] We can interpret from Weng's statements that in order to provide mechanisms for knowledge exchange, at a minimum, we need to identify a shared language which has commonly understood semantics. This is not strictly true, but alternative approaches which involve the maintenance of multiple terminology sources are much more complicated. As Weng declared, the mechanisms for their integration do not perform as well as we might hope. Unified Medical Language Systems and terminology maps have been developed to translate semantics among different terminologies, and so we incorporate this preexisting work into our architecture.

Independently of the question whether a single or multiple terminologies are used, these terminologies must be applied to clinical data. One way of doing so is by semantically annotating data with controlled terms. Semantic annotation essentially boils down to the association of terms, be they words in a narrative, labels in relational schema or notes in some other context, with alphanumeric codes which relate those terms to centrally negotiated and agreed upon definitions[50].

Having now discussed the language barrier issue which we introduced in our opening section, we place our approach to this problem within the context of knowledge exchange in translational research by providing a more detailed overview or our primary contribution.

1.7 Architecture Overview

Our primary contribution, illustrated in fig. 1.2, is a conceptual architecture for systems which can capture and subsequently exchange clinical research knowledge. Our secondary contribution, an implementation of a component of the first, is a visualization of a terminology server which provides terminological mappings between many controlled vocabularies in the field of translational research. This visualization is encapsulated in a semantic annotation interface. By introducing the conceptual architecture and the visualization, we provide ground work which can be adapted to develop systems which are better able to capture and exchange clinical knowledge in various clinical settings.

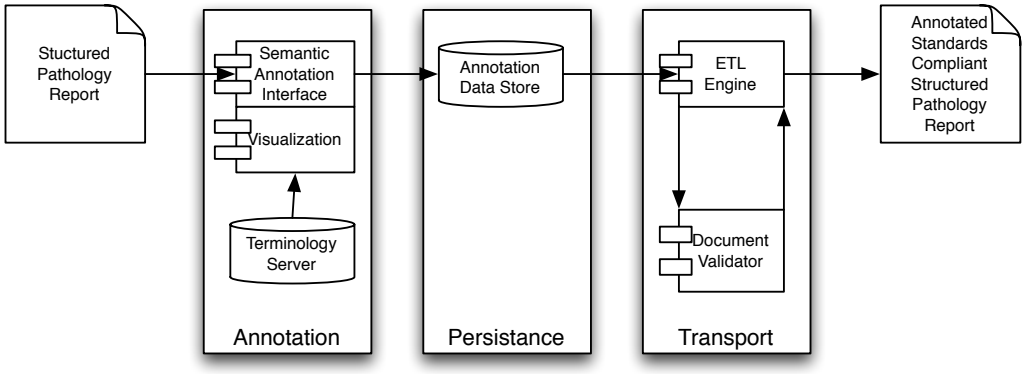


Figure 1.2: An architecture which facilitates the capture and exchange of clinical knowledge.

Our architecture is composed of three principal components: the annotation component, the persistence component and the transport component. The architecture takes raw text based anatomic pathology structured report (APSR) templates as input. As output, the architecture produces annotated, verified standards compliant APSRs. APSRs are electronic medical records (EMRs) which report on the properties of specimens (tissue samples) which have been extracted from oncology patients.

Using a system which employs our architecture, a clinical coder manually enters a raw text based APSR template using the semantic annotation interface component. This interface provides, as part of its feature set, a visualization of a terminology server. The clinical coder uses this visualization to select clinical concepts in the

terminology server with which to annotate the APSR. By providing these identifying concepts, the clinical coder encodes the fields and data contents of the APSR template with additional computable metadata which describes the intended meaning of the fields and data which are contained in the document.

These annotations which have been associated with the APSR are stored along side the APSR template and data in the persistence layer of the architecture. This persistence layer uses a relational schema which takes advantage of the semantics of the noun phrase labels that are used in the APSR templates and in medical observation more generally.

Finally, a multichannel export, transform and load (ETL) engine is used to extract the data from the persistence layer. The ETL engine uses the extracted data to form structured documents which it then validates using the document validation component. This document validation component is designed around two electronic medical record (EMR) standards. All documents which are generated are founded on the Health Level Seven Clinical Document Architecture. Documents which have encoded their data at a higher level of granularity will also rely on the Integrating the Healthcare Enterprise's Anatomic Pathology Structured Report Profile. Once the generated documents have been validated, the ETL engine exports them over an XML output channel.

1.8 Summary of Contributions

The primary contribution presented in this thesis is an architecture which may be employed in clinical research informatics systems that facilitate the exchange of knowledge which has been captured through clinical practice. We evaluate the architecture by examining each of its proposed components. We pay particular attention to the validity of the information that systems which employ the architecture would capture, and the reusability of that information by potential clients of those systems. By employing standards for communication protocols, syntax and content, and by evaluating the degree of adoption and reasons for adoption of those standards we develop a convincing argument to adopt the recommended architecture.

A secondary contribution is a prototypical implementation of a visualization interface to the Unified Medical Language System Metathesaurus. We describe how visualization theory and human computer interaction task lists[74][81] were used to evaluate this prototype. We also discuss briefly how this could be used in the context

of clinical coding to provide semantic annotations of data which was recorded in a CRI setting.

We have partially validated these components by having translational research experts at GenoLogics provide regular feedback on our implementations against the workflows which they have derived from their interactions with clients. We have also validated our contributions by relying on literature reviews to guide our design decisions. These literature reviews covered a variety of subjects including electronic medical record and scientific data informatics standards, more general informatics design, and visualization theories and task lists.

Having now provided an overview of the content of this thesis, we will now provide the reader with a synopsis of the thesis structure.

1.9 Thesis Structure

We will first discuss some of the work which is related to ours and on which we have in part based our architecture in chapter 2. We will then provide, in chapter 3, a discussion of foundational technologies and standards in the field of interoperability in clinical research informatics (CRI). In chapter 4 we discuss the industrial translational research informatics solution which has been developed by our collaborators, GenoLogics Life Sciences. This discussion will provide the context into which tools which employ our proposed architecture are perceived to fit. We then discuss our implementation experiences in chapter 5 and follow this discussion with an explicit discussion on the evaluation of our architecture in chapter 6. We close the thesis, in chapter 7, with a discussion on the elements of our architecture, and of the field of CRI more generally, which we believe deserve the most immediate attention in terms of additional research.

Chapter 2

Related Work

2.1 Overview

In this chapter we begin with a brief discussion on how the field of translational research has evolved from a collection of isolated research efforts to large multi-institutional collaborations. Architectures which do not have the capacity to satisfy the demands which will be placed on them are less likely to be implemented than those that do. By researching the environments for which our architecture is intended, we have begun to elicit some of the demands which might be placed on the systems which implement it. As we discuss these demands, we will introduce a number of standards development organizations (SDOs) which have made contributions to the translation research domain which are intended to address those demands. One such organization, and one whose efforts are dominant in the work we present in this thesis, is the cancer bioinformatics grid (caBIG). This organization has developed the standards which are used within the caGRID, one of the primary translational research collaboratives which are in operation today.

We will move from here into a discussion on the mechanisms used by caBIG to achieve interoperability between member systems of the caGRID research collaborative. We begin this discussion on the prescriptive accreditation process which is employed by caBIG and how it can interfere with the integration of the wealth of existing informatics systems within the translational research domain. In spite of this, there is value in some components of the caBIG architecture; consequently, we have incorporated elements of the caBIG design into our conceptual architecture. In the coming paragraphs we will provide a summary of the architecture proposed by

caBIG. We will later refer to this architecture in chapter 5 when we discuss our proposed architecture and contrast it against the one which is employed by member applications of the caGRID.

Two components of the caBIG architecture to which we pay particular attention are the enterprise vocabulary service (EVS) and the cancer data standards repository (caDSR). Collectively, these two components have a close analogue within our proposed architecture. The EVS and the caDSR are explored here in detail so as to allow comparison with our alternative approach in chapter 5.

We close with a discussion on the interfacing capabilities of caBIG systems. We provide this discussion so that we can later, in chapter 5, compare the interoperability interfaces provided by systems which employ our architecture to those that employ the caBIG architecture.

2.2 The Evolution of Translational Research

Frank's[24] single institution approach to translational research has become dated. Since 1995 when the work was performed the field has grown more collaborative. Research in the field is becoming less reliant on the work of singular researchers or even work which is being performed in single institution collaborations. As is described by Beaulah[6], scientific and medical questions in translational research are now being solved by large aggregates of research institutions rather than singular research efforts.

2.3 The caBIG Approach to Interoperability

2.3.1 Introduction

The cancer biomedical informatics grid (caGRID) is a network of interoperating biomedical informatics systems which have been established to facilitate the exchange of knowledge derived from cancer research and oncological practice[46]. The standards development organization (SDO) which maintains the standards used in this grid has the same name, but uses the acronym caBIG.

caBIG realizes the caGRID collaborative research community through a prescriptive accreditation process. At the outset of this research, of the three levels of acknowledged accreditation: Bronze, Silver and Gold, caBIG only provided Bronze level

certification for any software that had not been developed using its open source software development kit, the caCORE SDK[47]. Since then, caBIG has established a process through which external systems can now achieve Silver level accreditation.

Involvement within caBIG would have been required to achieve a significant level of accreditation from the organization. It was expected that the cost of this involvement would not have been justified from a business perspective for our industry collaborators at GenoLogics. In spite of our decision to not formally engage caBIG in our research, we did carefully consider the conditions they use to evaluate the level of accreditation which is assigned to a community member while we developed our architecture and prototypes. The criteria used by caBIG address important issues in interoperability. The four principle metrics caBIG uses in the evaluation of candidate systems are: the choice of information model, the choices of implemented ontologies, a description of the data managed by the system and finally a description of the programming and messaging interfaces which a given community member's system offers.[45]

2.4 caBIG from the Inside

2.4.1 Introduction

Komatsoulis[50] describes the process of developing a tool using native caBIG development methodologies and software components. The central component in this process is called the cancer common ontologic representation environment (caCORE) software development kit (SDK). Komatsoulis describes how the caCORE architecture addresses two of the primary challenges which must be overcome to achieve interoperability: "the ability to access data (syntactic interoperability) and understand the data once retrieved (semantic interoperability)".

The caCORE architecture is composed of three primary components. The first is a controlled terminology service, the enterprise vocabulary service (EVS), which provides a basis for semantic interoperability. The second is a standards-based metadata repository, the cancer Data Standards Repository (caDSR), which provides a basis for syntactic interoperability. Finally it includes an application programming interface (API) which was developed using a model driven architecture (MDA) methodology. This API facilitates message transmission and reception.

Over the course of the coming sections, we will first discuss Komatsoulis's archi-

tectural decisions. Next, we will discuss the interfaces which are provided by the systems which are constructed using the caCORE SDK. We will then explore the technical and practical aspects of the terminology server employed by caGRID informatics systems in detail. This component of the caBIG architecture is of significant importance to our discussion as we have made a conscious decision to deviate from the caBIG recommendation here and instead take a new path.

2.4.2 Architecture

The architecture used by Komatsoulis[50] in his caBIG reference implementation, which he calls cancer bioinformatics infrastructure objects (caBIO), is composed of a six components which can be clustered into three categories. The architecture itself is illustrated in fig. 2.1.

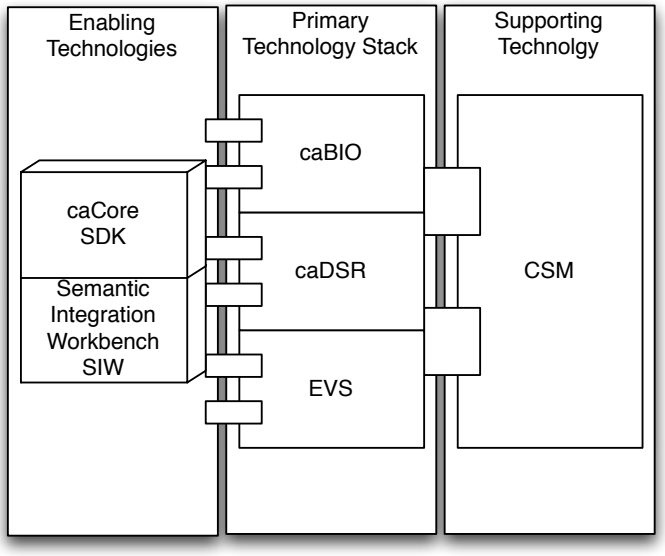


Figure 2.1: The major components of caCORE version 3. The primary technology stack contains a model driven, object oriented data system (caBIO in this example) and the metadata and controlled terminology services required to achieve semantic interoperability. Supporting this stack is a set of enabling technologies that simplifies the process of creating a caCORE-like system and a supporting technology stack that includes a Common Security Module (CSM) that can be readily implemented through the caCORE SDK.[50]

Firstly this technology stack includes Komatsoulis's reference implementation, caBIO, a system which is designed to record and exchange information regarding proteins, and genes. The reference implementation relies on Enterprise Vocabulary Services (EVSs) which supply the controlled terminology that is used to supply the semantic annotations necessary to identify the meaning of the data held in the system. The caDSR component of the system supplements the semantic meanings which are sourced from the EVS by providing an OWL/RDF like semantic supplement which provides context for the terms which are leveraged from the EVS. Collectively, these three components are classified as members of the primary technology stack.

These three architecture components are enabled by two principal technologies, the caCORE SDK which was used to implement them, and another distinct tool called the Semantic Integration Workbench (SIW) which is used to populate the caDSR with necessary metadata. These two components are classified together as enabling technologies. Finally, a security service, the Common Security Module (CSM), which spans the architecture is employed in tandem with the central interoperability components to control the flow of information into and out of the system through the management of user rights.

2.4.3 Interfaces

Komatsoulis[50] supplements this high level architectural view with a discussion of the technological aspects of the caCORE architecture where he informs us that "[d]ata systems in caCORE generally provide three or four interfaces to client systems. These APIs always include Java Bean and Web Services [interfaces], [and usually] include http interfaces ... The Java Bean and Web Services APIs are generated [by the caCORE SDK but the] http and other interfaces currently require manual coding. Wherever possible, these APIs are built on top of each other (i.e. the Web Services API translates requests into Java Beans) so that maximum API consistency can be maintained." Komatsoulis goes on to explain how the APIs which caCORE generates were initially based on Remote Method Invocation (RMI) technology but were translated in the version 3.0 caCORE SDK to rely instead on HTTP tunnelling.

2.4.4 A Closer Look at the EVS

Conceptual Overview

Komatsoulis[50] begins by describing the process of clinical coding or semantic annotation. This process involves formally binding words and phrases with alphanumeric concept codes which are invariant identifiers for a concept in a controlled terminology. In the caCORE environment "[t]his terminology is supplied by the National Cancer Institute's (NCI) EVS to indicate the meaning of the components of a [common data element] (CDE). The caDSR binds ISO 11179 components to controlled terminology [ISO 11179 is an ISO standard similar to OWL or RDF]." The concepts which are included in the EVS are sourced from the NCI Thesaurus and the NCI Metathesaurus. The first of these terminology sources represents a single terminology while the second contains an n-way mapping between multiple terminologies.

Interfaces

"Both [the NCI Thesaurus and the NCI Metathesaurus] are currently based on proprietary terminology service software... The caCORE client side API provides a unified open source interface to both EVS server environments. Applications that rely on EVS for terminology support make use of the caCORE API rather than the [proprietary interfaces]"[50] Komatsoulis addresses this issue directly. He states that at "the time EVS was originally constituted, there were few open source tools available for either terminology development or terminology server application, and none that appeared adequate for operational use in a large enterprise. The EVS was therefore constructed using proprietary commercial software. Consistent with NCICBs commitment to open software and content, EVS is moving to architecture that employs only open components. Currently, the EVS has migrated development of [the] NCI Metathesaurus ... [and] ... of [the] NCI Thesaurus to open software".

By electing to implement the unifying interface between the two semantic sources, the EVS and the caDSR, through a the Java based caCORE API which outputs XML, the caCORE development team does not tie their solution to a single technology. By electing to implement a strict XML interface, as opposed to a technology specific approach like remote method invocation (RMI) or remote procedure calls (RPC), they generalize their output thus making it more accessible to new systems which are integrated into the collaborating community of applications.

2.5 Summary

In this section we have discussed the architecture which caBIG prescribes and which is employed by members of the caGRID. This architecture is composed of a number of components. The enterprise vocabulary service (EVS) and the cancer data standards repository (caDSR) are the two which are of the greatest interest to our discussion as we will later discuss a component in our architecture which serves the same role as the aggregation of these two components. Finally, we closed with a discussion on the XML based interfaces with which caCORE based components communicate.

As will be discussed in chapter 5, we incorporate two primary elements of the caBIG architecture into our own. First, we incorporate a terminology server which we leverage to resolve terminological barriers between our various expected expert users. As our architecture is designed to satisfy users outside of the domain of clinical oncology and clinical cancer research, we use of a different terminology thesaurus than does caBIG. caBIG uses the National Cancer Institute (NCI) thesaurus and the NCI Metathesaurus. The terminology contained in these components is a subset of the terminology contained in the Unified Medical Language System Metathesaurus, the component which we use.

Chapter 3

Foundations: CRI Interoperability

3.1 Overview

The foundations chapter of this thesis has been written to satisfy the thesis's intended multidisciplinary audience; consequently, each section of this chapter discusses its topic in some depth. In this overview section we will provide guidance to the reader as to which sections will be of most value to them based on the background they possess.

We begin with a thorough discussion of the Health Level Seven (HL7) family of standards. Health Level Seven focuses on the exchange of information. Some of these standards also facilitate the exchange of semantic information. For readers who are familiar with the HL7 standards, we use of the HL7 version 3 Clinical Document Architecture (CDA) as the format for our message transmissions; we constrain the CDA using a Integrating the Healthcare Enterprise (IHE) profile which specifically addresses anatomic pathology structured reports (APSRs). For readers who are not familiar with the HL7 standards, we provide a discussion of the history of the HL7 standards organization.

We follow our discussion of the HL7 standards with an in depth discussion on the technologies which have been considered or have been incorporated into our architecture. For readers who are familiar with web technologies, we use semantic annotation in which take advantage of a translational research spanning terminology server from which we source our annotations. We access this terminology server through an XML interface which is built on a system with a hybrid service oriented architecture (SOA) / resource oriented architecture (ROA).

For readers who are unfamiliar with web technologies, this discussion begins with a brief introduction to the application of web technologies to the problem of information exchange by distributed collaboratives. From here we move to our in depth discussion of technology. We begin by introducing service and resource oriented architectures. We follow with an in depth discussion on the technologies which support these approaches. We then begin to discuss the cutting edge elements and philosophies of the modern day web. We focus here primarily on Tim Berners-Lees'[8] vision of the semantic web.

For readers familiar with modelling practices and the use of annotations in software engineering, many of the standards which we use are derived from knowledge or information models, as are the terminology servers which are employed in the domain. For readers who are unfamiliar with modelling practices and the use of annotations in software engineering, we move from here to a discussion on a variety of modelling techniques including the development of information models and knowledge models. We also discuss how these techniques relate to our earlier discussion on the semantic web. We conclude this section by establishing an evaluation criteria against which a terminology server could be validated. This criteria is later referred to in chapter 5 when we discuss our selection of the UMLS Metathesaurus as the terminology server which is used in our implementations of our architecture components.

3.2 Health Level Seven

Health Level Seven (HL7) is a non-profit standards development organization (SDO) who's focus prior to its HL7 v3 standard was primarily on developing transmission protocols for messages in the healthcare domain[69][7][37]. The new v3 standard includes the HL7 reference information model (RIM), the HL7 development methodology which is specified in the HL7 development framework (HL7 HDF), and the clinical document architecture (CDA), a schema outlining recommended contents of clinical documents.

3.2.1 Health Level Seven Version 2.x

The message standards in the HL7 2.x corpus are fine grained specifications for the transmission of healthcare information. Benson[7] describes the HL7 v2 standard, in short, as a tightly constrained delimited string based protocol for messaging between

healthcare applications.

As an example, HL7 v2 messages might be used by the computer system which is operated by a receptionist at a hospital. The receptionist would request information from a presenting patient and use it to fill a form within the system. Upon the submission of the form by the receptionist, the system would translate the populated form into an HL7 v2 message and would transmit the information to the system which supports the nursing staff. The nursing system would then decode the message. The message could be formatted and presented to a nurse, or the system could process the message automatically. The information contained in the message could be used to determine whether or not the presenting patient required a bed, and if so, what kind of bed.

3.2.2 Health Level Seven Version 3

The HL7 v3 standard covers more ground than does the HL7 v2 standard. Where the HL7 v2 standard is more of a framework for message content negotiation, the version three standard encompasses the whole process of interface development. One of the core elements of the version three standard is a development methodology called the HL7 development framework (HDF). This document provides a methodology which guides users through a specified process for developing software interfaces for software systems in the clinical and public health domain. The HDF prescribes that all software interfaces for health applications be based on an HL7 data model called the reference information model (RIM).

Another HL7 v3 standard, one which we use in the tools presented in this thesis, is called the clinical document architecture (CDA). Much like the HL7 v2 message specifications, the CDA specifies content and syntax for the transmission of clinical information. While the HL7 v2 message specifications are small and fine grained, the CDA bears more resemblance to an electronic medical record (EMR). The CDA has been derived using the HL7 v3 methodology. It uses the RIM as its base data model. It was developed in part as a generalization that was sufficient to capture the contents of many of the HL7 v2 messages.

The CDA is composed of two blocks, a header and a body. The header of a CDA compliant document is composed of a series of elements which are intended to provide the metadata required to express the context of the content in the remainder of the document. CDA compliant documents for example report a “recordTarget”, HL7

terminology which sometimes refers to a patient. They also report a "custodian", HL7 terminology for the entity which is responsible for maintaining the record in question.

The body of a CDA compliant document can be either structured or unstructured. This broad freedom which is provided in the CDA specification is provided based on the experience of the specifications authors. The history of the HL7 v2 standard, as will be described in the following subsection, implies that a large degree of freedom in the structure specified initially by the standard will lead to more pervasive adoption.

3.2.3 The Future of HL7

Corepoint stated that the patchwork data model employed by HL7 v2 is being stretched by new demands, a claim supported by Quinn[69]. Mead[59] explains how "[o]ne of the core issues with HL7 V2 is that although specific HL7 V2 messages may be semantically scalable, HL7 V2 in general is not." He begins by introducing computable semantic interoperability (CSI) which he describes as "[u]nambiguous data exchange". He states that "[t]he limitations of HL7 V2.x become most obvious when data exchange requirements cross inter-enterprise boundaries, thereby exposing conflicts or ambiguities in locally defined data semantics." Mead goes on to state that the four pillars of CSI are a "common information model that spans all domains of interest", "a computationally robust data type specification", "a sufficiently robust infrastructure for specifying and binding concept-based terminology values to specific message elements", and "a formal top down message development process". He outlines how HL7 V3 fulfills all of these shortcoming of the version 2 standard.

3.3 Communicating Collaborative Communities

3.3.1 Distributed Architectures

Kawamoto[49] describes how distributing software across multiple physical hardware units makes sense when the benefits gained from parallelization outweigh the cost of inter-machine communication. As has already been discussed, clinical research informatics is a highly collaborative field, and as with all collaborative fields, parallelization is pervasive throughout the clinical research informatics (CRI) domain. Consider for example that multiple specialists may want to access a cancer patient's

medical records simultaneously: the oncologist, the pathologist, the nurse, the primary physician and maybe even external scientific researchers.

Supplementary to this collaborative motivation for a distributed architecture is the computation complexity of the queries which will be requested of a CRI system. Queries in the CRI domain can be complex and sometimes data intensive. Consider for example the computational power which would be required to perform statistical analysis on the output of a next generation genome analyzer. Hey[39] claims that the genomic information generated about a single person in this context could easily be in the range of petabytes. Also consider the computational power required to perform a complex demographic query across the set of French biobanks which were studied by Hirtzlin[40]. Hirtzlin reported that collectively the French biobanks she contacted reported to be in possession of more than 18 thousand tissue samples and more almost 700 thousand blood samples. A set of cooperating services which can quickly respond to queries by employing parallelization will facilitate the rapid analysis of the data, information are captured, thus accelerating their evolution into knowledge.

3.3.2 Architectures for Web Solutions

In selecting an architecture for our solution we considered two possible styles: a service oriented architecture (SOA) and a resource oriented architecture (ROA). Our choice to evaluate these architectures was based primarily on the prevalence of SOA architectures on the web[62][55]. In the following sections we will provide an overview of each of the two investigated architectural approaches followed by a corresponding discussion of technologies which are commonly used to implement them.

3.3.3 Service Oriented Architecture

Erl[22] discusses how service oriented architecture (SOA) is a general term which describes projects which either employ web services technologies or use a web-centric variation of object-oriented design. Kawamoto[49] describes how it is generally agreed that systems which employ SOAs share the following core properties:

- the use of business-oriented services
- message-based interactions

- black-box service implementations
- communication over a network
- platform neutrality
- service description and discovery
- loose coupling between system components

Kawamoto goes on to discuss how a principle of the SOA paradigm is to decompose complex solutions into a collection of simple services and that this decomposition can lead to simpler software designs and implementations. He goes on to state that the composite nature of SOA solutions often leads to more frequent reuse of existing IT infrastructure and that the composite nature of SOA solutions also leads to greater business agility. Finally, Kawamoto states that SOA solutions are commonly expected to come at lower costs than monolithic enterprise solutions.

He[36] describes how the messages which are passed in SOAs are designed with three primary principles in mind. First, messages are descriptive and not instructive. They provide information but allow the message recipient to produce a service request solution in whichever way they choose. Second, messages are formatted. They are constrained to a particular schema and vocabulary which allows for standardized parsing mechanisms. Thirdly, the messages must be extensible. He goes on to say that if messages can not be extended, then services will be unable to provide backwards compatibility and that should the administrators decide to provide any additional information in the messages, then a redefinition of the communication schema and/or vocabulary will be required.

Simple Object Access Protocol

WS.* implementations of SOAs are common. Curbera[17] describes how these implementations are divided into three components: the simple object access protocol (SOAP) which enables communication among web services, the web services description language (WSDL) which describes those communications in a standardized format, and finally the universal description, discovery and integration directory (UDDI) which acts as a registry for web services. Curbera goes on to state that SOAP is an XML-based protocol for exchanging remote procedure calls (RPC).

3.3.4 Resource Oriented Architecture

The world wide web presents a network of interconnected uniform resource locator (URL) identifiable resources under a stateless and context free architecture based on representational state transfer (REST) principles. These resources are exposed as electronic representations of real world objects. Xu[87] describes how until recently, most resources could be found by addressing their uniform resource locators (URL) as in the case of web pages. He goes on to discuss how more recently, a broader range of resources are being exposed by systems which employ resource oriented architectures (ROA) under uniform resource identifiers (URI).

Representational State Transfer

The representational state transfer (REST) paradigm was first introduced by Fielding[23]. According to Fielding, the architecture can be described as having the following characteristics: a client-server architecture, statelessness, caching ability, interface uniformity and a layered design.

Fielding describes how the client-server architecture refers to an instance of separation of concerns, i.e. the user interface module is decoupled from the data storage module. He describes how the statelessness property requires that any message passed from the client to the server to contain all information necessary for its processing and that this provides scalability as the server no longer needs to store session data, but it also increases network traffic with repetitive information passing from the client to server in each transaction. He goes on to discuss how the caching property requires that the server explicitly label responses as cachable or not. The client is thus informed of its right to reuse the cachable responses. He states that the uniform interface characteristic enforces the principle of generality on component interfaces and that this leads to loose coupling and promotes independent evolvability, but comes at the cost of decreased performance due to the lack of interface tuning. He then discusses how the layered design allows the system to employ levels of indirection and that the incorporation of this well known software engineering principle allows system architects to encapsulate legacy components in wrappers which can be included in new systems.

A significant characteristic of RESTful architectures is an insistence on the use of uniform interfaces. This emphasizes the importance of using a limited set of operations for data manipulation. The operations in use in contemporary RESTful systems

are often restricted to the GET and POST operations defined in the HTTP 1.0 protocol, but are sometimes extended as far as the PUT and DELETE operations. This is in spite of the fact that the HTTP 1.1 protocol allows for the extension of this instruction set. This focus on a strongly constrained procedure set strongly contrasts against the simple object access protocol (SOAP) approach to web services where any procedure can be executed using SOAP as a tunnel for remote procedure calls (RPCs).

3.3.5 The Extensible Markup Language

The extensible markup language (XML) specification was authored by Bray[12] and others who comprised a world wide web consortium (W3C) working committee. This document describes the teams design goals could be summarized by stating that the team wished to build a standardized, adoptable, extensible, platform independent messaging syntax. A common issue with XML is that it isn't sufficiently restrictive for any specific application. It works well as a starting point, but further restriction of the syntax is required to achieve interoperability. XML schema documents (XSD) provided a first step in this direction.

3.3.6 XML Schema Documents

The XML schema definition language (XSD) normative specifications were written in two documents authored by Thompson[77] and Biron[9]. Thompson's efforts were later revised by Gao[26]. The XSD specifications describe a syntax which can be used to generate schema documents. The schema documents can then be used by computer applications to verify conformance of a given XML document. The approach to enforcement taken with XSD is to define the syntax to which compliant documents conform.

These initial efforts to provide a schema against which XML documents could be validated were later revealed to be insufficiently expressive. A particular example of this lack of expressivity is the fact that an XSD document can not be specialized in such a way as to restrict the content of a given element within a zero to many component of the sequence data type. This shortcoming arises from two statements made in the XSD specifications. The first of these statements is the element declarations consistent constraint (EDC). The EDC constraint, quoted below, has four components. The one of significance is the second:

> "If the *particles* property contains, either directly, indirectly (that is, within the *particles* property of a contained model group, recursively), or implicitly, two or more element declarations with the same expanded name, then all their type definitions must be the same top-level definition, that is, all of the following must be true:
>
> - All their declared *type definitions* have a non-absent *name*.
> - All their declared *type definition*s have the same *name*.
> - All their declared *type definition*s have the same *target namespace*.
> - All their *type tables* are either all absent or else all are present and have the same sequence of *alternatives* and the same *default type definition*[26]

The second of these statements describes how specialization occurs:

> Except for xs:anyType, every type definition is, by construction, either a restriction or an extension of some other type definition.[26]

To illustrate how these two requirements in concert preclude the specialization of a CDA while maintaining conformance to the CDA XSD consider the following example. A schema author wishes to implement a specialization of the HL7 CDA schema using a "restriction profile". The first step this schema author would need to take in authoring this specialization would be to include the CDA schema so as to include the base types for the new schema in the namespace. The author would then need to declare a new type with which to restrict the central type of the CDA, *ClinicalDocument*. The schema author would have the choice of attempting either a restriction or an extension. Now one of the elements of the *ClinicalDocument* complex data type is a *sequence* which contains another complex type called *component* who's cardinality is bound to be greater than one. Now imagine that the restriction profile requires that each document instance contain a *ClinicalDocument* data element. Further imagine that this instance of a *ClinicalDocument* is restricted to contain a *component* who's value set is restricted. The only way to restrict the value set is to use a restriction. The only way to generate a value restriction is to create a new type which restricts this component. This however, is disallowed by the EDC constraint which enforces the consistency of type on complex data types. As XSD is insufficiently expressive to accomplish this task, as series of new schema restriction tools were created including one called Schematron.

3.3.7 Schematron

Schematron is an xml schema definition and validation toolset which evolved through a series of steps from the early document type definition (DTD) schema specifications. In short, the first XML schema language was DTD, followed by XSD, relax NG came next, followed by Schematron[85]. Schematron takes a fundamentally different approach to schema validation from the definitive approach used by XSD. Schematron relies on transformation, rather than definition. This transformative strategy is what provides Schematron with the power to validate schema specializations which can not be managed by XSD.

3.3.8 Integrating the Healthcare Enterprise - Content Management

We have now discussed the XML message syntax and some of the validation techniques which are used in concert with this language. We have also discussed higher level architecture options for an interoperable informatics system including the service oriented architecture and resource oriented architecture approaches. We have also discussed how the Health Level Seven Clinical Document Architecture (CDA), a health specific example of an XML schema document, constrains, to a degree, the content which might be present in a transmitted compliant clinical document. The CDA however, only constrains the contents of compliant documents to the large set of data which might be contained in any electronic medical record. This general restriction may not do enough to facilitate the development of interoperable clinical systems. For this reason, organizations like Integrating the Healthcare Enterprise (IHE) which cooperates with HL7 to provide domain specific "profiles" which further restrict the CDA specification.

Abdel-Wahab[1] describes IHE's process in developing profiles for the radiology domain. These principles however are extended to the other domains which IHE targets. Abdel-Wahab describes the five core principles used by IHE. They can be summarized as identifying common interoperability issues encountered in clinical practice, "partner[ing] with vendors to develop solutions (also referred to as integration profiles) to the interoperability problems", "testing the broad application of these integration profiles across a variety of vender platforms at ... 'Connectathon' events, "facilitat[ing] demonstration of the application[s] by vendors of these integration profiles and how they facilitate seamless integration and transfer of patient data to the potential users

at the 'public demonstration' event and "publishing integration profiles to allow users to integrate these into requests for proposals and vendor contracts by institutions."

One pair of IHE collaboratives, the Anatomic Pathology Planning Committee and the Anatomic Pathology Technical Committee, have decided to tackle the transmission of anatomic pathology structured reports (APSR). Kussaibi[51] and other participants in this process engaged in the IHE process described above, supplemented by a study in which the Delphi method[54] was employed, to develop an integration profile for APSRs. Through this process they evaluated a variety of options, and in the end concluded that the base format for exchanging APSRs should be the HL7 CDA.

The specification of the content required for an APSR was released in a formal document called the APSR Integration Profile[42]. This document lays out in formal text how to constrain the CDA so that a complete APSR results. As the validation component of the document is not provided, the document, though formal, might still be misinterpreted. A Schematron validation is required to ensure the conformance of a given APSR document. The specific details of the IHE integration profile will be discussed in further depth in chapter 5.

3.4 Knowledge Exchange and the Semantic Web

Sir Tim Berners-Lee[8] conceived the semantic web. A central element to this vision is the codification and definition of concepts. GenoLogics, our industry sponsor has already begun to venture down this path with their industrial translational research solution, and so were very interested in pursuing this idea. In Berners-Lee's vision:

> "[t]he Semantic Web will bring structure to the meaningful content of Web pages, creating an environment where software agents roaming from page to page can readily carry out sophisticated tasks for users....[F]or the semantic web to function, computers must have access to structured collections of information and sets of inference rules that they can use to conduct automated reasoning. ...Two important technologies for developing the Semantic Web are already in place: eXtensible Markup Language (XML), and the Resource Description Framework (RDF)... Scripts, or programs, can make use of [XML] tags in sophisticated ways, but the script writer has to know what the page writer uses each tag for. In short,

> XML allows users to add arbitrary structure to their documents but says nothing about what the structures mean."

Berners-Lee advocates for the use of the Resource Description Framework (RDF) a technology which relies on concept identification. In order to identify a concept in a way which is semantically interpretable and universally accessible, Berners-Lee claims that at least two things are necessary. Firstly, there must be a level of abstraction between the terms that are used and their definitions. Secondly, the definitions must be publicly available.

RDF satisfies the first of these requirements by employing the same strategy used by the world wide web to identify webpages, the uniform resource locater (URL). In RDF, concepts are composed of triples which form a subject, action, target relationship. Each unit of each triple is identified by a universal resource identifier (URI). URIs are a generalization of the commonly known URL. These URIs are used to provide a level of indirection between the element itself and its definition. The second requirement is managed by the expectation that the publisher of the URI will maintain access to that resource. In other words, the publisher will ensure that the definition of their universal resource will be available for use by anyone who wishes to access it.

Of course, when multiple sources of information exist, redundant terminology will develop. Two different information sources may develop independent and different URIs to represent the same concept. Berners-Lee states that a "program that wants to compare or combine information across ... [multiple] databases has to know that [sometimes different] terms are ... used to mean the same thing. Ideally, the program must have a way to discover such common meanings for whatever databases it encounters. A solution to this problem is provided by the third basic component of the Semantic Web, collections of information called ontologies."

The term "ontology" has broad and diverse usage, and so for now, we will not define the term but rather will use Berners-Lee's implied definition that can be interpreted from his statement that "[t]he most typical kind of ontology for the Web has a taxonomy and a set of inference rules." "The taxonomy defines classes of objects and relations among them." "Inference rules in ontologies supply further power."

In other words, taxonomies are typically composed of simple hierarchical relationships while the inference rules in an ontology can be more complex. When we consider however that each information provider may expect that we use their ontol-

ogy, we can see how the introduction of ontologies in the plural doesn't resolve the problem of knowledge exchange in a collaborative environment. Weng[84] discusses the difficulties of aligning and integrating ontologies, and so for this reason we chose to avoid the dynamic resolution of controlled terminology. Instead, we use a common terminology server which natively contains ontologies which service much of the translational research domain. This terminology server however, presents this collection of ontologies as a single unified ontology through a singular mapping of all of its member ontologies. We do not address the complex problem of allowing dynamic use of collections of terminology servers.

3.4.1 Knowledge Models and Information Models

We now provide an introductory discussion on the types of models present in the domain of translational research to help the reader understand the roles each of these models play within the informatics domain. Schadow[73] states that "[a]n information model defines what we record and communicate about the world, whereas [a] reference ontology ... model[s] the world itself." He uses the HL7 reference information model (RIM) is an information model as an example of an information model. Schadow further clarifies the difference between information models and knowledge models when he describes why "the RIM does not seem to define the biomedical reality". He discusses how modelling the amount of a given drug in a given patient's blood system will not produce a useful reference for the exchange of clinical information about such a patient. Instead the RIM models the process by which this state is uncovered.

To summarize in a paragraph, the difference between a knowledge model and an information model can be understood by considering the purpose of the model. Knowledge models are intended to be representations of the real world. Information models are intended to be representations of the information which is collected through observation of the real world and then transmitted to interested parties.

3.4.2 Thesauri and Terminology Servers

In our section describing the semantic web we discussed how we use of central terminology server which provides a singular mapping of many ontologies as the shared ontology to be used by all systems in a research collaborative. This central terminology server is much like a thesaurus. van Rees[82] states that "[i]n principle, a thesaurus deals only with words, alternatives for those words, synonyms, translations,

[etc]...[t]his textual kind of information can be used by (or added to) ... an ontology. For instance, a pure thesaurus ... could be enhanced to an ontology, providing both the already available rich text information and formal definitions and properties." We concur with van Rees on this point and incorporate the Unified Medical Language System Metathesaurus (UMLS), a knowledge source which has done precisely as van Rees describes, into our architecture.

Barrett[5] states that "[i]nteroperability of health information systems requires common clinical terminologies and services that make those terminologies available on a shared infrastructure, i.e. Web-based terminology servers." This is an argument we have also been making. The critical element of Barret's study from our current perspective is his evaluation of the services required of a web terminology server. This element is of importance first in motivating our choice to use a terminology server, and second in motivating our selection of the UMLS Metathesaurus and Semantic Network.

One of Barrett's first claims, and one with which we concur, is that "[m]aking shared terminology services available requires an agreed upon set of services". In other words, partners in the collaborative and distributed collective need to agree on what it is exactly that their shared terminology server should do. Barrett goes on to propose a consensus set of services which he uncovered through a systematic literature review.

Barrett divides the collection of uncovered service requirements into two categories: management services and access services. He describes management services as the services which are required to curate the terminology, and describes the access services as those which are required to employ the terminology. Under the category of management services, Barrett lists the following service requirements:

- addition, and deletion of terms
- development, deprecation and replacement of terms
- verification of internal consistency
- security and privacy of sensitive information
- extensibility of the server architecture (software components)
- usage and resource monitoring, and resource allocation, distribution and scheduling

- terminology inventory

Under the category of access services Barrett lists the following service requirements:

- querying
- knowledge generation or inference
- natural language bridging
- synonym/lexical matching
- term completion
- semantic locality

With a framework now in place under which we might evaluate a terminology server we will be able to discuss our selection of the UMLS Metathesaurus in a critical manner in chapter 5. First however, let us discuss GenoLogics' translational research informatics solution so that we might provide the reader with a view of the context into which our architecture is intended to fit.

Chapter 4

An Industrial TRI Solution

4.1 GenoLogics Life Sciences

GenoLogics engaged us in this research collaboration partly in the hopes of gaining a greater understanding of the technical barriers to knowledge exchange in the translational research domain. We welcomed this invitation as we knew that there was much we could learn from GenoLogics as well. One of the primary benefits we gained from this collaboration was a first hand view of the suite of tools which GenoLogics sells under the umbrella of their translational research solution. We provide an overview of this solution in this chapter as a mechanism by which to explain the context into which our architecture is intended to fit.

One of our interests in this collaboration was to clearly identify some of the difficulties with knowledge exchange in translational research. We began our investigation by identifying how some of these exchanges occur by developing a use case using a model driven architecture methodology (MDA). As part of this process we relied on the personae that the product management team at GenoLogics has developed in order to better understand their clients. We will identify, over the course of the description of this use case, a number of areas of strength in the GenoLogics translational research solution. We will also identify two areas in which the solution might benefit from additional investment and which target with our primary contribution.

4.2 The Study Use Case - Preamble

Shortly before we engaged GenoLogics in this research project, GenoLogics had begun developing a solution to a use case which they had elicited from their clients. We have call this use case as the "Study Use Case", and will lay out the steps of this process in a forthcoming section. The Study Use Case describes how therapeutic cancer treatment processes are developed at many of GenoLogics Clients' sites. First however, we will introduce the players and artifacts involved in this use case. Below, in fig 4.1, we present an overview of the primary actors, artifacts and software components that are involved in the Study Use Case in a loosely UML based diagram representing this process. We follow this with a summary of the personae GenoLogics uses as part of their MDA development methodology to understand the actors in the Study Use Case.

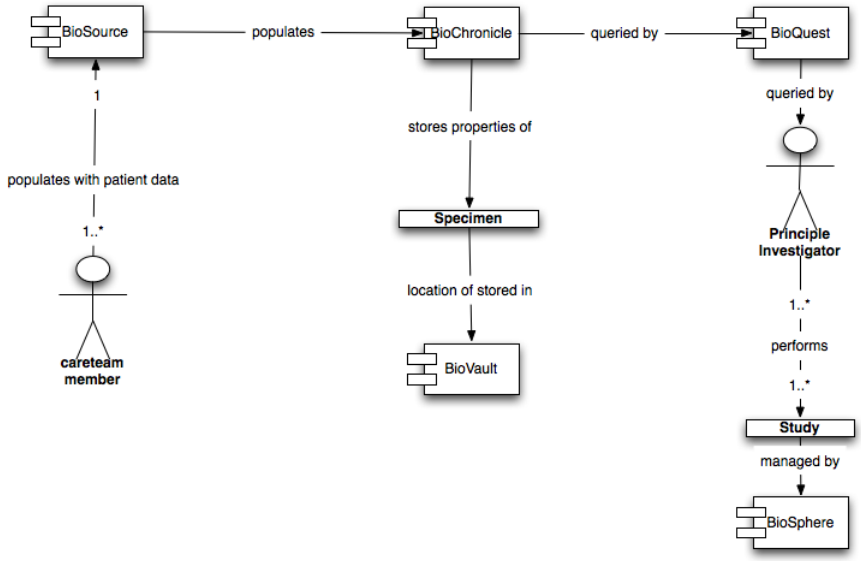


Figure 4.1: *An Overview of the GenoLogics TRI Solution*

4.2.1 The Primary Investigator

GenoLogics' models two variants on this actor. They perceive a significant divide between primary investigators who are internal to an organization, and those that are external. Much of this difference has to do with authority.

The Internal Primary Investigator

GenoLogics' persona for the internal primary investigator is named Dr. Jacques LaFleur. His primary responsibility is to manage the research projects which are underway at his institution. The primary artifacts for which Dr LaFleur is responsible are publications about the studies his institution performs. Dr LaFleur has a number of responsibilities including the decisions about which therapeutic compounds his institution will pursue and which they will abandon. Dr LaFleur is responsible for:

- Assembling and managing study participants
- Guiding the progress of research on therapeutic compounds
- Assembling publications which report on experimental results which characterize therapeutic compounds
- Assembling publications which report on patient responses to therapeutic compounds

The External Primary Investigator

GenoLogics' persona for the external primary investigator is named Dr. Immanuel Spock. He is an active university professor and as such has all the responsibilities associated with teaching and grading. His day to day research life mostly revolves around managing the work of his graduate and post graduate students. Though he lives to publish, this is mostly done through his students as he rarely engages directly in the lab work himself. Though Dr. Spock's job description is similar to that of Dr. LaFleur's in many ways, one primary difference is his access to specimens(tissue samples). While Dr. LaFleur has an insider track through which he can select specimens from an internal biobank, or can even request that new specimens be extracted from selected patients, Dr Spock is at the mercy of external biobanks from whom he must request specimens for his research.

Now that we have established the goals and capabilities of the two primary investigator personae, we can relate them to fig. 4.1. The primary investigator is represented on the right hand side of the diagram. He uses a tool which at its inception was called BioSphere to manage his studies. That is to say, he uses the tool to keep track of his study participants and the progress being made with his study. The tool, in essence, works as a feature rich customer relations management system (CRMS).

The primary investigator also uses the BioQuest component of the GenoLogics' TRI solution. GenoLogics' describes bioQuest in the following way

> BioQuest ... is a Web query portal that connects to BioChronicle, [GenoLogics'] integrated and centralized clinical annotations system. BioQuest provides a secure, external interface that allows researchers to access available biospecimens across their institution. Large research enterprises often house multitudes of valuable biospecimens within many biorepositories across their organizations, most of which are inaccessible to principal investigators...[A] secure Web query portal [can] enable investigators to search and query clinical annotations; request access, gain approval, and request and receive valuable biospecimens and de-identified clinical annotations, from disparate biorepositories. [This tool allows] federated access to all biospecimen repositories via a virtual biobank across an enterprise or multiple institutions."[27]

The BioQuest interface is the first component which might benefit from additional investment. Though a simple web-based front end like the one currently in place provides an intuitive interface to the biomedical knowledge source which the primary investigators wish to access, it is not machine processable. Supplementing BioQuest with an interface like the standards supported extract transform and load (ETL) engine component which we use in our architecture would provide a more computable interface to the biomedical knowledge source on which BioQuest sits.

GenoLogics describes BioChronicle as follows:

> BioChronicle ... is a cloud-based data management software system that aggregates clinical annotations from multiple biomedical informatics data sources for translational research, including hospital information systems (HIS), electronic medical records (EMR), clinical trials management systems (CTMS), biobank systems, electronic patient questionnaires, and flat file data inputs...The BioChronicle data management system can also automate data collection by using standardized import tools to capture information from other systems, such as images and other unstructured data types, or biomedical informatics research. The data model maps patients, studies, familial pedigree, time points, lifestyle, disease, treatment and outcomes. The BioChronicle data management system

> also handles identified and de-identified patient data with appropriate security mechanisms to segregate data, and only provides de-identified data when required." [28]

The data stored in BioChronicle which is of importance here is the data which is collected from biobank sytems. Of this data, the portion which is of significance to our use case describes the characteristics of the specimens which are owned by the controlling organization. These specimens, which are physical samples, are stored in a biobank. This biobank is basically just a large set of freezers which hold all the specimens, usually in micro-fuge tubes or on small glass slides. The state of the biobank is managed by the BioVault component of the solution.

In our architecture we propose the use of a persistence component which would store annotated data and metadata. We perceived the portion of this component which stored the data to operate much as BioChronicle does. The persistence layer in our architecture also includes a portion which is strictly devoted to storing metadata. This metadata portion was implemented and is described in chapter 5. The metadata portion of our architecture's persistence layer is designed to record the meaning of the labels which were used to record the information which is stored in the data portion of the persistence layer. This knowledge is what we expect will facilitate the subsequent integration of systems which are based on our architecture.

GenoLogics describes BioVault in the following way:

> "BioVault ... is a biospecimen data management software solution that offers flexible storage management; handles biospecimen and freezer management with volume and aliquot tracking; and access to tracking and configuration options.
>
> Access to biospecimens and well annotated clinical information is crucial to translational research. Unfortunately, biorepositories are often underserved with supporting biomedical informatics, and face many challenges in integrating clinical and phenotypical information, managing distributed collection sites, tracking biospecimen requests, and meeting regulatory requirements.
>
> For organizations with multiple collection sites, any biomedical informatics solution must be able to collect data across sites, track biospecimens, and administer multiple studies. A centralized biobank can then

> aggregate the biospecimen and clinical data from multiple sites, and facilitate Web-based queries, sample requests, and IRB approvals. [BioVault addresses the informatics components of these problems]"[29]

As was described above in the paragraphs which summarized the roles and responsibilities of the primary investigators, the primary investigator, if he is internal as is Dr LaFleur, could be a member of the care team for a given oncology patient. From the perspective of our UML diagram however, the primary investigator does not interact with the software system on this end. If we are considering the system from the data entry side, we instead need to focus instead on the Nurse, the Biobank Technician and the Lab Technician.

4.2.2 The Biobank Technician

GenoLogics' biobank technician persona is named Gary Hillenburg. Mr. Hillenburg's primary responsibilities are to receive samples, to derive aliquots from those samples and to enter the data regarding those samples into his institution's laboratory information management system (LIMS). If Mr. Hillenburg were to work in a hospital, he would act as the interface between the hospital's LIMS system and the care team.

4.2.3 The Lab Technician

GenoLogics' lab technician persona is named Abe Park. Mr. Park's primary goal is to accomplish the lab work he has to perform in his high throughput environment. Mr Park wants the laboratory information management system (LIMS) he used to "remove much of the burden of Good Laboratory Practices (GLP) and what it means to be compliant with FDA regulations." Day to day, Mr Park needs to "check on mass spectrometers, chromatography instruments and gel work that was setup to run overnight and ensure [that] the instruments have not malfunctioned." He needs to "check the batch database search to see whether all the proteins are being identified; if not, [he needs to perform] manual database searches on individual spectra." He also needs to "set up samples on various chromatography instruments, gels, and mass spectrometers." The end goal of Mr Park's work is to "gather results when a project is complete - cut and paste things into a Word document to prepare a final report [of the laboratory results] for [his] customers".

In Mr Park's day to day work, he is frequently entering data into the LIMS system performing many of the same duties as does Mr. Hillenburg. Mr Park, like Mr Hillenburg, is not a direct member of the care team in the left of the UML diagram 4.1 but rather acts as an interface between the care team and the LIMS. Unlike Mr. Hillenburg, Mr. Park also has very direct interactions with specimens. He performs biomedical assays of all types in his day to day work where as Mr. Hillenburg's work is much more focused on maintaining and tracking the specimens which arrive at his biobank.

4.2.4 The Nurse

In GenoLogics' model of the domain, another actor who is likely to enter data into the LIMS is the Nurse. GenoLogics' Nurse persona is named Nancy Baker. Ms. Baker's singular interest is the health of her patients. Her day to day activities revolve around maintaining her patients' health. One of the avenues through which she accomplishes this is by acquiring doctor ordered samples from those patients. These samples include blood samples, urine samples and swabs. When she collects these samples she will also often complete a questionnaire with her patient. The data from these questionnaires forms some of the data which is shuttled into the BioChronicle component of GenoLogics' TRI solution.

The interface used by each of these data entry technicians, the biobank technician, the laboratory technician and the nurse, was named BioSource or Collect at its inception. As GenoLogics understanding of industry demands has evolved, this project has changed in scope and scale. It has since transformed into a small and simple data entry mechanism who's primary use case is as follows:

1. Samples arrive at repository
2. Samples are registered, barcodes and worksheets printed
3. Samples are processed into derivative aliquots
4. Additional information is entered related to subject or samples

Having now provided anecdotal descriptions of the actors which GenoLogics' system is intended to service and the artifacts which the it is intended to manage, we can provide a concise description of the general use case for the system.

4.3 The Study Use Case

1. An internal primary investigator wishes to perform a study on a set of patients to verify a medical hypothesis. This primary investigator writes a document describing a study which would verify this hypothesis. To begin this study this primary investigator determines an information set which would be sufficient to disprove the hypothesis.

2. The primary investigator or one of his agents generates an information collection form to record this information. This form is very similar to the anatomical pathology structured reports (APSRs) described by Kussaibi[51].

3. The agents of the primary investigator, possibly including nurses, laboratory technicians or biobank technicians, use BioSource to create an interactive data collection service with which to capture the information necessary to complete the form.

4. The primary investigator or one of his agents assembles a collection of patients for the study and manages them with BioSphere. The primary investigator may use external means to find these patients or he may rely on queries he executes against a BioChronicle instance through a BioQuest interface.

5. The primary investigator or one of his agents, a nurse for example, then meets with each of the patients and uses the forms developed with BioSource to collect the required information.

6. BioSource then stores the collected information in a BioChronicle instance.

7. When the patients participate in the study they may also undergo biopsies in which they provide specimens. These specimens are stored in refrigeration units at a biobank who's samples are managed with a BioVault instance

8. These specimens, once they have been extracted, are characterized by pathologists and/or lab technicians. This data is input through the BioSource interface which pushes this information into a BioChronicle data store.

9. At some later date an external primary investigator may wish to perform a second study.

10. This external primary investigator wishes to assemble a collection of tissue samples which originated from a set of patients matching a certain criteria which he derives from the second study.

11. The external primary investigator or one of his agents uses BioQuest to search for samples matching the required criteria in BioChronicle.

12. BioChronicle reports on the samples managed by BioVault which were taken from the individuals matching the specified criteria.

13. The external primary investigator then uses the returned results to order the samples which were taken from the patients which matched his study criteria. These tissue specimens are ordered from the biobanks which hold them.

4.4 Conclusion

Based on the perceived strengths and weaknesses of the GenoLogics translational research suite, we decided to target three specific aspects of the cancer informatics interoperability problem in the context of this existing tool set. Firstly, the BioSource component could benefit from a more formal annotation mechanism. Secondly, if the data schema which is used in the bioChronicle component were formally annotated these annotations could facilitate the later automated integration with other disparate information systems. Thirdly, though the web interface to the bioQuest tool is a good interface for a human user, bioChronicle could benefit from an additional interface which provides a machine processable view of its contents. We describe in chapter 5 components of our architecture which begin to address these weaknesses. Though some components of our system are designed to integrate easily with the GenoLogics solution, the components themselves not designed explicitly to meet the needs of GenoLogics. Our architecture focuses much more on data capture and transmission techniques which could be leveraged for general interoperability than does the GenoLogics architecture.

Chapter 5

Component Implementations

5.1 Overview

After having synthesized a breadth of knowledge about the clinical research informatics domain and having learned of some of the shortcomings of preexisting approaches to the problem, we wished to explore an approach to interoperability in this domain which was founded on semantic annotation and the use of health informatics standards. In particular we use the HL7 version 3 CDA as it is prescribed by Canada Health Infoway[80] and by the NCI[44].

We elected to explore a potential CRI pipeline in breadth, delving into depth only where significant problems were foreseen. In this CRI pipeline, we use the UMLS Metathesaurus as the terminology server. This terminology server is a relatively open language source which encompasses many of the controlled terminologies which are of importance to our target domain. We also use the HL7 CDA in concert with the IHE APSR profile. Both of these standards again are open and accessible. They also have an engaged community of participants.

The pipeline is diagrammed in fig. 5.1. It is intended to service informatics experts who were entering pathology data into an information repository for future extraction by primary investigators. We focus in particular on the semantic annotation process. This use case is an expansion of a portion of GenoLogics' Study Use Case which is described in chapter 4. Specifically, it expands on the process by which members of the care team enter data into a component which is analogous to the BioSource component of GenoLogics' TRI solution, and a way in which that data might be exported from a component which is analogous to the BioChronicle component of

that solution.

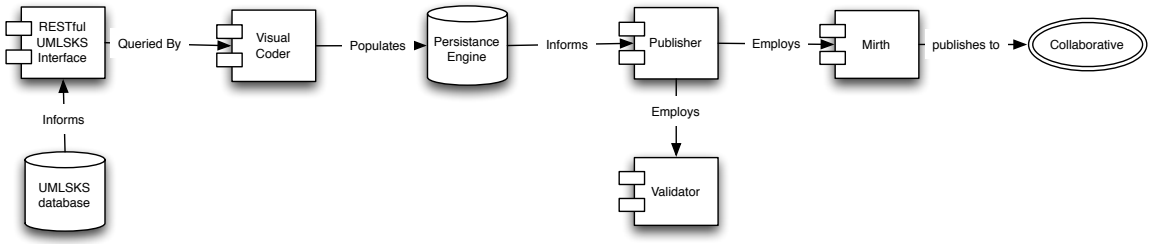


Figure 5.1: An overview of the proposed CRI interoperability pipeline. The left hand side of the diagram displays the information capture end of the architecture. The terminology server interface in concert with the Visual Coder component are used to annotate incoming reports with alpha numeric codes in the semantic annotation process. These codes are then recorded in the persistence layer. A publishing component then extracts the data sorted in the persistence layer and coordinates this information into standardized structured documents which it then validates using the Validator component. Finally, the publishing component employs the Mirth ETL to publish query results to the collaborative research community.

Having presented this overview of the pipeline, we will now provide a discussion on the development process of selected components: the UMLS RESTful terminology server, the visual coder, the annotation persistence engine, the validation mechanism and the Mirth export transform and load engine. First however, we provide a brief summary of the components in our conceptual architecture and associate them with their authors in table 5.1.

Component	Author(s)
UMLS Terminologies	Various SDOs
UMLS Relational Schema	UMLS staff[65]
Metamorphosys	UMLS staff[75]
Local UMLS Instance	generated by Fieran Mason-Blakley using Metamorphosys
RESTful interface to the UMLS	authored by Fieran Mason-Blakley using Libraries by Bill Burke[14]
Visualization of the UMLS	authored by Fieran Mason-Blakley using Prefuse Libraries[38] and employing visualization theories presented by Ware[83] and Tufte[83]
Persistance Engine	authored by Fieran Mason-Blakley employing algorithms of van Tulder[79] and noun phrase concepts of Friedman[25]
Mirth	Mirth Development Community[16]
HL7 CDA	HL7 Structured Documents Working Group[30]
IHE APSR Profile	Kussaibi[51] and the IHE Anatomic Pathology and Technical Committees

Table 5.1: A summary of the authorship of the components of the conceptual architecture

5.2 The Unified Medical Language System

We have discussed already how the inclusion of a semantic annotation service which uses ontologies can provide a common language which collaborating researchers can use to communicate in a distributed environment. If the mechanism of semantic annotation referred to by Berners-Lee[8] is facilitated by an ontology which is commonly used by all participants in the collaborative, the problem is relatively easy to solve. If this is not the case, two alternatives might be considered. Firstly, a secondary service which provides mapping between disparate ontologies could be employed to resolve terminological conflicts. Secondly, terminological differences might be resolved through the dynamic (on the fly) analysis of disparate ontologies used by the col-

laborative members. The approach we have taken in our proposed solution matches the first alternative.

In our proposed solution to interoperability in the anatomic pathology structured report subdomain of the field of clinical research informatics (CRI), we use an aggregation of the unified medical language system (UMLS) Metathesaurus and the associated semantic network as a common terminology server. We first discuss the suitability of this component for our purpose in terms of its technical merits by displaying how it provides the management and access services which were identified by Barrett and which we outlined in section 3.4.2. We then verify that the completeness and quality of the content of this component.

5.2.1 Evaluation of Management Services

Barrett[5] identified seven categories of service requirements which he categorized as management services. The first two of these categories revolve around the curation of the data stored in the terminology server. Explaining how the UMLS Metathesaurus provides these services requires us to first take a step back and consider the impact of curation on semantic interoperability in a collaborative distributed collective. Recall our discussion on Berners-Lee's[8] vision of a semantic web. In this vision Berners-Lee discussed the fact that if independent websites were allowed to create their own ontologies, that multiple different websites would likely model identical concepts. Furthermore, as these models were identified independently, not only would modelling differences likely exist, but the alpha numeric codes used to represent them would also be different. Without further intervention, this would lead to the necessity to perform dynamic mapping to achieve automated semantic interoperability. As we discussed in the previous section, we have taken a path which avoids this issue by choosing a centralized terminology server which contains mappings between a set of ontologies which service much of the translational research domain.

We propose a two tiered solution to this problem. First, local implementations of our proposed solution would use a two component terminology server complex. The first component of the complex would be the centralized UMLS Metathesaurus. The second component of the complex would be a homegrown supplemental terminology server. The UMLS Metathesaurus would be used for most communications, and would be the only system used when collaborating with "stranger" applications. To clarify, if no close relationship existed between two systems, they would use only the

accepted central terminology and would not include annotations which referred to terminology which was present in their supplementary terminology server. The local supplement would be used only for point to point communications between systems which were managed by closely collaborating researchers.

This two tiered approach to this problem is desirable because terminology evolves over the course of time, and because its adoption will usually occur at different rates at different institutions. This gap between the state of the art in terminology and local adoption displays the desirability of the use of intermediary, in other words local, solutions.

In this environment it is expected that the majority of the terminology used at each node of the distributed collaborative would exist in the centralized UMLS Metathesaurus. This assumption is based in part on the breadth of controlled terminologies included in the UMLS Metathesaurus. We also expect this to be true based on the intended purpose of this terminology server and the mandate of its authors. These points will be discussed further in our evaluation of the content of the UMLS. Local terms which did not exist within the UMLS Metathesaurus could be curated at local nodes of the grid on their local supplemental terminology server.

Even though some fields in TRI/CRI are well developed, cancer informatics being primary among them, these fields are rapidly evolving. If we recall the purpose of the TRI and CRI fields, to accelerate the propagation of knowledge between researchers in the lower level sciences and those in the clinical space, we can see that the field itself is framed in such a way as to encourage the evolution of terminology. That is to say, it should be expected that even the most mature disciplines in the field will develop new terminology which will need to be incorporated in to the collective language. This new terminology at first will be locally developed by nature. There will of course be an originator of the terminology. Over time however, as more point to point connections are made between the nodes of the research collective, the use of this terminology may become communal. At this point, the terminology can be standardized through a consensus process and can be incorporated into a formal ontology. It can then be mapped against the centralized terminology server using the same consensus process which has been used by the UMLS Metathesaurus authors to create the existing mappings.

We use this discussion of integrating new ontologies into the UMLS Metathesaurus to open a discussion of Barrett's second management service requirement, verification. Barrett states that "[t]erminology servers should incorporate automated mechanisms

for terminology verification, or at least facilitate manual verification.", and that "[t]his ensures the terminologies' consistency and soundness which are crucial for systems reliant on the terminology server". Though a discussion of all of the mechanisms by which such verification could be performed is beyond the scope of this thesis, a brief discussion on the mechanism by which the National Library of Medicine(NLM), the curators of the UMLS Metathesaurus, address this issue is not.

The process of verification performed by the NLM occurs more through the engagement of a community of experts than it is by an automated process. The NLM lays out their basic assumptions about the mapping process on the UMLS webpage[66]. The relevant basic assumptions outline a list of participants deemed to be required to produce an effective mapping. The following are a subset of the statements made by the NLM revolving around the validation of their terminological mappings:

> The producers of both vocabularies in any map must participate in the mapping effort to ensure that the result accurately reflects the meaning and usage of their vocabulary. At a minimum, both vocabulary producers must participate in defining the basic purpose and parameters of the mapping task, reviewing and verifying the map, developing the plan for testing and validation, and devising a cost effective strategy for building, maintaining, and enhancing the map over time[66].

> Target users of the map must participate in its design and testing to ensure that it is fit for its intended purpose[66].

> Production of mappings will be an iterative process, which must involve testing, validation, and use in real world settings. The functionality of mappings will improve over time as research, testing, and use determine (a) useful ways to construct and represent complex mappings (such as those that involve conditional rules) and (b) the extent they can be applied in the real world[66].

> When conflicting relationships between two concepts appear in different source vocabularies, both views are included in the Metathesaurus. Although specific concept names or relationships from some source vocabularies may be idiosyncratic and lack face validity, they are still included in the Metathesaurus[65].

The third element of the management services category outlined by Barret discusses security and privacy. The UMLS does not need to consider services of this nature because it is a strict terminology server. It contains no personal or otherwise sensitive information.

Next, Barret identifies architectural extensibility as a feature which is desirable for any clinical terminology server. We fully concur with Barret's assessment, and consequently engaged on a journey to create our own instance of the UMLS Metathesaurus and Semantic Network. When we first discovered the UMLS Metathesaurus and Semantic Network, we read about a variety of ways users could access the implementation of the Metathesaurus and Semantic Network which is maintained in Bethesda, Maryland by the NLM. These details were retrieved primarily from the UMLS Developers Guides which were provided after we signed a license agreement. Having now discussed some of the management services of the UMLS Metathesaurus, we will now turn to our evaluation of the UMLS Metathesaurus content.

5.2.2 Content Evaluation

In chapter 1 we discussed how the domain of clinical research informatics, or more broadly the field of translational research informatics is populated by experts from a variety of fields and disciplines. Each of these experts have their own terminological preferences, and many groups of these experts have already agreed to common standardized sets of terminology. A few sources of note in the realm of clinical research informatics (CRI) are the Systematic Nomenclature of Medicine Clinical Terms (SNOMED CT)[7], Logical Observation Identifiers Names and Codes (LOINC)[58] and the International Classification of Disease 10^{th} ed (ICD 10)[48]. These ontologies, taxonomies, and classifications are commonly used in both the Canadian Health System.

van Rees[82] discusses the differences between each of these types of controlled terminologies, and we in general concur with his conclusions as to their meanings. He claims via Gruber[31] that an ontology can be concisely defined as an explicit specification of a conceptualization. These models include entities and a rich set of relationships. van Rees describes taxonomies as very simple ontologies which include only strict hierarchies. These might include "is a" or part of relationships. Finally, classifications are simplest forms of taxonomies.

Kussaibi[51] identifies a series of controlled terminologies which are specific to the

field of anatomic pathology structured reports (APSRs). We use the term controlled terminology here to be the superset of all ontologies, taxonomies and classifications. Kussaibi uncovered through his study that the controlled terminologies of importance for APSRs in the United States are SNOMED-CT, ICD-O-3.

These controlled terminologies on their own however are inadequate to satisfy the TRI domain as a whole. As we previously discussed, disparate terminologies on their own are inadequate without mappings between them. Not only is there a need for mapping, but there is also a need for systems which can employ such a mapping to satisfy the semantic interoperability needs of an interdisciplinary audience. If such a system is to be adopted, a good first step might be to design it in such a way that it does not tax the knowledge or skills of its intended audience. The responsibility of performing semantic tagging on experimental results in clinical research informatics would belong to a person with a different vocabulary in a clinical setting than it would in a laboratory setting; consequently, a tool which is meant to facilitate interdisciplinary knowledge exchange must support the vocabulary and skill sets of many different expected user.

The Unified Medical Language System's Metathesaurus is the largest biomedical knowledge source of its kind[33]. It provides mappings between more than 100 biomedical ontologies, taxonomies and controlled terminologies including SNOMED CT[7], LOINC[58], and ICD-9/10[48]. Sadly, it does not currently include ICD-O-3, but the infrastructure to include these vocabularies is already in place as was described in section 5.2.1. Consequently, when the need for the inclusion of these terminologies presents itself, a process to do so will already be in place.

Though this argument addresses the terminological needs of the participants in the TRI domain, it does not address their skill sets. We will discuss this later point when we introduce the visualization element of the semantic annotation component of our architecture. Before we begin that discussion however, we will present an evaluation of the technological aspects of the UMLS Metathesaurus.

5.3 Technological Summary and Evaluation of the UMLS Metathesaurus

In this section we will begin by providing an overview and evaluation of the current national library of medicine (NLM) provided implementation of the UMLS

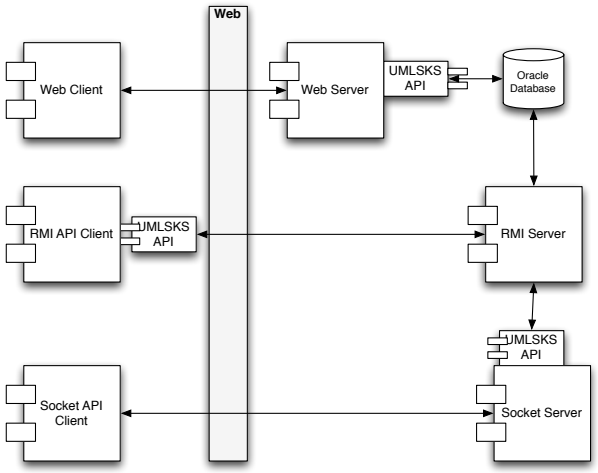
Metathesaurus and the interfaces the NLM provides to this implementation. We will first discuss the various interfaces to the UMLS Metathesaurus and the UMLSKS in general. We will then discuss the shortcoming of the current implementations of these interfaces and will use this discussion to motivate the forthcoming section on our implementation of a resource oriented architecture (ROA) based interface to the UMLS Metathesaurus. We begin this discussion with a description of the publicly available interfaces to this terminology server.

5.3.1 The NLM Interfaces to the UMLS Metathesaurus

The UMLS Knowledge Sources which include the UMLS Metathesaurus, the Semantic Network and a component called the Lexicon are implemented as a single relational database. Each of these three components are segregated to their own set of relational tables. The public version of the implementation of the UMLSKS was initially made available in March 2002. This release included new features including access to the UMLS Knowledge Sources through a public Web interface, incorporation of XML support for programmers both in requesting and returning data, inclusion of a Java-based Object Model of the UMLS Metathesaurus data, and incorporation of a TCP socket-based interface for non-Java programs.[66]

As a consequence of usage, the project has continued to evolve. We will provide in this paragraph an overview of the logical architecture of the NLMs publicly available interface to the UMLSKS. In the version 5.0 documentation, the features described in the previous paragraph were displayed in a diagram which provided an overview of the logical architecture of the system. That diagram has been reproduced in fig. 5.2.

Figure 5.2: An overview of the logical architecture of the version 5 UMLSKS release



The NLM implementation of a UMLSKS interface as a whole runs on an remote method invocation (RMI) paradigm with an Oracle database back end. Queries from the web client hosted on the UMLSKS intranet portal, a socket application programmer interface (API) client and an RMI API are all processed by an RMI server which acts as a gate keeper to the Oracle database which stores the UMLSKS data.

The Web Interface

The web interface is available on the UMLS intranet portal. There are five independent browsing mechanisms available for site members. The UMLSKS web client provides a simple interface to the UMLSKS which displays a set of tabs which provide access to the various knowledge sources. The three principle knowledge sources: the Metathesaurus, the SPECIALIST Lexicon and the Semantic Network each have their own individual access tab. In addition to these tabs, two additional features are included: a tab for source discovery called the UMLS and Source View and the Tree Browser. As these interfaces are not easily employed in an automated fashion, they were not explored in detail.

The RMI API Client

The remote method invocation application programmer interface (RMI API) is the native interface to the UMLSKS which was explored in the most detail. The

process required to setup an environment in which a developer can use the RMI interface is relatively simple and is documented in a developer's guide available to internal members of the UMLSKS via the UMLSKS intranet portal. These steps are summarized below:

1. Obtain a Simple Object Access Protocol (SOAP) 1.2 compatible engine for webservice message routing
2. Download the Authentication Web Services Description Language (WSDL) for the CAS Authentication web service
3. Download the WSDL for the UMLSKS web service
4. Generate the client side stubs using Axiss WSDL2Java utility for both WSDLs

For the first step we elected to employ the Apache Software Foundations Axis 2 Library and found it to be suitable. Acquiring the CAS Authentication web service was equally uneventful. The developer's guide which outlines the setup process directs developers to that resource. The final two steps were performed with guidance provided by the developer's guide.

The next hurdle that developers using the UMLSKS RMI interface will face is the use of the proxy granting ticket provider. The UMLSKS requires users to agree to the license terms and agreements. Once a licensee has agreed to this contract they are granted a username and password. This authentication information is provided through a Java API to a UMLS service which returns a proxy granting ticket. The ticket which is provided in exchange for the authentication information is subsequently used to acquire proxies which are required to make query requests again the UMLSKS services. This process is replicated below:

1. Establish a connection to the Authentication web service
2. Obtain a proxy granting ticket (PGT) from the Authentication web service
3. Using the PGT, obtain single-use tickets from the Authentication web service to embed in each UMLSKS web service call
4. Build the call argument(s) to the UMLSKS
5. Request the data from the UMLSKS web service

The UMLSKS provides a variety of predefined queries through their RMI API. They also provide the capacity to perform custom queries against the raw UMLSKS data.

The Socket API Client

The socket API Client was the first programatically accessible and hosted view to the UMLSKS to be published. A number of previously constructed applications use this access point including some work done by Hubmed.org[41]; however, since this was an older version of the UMLSKS Portal, and since the available resources left the impression that some resistance was encountered while developing with them, this tool was also not deeply inspected. In addition to the difficulties encountered while researching the socket protocol, we were motivated to drive the research in the direction of the RMI protocol by our own and our industrial sponsor's experience with Java technologies.

The Supporting Physical Architecture

In the UMLSKS version 5 documentation an overview of the physical architecture which supports the UMLKS is provided. That diagram has been reproduced in fig. 5.3. In addition to this diagram, a table providing technical specifications about the machines in the diagram was included. This table is reproduced in tab. 5.2.

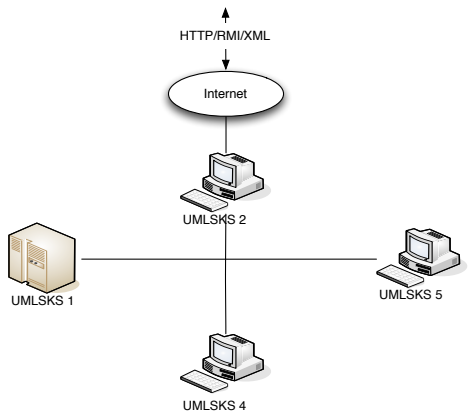


Figure 5.3: An overview of the physical architecture of the UMLSKS servers

Machine	Hardware Configuration	Functions
umlsks1	Enterprise 4000 4 CPU Sun machine with 2 GB RAM	Resonate Back-up Server Web Server UMLSKS Server/Oracle Database
umlsks2	SunSparc Ultra 2 workstation	Primary Resonate Server MySQL User Accounts Database
umlsks4	SunSparc Ultra 2 workstation 2 GB RAM	Web Server UMLSKS Server/Oracle Database
umlsks5	SunSparc Ultra 2 workstation 2 GB RAM	Web Server UMLSKS Server/Oracle Database

Table 5.2: An technical specification of the physical architecture supporting the NLM's deployment of the UMLSKS.

5.3.2 Shortcomings of the NLM UMLSKS Implementation

Two primary issues were uncovered while testing an early prototype of our UMLS exploration interface. The first was the fact that the continuity of service to the UMLS interfaces was insufficient for production use. While testing our early prototypes we found that the NLM interfaces were only operational about half of the times we attempted to access them. In addition to this problem, when the service was responding to queries, the responses took between 10s and a few minutes to respond. Even during these periods of service up time, it was found that approximately 10% of queries had not responded after 10 minutes.

Regardless of the sources of the performance and continuity of service issues, as a consequence of their existence, the native implementations of the UMLKS which are published by the NLM can not feasibly be incorporated into a production system where queries against this data source are a central part of a high throughput semiautomated workflow like the one we will describe in our codification workflow.

5.4 An ROA Interface to the UMLS Metathesaurus

A number of other tools which employ the UMLSKS services exist, most notably the MetaMap Program which is powered by the MMTx Java library[67]; however, in our experience these projects are typically purposed to very specific tasks and the interface

to the UMLS Metathesaurus, or the UMLSKS generally, were not modularized in such a way as to be accessible and adaptable to our application. Consequently we opted to develop our own interface and to design it in such a way that it could be incorporated into the architecture of our industrial partner's translational research informatics (TRI) solution.

In chapter 4 we discussed GenoLogics' industrial TRI solution. One of the properties of this solution is that it is based on a resource oriented architecture (ROA). The data exchange mechanisms used in this architecture are based on extensible markup language (XML), and are produced by a Java library called RESTeasy which is produced by Burke[14]. We elected to follow this same path as it was well explored, used technologies with which we were familiar and because we were surrounded by experts with these tools during the time we developed our interface implementation.

We began the development of our local implementation by downloading the UMLSKS data files and the Metamorphosys program which is provided by the NLM to licensees through their intranet portal. The Metamorphosys program provides an interface for customizing and loading the UMLSKS data into a local database. We elected to include only the SNOMED-CT, ICD-9/10 and LOINC vocabularies in our deployment. This decision was based on Kussaibi's[51] recommendations with regard to terminologies used in anatomic pathology structured reports, and based on our experience with the Health Level 7 Clinical Document Architecture.

Once we had acquired the dataset, we were able to generate a local instance of the UMLSKS database. Upon querying this local database with queries identical to the queries we had executed against the NLM provided interface with our early prototypes we noticed a significant improvement in performance. Indexing the database resulted in an improvement in performance against the NLM interface which was in the range of 3 orders of magnitude. Initial queries which might have taken three minutes were now taking tens of microseconds.

From here we created a object oriented model of the UMLS content which would satisfy the needs we had of the terminology server. To complete our component architecture, we developed two supplemental components: a server component which lifted objectified representations of the data from the database and subsequently marshalled that data for transmission, and a client component which unmarshalled that serialized data and subsequently provided an interface for its consumption. An overview of this architecture is presented below in fig. 5.4.

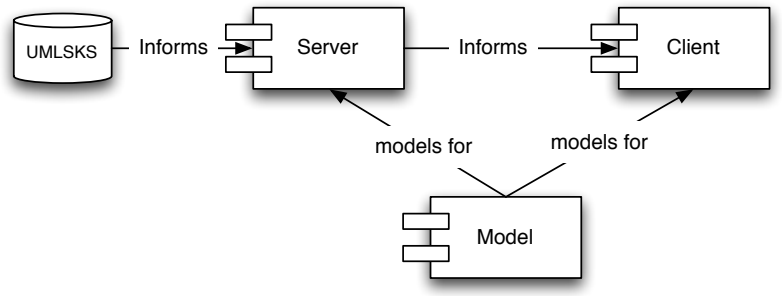


Figure 5.4: An overview of the RESTful UMLSKS interface architecture

5.5 Modelling Pathology Reports

Throughout the development of the prototypical implementations of the various pieces of our conceptual architecture we assumed that anatomic pathology structured reports (APSRs) could be modelled as hierarchical documents. This assumption was grounded both on the knowledge that APSRs have traditionally been designed as human readable documents, and on the expertise of both our collaborators at GenoLogics and of the developers of the integrating the healthcare enterprise APSR profile development team. Relying on this assumption we designed a simple model of an APSR which consisted of sections, subsections and finally leaves. This tree based structure has been sufficient to represent the generalization of the APSR class of documents described by IHE in the IHE APSR profile. We would include the APSRs provided to us by GenoLogics, but they were offered to us under a non-discloser agreement. Interested readers may contact Genologics (http://genologics.com) directly about possibilities to gain access to the documents.

We initially developed an XSD that specified a simple exchange format which could be used within an organization to validate basic electronic APSRs. This choice was motivated by an expectation that in future iterations of the prototypes we would wish to create mappings between this internal structure and the HL7 CDA via the Mirth extract transform and load (ETL) engine. In order to develop a software model to represent the instances of this XSD we employed the jaxb-xjc java package authored by Sun Microsystems. With little effort we were able to generate the data model for our hierarchical report from our hand crafted XSD.

5.6 An Interface for Creating Electronic APSRs

Our interface for creating and editing electronic anatomic pathology structured reports (APSRs) is a tool built using Java Swing technology. It is designed to manage a tree structure which represents an APSR. Through this interface a user is able to open manually created structured text based APSR templates. The interface then allows the user to modify the template by adding new sections, subsections and leaves to the template.

A number of different options were explored which could each have satisfied the practical requirements for creating and viewing APSR templates. When this work had just begun, our association with GenoLogics was quite close. GenoLogics wished to explore the possibility of using an XForms based technology for providing a feature like this. Consequently we explored a tool called Orbeon Forms. This tool seemed as though it would be suitable for the purpose of viewing and modifying data collection forms like those GenoLogics expected to encounter. GenoLogics perspective however, was very data-centric. The company expected that by requiring the use of forms, that the granularity of data units in their system would be too large. They expected that as a consequence of choosing Orbeon Forms, they would not maintain the flexibility they expected to maintain by choosing an option which induced fewer constraints on the data collection process. In the end, GenoLogics adopted a suite of tools which rely on cloud technology which are authored and maintained by salesforce.com.

This was a significant moment of divergence between the direction taken by GenoLogics and the direction which we adopted. We elected to continue with the use of the swing interface for our anatomic pathology structured report generation interface, where as GenoLogics elected to build their form construction interface using salesforce.com technology. Rather than interface with cloud technology as did GenoLogics, we instead adopted a local persistence layer which we designed to facilitate natural language processing. GenoLogics aims to meet their clients high data volume, and intermittent processing demands. Their clients only use available architecture in brief intervals, but when these clients need to perform processing, the system with which they interact must respond quickly. Our research efforts on the other hand focus more on the metadata - the annotations. There is far less metadata than there is data, and so the use of cloud technology would not be of great benefit in our work. It would have, however, greatly complicated our efforts.

5.7 Coding with the UMLS Metathesaurus

In order to address the disparity in skills possessed by our intended interdisciplinary user base, we constructed our clinical coding tool around a visualization which is based in large part of theory which is presented by Ware[83] and to a lesser extent, that presented by Tufte[78]. The approach we have taken with the visualization component of our solution is an extension of the works of Sundvall[76] and Lin[53] who built visualization tools for SNOMED CT and the UMLS Metathesaurus respectively. We first explored these tools and the publications describing them. We then, with the guidance of the publications of Ware[83] and Tufte[78], developed a visualization which more directly addresses the needs of users who wish to perform clinical coding with the assistance of a terminology server.

The terminology exploration feature of our clinical coding tool allows coders to inspect the relationships between and within the terminologies with which they are familiar and those used by other researchers. By employing the UMLS Metathesaurus as a communal knowledge source while performing clinical coding of anatomic pathology structured report (APSRs), semantic annotations which facilitate interdisciplinary knowledge exchange can be chosen. By employing visualization we have allowed these experts to perform this task in a more exploratory way, accelerating the development of their understanding of diverse terminologies.

Presented in fig. 5.5 is a screen shot of this visualization interface. A user first opens an APSR for annotation. Our prototype then parses the APSR into the tree structure labeled as A in the figure. Once the user has finished modifying the APSR template by adding and removing sections, subsection, leaf nodes, etc., the user can use the search tools labeled as B to uncover concepts which might potentially be used for annotation. The user can then continue to explore the uncovered concepts using the visualization panel labeled as E with its corresponding controls labeled as C and D, and the definitions panel labeled as F. Annotated documents may be saved for future retrieval.

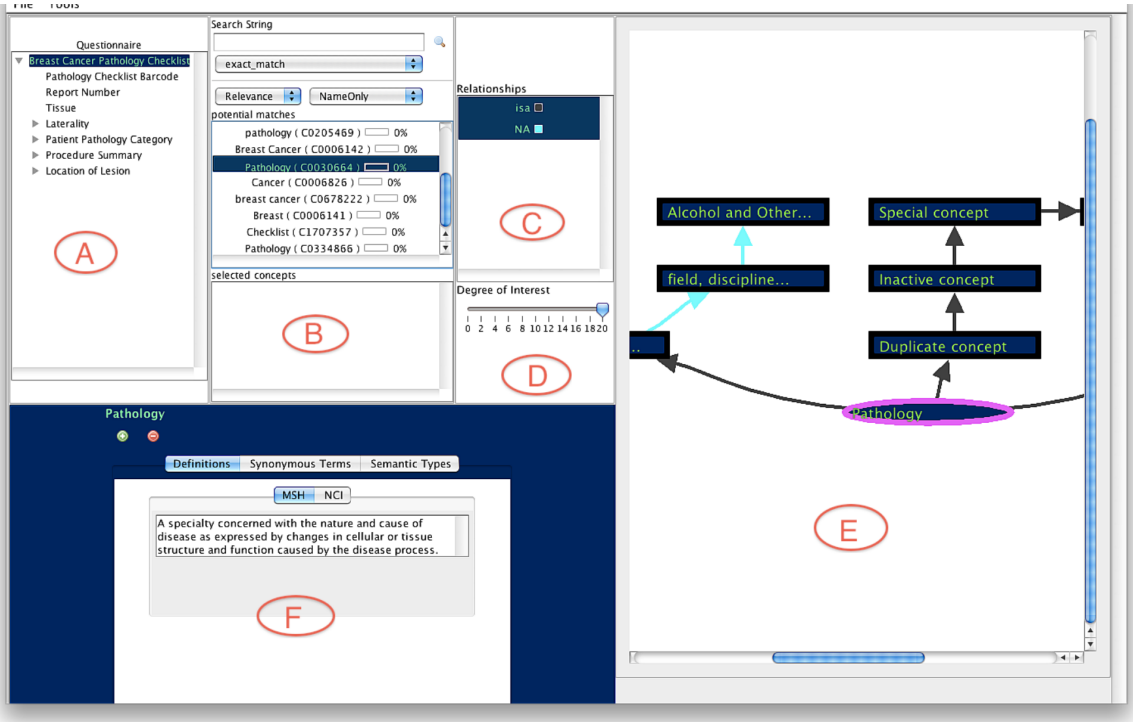


Figure 5.5: A screen shot of the visual coder application with an open anatomic pathology structured report.

5.7.1 Clinical Coding with a Terminology Visualization Tool

We demonstrate the efficacy of our visual coding interface by evaluating it on a theoretical basis by employing the visualization theories described by Ware[83] and Tufte[78] in the context of a hybrid of Shneiderman's[74] and Van Ham's[81]visualization task lists. As an example of visualization theories which have been employed, we use no more than eight colours to represent relationships between concepts in the visualization. This is because Ware indicates that human cognitive abilities are insufficient to process such a complicated display[83][p. 113]. We also attempted to make use of Tufte's theories on color[78][ch 5].

Visual Coder is evaluated in the context of a targeted usage scenario which is described in tab. 5.3. We evaluate the visualization by showing that the tasks related to the steps of this workflow are supported. Shneiderman's visualization task list[74] was found to be insufficient for use as a representation of the visualization tasks which would be required to accomplish the steps presented in the workflow, and so we have

constructed a hybrid task list in which we also incorporate tasks which are suggested by van Ham[81]. This task list is presented in tab. 5.4.

Use Case Steps
1. Open an APSR template
2. Inspect a section title or field title from the APSR
3. Use the terminology exploration tool to find the concept which was intended by this title
4. Annotate the selected field with the intended concept
5. Repeat this process until the document is sufficiently annotated to provide the desired degree of semantic interoperability with collaborating researchers
6. Persist the annotations to an information repository

Table 5.3: A clinical coding use case

Task	**Description**
Search	Isolate a generalized subset of the terminology for refinement
Show Context	Show relationships between a core concept and its conceptual context
Filter	Hide uninteresting items
Expand on Demand	Allows the user to broaden over specified searches and also provides additional information about selections
Zoom	Focus on details which are pertinent to the task
Extract	Extract data subsets and query parameters
History	Rely on the system to persist actions previously taken for future reuse

Table 5.4: A hybrid visualization task list for semantic annotation[74][81]

5.7.2 Satisfying the Use Case

Initially a clinical coder opens an APSR which is represented in a local computable syntax. The APSR is displayed in A of the working area as shown in Fig 5.5. Visual Coder will automatically select the first APSR node (i.e. a section title, a data label, or a data value) for the user to annotate.

Search

Visual Coder then automatically searches the knowledge source for concepts which match this first selected node using pre-configured search strategies which are provided in B. It then automatically populates the list of potential matches in the list

shown directly above B. The first of these concepts is automatically selected for display in the visualization panel labeled E. The display shows a node link graph which places the selected concept as the root. The graph portrays the conceptual hierarchical context of the selected concept.

The user can then use the contextual information provided by the visualization to determine the suitability of the suggested concept for the annotation of the selected APSR node. If the user is not satisfied with the first concept in the list of concepts proposed by the system for annotation, he or she can either select another concept in B, or even choose one of the other concepts which appeared in the visualization, E, as an annotation. If the user exhausts all of the system's prefetched suggestions, then he/she can use the search strategies provided in B using his/her own input strings as opposed to the exact APSR node title which is used by default. This process again will populate the potential matches list with the concepts returned by the knowledge source in response to the user's queries.

Show Context

The relate task is fulfilled by the node link graph which is used to represent suggested concepts within their semantic context in E. Using nodes to represent concepts and arrows to represent relationships between them takes advantage of users' precognitive faculties to recognize association[83]. Curved arrows were used to represent the links between nodes as they are more more likely to be perceived as part of the same entity[83]. The various relationship types are categorized and aggregated using colour. The display uses only eight colours so as not to overwhelm the pre-coginitive recognition of the user[83]. These colours are linked to the relationship filter in C by a legend of associated colour blocks. The colour legend shown in the screen shot displays a yellow square associated with the isa relationship type. This yellow square corresponds to the yellow arrows used in E to represent concept relationships. In more complex graphs however, a more extensive legend is shown which uses different colours for each relationship type. The inclusion of a brushing and linking feature has been considered as well. This feature would highlight elements in the display panel as the user passed the mouse cursor over the associated relationships in the visualization control panel.

Filter

Once a user has found a concept which shows promise as a potential APSR node annotation, he/she can take advantage of the filtering options provided in C and D to refine his or her search. Currently, a set of relationships are displayed to portray the conceptual hierarchical relationships of the selected concept. The visualization control panel allows the user to show or hide concepts based on their associated relationship types. Currently, the graph does not include children of the root concept along the hierarchy paths. This is as a consequence of technical issues with the underlying relational model which have resulted in performance issues. In future work, it is expected that child, co-occurrence and other relationship sets will also be extracted from the knowledge source in addition to the hierarchical relationships. In addition to the relationship type filtering feature in C, a degree of interest filter, labeled D, is provided. With this feature a user can specify the desired volume of context they wish to view by limiting the concepts by their distance in edges from the graph's root concept.

Expand on Demand

The next task is primarily addressed by the definitions panel, labeled F. As has been described, when a user selects a concept from the selected concepts list in B, the display, labeled E, is populated by the node link graph which represents the conceptual context of the selection. The selected concept is designated as the root of this diagram and is selected by default. The definitions panel, labeled F, provides a more detailed description of the selected concept than is provided in the visualization panel, labeled E. The display panel only provides a fixed length abbreviation of concept names with the full name being available through a mouse over tool tip (in addition to the context information of course). The definitions panel provides the full concept name, concept definitions, synonymous terms to the concept title and semantic categorizations. In other words, it expands the number of displayed details about a given concept on demand.

By default, the root concept of the node link graph is selected in the visualization. Consequently, the information published in the definitions panel describes this root concept. With the click of a mouse the user can select other nodes within the graph. This action will automatically repopulate the definitions panel with details about the newly selected concept. Concepts which have been selected in the visualization panel

can be added as annotations to the concept which has been selected in the APR tree by using the add annotation button in the definitions panel (the green plus icon). This action will move either move the selected concept from the potential matches list to the selected concepts list in B if the chosen concept is present there, or will simply add the concept to the selected concepts list if it is not.

Zoom

The zoom task is supported by the navigation features available through the display panel in E. By using the mouse and various command keys, the user can pan across the displayed concept graph and can also zoom in and out of the display when they wish to see a particular corner of the graph in more detail. Additional features like a semantic zoom with which a user could change the root of their concept map based on a double clicking action will likely be considered in a future version.

Extract

The extract task is not currently supported by this prototype; however, the use of the Mirth extract, transform, and load (ETL) engine in concert with the persistence layer of our architecture is presented later in this chapter.

History

Some history tasks are supported. For example, selected annotations are stored in the short term as selected terms, and can later be stored in the long term through the persistence layer. The history of perviously performed searches however is not supported. A feature like this could be facilitated in future by the implementation of caching on the RESTful server, and also by using a more elaborate querying mechanism for prefetched annotation suggestions. Currently a variety of search routines are employed to provide the initial annotation suggestions. By monitoring user selections and the queries which uncovered them, a bias towards the better search strategies could be integrated into the application.

5.8 Persistance

Conceptually we divided our information storage needs into four distinct components. First, we needed to store the structures of our APSRs. Second, we needed to store

the annotations which provided the semantic meaning of our APSRs' fields and titles. As an example, the synoptic report provided to us by GenoLogics which we used to generate our computable APSR had a title, "Breast Cancer Pathology Checklist". (We would provide this sample synoptic report as an appendix, but it was provided to us under a non disclosure agreement. Interested readers may contact Genologics (http://genologics.com) directly about possibilities to gain access to the documents.) This title would be annotated. Each section and subsection title in addition to each field label of the pathology report would also be annotated. The sections would sometimes contain a set of labeled checkboxes which were used to classify the tissue sample in question. These labels would be annotated as well. Third and fourth, we recognize that we eventually need to capture the data which is recorded using those APSRs and we will also likely need to maintain a record of the meta data surrounding the APSR templates (authorship etc).

5.8.1 APSR Template Structure

In order to represent our APSR templates in the database, as opposed to our initial computable syntax, we considered two potential algorithms: an adjacency list algorithm, and a modified pre-order traversal algorithm both of which were outlined by van Tulder[79]. van Tulder, in his examples, wishes to store the tree in fig. 5.6 in a relational database. APSRs like many human generated documents have a hierarchical structure. Documents have a title, sections, subsections, and in the case of APSRs, the data entry elements could be considered leaves. Because there is such great similarity between the structure of an APSR and the structure of a tree, we elect to represent the APRS templates as trees in our persistence layer.

Now, as with many pairs of algorithms, the adjacency list algorithm and the pre-order traversal algorithm each have strengths and weaknesses. In the following three sections, we will first outline line the adjacency list algorithm, next we will outline the pre-order traversal algorithm, and finally we will provide a contrast between the strengths and weaknesses of each approach.

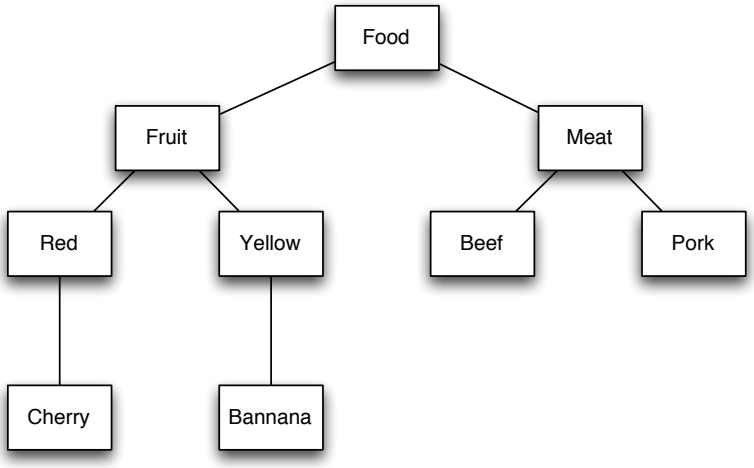


Figure 5.6: A hierarchical structure which might be stored in a relational database

The Adjacency List Algorithm

Here we will outline the adjacency list algorithm described by van Tulder[79] for storing hierarchical data in databases. In order to provide this outline we will illustrate how the algorithm works by employing the algorithm on the same hierarchical dataset, displayed in fig. 5.6.

In the adjacency algorithm, a database table is used to store child parent pairs. To demonstrate this pairing we refer to tab. 5.5 which displays how the tree in fig. 5.6 would be stored in such a table.

parent	**title**
	Food
Food	Fruit
Fruit	Red
Red	Cherry
Fruit	Yellow
Yellow	Bannana
Food	Meat
Meat	Beef
Meat	Pork

Table 5.5: The hierarchical structure in fig. 5.6 represented in a relational database table using the adjacency list algorithm[79]

If the adjacency algorithm is used, then a traversal of the tree or any subtree can be executed through recursive querying as displayed in the php/sql code example provided in fig. 5.8.1. Using a very similar algorithm, van Tulder also explains how a similar recursive querying approach can be used to acquire the path to any given node.

```
<?php
// $parent is the parent of the children we want to see
// $level is increased when we go deeper into the tree,
//        used to display a nice indented tree
function display_children($parent, $level) {
   // retrieve all children of $parent
   $result = mysql_query('SELECT title FROM tree '.
                         'WHERE parent="'.$parent.'";');
   // display each child
   while ($row = mysql_fetch_array($result)) {
       // indent and display the title of this child
       echo str_repeat('  ',$level).$row['title']."\n";

       // call this function again to display this
       // child's children
       display_children($row['title'], $level+1);
   }
}
?>
```

Figure 5.7: This recursive php function, reproduced from [79], will perform a preorder traversal of the hierarchical structure, shown in fig. 5.6, which tab. 5.5 represents

The Modified Pre-Order Traversal Algorithm

Here we will outline the modified pre-order traversal algorithm described by van Tulder[79] for storing a hierarchical tree in databases. In order to provide this outline we will illustrate how the algorithm works by employing the algorithm on the tree displayed in fig. 5.6. We display a labelled replication of the initial tree shown in fig. 5.6 in fig. 5.8 to illustrate the mechanism used in this algorithm.

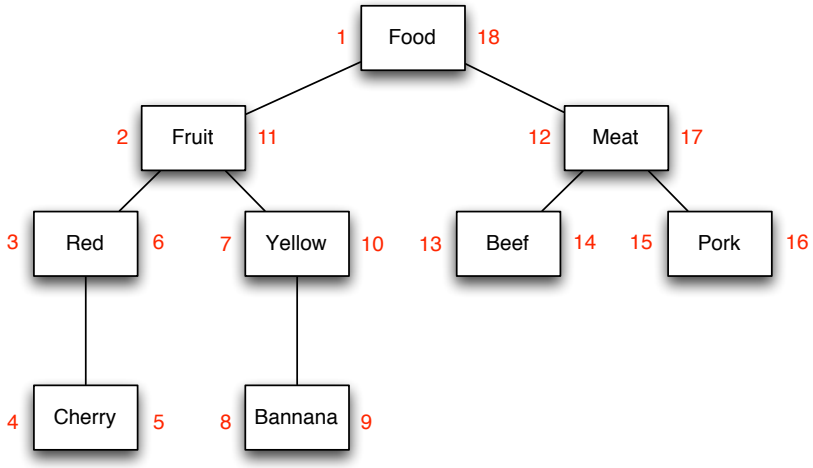


Figure 5.8: An illustration of the tree in fig. 5.6 reproduced from [79] which has been labeled using the modified preorder traversal algorithm.

Using the modified pre-order traversal algorithm, a database table is used to store triples which include a node value, and numeric left and right labels. Tab. 5.6 displays how the tree in fig. 5.8 would be expressed in such a table.

parent	**title**	**left label**	**right label**
	Food	1	18
Food	Fruit	2	11
Fruit	Red	3	6
Red	Cherry	4	5
Fruit	Yellow	7	10
Yellow	Bananna	8	9
Food	Meat	12	17
Meat	Beef	13	14
Meat	Pork	15	16

Table 5.6: A table reproduced from [79] which illustrates how the tree in fig. 5.6 could be represented in a relational database table using the modified preorder traversal algorithm.

If we elect to use the modified pre-order traversal algorithm, we can traverse the tree in two subsequent steps. First we need to execute a single query against the database to acquire a listing of the tree nodes, and second we need to use a function

which will employ a stack to parse the query result and generate the tree structure using the left and right node labels. In fig. 5.8.1 we provide a sample block of php code which performs this operation, but also uses an initial database query to fetch the left and right labels of the root node.

```
<?php
function display_tree($root) {
   // retrieve the left and right value of the $root node
   $result = mysql_query('SELECT lft, rgt FROM tree '.
                         'WHERE title="'.$root.'";');
   $row = mysql_fetch_array($result);
   // start with an empty $right stack
   $right = array();
   // now, retrieve all descendants of the $root node
   $result = mysql_query('SELECT title, lft, rgt FROM tree '.
                         'WHERE lft BETWEEN '.$row['lft'].' AND '.
                         $row['rgt'].' ORDER BY lft ASC;');
   // display each row
   while ($row = mysql_fetch_array($result)) {
       // only check stack if there is one
       if (count($right)>0) {
           // check if we should remove a node from the stack
           while ($right[count($right)-1]<$row['rgt']) {
               array_pop($right);
           }
       }
       // display indented node title
       echo str_repeat('  ',count($right)).$row['title']."\n";
       // add this node to the stack
       $right[] = $row['rgt'];
   }
}
?>
```

Figure 5.9: This php function, reproduced from [79], will display a modified preorder traversal of the hierarchical structure, shown in fig. 5.6, which tab. 5.6 represents.

With the modified pre-order traversal algorithm, acquiring the path to a node can be achieved with a function which unlike with the adjacency list algorithm, is in fact quite different from the display function. The SQL query listed in fig. 5.10 will provide the nodes in this path.

```
SELECT title FROM tree WHERE lft < ? AND rgt > ? ORDER BY lft ASC;
```

Figure 5.10: A query, modified from [79], which will acquire the rows representing the path to a given node in the tree structure represented by tab. 5.6.

A Comparison of the Adjacency List and Modified Pre-Order Traversal Algorithms

Query frequency is expected to be the factor which has a significant impact on system performance in our environment since we are building what is expected to be a geographically widely distributed network; consequently, we want to ensure that network traffic is kept to a minimum. For this reason, using the modified pre-order traversal algorithm is expected to be faster than the adjacency list algorithm. This is because the process for retrieving structural information when the adjacency list algorithm is used requires multiple queries against the database, while when the pre-order traversal algorithm is used only a single query is required. Once this query has been performed an iterative function can be used to recreate the tree structured from the raw data. The fact that this function would be iterative rather than recursive is expected to provide further run time benefits as the languages which are currently commonly used in system construction: Java, C++, C#, are procedural rather than functional and thus are not tuned to provide recursion in such a way as to make effective use of memory[79].

5.8.2 Storing Annotations to Accomodate NLP Processing

The relational model used to store semantic annotations of APSR template titles, section and subsection titles and field labels in our system, shown in fig. 5.11, is designed to facilitate the integration of the data captured using those templates using natural language processing assisted automated mechanisms.

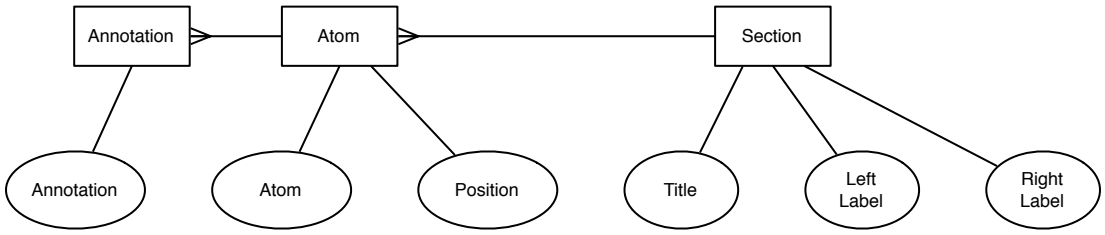


Figure 5.11: An Entity Relationship Diagram describing the database schema used to store APSR template annotations

Through our experience, which is supported by the IHE APSR Profile, APSR template titles, section and subsection titles and field labels are frequently structured

as noun phrases[25]. These noun phrases are composed of a trailing "head" noun preceded by a series of qualifying adjuncts. For example, consider the title of the APSR template provided to us by GenoLogics: "Breast Cancer Pathology Report". The trailing head noun identifies the object. In this case, we are dealing with a report. We also see that there are some qualifying left adjuncts. These left adjuncts could be considered in their initial state or by using compound nouns which were composed of the atoms. For example, perhaps this is a pathology report, or perhaps it is a cancer pathology report. On the other hand, perhaps the pathology in question is a cancer, and more specifically a breast cancer.

The degree of apparent freedom in this type of atomization of the APSR template titles, section and subsection titles and field labels lead us to the conclusion that rather than prescribe a naïve interpretation of the semantic intent of the template author, we should instead provide the subordinate clinical specialist of this author the ability to annotate the template in an atomized way. Rather than requesting that a user annotate a title, section or subseciton title, or a field label as a whole we request that they annotate atoms within the title or label. To provide this feature we take a composite noun approach. We automatically identify the trailing noun as the head noun, and then provide a qualifying "degree of distance" attribute to each of the preceding nouns.

5.9 Transporting CRI Knowledge

Many legacy systems are already in place at hospitals, clinics and research facilities around the world. Currently many middleware solutions are used as mechanism to establish lines of communication between these systems. The Mirth Project[10] is one such middleware system. Mirth is an extract transform and load (ETL) engine which has a strong health focus. It provides extensive HL7 support and addresses a number of other medical communication protocols as well. Given that the IHE APSR group advocates for the use of the HL7 Clinical Document Architecture (CDA) in their profile, and given that the HL7 CDA is prescribed by both Canada Health Infoway[80] and adopted by the National Cancer Institute (NCI)[44], our choice to employ Mirth was natural. Though a number of other projects provide similar services to those provided by Mirth, our exploration of the Mirth community and tools was so rewarding that we did not pursue other options. We were able to accomplish each task we set out to perform with Mirth and so the need to explore

alternatives did not arise. Interested readers however may wish to read Brauer's[11] discussion of alternatives.

5.9.1 Exploring Mirth

In order to validate our suspicion that Mirth could provide an effective communications backbone for the architecture of a collaborative research network we employed a small scale experiment against GenoLogics BioChronicle component. This system provides a RESTful interface using XML technology and HTTP protocols. Over the course of a few working days we were able to establish an authenticated connection, author a transformation script using XSLT, another XML technology and output the transformed data. Specifically, we fetched a string of specimen annotations from bioChronicle and relabelled the wrapping syntax with synonymous vocabulary. Though this experiment was not performed within the full fledged context of our proposed interoperability architecture, it still demonstrates the feasibility of this approach.

Connectors

Bortis[10] explains how the Mirth interface architecture is designed around a source connector, filter, transformer, and destination connector model. The use of this model has some important ramifications in practice and some which are of particular interest for our APSR focused use case. When using Mirth, a user is not restricted to the incorporation of a single source connector for a transformer, nor are they are they restricted to a single destination connector.

Considering as an example a sample APSR with its multiple sections including the macroscopy, microscopy and pathology sections[51], we can see how each of these sections may be have been completed by a different member of a patient's care team. Each of these care team members might then use their individual information systems to submit their portion of the APSR. This information can then be aggregated and manipulated by a transformer module. This transformer module could populate an HL7 CDA from the data collected in the collection of source connectors. This HL7 CDA could then be shipped by the transformer over a destination connector to a source connector which is received by the supervising physician's system and the systems of other authorities who would be responsible for validating the document content's validity, approving the document and subsequently completing the HL7

CDA header information before the document was saved.

This kind of content validation however, is insufficient. Our aim is not to provide human interoperability. This exists already. We aim to provide machine interoperability, and to facilitate this we need to provide rigorous rules based automated validation of the syntax and content of the document, a topic which we will now discuss further.

5.10 Validating an APSR

To validate a completed APSR within our architecture we need to ensure that the syntax of the document is compliant with the HL7 CDA in simple cases or that it is compliant with IHE APSR profile in more advanced case. Using our architecture, this process would occur by first establishing a connection between our publishing component and Mirth. In the simple cases, the source connectors which our publishing component would use to push data into Mirth would be consumed by a destination connector which was connected to a validation service. This validation service could then use known software packages to validate the compliance of the document which was proposed for publishing against the HL7 CDA XSD. The more complicated case would operate in a similar way. Instead of validating the document which was proposed for publishing against the HL7 CDA XSD, the validation component would instead use different software packages to validated the document using a Schematron generated XSLT and report any transformation failures.

XSD

In order to ensure that APSR documents could be validated we were able to employ java libraries to validate XML documents against XSD schemas. Though we did not go so far as to validate fully completed HL7 CDA documents, we have demonstrated that the automated validation of XML documents against XSD schemas is not only feasible, but that the implementation of such a system is well supported with current software packages.

This validation would ensure that documents which are produced by the system are compliant with the syntax demanded by the HL7 CDA. An initial benefit of this validation is the insurance that documents which have been generated include critical metadata. The document for example, must identify its author, its curator, the

patient about whom it speaks, its legal authenticator and a number of other parties and entities. The data content of the document however is not thoroughly addressed by the CDA, further more, XSD is insufficiently expressive to provide validation of specializations of all of its templates. This was discussed in detail in chapter 3. It is for this reason that when we began to work with the IHE APSR profile, we began to explore Schematron and its associated tooling.

Schematron

As was discussed in our background material, not all restrictions on the contents of APSRs induced by the IHE APSR profile can be validated by employing the XSD standard. One example of this shortcoming is the requirement within the APSR profile that APSRs contain specific sections which would translate to specializations of the Component type in the central ClinicalDocument type expressed in the HL7 CDA. By employing the transformative approach of Schematron as opposed to the declarative approach used by XSD this issue can be resolved. As a reminder, XSD validation relies on a declaration of the structure of a document as the point of validation. Schematron on the other hand makes the statement that all documents which are compliant to the given template can be successfully transformed by an XSLT which is generated based on the Schematron syntax.

The IHE profile in its published state does not include the Schematron file required to generate the required XSLT. We started writing this file using the Schematron language and completed enough of the process to understand how the entire profile could be written. This included the first twenty four elements in the profile specification which are listed beginning on page 46 of the IHE APSR Profile[43]. Examples of the rules we implemented are provided below:

- The Structured Body must contain no more than one Clinical Information Section
- The templateId for a Clinical Information Section must be 1.3.6.1.4.1.19376.1.8.1.2.1
- The code for a Clinical Information Section must be 22636-5
- The displayName for a Clinical Information Section must be "Pathology report relevant history"
- The codeSystem for a Clinical Information Section must be 2.16.840.1.113883.6.1

- The codeSystemName for a Clinical Information Section must be LOINC
- A Clinical Information Section must include a narrative block in a text element
- A Clinical Information Section must contain no more than one Coded Reason for referral [Reason for AP procedure] section
- A Clinical Information Section must contain no more than one History of present illness section
- A Clinical Information Section must contain no more than one Active Problems section
- The templateId for a Reason for Referral Section must have be 1.3.6.1.4.1.19376.1.5.3.1.3.2
- The code for a Reason for Referral section must be 42349-1
- The codeSystem for the Reason for Referral section must be 2.16.840.1.113883.6.1
- The codeSystemName for the Reason for Referral section must be LOINC
- The displayName for the Reason for Referral section must be Reason for Referral
- etc.

It became clear that the process was not academically challenging but merely labour intensive. At the point where we recognized how the process would proceed, we decided to turn our attention to other aspects of the architecture instead. Our expectation is that the IHE APSR group will eventually release the Schematron file which is required to verify the conformance of an APSR instance against the specification in their APSR profile.

Chapter 6

Evaluation

6.1 Evaluation: Validation and Verification

Evaluation of software engineering research typically deals with verification and validation as the process of checking that a software system meets specifications (verification) and that it fulfils its intended purpose (validation). We provide a verification of our architecture against its specifications. We provide an overview of our evaluation process in fig. 6.1. This overview links the principle components and artifacts of our architecture with the corresponding methods we used to verify them. Our process as a whole used as an agile model driven architecture methodology[2]. We supplemented this methodology with the evaluation techniques which are labeled in fig. 6.1. The details of how we incorporated each of these methodologies are described throughout the remainder of this chapter.

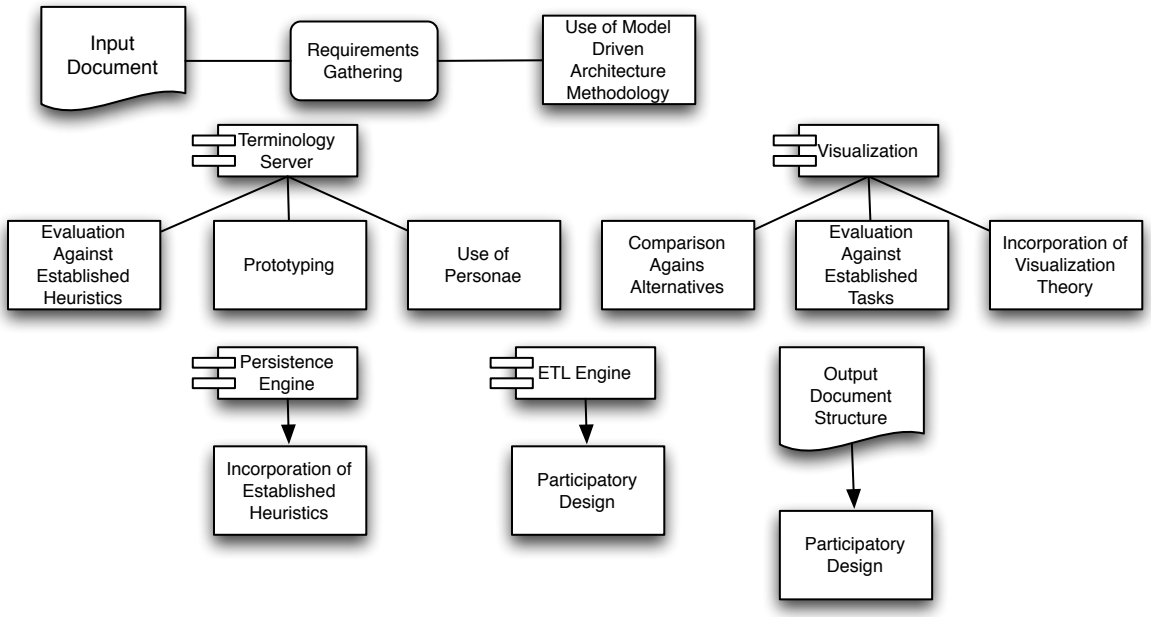


Figure 6.1: A diagram associating the components of our architecture with the techniques used to evaluate them

A shortcoming of our evaluation is that we do not rigorously address the question of whether the annotations which could be recorded with it can in fact be effectively leveraged for automated, or semiautomated system integration. The incorporation of validation techniques like user studies such as these is a persistent and primary difficulty in translational research informatics (TRI). Both older research efforts, like those of Frank[24] and more recent ones like those of Glez-Peña[61] struggled with this issue. The difficulty with performing user studies in the domain arrises as a consequence of a number of factors including but not limited to the availability of the experts for whom systems are designed, the privacy and security issues which need to be addressed in health care order to pursue such studies and the time required to perform studies of this nature which will yield meaningful results. In the domain of clinical research informatics there is also significant disparity in the processes which are used to manage cancer informatics data between different health authorities. As we have decided to relegate user studies which evaluate our architecture to future work, here we focus instead on the verification efforts we have undertaken.

We will begin our verification by critically considering our requirements gathering process. This will act as a bridge to the evaluation of our primary contribution which we present in the intermediate sections. In the final section of this chapter we will verify our tool based on domain specific criteria which were laid out by Embi[21].

6.1.1 Requirements Engineering

The use case described in chapter 5 which was used as the basis for many of our evaluations was derived through an agile model driven architecture design process[2] which took advantage of personae, an interaction design technique described by Grudin[32]. We also engaged in traditional requirements engineering techniques, prototyping activities, and other model-driven techniques[63] with experts at GenoLogics in regular design and evaluation meetings. We applied the knowledge we acquired through these processes to our models which we usually represented using informal syntax, but which we occasionally formalized using the unified modelling language (UML). The use case which we described in chapter 4 was derived during this process. The expertise which GenoLogics provided was derived from their experience with a breadth of clients within the translational research domain. GenoLogics' understanding of the actors in this domain is expressed in the personae which were mentioned just mentioned and which are outlined in chapter 4.

By employing personae, as described by Grudin[32], we ensured that we had empathy for the potential end users of our architecture implementations and that we were designing explicitly for them. A criticism which could be levelled against our approach to requirements gathering would be that it did not put us in direct contact with out expected end users, this technique might have more closely resembled what Grudin refers to as participatory design[32]. We acknowledge this, but we also propose that by taking advantage of the expert knowledge that GenoLogics provided, we were able to focus more on the literature review portion of our requirements gathering process than we would have had we performed interviews with each of these potential users.

6.1.2 Evaluation against Established Heuristics: The Terminology Server

In chapter 5 we discussed an evaluation of the UMLS Metathesaurus against a checklist of terminology server properties which was authored by Barrett[5]. To summarize, Barrett stated that a terminology server could be evaluated based on the services it provides for curation, i.e. addition and deletion of terms, and development, deprecation and replacement of terms. The UMLS Metathesaurus and its curation are managed by the National Cancer Institute (NCI). In a general sense, each of these features is provided, though the manager of a system which was constructed based on our architecture would be required to rely on the NCI to provide these services when they wished to make a modification to the central as opposed to local portion of their terminology server complex.

At first glance, this may appear to be a bad thing. Consider however, that the terminology contained in this system is intended to promote interoperability, and by Weng's[84] estimation, many approaches to semantic interoperability which relied on semantic annotation and which did not rely on a central terminology server have been less effective than desired. Also recall the fact that we use a supplemental local terminology server in our architecture. This supplement is intended to be used for point to point communications between users who have isolated communication needs, thus addressing, in part, users immediate need to curate their annotation terminologies. Given the availability of this supplemental terminology component, the UMLS Metathesaurus is, at least when evaluated against these metrics, a good choice of terminology server for our application.

Barrett also states that "[t]erminology servers should incorporate automated mechanisms for terminology verification, or at least facilitate manual verification.", and that "[t]his ensures the terminologies' consistency and soundness which are crucial for systems reliant on the terminology server". We discussed in chapter 5 how the process of verification performed by the National Library of Medicine (NLM), rather than being automated, is focused on the engagement of a community of experts who satisfy the same goal.

Barrett also identifies architectural extensibility as a feature which is desirable for any clinical terminology server. When we first uncovered the UMLS we implemented a number of prototypes which employed the native UMLS Metathesaurus interfaces provided by the NLM. These are discussed briefly in chapter 5. In summary, the

native interfaces to the UMLS Metathesaurus did not provide an acceptable architecture. The interfaces were not sufficiently generic to be incorporated by a sufficient breadth of clinical architectures, and the native interfaces performance and quality of service were also insufficient for use in production systems. Consequently, we designed a new interface to the UMLS Metathesaurus. This new interface is takes advantage of a software engineering design called the factory pattern to provide an easily extensible interface to a local instance of the UMLS Metathesaurus. This use and fine tuning of the local instance of the terminology dataset on a local database resolved the quality of service and performance issues.

In terms of content, we refer first to Gschwandtner[33] who informs us that the UMLS is the largest thesaurus of its kind. We inform the reader that the UMLS Metathesaurus includes principal terminologies used for semantic annotation in medicine including SNOMED CT, LOINC, and ICD-9/10. It also includes the Gene Ontology which, according to Rojas[70], is the primary terminology used for semantic annotation in the basic biomedical sciences.

Of course, just because the terminology server includes the most vocabulary does not necessarily mean it will be the best choice for our application. Supplementing the breadth of information which is provided by the UMLS Metathesaurus is the fact that it provides features for both filtering the data contained in a local implementation of the database, as well as features for indicating preferences for certain terminologies. A user can for example modify settings in the UMLS Metathesaurus installation program to indicate that SNOMED CT codes are more important to them than are ICD-9/10 codes.

Given that the UMLS Metathesaurus, when supplemented with our RESTful interfaces, provides an extensible interface, provides more of the terminology which spans the translational research domain than other options and given that it provides customization options which facilitate the maintenance of performance and intelligibility, we conclude that it is a good choice of terminology server for the purpose of providing the terminology necessary for semantic annotation in an interoperable electronic biomedical research collaborative.

6.1.3 Prototyping, Personae and the Coding Interface

One of the mechanisms we used to evaluate our implementations, and transitively our architecture was to evaluate the suitability of our interface for the personae[32]

developed by GenoLogics, and specifically the use of personae as a target audience for our tool. One example of the benefits of using personae is that by incorporating an understanding of a given persona's willingness to learn new technology, we can gauge how intuitive an interface needs to be in order for it to be useful to its target audience. We acknowledge that without explicit metrics to measure intuitiveness, that it is difficult to make any strong claims here, but by relying on the evaluations which were provided by the technical managers and product managers at GenoLogics, we can claim that at the very least on the right track. As an example of how these evaluations were used we can relate that they were used to discard a number of early prototypes which attempted to use abstract information models as the basis for the annotation process. This approach was deemed to be too foreign to the personae who were expected to use tools based on our architecture and so it was abandoned.

6.1.4 Comparison of the Visualization to Existing Alternatives

In this section we compare our visualization interface to the previously created visual biomedical knowledge source exploration tools of Sundvall[76] (TermViz) and Lin[53](Visual Concept Explorer (VCE)) using three generalized tasks each of which relate to the intended purpose of our visualization implementation: querying, exploration, and scope management. We make a division between general exploration and scope management because the techniques used to manage the scope of view is explicitly explored in visualization and so we can provide a more thorough investigation with respect to this metric.

We selected these three tasks for explicit inspection because they are all central to the semantic annotation process. Given a specific title, section or subsection title or field label, a user will first query the interface to uncover the potential meanings of the terms in that title or label. Based on the results of this query, the user might either explore the terms in the surrounding context to verify his or her suspicions that he or she has found an appropriate annotation. To accomplish this task, the user may wish to manage the scope of terms which he or she is currently viewing. On the other hand, a user may recognize that his or her initial query has returned results which do not match the users intended meaning. In this case, the user may wish to attempt a new query.

Both Lin and Sundvall's tools incorporate a number of visualization techniques to

facilitate the exploration of a knowledge source. Lin's tool, the one we will explore in the most detail suffers from usability issues. Sundvall's tool was not available in a form that allowed us to perform an in depth evaluation and so we discuss it based solely on his publication; consequently, our coverage of his tool will not be as comprehensive. Let us now then begin with an investigation of Sundvall's tool. We will continue afterwords with a discussion on Lin's tool.

6.1.5 TermViz

Sundvall's TermViz[76] is a visualization tool which was developed to explore the SNOMED-CT[7] knowledge source and the ICD-10 codification standard. The tool was developed using the Prefuse[38] graphics package and was implemented in Java. Supplementary to this module, the Lucene[35] free text indexing engine was used to facilitate Google style querying of these knowledge sources.

Sundvall motivated the creation of a flexible visualization tool for knowledge source exploration with a number of arguments. To motivate the creation of a visualization tool they point to the volume of data stored in typical terminology servers as a major challenge for users. To motivate the use of a flexible visualization they explain how typical tree based structures like those found in computer file systems are only appropriate for certain user tasks. They also illustrate that tree based structures are unable to represent common knowledge source data structures like multiple inheritance and concepts which are linked together by multiple relationship types.

To overcome the shortcomings of tree based visualizations, Sundvall employed a node graph visualization technique. To display the nodes in an orderly fashion, they used a force based layout option provided by the Prefuse library. This force based layout algorithm is based on the works of Barnes and Hut[4]. This layout option is successfully used as a graph drawing algorithm. The visualization behaves in an intuitive way and produces aesthetically pleasing and intelligible graphs. Unfortunately, as we discovered in our work, this graph drawing technique does not scale well with the number of nodes displayed.

Sundvall describes a tool which supports a suite of tasks: querying, graph exploration, zooming and panning, overviewing, and viewing details on demand. The selection of this set of tasks for implementation was inspired by Schneiderman[74]. In the following sections, we will compare our implementations of these tasks to those described by Sundvall.

Querying

The querying functionality described by Sundvall involves the use of query templates. They provide the description of a sample template which "[r]eturn[s] all SNOMED CT concept nodes that have any description containing the word supplied in the search field"[38] The system provides advanced users with the power to create their own query templates.

In our implementation we also provide template based queries; however, our templates are quite different than those used by Sundvall. In our queries, we request a search string and then provide the user with a choice of search strategies. We then provide the visualization as a means of conceptualizing the search results. We have the intention, for future work, of allowing the user to interact with the visualization itself to refine the search, but we have not yet developed these features. Through our implementation is not feature complete, the extensible architecture of the software and the use of the factory pattern for these search strategies makes our implementation easily adaptable to user needs.

Exploration

The visual exploration functionality provided by Sundvall is based on a degree of interest metric. Given a node which is returned by a query, a "fetch" slider is inspected to determine the "degree of interest" to be fetched on the node in question. Given a degree of interest, the hierarchy or hierarchies in which the node is present are climbed by a number of links equal to the degree of interest. Each of the nodes explored in this way are displayed in the visualization.

Though we employed a different graph layout algorithm than the one used by Sundvall, our approach was not that different. We still use a node link graph to represent concepts in our implementation. We also provide a degree of interest filter, as did Sundvall. The principal difference between our implementations in this respect is that rather than fetching nodes based on the degree of interest as is described by Sundvall, we fetch all of the nodes available in our queries regardless of the degree of interest. We then rely on the degree of interest to govern which nodes we present in the visualization and which we do not. Though this results in a performance drawback in the initial query performed with our tool, it has a performance advantage over Sundvall's approach during the exploration phase which follows. The reader will recall that our system is designed for a distributed multiuser collaborative; consequently, it

was important to minimize the number of queries which were sent to the underlying terminology server and to keep network traffic to a minimum.

Managing Scope

Sundvall's tool also offers a semantic zooming feature which either displays or conceals node labels at a degree of interest distance from a highlighted node based on the position of a "label" slider. The tool does not however, provide a more meaningful semantic zoom which might expand a node by either displaying more of a focused node's relatives or display nodes for concepts which co-occur with the concept represented by the focused node. Sundvall did however describe the potential of using clustering. Clustering could be implemented by grouping collections of objects which had a low degree of interest into single aggregates. These aggregates could subsequently be expanded and explored when their contents once again became objects of interest.

Though we recognize these shortcomings in Sundvall's work, we have not addressed them in ours. We have however replicated much of the zooming functionality available in Sundvall's implementation. We elected not to reproduce the "label slider" functionality as we discovered that it was not as useful as might be expected. Given a root node like "SNOMED CT Vocabulary", and a leaf node in a hierarchy called "haemoglobin", it is difficult to understand the scope of the hierarchy. In particular it is difficult to gauge the granularity of the intermediate divisions in the hierarchy. The purpose of the label feature was to act as a details on demand mechanism that would help the user understand the context of their query results. Given the example above however, it becomes clear that the context is quickly lost when this approach is taken.

6.1.6 Visual Concept Explorer

The intent of the Visual Concept Explorer (VCE)[53] was to make the implicit knowledge which can be extracted from semantic networks explicitly available to the tool's user. VCE was initially designed to explore the Unified Medical Language Sysem Knowledge Sources (UMLSKS)[3] controlled terminology by enlisting the assistance of the PubMed search engine as an interface to the MEDLINE dataset. VCE provides two principle views of the semantic networks stored in the UMLSKS. The first is a node graph much like the one provided by TermViz which we will explore

in detail, and the second is a "concept map" which we will not investigate here as it is quite dissimilar to the approach we have taken to terminology exploration, and is also quite different from the approach taken by Sundvall with his TermViz tool.

The node graph layout provided by VCE, see fig 6.2, uses a layout algorithm called Pathfinder which could be considered less intuitive and less intelligible than the force directed layout used by TermViz. It also incorporates the use of colour to cluster nodes by semantic type. This feature would have been more powerful however, had the authors also provided the capacity to filter on semantic type, one of the pieces of classification information which is extracted from the UMLSKS by the VCE interface.

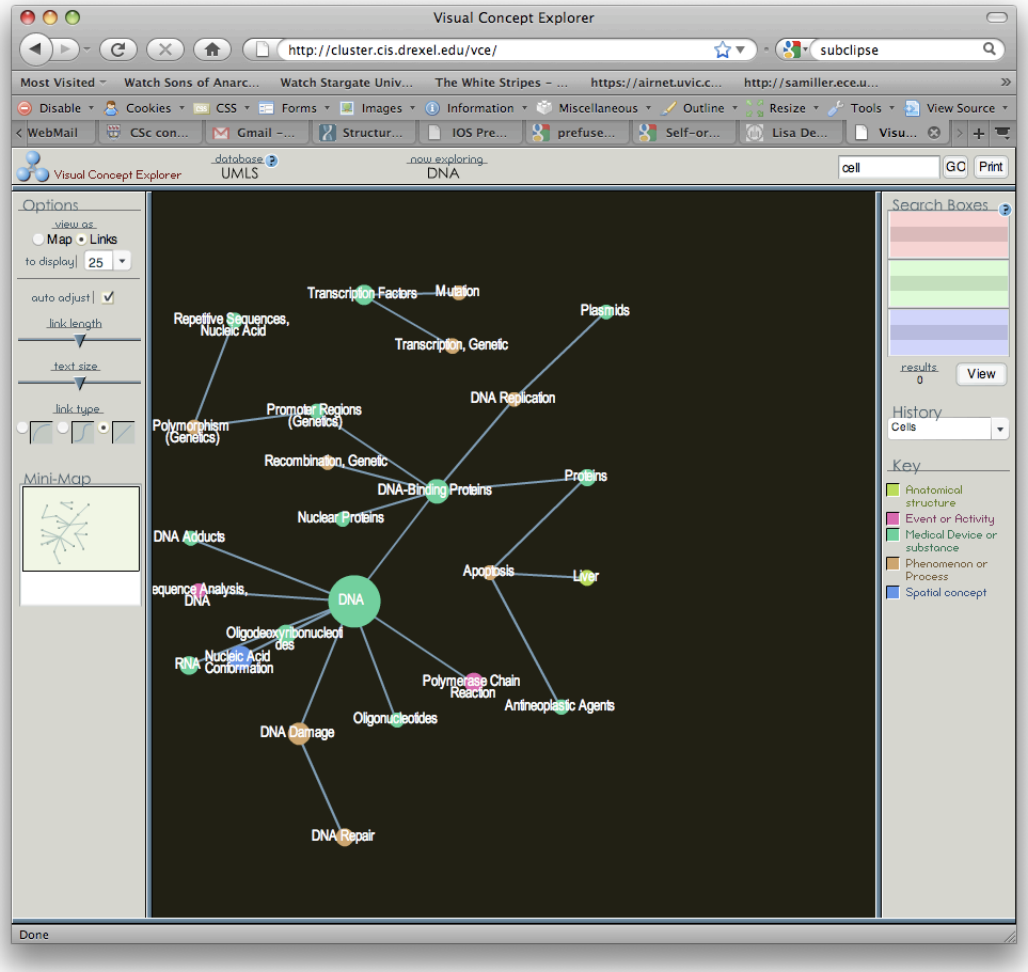


Figure 6.2: A screenshot of the node graph view from VCE[53]

Querying

One of the aspects of VCE which shows great promise but seems underdeveloped is the capacity to use the PubMed search engine directly from within the tool. VCE provides a logical query tool which posts queries to the PubMed search engine and which returns lists of links to articles relating to the search terms.

Exploration

VCE also provides a number of visual controls in its interface. Some are quite helpful while others are less well conceived. The interface for the node graph view provides a feature which will adjust the length of the links between nodes and another that will adjust the size of the displayed text. Together these features provide a dynamic control which will allow the user to mitigate occlusion, a significant problem with the display. Using a more sophisticated layout algorithm and pre-computing and avoiding instances of text occlusion seems a wiser solution, and one which we have elected to incorporate into our terminology server visualization implementation.

A second set of controls allows the user to select the style of links they wish VCE to display between nodes. Upon some experimentation however users are likely to converge on a single universal preference for the simple curved link as this is, according to Ware[83] (p.193), the most easily perceived linkage.

The tool also provides another control which allows users to stow panels which they are not currently using. This feature provides a dynamic workspace that allow the user to use as much space as he/she desires for the visualization. This is a useful feature that we would consider adding to our implementation should user demand be uncovered during a user study.

Scope of View

Unlike TermViz, VCE provides an explicit overview feature. The feature displays an unlabelled edge graph of the nodes present in the visualization. This feature also provides a refocus feature which centres the main viewport on the location selected in the overview display. The overview itself provides no consumable information as it contains no labels whatever. The principle issue which is highlighted here is that it is difficult to provide an overview of a complex semantic network when one has no specific task in mind, and when one doesn't have a clear grasp of the topography of the network in question. Supplementary to this reasoning, we can return to our

use case. We are attempting to provide a tool which allows a user to select semantic annotations for very specific concepts. An overview of an entire terminology is not particularly useful for this task. A focus on providing a feature rich exploration interface for localized subsets of the terminology is more likely to provide utility.

VCE took a different approach to details on demand than TermViz. In VCE this feature is activated using the same double click gesture, but instead of opening a new panel describing the selected term, the tool opens a new visualization which has as its root the selected term. This central root, as before is surrounded by co-occuring concepts. The consequence of this behaviour can be unsettling, but more importantly, there is no immediate way to return the viewport to its previous state. Of course, this can be affected by using the history tool; however, given that this is likely to be a frequently accessed feature, and that users will frequently wish to reverse this action, a mechanism for reverting the viewport to its previous state should be more accessible.

VCE appears to provide only two mechanisms to filter results. The first of course is the search functionality, and the second is the tab at the top left corner of Fig 6.2. This "to display" feature seems to imply that only the selected number of concepts will be displayed in the viewport. A reasonably thorough investigation of the interface and the paper describing the tool however indicate that the feature doesn't in fact work.

The color key based semantic clustering feature offered as a legend on the right hand side of both the Map and Node Graph views of VCE seems an ideal candidate for a filtering option, but this does not appear to be implemented. However, a brushing and linking feature relying on the same legend was implemented.

The problems of occlusion faced by VCE are not unique to this tool. A principal issue with the graph based layout approach to information visualization is occlusion. Though force based layouts like the one used in TermViz help to mitigate this issue, a more rigorous mathematical approach to graph drawing like the one taken by Dwyer[20] may provide more readable results. Dwyer's algorithm performs faster, and does a better job of mitigating occlusion than force based layouts.

Within our application we first elected to use a force based layout, but upon uncovering difficulties with scalability with the number of displayed nodes, we changed direction and elected instead to use a different node graph layout available in the same library. This yielded relatively easily interpretable results and so we did not investigate further. Though a more sophisticated approach to the graph drawing

task may improve the usability of our tool, it is expected that the benefit gained from these improvements would provide a low marginal return when compared against efforts which might be invested on other features.

The work of Haraguchi et al[34] on the other hand provides a much more compelling value proposition. This work focuses on drawing multiple related graphs together using perspective. This kind of approach could provide valuable cognitive support when performing tasks which are more complex and which require a deeper understanding of the interrelationships between concepts or terminologies.

6.1.7 Incorporation of Established Heuristics: The Persistence Engine

We have not thoroughly evaluated the persistence engine implementation in our architecture. We based the architecture on two principles which have been described in this thesis already. Firstly, we take advantage of the fact that the labels used in APSRs are frequently noun phrases. Secondly, we use the structure of these noun phrases to provide a semantically rich annotation set which is associated with each of the labels in the APSR templates.

6.1.8 Participatory Design: The Message Structure

We rely on a study by Kussaibi[51], which implemented participatory design for the construction of our systems output document structure. The architecture is designed in general to exchange biomedical knowledge. Our implementations however targeted breast cancer informatics specifically, and relied on anatomic pathology structured reports(APSRs) as the message structure for knowledge exchange. If the reader accepts that this is a reasonable choice of message structure, then our selection of APSR template is verified by its construction.

The APSR template we elected to use was the one presented in the Integrating the Healthcare Enterprise (IHE) APSR profile. This profile is a synthesis of the APSR templates used by the College of American Pathologists (CAP) and French Society for Pathology (SFP) which is deployed in the context of a Health Level Seven Clinical Document Architecture (HL7 CDA) compliant XML document and which was derived by the Delphi method[51]. The Delphi method, in short, is a process by which requirements can be gathered and synthesized from a large group

of domain experts using a consensus building process[54]. It falls under the scope of participatory design as the guiding principle of this methodology, as described by Muller[60], is the incorporation of users into the process of designing the tools they will later employ.

With respect specifically to our choice of terminology server, we acknowledge some weakness. Kussaibi indicated that the most common terminologies used in the annotation of APSRs were SNOMED CT, ICD-O-3 and in France, ADICAP. These discoveries were made during the same Delphi process as was used to synthesize the APSR templates. Of these three terminologies however, the UMLS Metathesaurus currently provides only SNOMED CT. That said, the process which is in place to introduce new terminological mappings to the UMLS Metathesaurus could be used, should the demand exist, to incorporate ICD-O-3 or ADICAP at some future date. Supplementary to this is the fact that we are designing a translational research solution. Our solution must satisfy users outside of the cancer informatics domain if it is to be a truly translational solution. The participatory design element of our selection of terminology server arises from the expert driven process which is used to create not only the controlled terminologies which are incorporated in the UMLS Metathesaurus (these are created in large part by Standards Development Organizations (SDOs)), but also to create the mappings between the incorporated controlled terminologies (the National Library of Medicine (NLM) uses processes which engage teams of experts in consensus building processes to accomplish this as was discussed in chapter 3).

6.1.9 Participatory Design: Mirth

Mirth is the export, transform and load engine (ETL) which we incorporate in our architecture implementations. It is designed and used by its health focused user community as an open source, though commercially driven software application. We chose the Mirth ETL engine because of its health focus and healthcare based user community. It also supports the Health Level Seven (HL7) standard which we chose as our message structure for transmission from our system. Though other ETL engines are available, the health focus of Mirth distinguishes it from other choices. Also, as we described in chapter 5, Bortis[10] and Brauer[11] advocate for the use of Mirth over other ETLs in health base applications for similar reasons.

6.2 Informatics Needs in Translational Research

Embi[21] describes the current needs in translational research by providing a number of focus areas of the domain including:

- developing effective workflows and tooling to support them
- providing human-computer interaction mechanisms within the computerized tooling which facilitate the user's objectives
- providing effective information capture and data flow in translational research which facilitate interdisciplinary knowledge exchange
- performing knowledge engineering and standards development in translational research which will facilitate the exchange of data, information and knowledge between disparate researchers and institutions
- providing data mining, data analysis and knowledge integration tools which facilitate communication between clinical and research information systems

We will discuss how our proposed architecture and how our individual implementations address each of these five issues.

6.2.1 Workflow

The workflow we propose for our architecture has its basis in the experience of the product managers at GenoLogics Life Sciences. This expected workflow is a generalization of the workflows used by GenoLogics' CRI clients and as such is implicitly verified to be an existing workflow. Our tool, as was described in chapter 5, is explicitly designed to satisfy this workflow.

6.2.2 Human Computer Interaction

Our secondary contribution, the visualization interface to the Unified Medical Language System (UMLS) Metathesaurus, was designed explicitly to provide an good human-computer interaction with our application. We have already discussed, in chapter 5, how the workflow established with GenoLogics is serviced by our visualization and its encapsulating clinical coding interface.

6.2.3 Information Capture and Data Flow

In order to provide effective information capture, we verify that transmitted information is sufficient to fulfill the requirements of recognized electronic medical record (EMR) standards which govern document contents like the "profiles" produced by Integrating the Healthcare Enterprise (IHE). To service the need for data flow, we include a communications backbone called Mirth in our architecture. Mirth is a health focused, EMR and software standards driven web application which relies on an enterprise service bus to ship channels of data between endpoints. By facilitating clinical coding, we allow users to record information in a way which is not only semantically rich, but is also computable.

6.2.4 Knowledge Engineering

By facilitating the consumption of existing standards like the Integrating the Healthcare Enterprise (IHE) profiles, we contribute to the development of knowledge engineering and standards development by users by facilitating the growth of the client base for these standards. As more systems are built using our architecture, the users of these systems will uncover weaknesses in these standards and will begin to participate in the communities which created them by engaging in the refinement processes which are used to improve the standards.

6.2.5 Data mining, Data Analysis and Knowledge Integration

To service the need for data mining, and analysis, our architecture includes a persistence layer which acts as a knowledge source by providing annotated data through two distinct mechanisms. The first, direct access, provides the facility to reconstruct compound nouns from the stored atoms as one approach to the data analysis and integration where client applications can formulate their own aggregations of knowledge units from the knowledge atoms which are stored in the persistence layer. The second, electronic medical record (EMR) transmission, takes a pre-coordinated approach which relies on EMR standards to govern the contents and structure of the knowledge which is communicated by the system. To facilitate data integration, we provide the persistence layer which stores semantic annotations. These annotations will facilitate the automated integration of data between disparate systems.

6.3 Retrospective

Though our research approach has lead to the exploration of a breadth of areas of difficulty in translational research, and specifically clinical research informatics (CRI), it suffers in many areas from a lack of depth. We could have spent more time evaluating the features and content of the UMLS Metathesaurus. We might also have spent much more time on developing a thorough feature complete visualization of this terminology server. Each of these choices however would have detracted from the breadth we have achieved with the process we have chosen.

Though we propose that much has been accomplished through our approach, the research we present here would benefit from a more in depth exploration of a number of components of our architecture. In particular, a more in depth exploration of the UMLS Metathesaurus and the queries which are likely to be presented by users could yield a much more effective clinical coding interface. In this same vein, developing a more feature complete visualization of the terminology server might yield a much more comfortable exploration experience for the clinical coder we attempt to service. Finally, our choice not to thoroughly evaluate the effectiveness of semantic annotation, but rather to focus on facilitating the process may at first seem to cast doubt on the value of the research we have performed. When one considers however, that much of the work we have done would be necessary to effectively generate the annotations which are to be used, it can be understood that steps similar to those we have taken must necessarily precede an exploration of the utility of the annotations which can now be generated.

Chapter 7

Future Work

Our progress in interoperability in translational research informatics, and the T2 block of clinical research informatics (CRI) more specifically, is only one more piece of a larger puzzle. In this chapter we will express the directions in which we feel our research should be explored next. We will first traverse our architecture from the information capture end to the knowledge transmission end. We will then discuss some more serious issues with our research and we will suggest methods by which they might be remedied.

7.1 The UMLS Metathesaurus

Though we are pleased in general with the feature set, semantic richness and breadth of the UMLS Metathesaurus, the tool has some significant weaknesses with respect to our chosen application. In addition to the inherent weaknesses of the tool, our application of the tool was not as complete as it might have been had we focused more on this aspect of our project.

7.1.1 The Hierarchy Table

Some of the inherent issues we perceive with the UMLS Metathesaurus include lingering performance issues which arise in part from its native relational schema. Rearranging the schema of this tool may tune the terminology server to our particular needs. One specific example of this type of issue is the mechanism the UMLS Metathesaurus hierarchy table uses to represent the hierarchy of its member concepts. The hierarchical relationship between concepts in represented by a delimited string which

indicates the path from the given concept to its parent. As a consequence of this approach to the representation of hierarchies, ancestors of a given concept can be rapidly and efficiently extracted from the database. On the other hand, fetching children requires the use of a SQL LIKE clause which is known to be slow. Even when these queries were restricted to "starts with" style clauses, the query time of the terminology server remained unacceptably long.

Initially one might expect that the application of a pre-order traversal algorithm might alleviate this issue. Though this idea holds some merit, we are not convinced that this approach will be effective. This is because the hierarchies represented in the UMLS Metathesaurus are not strict hierarchies. Some of the hierarchies are in fact cyclic graphs. Now one might resolve this issue by duplicating nodes in these cycles, but as the database is already large (the hierarchy table contains more than 12 million rows), if the cycles appear early in the hierarchy, this may produce a substantial increase in the number of rows in the hierarchy table. It is for these reasons that we feel an exploration various approaches to the representation of the UMLS Metathesaurus hierarchy table could produce valuable results.

7.1.2 Content

Though we are pleased in general with the breadth of controlled terminologies which are included in the UMLS Metathesaurus, the absence of two of the three terminologies which are central to clinical cancer informatics leads us to question the completeness of the included terminology set. To remind the reader again, the terminologies which are common in clinical cancer informatics are SNOMED CT which is included in the UMLS Metathesaurus, and ICD-O-3 which is not. A significant piece of future work which may increase the utility of the Metathesaurus would be the incorporation of additional terminologies. One way to approach this work would be to gather survey papers on the use of controlled terminologies in translational research and to collect from these a list of important terminologies which are not already included in the Metathesaurus and to subsequently map them in.

7.2 Exploring the Persistence Model

One of the elements we have not explored in great deal in our research is our persistence engine. Though visionaries like Tim Berners-Lee may foresee a future where

semantic annotation solved the interoperability woes of the world wide web and leads to machines which make intelligent, yet automated decisions, this vision has yet to come to pass. Finding problems which might be solved using these annotations specifically in translational research would of course be a necessary preliminary step, but once these problems have been identified, there may be great value in developing systems which attempt to leverage the annotations which are stored in systems like those which might be based on our architecture. It is only once the tools we have created are used that we can validate their utility.

7.3 Application to Other Medical Reports

One of the ways in which we chose to limit the scope of our research was to focus it specifically on cancer informatics, and more specifically on the use of anatomic pathology structured reports as a mechanism for knowledge exchange. Since the medical informatics standards used to verify the content and structure of these documents is formalized in Integrating the Healthcare Enterprise (IHE) profiles it would be relatively easy to expand the scope of our current solution to service other domains of the clinical and clinical research domains.

7.4 Conclusion

Saltz[72] states that "[g]iven that interoperability is a broad subject and spans many possible scenarios, the focus should be on practical usage scenarios. The verification and validation of usage scenarios and interoperability evaluation metrics should also be addressed." This has been our perspective from the nascent stages of our research. If our research is to be useful, it must be continually validated. A research approach like participatory design may help maintain this type of focus in future work. One of the primary weaknesses we perceive in our work so far is the lack of validation. Pushing the participatory design method much harder than we have so far seems a good way to resolve this weakness in future work. A principal way in which we expect that our research could be validated would be through user studies. Though Frank[24] and Beaulah[6] both warn of the challenges with user studies in translational research, and particularly in clinical research, the need to validation is paramount. Consequently, it is necessary to find ways to alleviate the difficulties which Frank and Beaulah identify in order to pursue research in this direction.

Appendix A

Additional Information

A.1 An Example in Oncology

McConnell [57] provides an insightful example of the "bench to bedside"/"bedside to bench" paradigm which illustrates how difficult it can be for experts to exchange knowledge. This example will be used to introduce the reader to the principal actors and artifacts in clinical cancer research. It will also be used to illustrate the ways in which tools based on our proposed architecture could facilitate the exchange of knowledge between experts in clinical cancer research. The example, as presented by McConnell, proceeds as follows:

> "[A] patient with a breast lump approach[es] an oncologist for treatment at a tertiary care facility. The patient has been told by her local physician that the tumor she has is not uncommon, but difficult to manage. The oncologist reviews the medical documents from the outside institution and recognizes an entity that is well-equipped to handle a lobular carcinoma of the breast. What is slightly odd is that this tumor has an intermediate nuclear grade and has been further designated as a 'pleomorphic lobular carcinoma'. The oncologist is familiar with the medical literature on the topic and recognizes that in certain circumstances such an entity may exhibit biologic behavior akin to an invasive ductal tumor (rather than that of a typical lobular). Paralleling the odd histologic finding is an additional finding that the tumor is negative for estrogen receptor (ER) expression, negative for progesterone receptor (PR) expression, but strongly positive for human epidermal growth factor receptor 2 (Her-

> 2/neu) overexpression. This profile is very uncharacteristic for a lobular carcinoma of the breast. Thinking that the tumor may have been misclassified, the oncologist has the pathology material reviewed again and re-runs the molecular studies to confirm the receptor studies reported from the outside institution. To complicate matters, the reviewing pathologist reclassifies the tumor as a typical lobular carcinoma (not pleomorphic), but the receptor study profiles (ER, PR) and Her-2/neu status remain unchanged. Now there is no diagnostic correlate for the receptor profile results. Lobular tumors (particularly typical ones) are nearly always positive for ER and PR receptor (and usually strongly so) and negative for Her-2/neu overexpression."

In McConnell's example, the care team, i.e. the local physician, the oncologist and the pathologist, intends on using the patient's phenotype as a causal diagnostic. Brock[13] states that a phenotype is "the observable characteristics of an organism". In layman's terms, a person's phenotype could be considered synonymous with their pattern of protein expression. The patient in the example's phenotype then, is described as their relative expression of the estrogen receptor(ER) protein, the progesterone receptor(PR) protein and the epidermal growth factor(Her2/neu) protein. This collection of clinical observations could be considered as a unit of knowledge. Though each observation on its own provides a small piece of information, when the observations are aggregated they provide a much more powerful insight into the patient's state of health.

The specific pieces of the patient's phenotype in which the care team takes interest are called biomarkers. In McConnell's example the specimen (tissue sample) did not show expression of ER or PR. In other words, it did not display either of these two biomarkers. These expression patterns are not indicative of a diseased state. However, the specimen did show Her2/neu overexpression, a third biomarker, which is indicative of a diseased state. An additional element to the Her2/neu overexpression pattern is the character of the specimen. This is described by the cellular composition of the specimen which includes the nuclear grade and the designation of pleomorphic or non-pleomorphic. These additional factors, in effect, intensify the Her2/neu overexpression biomarker. They are additional evidence of the diseased state of the specimen. The oncologist must consider all of these factors in assigning a diagnosis(lobular carcinoma or invasive ductal tumor) and in choosing a treat-

ment plan. Again here we see how individual observations though valuable, do not provide the same insight as their aggregation. We see how units of knowledge when aggregated become more than the sum of their parts.

With some of the terminology now explained, we can better address the communication needs of the actors in McConnel's example. We identify three opportunities for knowledge exchange in the scenario which could have been better exploited. The first was when the patient's primary physician completed medical records for the patient that were subsequently forwarded to the oncologist. The second opportunity was when the oncologist requested that the pathologist reevaluate the specimen. Finally, another opportunity presented itself when the pathologist responded to the oncologist's request for a reevaluation of the specimen.

It could be interpreted from the apparent confusion of the actors in this scenario that knowledge transfer has not occurred to the fullest extent and that their interactions might be facilitated by a better knowledge exchange system. The oncologist was clearly able to understand the information which was sent to him by the patient's primary physician, and so the communication between these two parties could certainly be considered information exchange. The information which was passed to the oncologist however seemed to be incomplete, or possibly incorrect. This might indicate a failure on the physician's part to impart his knowledge to the oncologist. On the other hand it might be an expression of the physician's lack of knowledge, the oncologist being the specialist. If a more complete information set, and one which had been validated had been passed to the oncologist by the primary physician, perhaps the oncologist would have been less confused.

A similar failure to exchange knowledge occurred between the oncologist and the pathologist. The pathologist understood what was requested of him when the oncologist requested the reevaluation of the specimen; however, from the oncologist's confusion at the pathologist's response, it seems likely that the oncologist failed to indicate to the pathologist why the specimen had been sent for reevaluation. Consequently, the pathologist was not able to convey sufficient knowledge to satisfy the oncologist with the reevaluation. Again in this situation, the application of a standards based knowledge exchange system may have alleviated some of the difficulty in the exchange of knowledge during this interaction.

Bibliography

[1] M Abdel-Wahab, R Rengan, B Curran, S Swerdloff, M Miettinen, C Field, S Ranjitkar, J Palta, and P Tripuraneni. Integrating the healthcare enterprise in radiation oncology plug and play-the future of radiation oncology? *International Journal of Radiation Oncology Biology Physics*, 76(2):333–336, 2010.

[2] S W Ambler. *The Object Primer: Agile Model-Driven Development with UML 2.0*. Cambridge University Press, New York, NY, USA, 2004.

[3] A Bangalore, K E Thorn, C Tilley, and L Peters. The umls knowledge source server: an object model for delivering umls data. *American Medical Informatics Association Symposium Proceedings*, pages 51–55, 2003.

[4] J Barnes and P Hut. A hierarchical o(n log n) force-calculation algorithm. *Nature*, 324(6096):446–449, 12 1986.

[5] N Barrett and J Weber-Jahnke. ehealth interoperability with web-based medical terminology services - a study of service requirements and maturity. *Journal of Emerging Technologies in Web Intelligence*, 1(2), 2009.

[6] S A Beaulah, M A Correll, R E J Munro, and J G Sheldon. Addressing informatics challenges in Translational Research with workflow technology. *Drug Discovery Today*, 13(17-18):771–777, 2008.

[7] T Benson. Hl7 version 2. In *Principles of Health Interoperability HL7 and SNOMED*, Health Informatics, pages 91–106. Springer London, 2010. 10.1007/978-1-84882-803-2_6.

[8] T Berners-Lee, J Hendler, and O Lassila. The semantic web. *Scientific American*, 284(5):34–43, May 2001.

[9] P Biron. http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/datatypes.html. [last accessed:Nov 2010].

[10] G Bortis. Experiences with mirth: an open source health care integration engine. In *ICSE '08: Proceedings of the 30th international conference on Software engineering*, pages 649–652, New York, NY, USA, 2008. Association of Computing Machinery.

[11] J Brauer. Mirth: Standards-based open source healthcare interface engine. http://www.osbr.ca/ojs/index.php/osbr/article/view/777/748. [last accessed: Nov 2010].

[12] T Bray. http://www.w3.org/TR/xml11/. [last accessed: Nov 2010].

[13] T D Brock. *Brock biology of microorganisms*. Prentice Hall/Pearson Education, Upper Saddle River, N.J., 2003.

[14] B Burke. Resteasy 2.0.0 released! http://bill.burkecentral.com/2010/07/19/resteasy-2-0-0-released/. [last accessed: Nov 2010].

[15] R Chen, G O Klein, E Sundvall, D Karlsson, and H Ahlfeldt. Archetype-based conversion of EHR content models: pilot experience with a regional EHR system. *BioMed Central Medical Informatics Decision Making*, 9:33, 2009.

[16] Mirth Corporation. Mirth. http://www.mirthcorp.com/community/overview. [last accessed: Dec 2010].

[17] F Curbera, M Duftler, R Khalaf, W Nagy, N Mukhi, and W Weerawarana. Unraveling the web services web: an introduction to soap, wsdl, and uddi. *Institute of Electrical and Electronics Engineers Internet Computing*, 6(2):86–93, 2002.

[18] C Daniel. http://www.linkedin.com/pub/christel-daniel/23/648/731. [last accessed: Nov 2010].

[19] 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*, 13:30–39, 2006.

[20] T Dwyer, K Marriott, and P J Stuckey. Fast node overlap removal. In *GD2005: Proceedings of the 13th International Symposium of Graph Drawing 2005*, volume

3843 of *Lecture Notes in Computer Science*, pages 153–164, Heidelberg, 2006. Springer.

[21] P J Embi and P R Payne. Clinical research informatics: challenges, opportunities and definition for an emerging domain. *Journal of the American Medical Informatics Association*, 16:316–327, 2009.

[22] T Erl. What is soa. http://whatissoa.com. [last accessed: Nov 2010].

[23] R T Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine, 2000. Chair-Taylor, Richard N.

[24] J Frank, B Rupprecht, and V Schmelmer. Knowledge-based assistance for the development of drugs. *Institute of Electrical and Electronics Engineers Expert: Intelligent Systems and Their Applications*, 12(1):40–48, 1997.

[25] C Friedman and S B Johnson. Natural language and text processing in biomedicine. In K J Hannah, M J Ball, E H Shortliffe, and J J Cimino, editors, *Biomedical Informatics*, Health Informatics, pages 312–343. Springer New York, 2006.

[26] S Gao. http://www.w3.org/TR/2009/WD-xmlschema11-1-20090130. [last accessed: Nov 2010].

[27] GenoLogics. http://genologics.com/solutions/biomedical-informatics/biospecimen-web-query_bioquest. [last accessed: Nov 2010].

[28] GenoLogics. http://genologics.com/solutions/biomedical-informatics/clinical-data-management_biochronicle. [last accessed: Nov 2010].

[29] GenoLogics. http://genologics.com/solutions/biomedical-informatics/biorepository-data-management_biovault. [last accessed: Nov 2010].

[30] HL7 Structured Documents Working Group. Structured documents working group wiki. http://wiki.hl7.org/index.php?title=Structured_Documents. [last accessed: Dec 2010].

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

[32] J Grudin and J Pruitt. Participatory design and product development: an infrastructure for engagement. In *Proceedings participatory design conference 2002*, pages 144–161, 2002.

[33] T Gschwandtner, K Kaiser, P Martini, and S Miksch. Easing semantically enriched information retrieval–an interactive semi-automatic annotation system for medical documents. *International Journal of Human-Computer Studies*, 68(6):370 – 385, 2010. Human-Computer Interaction for Medicine and Health care (HCI4MED): Towards making Information usable.

[34] K Haraguchi, S Hong, and H Hagamochi. Visual analysis of hierarchical data using 2.5d drawing with minimum occlusion, 2009.

[35] E Hatcher and O Gospodnetic. *Lucene in action*. Manning Publications Co., Greenwich, CT, USA, 2004.

[36] H He. What is service oriented architecture. http://www.xml.com/pub/a/ws/2003/09/30/soa.html, April 2010. [last accessed: Nov 2010].

[37] Corepoint Health. The hl7 evolution:comparing hl7 version 2 to version 3, including a history of version 2. http://www.corepointhealth.com/sites/default/files/whitepapers/hl7-v2-v3-evolution.pdf. [last accessed: Nov 2010].

[38] J Heer, S K Card, and J A Landay. prefuse: a toolkit for interactive information visualization. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 421–430, New York, NY, USA, 2005. Association of Computing Machinery.

[39] A J G Hey and A E Trefethen. The data deluge: an e-science perspective, 2003.

[40] I Hirtzlin, C Dubreuil, N Preaubert, J Duchier, B Jansen, J Simon, P L De Faria, A Perez-Lezaun, B Visser, G D Williams, and A Cambon-Thomsen. An empirical survey on biobanking of human genetic material and data in six EU countries. *European Journal of Human Genetics*, 11(6):475–488, 2003.

[41] Hubmed.org. Accessing the umlsks soap web service using php5. http://hublog.hubmed.org/archives/001591.html. [last accessed: Nov 2010].

[42] IHE. http://www.ihe.net/Technical_Framework/index.cfm#pathology. [last accessed: Nov 2010].

[43] IHE. Anatomic pathology structured reports (apsr). http://www.ihe.net/Technical_Framework/uploadIHE_PAT_Suppl_APSR_Rev1-0_PC_2010-07-30.pdf. [last accessed:Dec 2010].

[44] National Cancer Institute. http://cabig.cancer.gov/action/health20/soa/. [last accessed: Nov 2010].

[45] National Cancer Institute. cabig. https://cabig.nci.nih.gov/training/cabigessentials/player.html. [last accessed: Nov 2010].

[46] National Cancer Institute. Cancer biomedical informatics grid™. https://cabig.nci.nih.gov/. [last accessed: Nov 2010].

[47] National Cancer Institute. Frequently asked questions about the bronze certification program. https://cabig.nci.nih.gov/guidelines_documentation/bronze/faq.html. [last accessed: Nov 2010].

[48] G Jiang, J Pathak, and C G Chute. Formalizing icd coding rules using formal concept analysis. *Journal of Biomedical Informatics*, 42(3):504–517, 2009.

[49] K Kawamoto and D F Lobach. Proposal for fulfilling strategic objectives of the u.s. roadmap for national action on decision support through a service-oriented architecture leveraging hl7 services. *Journal of the American Medical Informatics Association*, 14(2):146–155, March 2007.

[50] G A Komatsoulis, D B Warzel, F W Hartel, K Shanbhag, R Chilukuri, G Fragoso, S de Coronado, D M Reeves, J B Hadfield, C Ludet, and P A Covitz. cacore version 3: Implementation of a model driven, service-oriented architecture for semantic interoperability. *Journal of Biomedical Informatics*, 41(1):106–123, 2008.

[51] H Kussaibi, F Macary, M Kennedy, D Booker, V Brodsky, T Schrader, M Garcia-Rojo, and C Daniel. HL7 CDA implementation guide for structured anatomic pathology reports methodology and tools. *Studies in Health Technology and Informatics*, 160(Pt 1):289–293, 2010.

[52] M Liebman. Translational research and biomedical informatics. *Methods in Molecular Biology*, 563:369–78, 2009.

[53] X Lin and D Zhang. Visualization of knowledge structures. *Proceedings of the 11th international Conference information Visualization*, IV:476–484, July 2007.

[54] H A Linstone. Futures research methodology-version 3.0. *Technological forecasting and social change*, 76(7):999 – 1001, 2009.

[55] H Luthria and F Rabhi. Organizational constraints to realizing business value from service oriented architectures: an empirical study of financial service institutions. In *ICSOC '08: Proceedings of the 6th International Conference on Service-Oriented Computing*, pages 256–270, Berlin, Heidelberg, 2008. Springer-Verlag.

[56] J A Maldonado, D Moner, D Tomas, C Angulo, M Robles, and J T Fernandez. Framework for clinical data standardization based on archetypes. *Studies in Health Technology and Informatics*, 129(Pt 1):454–8, 2007.

[57] P McConnell, R C Dash, R Chilukuri, R Pietrobon, K Johnson, R Annechiarico, and A J Cuticchia. The cancer translational research informatics platform. *BioMed Central Medical Informatics Decision Making*, 8:60, 2008.

[58] C J McDonald, S M Huff, J G Suico, G Hill, D Leavelle, R Aller, A Forrey, K Mercer, G DeMoor, J Hook, W Williams, J Case, and P Malone. Loinc, a universal standard for identifying laboratory observations: a 5-year update. *Clinical Chemistry*, 49:624–633, 2003.

[59] C N Mead. Data interchange standards in healthcare IT–computable semantic interoperability: now possible but still difficult, do we really need a better mousetrap? *Journal of Healthcare Information Management*, 20(1):71–78, 2006.

[60] M J Muller, D M Wildman, and E A White. Taxonomy of pd practices: A brief practitioner's guide. *Communications of the Association of Computing Machinery*, 36(6):26–28, 1993.

[61] D Glez-Pe na, F Diaz, J M Hernandez, J M Corchado, and F Fernandez-Riverola. geneCBR: a translational tool for multiple-microarray analysis and integrative information retrieval for aiding diagnosis in cancer research. *BioMed Central Bioinformatics*, 10:187, 2009.

[62] P M Nadkarni and R A Miller. Service-oriented architecture in medical software: promises and perils. *Journal of the American Medical Informatics Association*, 14(2):244–246, March 2007.

[63] B Nuseibeh and S Easterbrook. Requirements engineering: a roadmap. In *Proceedings of the Conference on The Future of Software Engineering*, International Conference on Software Engineering '00, pages 35–46, New York, NY, USA, 2000. Association of Computing Machinery.

[64] National Institute of Health. Translational research. http://nihroadmap.nih.gov/clinicalresearch/overview-translational.asp. [last accessed: Nov 2010].

[65] National Library of Medicine. http://www.nlm.nih.gov/pubs/factsheets/umlsmeta.html. [last accessed: Nov 2010].

[66] National Library of Medicine. http://www.nlm.nih.gov/research/umls/knowledge_sources/metathesaurus/mapping_projects/index.html. [last accessed: Nov 2010].

[67] National Library of Medicine. Metamap. http://metamap.nlm.nih.gov/. [last accessed: Nov 2010].

[68] S D Patterson. Data analysis-the achilles heel of proteomics. *Nature Biotechnology*, 21(3):221–222, 2003.

[69] J Quinn. An HL7 (Health Level Seven) overview. *Journal of the American Health Information Management Association*, 70(7):32–34, 1999.

[70] I Rojas, E Ratsch, J Saric, and U Wittig. Notes on the use of ontologies in the biochemical domain. *In Silico Biology*, 4(1):89–96, 2004.

[71] A Ruttenberg, T Clark, W Bug, M Samwald, O Bodenreider, H Chen, D Doherty, K Forsberg, Y Gao, V Kashyap, J Kinoshita, J Luciano, M S Marshall, C Ogbuji, J Rees, S Stephens, G T Wong, E Wu, D Zaccagnini, T Hongsermeier, E Neumann, I Herman, and K Cheung. Advancing translational research with the semantic web. *BioMed Central Bioinformatics*, 8 Suppl 3(NIL):S2, 2007.

[72] J Saltz, S Hastings, S Langella, S Oster, T Kurc, P Payne, R Ferreira, B Plale, C Goble, D Ervin, A Sharma, T Pan, J Permar, P Brezany, F Siebenlist, R Madduri, I Foster, K Shanbhag, C Mead, and N C Hong. A roadmap for caGrid, an enterprise Grid architecture for biomedical research. *Studies in Health Technology and Informatics*, 138:224–237, 2008.

[73] G Schadow, C N Mead, and D Mead Walker. The HL7 reference information model under scrutiny. *Studies in Health Technology and Informatics*, 124:151–156, 2006.

[74] B Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. *Proceedings of the 1996 IEEE Symposium on Visual Languages*, pages 336–343, September 1996.

[75] UMLS staff. Metamorphosys. http://www.nlm.nih.gov/pubs/factsheets/umlsmetamorph.html. [last accessed: Dec 2010].

[76] E Sundvall, M Nystr´om, H Petersson, and H Ahlfeldt. Interactive visualization and navigation of complex terminology systems, exemplified by snomed ct. *Studies on Health Technology and Informatics*, 124:851–856, 2006.

[77] H Thompson.]http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/structures.html. [last accessed: Nov 2010].

[78] E R Tufte. *Envisioning information*. Graphics Press, Cheshire, Conneticut, 3rd printing with revisions edition, 1992.

[79] G Van Tulder. Storing hierarchical data in a database. http://articles.sitepoint.com/article/hierarchical-data-database/2. [last accessed: Nov 2010].

[80] Dalhousie University. Getting started with hl7 v3. http://healthinfo.med.dal.ca/HL7Intro/gettingstarted.html. [last accessed: Nov 2010].

[81] F van Ham and A Perer. Search, show context, expand on demand: supporting large graph exploration with degree-of-interest. *Institute of Electrical and Electronics Engineers Transactions on Visualization and Computer Graphics*, 15:953–960, 2009.

[82] R van Rees. Clarity in the usage of the terms ontology, taxonomy and classification. *Proceedings of the CIB W78's 20th International Conference on Construction IT, Construction IT Bridging the Distance*, Conseil International du Batiment Report 284:432–440, 2003.

[83] C Ware. *Information visualization: perception for design*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[84] C Weng, J H Gennari, and D B Fridsma. Methodological review: user-centered semantic harmonization: a case study. *Journal of Biomedical Informatics*, 40(3):353–364, 2007.

[85] Wikipedia. http://en.wikipedia.org/wiki/XML_Schema_Language_Comparison. [last accessed: Nov 2010].

[86] S H Woolf. The meaning of translational research and why it matters. *Journal of the American Medical Association*, 299(2):211–213, 2008.

[87] X Xu, L Zhu, Y Liu, and M Staples. Resource-oriented architecture for business processes. In *APSEC '08: Proceedings of the 2008 15th Asia-Pacific Software Engineering Conference*, pages 395–402, Washington, DC, USA, 2008. Institute of Electrical and Electronics Engineers Computer Society.