Model Driven Engineering for Large-Scale Clinical Research

Jim Davies, Jeremy Gibbons, David Milward, Seyyed Shah, Monika Solanki, and James Welch
Department of Computer Science, University of Oxford
Email: Firstname.Lastname@cs.ox.ac.uk

***Abstract*—Scientific progress in medicine is increasingly dependent upon the large-scale integration of data from a range of sources. This data is acquired in different contexts, held in different systems, and subject to different processes and constraints. The value of the integrated data relies upon common data points having a consistent interpretation in each case; to achieve this at scale requires appropriate informatics support. This paper explains how a model-driven approach to software engineering and data management, in which software artefacts are generated automatically from data models, can be used to achieve this. It introduces a simple data modelling language, consistent with standard object modelling notations, together with a set of tools for model creation, maintenance, and deployment. It reports upon the application of this approach in the provision of informatics support for two large-scale clinical research initiatives.**

I. Introduction

Data Quality and Data Integration are two major problems which are becoming critical for health care providers everywhere. Research data is held on different systems, in different locations and the data itself is heterogeneous, collected without a common syntax or semantics. Data quality within the enterprise is dependent on successfully managing heterogeneous data sources, and there are variety of approaches to solve the problems associated with data integration. In this paper we show that the successful application of metadata registries combined with model driven engineering principles has been demonstrated to have a significant impact on both data quality and data interoperability within two major healthcare projects within the UK's National Health Service (NHS).

The UK's National Institute for Health Research (NIHR [1]) has sponsored the Health Informatics Collaborative (NHIC) which is backing a cross-trust programme involving 5 key NHS hospital trusts in the UK in order to set up a flexible and responsive governance framework, whereby research outcomes can rapidly be exploited by the NHS community. The work is currently limited to 5 clinical areas, but is expected in time to be extended. One of the aims of the programme is to develop tools and services for research, so that researchers and clinicians can have access to a wider cross Biomedical Research Centre (BRC) dataset. The programme has been working on developing a federated metadata registry, based on the ISO standard for metadata registries ISO11179[4], for use as a basis for data interoperability for research data from clinical trials but also with a view to integrating this capability with Electronic Patient Records (EPR) datastores within the trusts.

Genomics England(GEL [2]) is a company owned by the Department of Health and was set up to deliver the 100,000 Genomes Project. This flagship project will sequence 100,000 whole genomes from NHS patients by 2017, the project has four main aims: a)to bring benefit to patients, b) to create an ethical and transparent programme based on consent, c) to enable new scientific discovery and medical insights, d) to kickstart the development of a UK genomics industry. The project is focusing on patients with rare diseases, and their families, as well as patients with common cancers.

A. Background

Interoperability is major problem in electronic information systems in 2 (arguably) orthogonal respects, firstly that domain concepts and terms in multiple systems are used to mean slightly different things, and secondly that the same concept meaning in different systems very often are given a different representation, so that speed may be represented by an Integer value in one systems and by a Double in another system.

Metadata Registries are needed in organizations to ensure data consistency. Metadata Registries are capable of analysing, administering and classifying metadata and very often in practise they also function as repositories for metadata, storing the schema and blueprints for data types and structures. The ideas and concepts that are stored in data systems form the basis of executable software systems, however many of the details are peculiar to particular disciplines.

Metadata Registries are normally found in Data Warehouses and Enterprise data mining systems, where vast quantities of data need to be administered and managed. It is likely that metadata registries will become more common as it becomes more and more necessary to deal with the recent internet-driven data explosion, especially as much of this data is unstructured, and can only be processed by relatively inefficient techniques.

Metadata Registries contain the ability to examine both how a data element is represented as well as to record it means, and it is this relationship that is embodied in the International Standard 11179, which we will examine in the next section. There are other standards which are concerned with metadata registries namely ISO15000 (part 3 and 4) which relates to ebXML, however they are more concerned with storing and

accessing metadata rather than classifying it and relating the semantic and representational aspects of that metadata.

B. Related Work

State-of-the-art is the research electronic data capture (REDCap) [**?**] web-based application to support translational research. The tool support data capture for research studies providing an online interface for data entry, audit trails for tracking data, export to common statistical packages and data import from external sources. Like the Model Catalogue, the REDCap system focuses on the modes of clinical metadata. However, REDCap differs from the current Model Catalogue in two important aspects. Firstly, REDCap is a closed system, any data must be imported into and stored within the REDCap database, whereas the Model Catalogue aims to provide platform to create software for use in clinical environments. Secondly, the meta-data management (creation, revision, sharing) in REDCap is considered a specialist task, requiring modelling experts to initialise a REDCap installation per-study. Whereas in the Model Catalogue, clinical users have capability to adapt and modify the metadata as needed. Effectively, the Model Catalogue follows the same metadata workflow as REDCap, but without the need for modelling experts to adapt or change the meta data models. Because the Model Catalogue works at the meta-data level, it can be used to share similar models between organisations and to derive software artefacts directly from user created metadata models.

Data warehousing [**?**] provides a framework for reporting and analysis tasks across disparate enterprise data systems, for decision support. In data warehousing meta data is modelled to extract information from business systems into a centralised data warehouse. The warehouse is used to create reports and analyse business performance. Data warehouse models can be arranged in normalised form, following the relational approach [**?**], or 'dimensional form [**?**] as quantifiable *facts* and *dimensions* which denote the context of facts. Data warehousing relies on fixed data models and structures, which is in contrast to the more general approach taken in ISO11179, where models are expected to change over time.

The Common Warehouse Metamodel (CWM) [5] from the Object Management Group is a UML based framework to enable data warehousing in practice. However, as with data warehousing, CWM is focused on write-once models, and for working with rigidly structured data. The models and data warehouse structure in CWM are not intended to change after creation and data is copied into the data warehouse from separate systems. The standard consists of 22 parts and the core meta model has overlap with the the *concept* and *value* elements of the ISO11179 meta model.

Several efforts have addressed the representation of ISO11179 as ontological models for enabling data integration across metadata registries (MDRs). In [6] the authors describe a federated semantic metadata registry framework where CDE (Common Data Elements) are exposed as LOD resources. CDEs are described in RDF, can be queried and interlinked with CDEs in other registries using SKOS. An ISO1179 ontology has been defined as part of the framework and the Semantic MDR has been implemented using the Jena framework. In [3] the authors present the Clinical Data Element Ontology (CDEO) for unified indexing and retrieval of elements across MDRs. Concepts in CDEO have been organised and represented using SKOS. In [7] the authors present case studies for representing HL7 Detailed Clinical Models (DCMs) and the ISO11179 model in OWL. A combination of UML diagrams and Excel spreadsheets were used to extract the metamodels for 14 HL7 DCM constructs. A critical limitation of this approach is that the transformation from metamodels to their ontological representation in OWL is based on a manual encoding.

Ontology repositories can be considered analogous to model catalogues they provide the infrastructure for storing, interlinking, querying and visualising

C. Objectives

This paper is an experience paper showing the results obtained from applying Model Driven Engineering principles to data interoperability problems in two instances in the medical domain. In both cases the aims of the clinicians were similar, namely to integrate existing clinical trials data and to ensure that future artefacts, such as the forms used to capture datasets resulted in high data quality.

To measure data quality we looked at how the data forms used in capturing information conformed to the standards relating to the datasets used, in particular was the terminology uniform, did captured data elements mean the same thing, and did equivalent data elements capture the same values across different systems.

The paper is split into 5 sections, this section introduces the problem and gives an overview of the work, the next section looks at the core language or model used, section 3 deals with the implem entation details, section 4 details the experience gained, in essence the results achieved and section 5 concludes.

II. The Models Language and Architecture

In the past Data Dictionaries were used to store details of database record structure or application data structure on a local per application basis, a metadata registry provides a similar capability but on a system or organisation-wide basis. It also provides features that are commonly included in a *Thesaurus*, a *Taxonomy*, and an *Ontology* . These features include the ability to classify terms in relation to one another, record relationships such as synonyms, and classify hierarchical relationships. Ontologies have proved effective in matching what we have defined as the first problem, that is how domain concepts can be matched, however they are quite unwieldy to use when tackling the second problem of how to match and manage different representations.

Metadata Registries are normally found in Data Warehouses and Enterprise data mining systems, where vast quantities of data need to be administered and managed. It is likely that metadata registries will become more common as it

becomes more and more necessary to deal with the recent internet-driven data explosion, especially as much of this data is unstructured, and can only be processed by relatively inefficient techniques.

This section takes key concepts from the ISO11179 standard and explores ways in which these concepts can be integrated with concepts and ideas embodied in model-driven engineering principles in order to design a metadata registry, which will both serve the practical purpose of allowing datasets to be matched, compared and managed, and allow for the development of a unified treatment of data in heterogeneous systems. The first part looks at the ISO standard and the ideas behind MOF, the second takes these ideas and uses them as the foundation for a specification of a metadata registry. The third section looks at how the metadata registry can be used to provide basic services for data-oriented model driven programming. In later chapters the practical and theoretical problems are examined in detailed to see in what ways full automated semantic interoperability can be achieved.

A. ISO11179

ISO11179 is the international standard relating to metadata and in particular metadata registries, and although there are a few other related standards which have informed the specification of the toolkit we describe ISO11179 provides the most exhaustive description of a metadata registry. It is therefore a key reference in this specification.

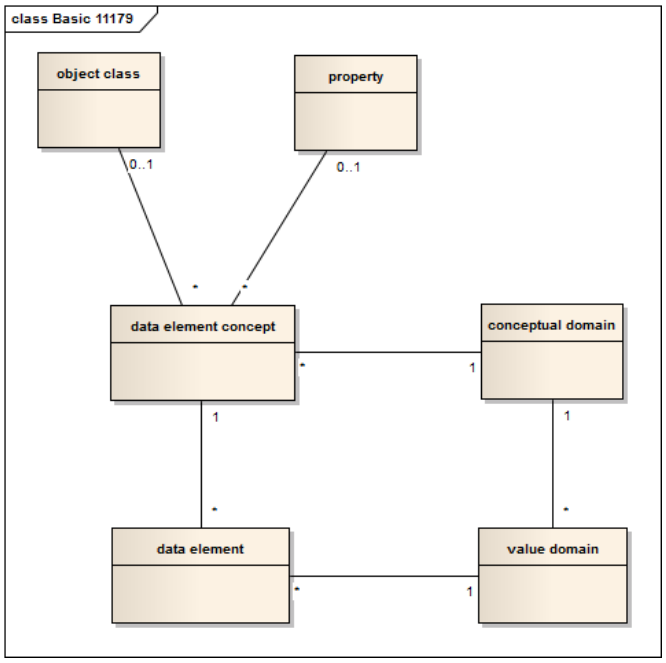

Fig. 1. Core model for ISO11179 Metadata Registry

The core ideas from the ISO11179 standard can be extracted to give us the notion of a *data element concept*, *a data element*, *a value domain*, and a *conceptual domain*. The standard currently confines itself to the detailed level of concepts and data elements and has no notion of collections of data elements or data element concepts, but instead attaches two attributes: an *object class* and a *property* to each DEC, these attributes allow DEC's to be aggregated or classified. This core model of the ISO11179 is illustrated in figure 1.

B. Metadata Registries - An Overview of Standards and Ideas

C. ISO11179

ISO11179 is the international standard relating to metadata and in particular metadata registries, and although there are a few other related standards which I have examined in the course of specifying a metadata registry ISO11179 provides the most exhaustive description of a metadata registry. It is therefore a key reference in this specification. If we abstract the core ideas from the ISO11179 standard we have the notion of a *data element concept*, *a data element*, *a value domain*, and a *conceptual domain*. There is a four-way set of relationships between CDs, VDs, CDEs, and DECs which draws a distinctions between the conceptual or semantic *level* and the representational *level* in which these entities live, illustrated in figure2.

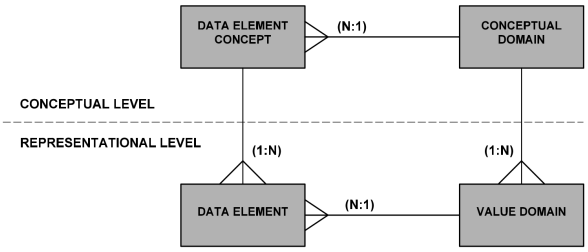

Fig. 2. Overview model for a Metadata Registry

We can get some more hints as to the theory behind these guidelines in a related ISO standard *ISO 20943-1: the standard for consistency in meta-data registries)* which states that the data element concept may relate several data elements that record data about that concept with different representations, implying that a single data element concept may have many value domains. It would appear therefore that what is intended is that a common data element in ISO11179 is really a relationship, or mapping between the set of data element concepts and the set of value domains.

For example an Integer data-type in a programming language may be used to represent inches in a measurement program, it may also be used to count vehicles in a logistics application. A data element is said to be comprised of a data element concept(DEC) which is its meaning and a value domain(VD) which is its representation.

The simple mapping between DEC's and VD's is perhaps better replaced with a functional relationship, and in section 2 we will explore the idea of a functional relationship between the two, its advantages and disadvantages. For now we will just try and describe the MDR according to the ISO specification.

Entity	ISO Definition	ISO11179 Implementation Guidelines
Data Element Concept(DEC)	An idea that can be represented in the form of a data element, described independently of any particular representation.	A concept that can be represented in the form of a Data Element, described independently of any particular representation.
Common Data Element(CDE)	A unit of data for which the definition, identification, representation, and permissible values are specified by means of a set of attributes.	A unit of data for which the definition, identification, representation and Permissible Values are specified by means of a set of attributes.
Value Domain (VD)	The description of a value meaning.	A description of a Value Meaning.
Conceptual Domain (CD)	A set of valid value meanings, which may be enumerated or expressed via a description.	A set of valid Value Meanings.

If we consider a single model of a system then we are putting constraints on the extent of the VD's, or if a set of VD's is considered to be a conceptual domain then we are constraining the model to a single CD. By doing this we can model the system using functional relationships, so make a total function relating the model DEC's to the VD's *in a particular conceptual domain*, since all the DEC's will have a mapping to a VD. It is not an injective function because of course more than one DEC can use the same representation or VD, and it is not surjective because there can be VD's in that particular Conceptual Domain which are not used in the model. It may be that a more useful model can be built by using the term Conceptual Domain to mean just those elements related to a *group* of data elements, in which case we could specify a surjective relationship. The ISO standard does not specify anything about *groups* of data elements, so it would be perfectly reasonable to introduce just such a constraint.

Consider the definition of a *Conceptual Domain*:

> A conceptual domain is a set of value meanings. The intention of a conceptual domain is to detail the model's value meanings. Many value domains may be in the extension of the same conceptual domain, but a value domain is associated with one conceptual domain. Conceptual domains may have relationships with other conceptual domains, so it is possible to create a concept system of conceptual domains. Value domains may have relationships with other value domains, which provide the framework to capture the structure of sets of related value domains and their associated concepts.

Conceptual domains comprise sets of value domains, and enable value domains themselves to be "re-used" between difference Data Elements.

D. A Metadata Language and Abstract Architecture

The ISO11179 specification illustrates different aspects of the standard using UML class diagrams, we have used these to inform the development of a very simple domain specific language, which we are calling **Elm DSL** based on the standard. The main differences are that we have added in containers to handle data element collections, calling these *Classes* and to handle collections of these *Classes* which we have called *Models*.

A simplified overview model, without attributes and methods, showing the Ecore model for the Elm DSL is shown in Figure 3.

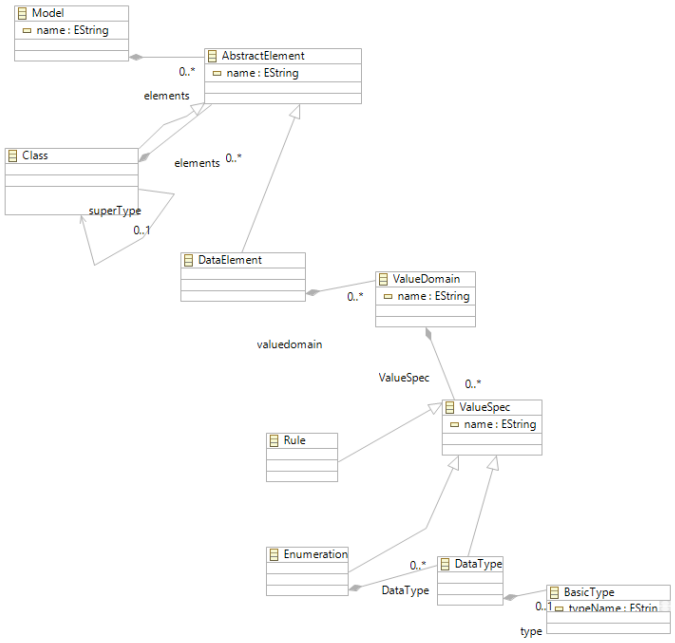

Fig. 3. Overview Elm DSL for Model Catalogue

1) Model: A model is a grouping or containment entity which groups a set of *Classes* together. Models can be thought of as datasets, or even database schemas, very often in the medical domain they are defined either by XML Schema definition files, or by equivalent schemas written in Excel. Models are collections of either *Data Elements* or *Classes*. This aspect is captured in the *Emfatic* representation by the *AbstractElement* which can be either a *Data Element* or a *Class* There is no real notion of composition or multiplicity, a instance of a Model can contain an instance of a Data Element or not as required by the instance. Models are named, have a description and have a version identity.

2) Class: A Class is a grouping or collection of *attributes* which can be data elements or classes, the attributes are currently *mandatory*, so that class with 5 attributes must have those 5 attributes instantiated in an instance for it to be considered of that class. Classes represent *Concepts*, (***This is not true present****)and can be *Generalized* into a hierarchy(**but we might get it implemented before the paper comes out**). The idea of a *Data Element Concept* was omitted from the initial prototype, although it has been added to later versions of the language.

E. Data Elements

Data Elements can also represent *Concepts* and are by their nature *atomic*. Each data element is related to a value domain on a one-to-one basis, and the relationship is a two-way relationship.

F. Value Domain

A Value Domain is the domain in which the data element is represented, it can consist of one or more *ValueSpecs*.

G. ValueSpec

A *ValueSpec* can be a simple datatype, an enumeration of datatypes, or a rule - such as a regular expression - which defines the way in which a series of characters is formed into a string attribute.

III. Implementation

Groovy was chosen as a language for implementation for two reasons, firstly it has a very efficient web framework called Grails built on the Spring framework, which is not only proven to be very robust and scalable, but is also relatively easy to implement and so enables quick agile development cycles. Previous implementations using Java/Spring and Java/Roo have proved very time-consuming to experiment with, whereas Grails has proven to be more flexible and easier to experiment with. The Groovy language also offers both some functional capability, as well as dynamic meta-programming capability. Domain specific languages (DSLs) can be easily built on this framework, and this capability offers some scope to build a DSL based on the DataMOF DSL specification and meta-model expanded in the previous section.

The Model Catalogue design discussed here has been implemented by building a core Models Catalogue using Groovy within the Grails framework. Prototype implementation was carried out using the EMF/ECore framework, however the Grails framework offered a robust web-based alternative, with enough Model-Driven capability to test out the core ideas expressed here using a maintainable java-based stack that could be worked on by mainstream developers without a detailed knowledge of the Model Driven Engineering.

The front end user interface was implemented using a combination of HTML with Javascript and CSS, the principal framework used being Angular JS. Communication with the client was carried out using a REST controller, enabling a variety of clients potentially to link up with the Model Catalogue.

GORM was used as a persistence mechanism, with a MySQL relational database as storage, although different GORM adapters made it possible to attach NO SQL datastores such as Neo4J and MongoDB. The full architectural stack is shown in figure4

The Grails/GORM framework enabled the Ecore model to generate the basic Grails Domain model, and from that a Groovy DSL was built to handle transformations internally between different representational languages such as XML and excel. A series of importers was built, although due to the simplicity of the DataMOF DSL it was not possible to build exporters for every format. The internal domain model used a basic *Catalogue Element* which was able to link elements via the *relationship* and *relationshipType* classes. Any catalogue element is able to be related to any other catalogue element through a relationship class, this relationship is constrained by the relationshipType object which can prevent different catalogue element types being related, so that a Model cannot directly be related to a say a Datatype Enumeration. Relationship types can be added to the Model dynamically, so that even though the relationship between a Model and Datatype enumeration is prevented initially, a new type could be introduced by an administrator or super user to add in that relationship. Some of the main relationships that are currently modelled in the *Models Catalogue* are as follows:

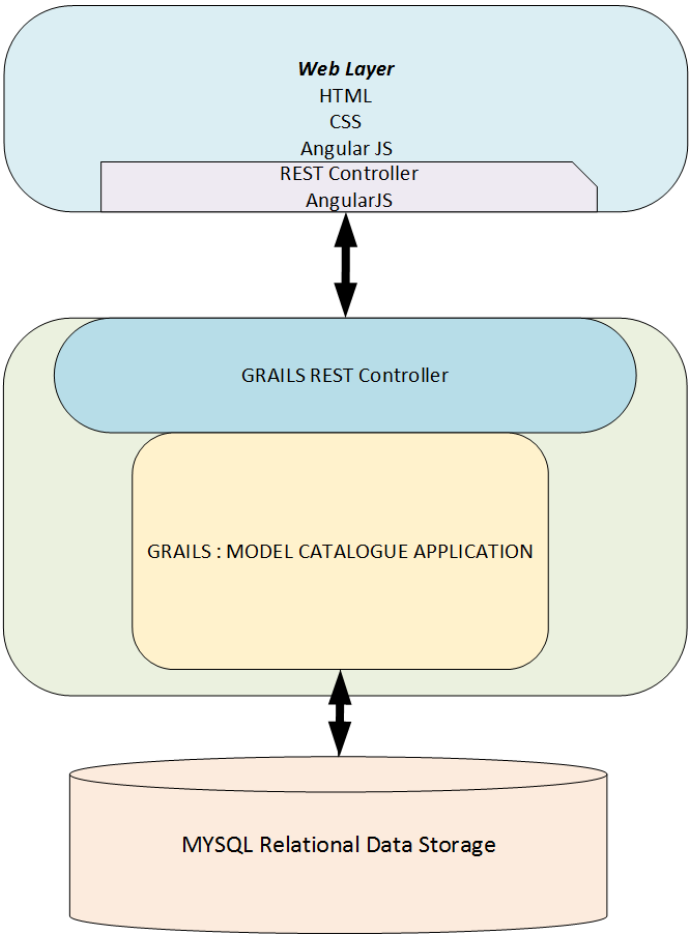

Fig. 4. Overview Architecture

Source	Relationship	Destination
Model	containment	DataElement
Model	containment	Class
Model	hierarchical	Model
DataElement	supersession	DataElement

The Core architecture can seen in figure4

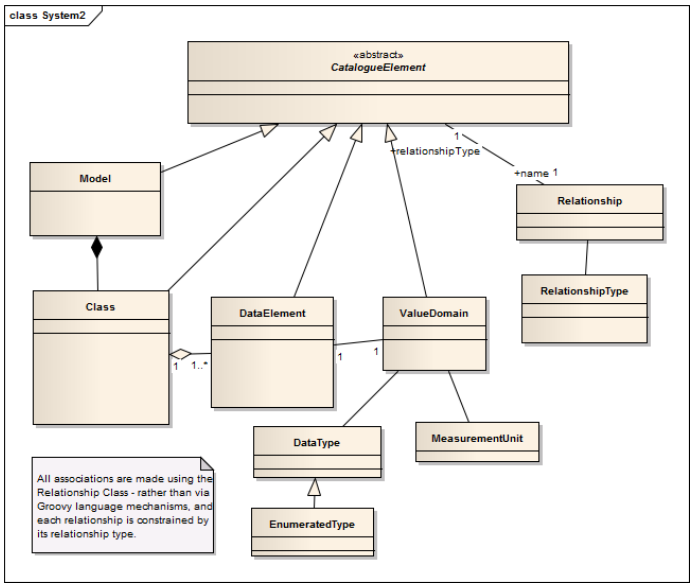

Fig. 5. Overview Architecture

A. Modelling Overview

IV. Experience

A. NHIC

B. Genomics England - reuse of HIC Models

C. Genomics England - deployment

V. Conclusion

A. Ongoing work

References

[1] National institute for health research.

[2] Genomics england limited, 2014.

[3] Paul A Harris, Robert Taylor, Robert Thielke, Jonathon Payne, Nathaniel Gonzalez, and Jose G Conde. Research electronic data capture (redcap)a metadata-driven methodology and workflow process for providing translational research informatics support. *Journal of biomedical informatics*, 42(2):377–381, 2009.

[4] W. H. Inmon. *Building the Data Warehouse*. QED Information Sciences, Inc., Wellesley, MA, USA, 1992.

[5] S. Jeong, H. H. Kim, Y. R. Park, and J. H. Kim. Clinical Data Element Ontology for Unified Indexing and Retrieval of Data Elements across Multiple Metadata Registries. *Healthc Inform Res*, 20(4):295–303, Oct 2014.

[6] Ralph Kimball and Margy Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, Inc., New York, NY, USA, 2nd edition, 2002.

[7] International Standards Organisation. International standard for metadata registries. http://metadata-standards.org/11179/.

[8] John Poole, Dan Chang, Douglas Tolbert, and David Mellor. *Common Warehouse Metamodel developer's guide*, volume 24. John Wiley & Sons, 2003.

[9] A. Anil Sinaci and Gokce B. Laleci Erturkmen. A federated semantic metadata registry framework for enabling interoperability across clinical research and care domains. *Journal of Biomedical Informatics*, 46(5):784 – 794, 2013.

[10] C. Tao, G. Jiang, W. Wei, H. R. Solbrig, and C. G. Chute. Towards Semantic-Web Based Representation and Harmonization of Standard Meta-data Models for Clinical Studies. *AMIA Jt Summits Transl Sci Proc*, 2011:59–63, 2011.