Domain Specific Modelling for Clinical Research

Jim Davies, Jeremy Gibbons, Adam Milward, David Milward,
Seyyed Shah, Monika Solanki, and James Welch

Department of Computer Science, University of Oxford
firstname.lastname@cs.ox.ac.uk

Abstract

The value of integrated data relies upon common data points having an accessible, consistent interpretation; 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, and models are used as metadata, can 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.

Categories and Subject Descriptors D.2.12 [*Interoperability*]: D.3.3 Programming Languages: Specialized application languages

Keywords domain-specific modeling languages, interoperability, mapping, code generation, model transformation, meta-modeling

1. Introduction

To obtain the evidence required to support the development and introduction of a new treatment, or a new diagnostic tool, we need to consider the results of detailed, clinical observations of a large number of individuals. These observations need to be made, and the results recorded, in a consistent fashion.

The usual way of achieving this is through prior agreement upon a study protocol: a detailed specification of the information required, and an account of the proposed analysis. Clinical staff receive training and support to ensure that data collection proceeds according to the protocol, and data is recorded using a single set of 'case report forms'.

There are two problems with this. The first is the cost of manual data collection, of additional training, and of bespoke systems development. The second is a lack of any guarantee of consistency across studies: even where two studies require what is essentially the same information, differences in specifications may mean that the data collected is incompatible.

For example, in breast cancer, 'histological type of tumour' is an common piece of information. However, one study might record this against an enumeration such as

in-situ ductal only | tubular/ cribriform |
ductal grade unknown | mixed

whereas another might offer a choice of

invasive ductal or no specific type | tubular |
mucinous | invasive cribriform

Even if we assume that the clinical staff have the same interpretation of these technical terms, the resulting data cannot be combined without additional effort and some loss of information.

We can reduce the cost of new studies by re-using data already collected: in earlier studies, or in the clinical information systems used to support patient care. We can reduce the cost of bespoke systems development by generating case report forms, queries, databases, and workflows from the detailed specifications in the protocol. We can increase consistency across studies, and faciliate re-use of the data, by coordinating the design of specifications.

To do this in practice, and at scale, requires effective, domain-specific modelling. We need models that describe study data: the detailed data specifications mentioned above. We need models that describe the relevant contents of clinical information systems. We need models that describe forms, queries, databases, and workflows used for data collection, transmission, and integration.

We need also a mechanism for relating the declarations of individual data items in different models. We need to be able to record the fact that two items, declared in different models, represent the same information: that any value assigned would have the same interpretation in both contexts. This is precisely what is needed if we are to re-use data from different systems, or combine data from different studies.

In this paper, we describe the notion of a *model catalogue*: an application that stores and presents models, links data declarations, and supports the generation of artifacts such as case report forms and data schemas. We introduce the domain-specific language that describes the catalogue contents. We then report upon the experience of deploying the catalogue, and the domain-specific modelling language, in the development of national infrastucture for clinical research.

2. Data Models

2.1 Data sets and data standards

A data set definition for clinical research study will consist in a number of different parts, each of which declares a set of related data items. Typically, this will be a set of data items that would be collected together: the results of a particular kind of observation, or the account of a particular kind of intervention. These parts may be 'repeating': the same kind of observation may be made many times of a single study participant.

A data item declaration should explain not only the name under which values are to be stored, but also the type of those values. If the type is numeric, then the unit of measurement should be given. If the type is an enumeration, then the intended interpretation of each value should be explained. Finally, the parts of the dataset may be connected or related to one another, and these relationships may have constrained multiplicities.

It should be clear that a dataset definition can be represented as a class diagram or object model. Data items can be introduced as attributes, parts of the model as classes, and relationships between classes as associations—complete with multiplicities. Data types and enumerations can be used to support attribute declarations.

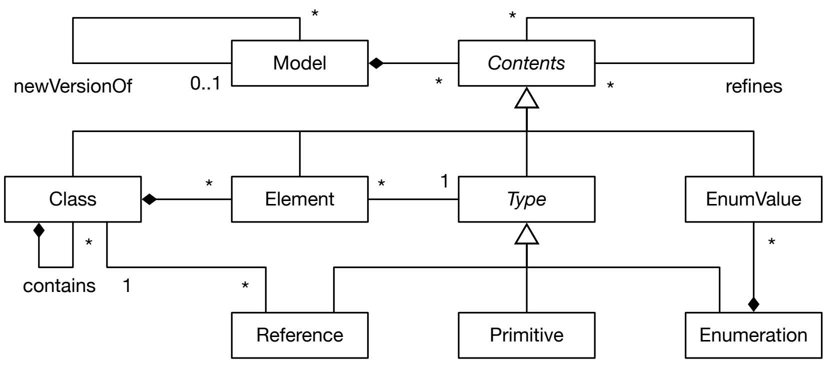

Figure 1. Generic data modelling language

A data set definition may apply to more than one study. It may also be used as a data standard for communication between information systems used in healthcare. The UK National Laboratory Medicines Catalogue, for example, provides a set of standard definitions for pathology reports, to facilitate safe, effective data transfer across different systems.

2.2 Studies, forms, schemas, and databases

A data set definition will not contain enough information to completely characterise a study. The study protocol document will contain precise information about study timetables, workflows, and procedures, as well as a considerable amount of free text explanation. A domain-specific modelling language for studies would be more expressive than a domain-specific language of data set definitions.

Similarly, a domain-specific modelling language for case report forms will support the description of form structures, sections, and 'skip logic'; for example, 'if yes, then go to Question 5'. A modelling language for XML schemas will support the description of schema structures, choices, and complex types, and a modelling language for databases will include information about queries and constraints.

A model of a case report form will contain a number of data item declarations, each with the same information content as a data item definition in an abstract data model. We might be forgiven, then, for thinking that we might not need an abstract data modelling language: we could simply consider, relate, and re-use data definitions from models of 'real' artefacts: studies, forms, schemas, and databases.

However, the fact that a model corresponds to a particular artefact, or even a particular kind of artefact, provides additional context for the data definitions it contains. Whether or not there are additional, explicit constraints upon a data item, the fact that any data collected will have been entered into an implementation of the form tells us strictly more about it—narrowing the interpretation of the data definition.

2.3 Models as metadata

A domain-specific model used in the generation or documentation of an artefact represents valuable metadata about that artefact, and also about any data that the artefact is used to collect or produce. A domain-specific data model—or a data component of any model, for that matter—can be used as metadata about other models. In this way, we can relate artefacts described by different models, and hence the data collected by different studies, forms, or schemas.

To use a model as metadata for an artefact, we have only to create a link between the artefact and a published instance of the model, held in a repository or *model catalogue*. This link may be created automatically if the artefact is generated from the model, or if the model is generated from the artefact: for example, we may generate a more abstract data model from the schema of a relational database.

To use a model as metadata for another model, we create links between the two models. Typically, these will be links between individual data items: for example, an attribute in a model of a form, labelled *height*, could be linked to an attribute in a model of a data set or data standard, labelled *patient's height in cm, measured without shoes*, to indicate that form attribute has all of the properties described in the data set definition.

As we argued above, the form attribute will be further constrained by the remainder of the form model, so the relationship between the two is asymmetric. For this reason, we refer to such a link as an 'implements' relationship. If a pair of attributes are 'implementations' of each other, then we refer to the pair of links as a 'same as' relationship.

In general, we do not expect to find 'same as' relationships between attributes. Different models will add different constraints to the definition of a data item. Instead, consistency of data definitions between studies, forms, or schemas will be represented by 'implements' links to the same data set or data standard. The data definitions are not identical, but are consistent as far as the constraints of the data standard are concerned.

3. Implementation

3.1 Generic data modelling language

A model for a generic data modelling language is shown in Figure 1. All data classes, data elements, and data types are declared and managed within Models. A Class may contain many Elements, and may have other classes as components—corresponding to the UML concept of composition. An element has a unique Type, which may be reference-valued, a Primitive type, or an Enumeration. Reference types correspond to class names within this or some other model. EnumValue, enumeration values, are managed as separate, identified items.

A model may be declared as a new version of an existing model. Any of the items within a model may be declared as a refinement of an existing item. This indicates that its interpretation or semantics should be seen as an extension of those associated with that other item. Typically, this will correspond to the author of the model recording that a particular data class or data element is intended to conform to some existing, published standard.

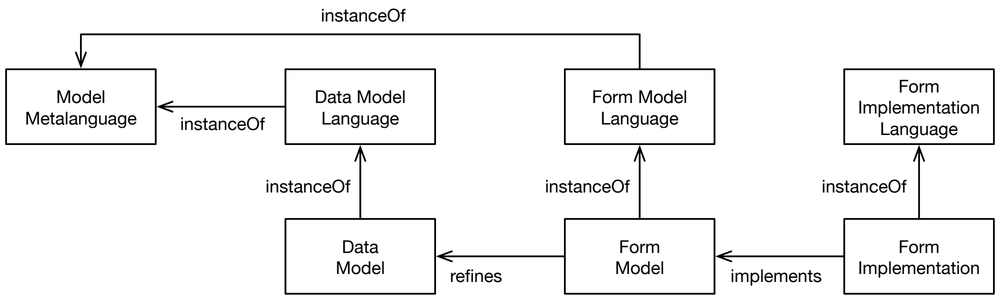

Figure 2. Domain specific data modelling

3.2 Domain-specific data models

The current model catalogue implementation supports the creation, storage, and management of data models in the language of Figure 1. Domain-specific models of studies, forms, and schemas are generated from these models, using different sets of heuristics, but are not themselves managed as catalogue items. A more comprehensive approach would involve persisting, managing, and editing these models alongside the generic data models that they correspond to: the generation process could then work in both directions.

Figure 2 shows the relationship between domain-specific data models—in particular, models of form designs—and generic data models. It shows also the implementations derived from the form designs using a model-driven approach. In the terminology of MDA [18], we may see the form model as a platform-independent entity (at the M1 metamodelling level) and the form implementation as a platform-specific entity (also at M1).

The data model language and the form model language are both entities at the M2 level. The catalogue would support both of these as instances of a data metamodelling language (or model metalanguage) at the M3 level. The relationship between the generic data model and the corresponding domain-specific model would be one of data refinement, in the sense of [19]: in the case of a form model, this would be a simple correspondence between classes, elements, and datatypes; for a workflow or process model, the ways in which data is exposed through transactions and events would need to be considered.

The advantage of this more comprehensive approach is that aspects of form design and implementation can be introduced and managed directly, through editing of Form Models, rather than being encoded as options in a generation pipeline. The corresponding generic data model may be abstracted automatically from the form model; alternatively, the form model may be partially (re-)generated from the generic data model, in the sense of [6].

An alternative approach is shown in Figure 3, in which the generic data model is used as a metamodel for domain-specific data modelling languages. Existing language and tool support for the use of models as metamodels in this context does not allow for the definition and maintenance of the ‘instance of’ relationships in the diagram, and the catalogue implementation under development follows the approach of Figure 2. However, this alternative approach would remove the need to maintain separate, generic data models, and it remains the subject of active investigation.

3.3 Catalogue Implementation

In designing the catalogue, we paid considerable attention to the ISO/IEC 11179 standard for metadata registration, which sets out a design for metadata catalogues. Some difficulties have been encountered in the practical application of ISO/IEC 11179 at scale: see, for example [13] and [14].

The principal complaint is that there is no structuring mechanism for data definitions: data items can be associated only at the conceptual level. As a result, each item has to be defined separately: there is no opportunity to add the same information to several data item definitions at once, whether this is within the model, or as a link to another model.

The approach taken in our implementation of the model catalogue is more general: the use of tagging supports multiple classification schemes, and allows the representation of relationships as well as simple taxonomies. However, it should be clear that our catalogue could be used, under suitable constraints, as an effective implementation of the ISO/IEC 11179 standard.

This applies also to the processes of registration, versioning, and publication. Every object stored in the catalogue is managed as an *administered item*, in the language of the standard. The notion of linking in the catalogue implementation allows us to exploit this administrative information in the automatic creation and maintenance of semantic links, and the administrative processes are generalised to provide support for collaborative development.

The existing model catalogue is built using the Groovy/Grails framework, which takes a model-view-controller approach to data management and presentation. The key advantage of this platform has been the ability to revisit the underlying data representation—the domain model—without needing to re-implement the presentation layer, and vice versa. As the software was developed in the course of application, this was particularly important.

A ‘discourse’ plugin provides support for collaborative development of models and data definitions, with users able to contribute to a comment history for each administered item, prompting responses from other users as necessary. This proved particularly important given that many of the clinical scientists were contributing to the dataset development in their spare time.

3.4 Generation pipelines

The existing implementation has been used to generate several different types of artefact, including:

Case report form models for consumption by the OpenClinica clinical trials management system. These take the form of Excel spreadsheets with columns specifying form structure, question text, response types, logical constraints (including skip logic), and presentation controls. These models are generated from form models in the data modelling language by way of a complex transformation: the hierarchical structure of the data model is flattened to produce lists of sections, repeating groups, and questions. Default values and implementations are included as part of the transformation: for example, we provide custom validation for textual fields that are tagged with constraints in the form of regular expressions.

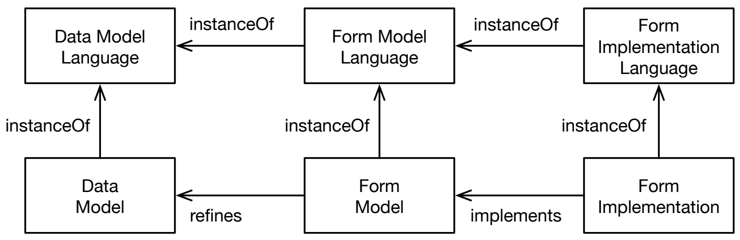

Figure 3. Domain specific data metamodelling

Database triggers To support the automatic processing of data received from the clinical trials system, we require a collection of triggers for the underlying database. These ensure that the combination of existing and newly-received data is properly normalised. This is particularly important where data is being collected against different versions of the same form.

XML schemas for electronic document submission This is a more straightforward transformation, as the structure of the XML schemas is closer to that of the data models. However, additional processing is required to produce normalised, readable schemas. For example, if there are several data elements sharing the same datatype, we would wish to include that datatype only once within the schema.

Tools for creating and validating .csv files For some of the systems that we are working with, the easiest way to import or export information is in comma-separated value format. In this case, we are not generating a specification of the data format in some implementation language; we are instead generating tools that will ensure that the values presented in a file comply with the model constraints.

Data manuals Datasets and data standards in health informatics are communicated through documents in which each data point is listed along with its intended interpretation. These manuals are automatically generated, ensuring consistency between the information that they present and the tools used for data acquisition and processing.

4. Experience

4.1 Re-use of data from clinical information systems

The UK National Institute of Health Research (NIHR) is funding an £11m programme of work across five large university-hospital partnerships: at Oxford, Cambridge, Imperial College London, University College London, and Guy's and St. Thomas'. The aim of the programme is to create the infrastructure needed to support data re-use and translational research across these five institutions.

The programme, the NIHR Health Informatics Collaborative (HIC), was initiated in 2013, with a focus upon five therapeutic areas: acute coronary syndromes, renal transplantation, ovarian cancer, hepatitis, and intensive care. The scope was increased in 2015 to include other cancers—breast, colorectal, lung, and prostate—and other infectious diseases, including tuberculosis.

The key component of the infrastructure consists in repositories of patient data within each of the five institutions. The intention is that these repositories should hold a core set of data for each therapeutic area, populated automatically from clinical systems, together with detailed documentation on the provenance and interpretation of each data point.

Researchers can use the documentation to determine the availability and suitability of data for a particular study. They can use it also to determine comparability across institutions: whether there are any local differences in processes or equipment that would have a bearing upon the combination and re-use of the corresponding data. Once a study is approved, the repositories act as a single source of data, avoiding the need for data flows from individual clinical systems.

The development of the infrastructure required the development of a 'candidate data set' for each therapeutic area, as a core list of data points collected in the course of routine care that would have value also in translational research. Each institution then set out to determine which information systems, within their organisation, could be used to populate each of the candidate data sets: this was termed the 'data exploration exercise'.

The results of the exercise informed further development of the data sets, and data flows were established. To demonstrate and evaluate the new capability, 'exemplar research studies' were initiated in each therapeutic area, using data from all five institutions.

Each institution had a different combination of existing systems, a different approach to data integration, and a different strategy for informatics development. It was not feasible or appropriate to develop a common 'data repository' product for installation. Instead, a set of data models were distributed, and each institution worked to implement these using their own messaging, business intelligence, or data warehousing technologies.

None of the institutions had the capability to provide documentation on the provenance and interpretation of their data in any standard, computable format; the model or metadata aspect of the infrastructure was entirely new. It was this that drove—and continues to drive—the development of a comprehensive model catalogue application.

At the start of the project, teams of clinical researchers and leading scientists were given the responsibility of creating the candidate data sets for each therapeutic area. They did this by exchanging spreadsheets of data definitions in email. This proved to be a slow process, and face to face meetings were needed before any real progress could be made.

It proved difficult to properly represent repeating sections of the dataset—corresponding to investigations or interventions that may happen more than once for the same patient. Researchers resorted to Visio diagrams to try to explain how observations fitted into clinical pathways or workflows—and discovered that there were significant differences between pathways for the same disease at different institutions.

In one therapeutic area, these differences had a profound effect upon the interpretation of certain observations, and the candidate dataset was extended to include additional information on the pathway. Due to the complexity of the pathways involved, this was a time-consuming and error-prone process. Furthermore, the spreadsheets quickly became inconsistent with the Visio diagrams.

The candidate datasets were distributed to the informatics teams at the five institutions in the form of XML schemas. At first, these were created from scratch, rather than being generated. There were many requests for changes to the schemas; these proved difficult to track and coordinate.

The exploration exercise was reported by adding columns to the distributed versions of the candidate dataset spreadsheets, listing the information systems containing the data points in question, or suggested alternatives where there were significant differences due to local systems and processes.

This was despite the availability of an initial version of the model catalogue. Researchers and local informatics teams preferred to work with spreadsheets, having little or no knowledge of modelling languages such as UML and no automatic support for model creation and maintenance. It fell to the software engineering team at the coordinating centre to record the datasets and variations in the catalogue.

While it was disappointing to have the researchers still working in spreadsheets, the ability to generate XML schemas from models, and to manage relationships between data items in different models and different versions, proved invaluable. In the second phase of the project, researchers are starting to abandon the spreadsheet mode of working, and are instead maintaining the datasets as data models, in the catalogue.

4.2 Coordination of clinical data acquisition

The UK Department of Health, through the NIHR and the National Health Service (NHS), is providing funding for the whole genome sequencing of blood and tissue samples from patients with cancer, rare disorders, and infectious disease. A network of regional centres is being established to collect samples and data, and to provide access to genomic medicine across the whole of the country. The funding committed to date is approximately £300m.

The results of the whole genome sequencing will be linked to detailed information on each participant: clinical and laboratory information drawn from health records, ontological statements regarding abnormal features or conditions, and additional information obtained from the participant or their representatives. The information required will depend upon the nature of the disease that the patient is suffering from. For example, information on breast density is required in the case of breast cancer, but not for other diseases.

131 different diseases have been included in the sequencing programme thus far. Each disease corresponds to a different combination of clinical and laboratory data points, a different set of ontological statements, and a different set of questions for the participant. There are, however, significant overlaps between diseases: for example, many different rare diseases will require the same information on kidney or heart function.

The modelling task is at least an order of magnitude greater than that required for the NIHR HIC, and yet candidate datasets have already been created for more than half of the diseases included. This is due partly to the availability of the model catalogue application from the start of the project, and partly to the availability, within the catalogue, of the full complement of HIC-defined data models and related data sets—including the national NHS data dictionary and the national cancer reporting datasets.

Two routes are available for the provision of data from the network of contributing centres: direct data entry into electronic case report forms, in a on-line clinical trials management system; and electronic submission of data in XML format. The intended interpretation of the data required is explained in a regularly-updated set of data manuals.

It is important that the forms used for direct data entry, the schemas used for XML submission, and the data manuals are properly synchronised. An initial approach to this, in which a single model was used as the basis for the generation of all three kinds of artefact, proved inconvenient in practice. Although the same data points were to be collected in each case, the distribution of these data points across classes and sections was different.

Accordingly, the model catalogue is used to store three different data models for each dataset: one for the generation of the forms, another for the generation of the XML schemas, and one for the generation of the data manual. These models are semantically-linked. If one is updated, then the fact that the others may now be inconsistent will be flagged to the user.

The same linkage is made with regard to existing reporting datasets and clinical audits. To avoid duplication of effort, the reporting datasets for the genomic medicine programme have been aligned with these activities. The existing datasets have been modelled, and updates to them will be tracked in the catalogue: again, potential inconsistencies can be flagged.

5. Related Work

The work described in this paper has evolved from the CancerGrid project [8], where an ISO/IEC 11179-compliant metadata registry was developed for curation of semantic metadata and model-driven generation of trial-specific software [5, 7]. The approach to generating forms in the CancerGrid project has been generalised significantly with the introduction of a data modelling language and a broader notion of semantic linking.

Another effort to develop an implementation of ISO/IEC 11179 is found in the US caBIG initiative [12]; however, their caCORE software development kit [11] applies model-driven development only to generate web service stubs, requiring developers to create application logic by hand, whereas our technique integrates with existing clinical Electronic Data Capture tools and workflows, such as OpenClinica [4].

Several efforts have addressed ontological representations for enabling data integration across metadata registries (MDRs). Sinaci and Erturkmen [16] describe a federated semantic metadata registry framework where Common Data Elements (CDEs) are exposed as Linked Open Data resources. Jeong *et al.* [10] present the Clinical Data Element Ontology (CDEO) for unified indexing and retrieval of elements across MDRs; they organise and represent CDEO concepts using SKOS. Tao *et al.* [17] present case studies in representing HL7 Detailed Clinical Models (DCMs) and the ISO/IEC 11179 model in the Web Ontology Language (OWL), but do not present any systematic metamodelling or language definition framework.

Ontology repositories can be considered closely analogous to model catalogues, they provide the infrastructure for storing, interlinking, querying, versioning, and visualising ontologies. Relationships capturing the alignments and mappings between ontologies are also captured, allowing easy navigability. Linked Open Vocabularies [2] provides a service for discovering vocabularies and ontologies published following the principles of linked data.

In the Model Driven Health Tools (MDHT) [3] project, the HL7 Clinical Document Architecture (CDA) standard [9] for managing patient records is implemented using Eclipse UML tools [1]. In principle, this is similar to our Model Catalogue approach, where the CDA metadata can be represented and implementations derived. However, MDHT supports only the CDA standard, whereas the Model Catalogue can interoperate with any metadata standard. The CDA standards are large and complex: Scott and Worden [15] advocate a model-driven approach to simplify the HL7 CDA.

6. Conclusion

The experience of applying the data model language, the model catalogue, and the associated generation tools in the context of clinical research informatics has led to the following suggestions.

A data dictionary is not enough. A simple, flat list of data definitions does not support re-use at scale: it requires the user to place all of the contextual information into the definition of each data item, and mitigates against the automatic generation and application of definitions. Instead, a compositional approach is required, in which data elements are defined in explicit context.

A catalogue is not enough. The models in the catalogue must be linked to implementations, and to each other, with a considerable degree of automatic support. If the models are out of sync with the implementations, and with the data, then their value is sharply diminished. If you are going to manage data at scale, you need a data model-driven approach.

The tools must be usable by domain experts. To have the processes of model creation and maintenance mediated by software engineers is problematic: there may be misunderstandings regarding interpretation, but—more importantly—there are not enough software engineers to go around. An appropriate user interface, that closely matches the intuition and expectations of domain experts, is essential.

There will be more models than you think. Different models will be required for different types of implementation, and—in any research domain, at least—data models will be constantly evolving, with data being collected against different versions.

Intelligent, automatic support is essential. The information content of precise data models is considerable, and there may be complex dependencies between data concepts and constraints. A considerable degree of automation is required if users are to cope with this complexity.

The model catalogue and the associated toolset should, as far as possible, automatically: create or propose links, including classifications; manage model versioning, and the consequences for linked data concepts; manage dependencies, including those between different models for same dataset, targeted at different implementation platforms.

This should come as no surprise. If, as Warmer and Kleppe [18] suggest, the model-driven approach is about "using modelling languages as programming languages rather than merely as design languages" then we should aim to provide modellers with the same kind of support that programmers have come to expect from modern integrated development environments.

Acknowledgments

We would like to acknowledge the support of the National Institute for Health Research (NIHR), and in particular that of the NIHR Health Informatics Collaborative, and the financial support of the Medical Research Council. We would like to acknowledge also the support of the European Union Horizon 2020 Project ALIGNED, Project Number 644055. We are very grateful for the contributions made by colleagues at the University of Oxford: in particular, Steve Harris and Charles Crichton.

References

[1] Eclipse MDT UML2 tools. URL `https://eclipse.org/modeling/mdt?project=uml2`.

[2] Linked Open Vocabularies. URL `http://lov.okfn.org/dataset/lov`.

[3] Model driven health tools. URL `https://www.projects.openhealthtools.org/sf/projects/mdht`.

[4] OpenClinica. URL `http://www.openclinica.com`.

[5] D. Abler, C. Crichton, J. Davies, S. Harris, and J. Welch. Models for forms. *Proceedings of the 11th Workshop on Domain-Specific Modeling*, October 2011.

[6] K. Czarnecki and U. Eisenecker. *Generative Programming: Methods, Tools, and Applications*. Addison Wesley, 2000.

[7] J. Davies, J. Gibbons, R. Calinescu, C. Crichton, S. Harris, and A. Tsui. Form follows function: Model-driven engineering for clinical trials. In *Foundations of Health Informatics Engineering and Systems*, pages 21–38. Springer, 2012.

[8] J. Davies, J. Gibbons, S. Harris, and C. Crichton. The CancerGrid experience: Metadata-based model-driven engineering for clinical trials. *Science of Computer Programming*, 89:126–143, 2014.

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

[10] 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. *Healthcare Informatics Research*, 20(4):295–303, Oct 2014.

[11] 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, et al. caCORE version 3: Implementation of a model driven, service-oriented architecture for semantic interoperability. *Journal of Biomedical Informatics*, 41(1):106–123, 2008.

[12] I. Kunz, M.-C. Lin, and L. Frey. Metadata mapping and reuse in caBIG. *BMC Bioinformatics*, 10(Suppl 2):S4, 2009.

[13] S. M. Ngouongo, M. Löbe, and J. Stausberg. The ISO/IEC 11179 norm for metadata registries: Does it cover healthcare standards in empirical research? *Journal of Biomedical Informatics*, 46(2):318 – 327, 2013. ISSN 1532-0464. . URL `http://www.sciencedirect.com/science/article/pii/S1532046412001827`.

[14] G. Richardson and E. Schwarz. A Prototype ISO/IEC 11179 Metadata Registry. *CEUR Workshop Proceedings*, 966, 2012.

[15] P. Scott and R. Worden. Semantic mapping to simplify deployment of HL7 v3 Clinical Document Architecture. *Journal of Biomedical Informatics*, 45(4):697–702, 2012.

[16] A. A. Sinaci and G. B. L. 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. ISSN 1532-0464. . URL `http://www.sciencedirect.com/science/article/pii/S1532046413000750`.

[17] 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 Summits on Translational Science Proceedings*, 2011:59–63, 2011.

[18] J. Warmer and A. Kleppe. *The Object Constraint Language: Getting Your Models Ready for MDA*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2 edition, 2003. ISBN 0321179366.

[19] J. Woodcock and J. Davies. *Using Z: Specification, Refinement, and Proof.* Prentice Hall, 1996.