Mapping OBI and XPDL to a MDE Framework for Laboratory Information Processing

Alessandro Maccagnan*, Nicola Cannata‡, Giorgio Valle†, Tullio Vardanega*
*Department of Pure and Applied Mathematics, University of Padua, via Trieste 63, 35121 Padova, Italy
e-mail:{maccagnan, tullio.vardanega}@math.unipd.it
‡School of Science and Technology, University of Camerino, Via Madonna delle Carceri 9, 62032 Camerino, Italy
e-mail: nicola.cannata@unicam.it
†CRIBI Biotechnology Centre, University of Padua, viale G. Colombo 3, 35121 Padova, Italy
e-mail: giorgio.valle@cribi.unipd.it

***Abstract*—Biomedical analyses are becoming increasingly complex, both for the type of data produced and the procedures necessary to obtain them. This trend is expected to continue; therefore the development of suitable systems for information and protocol management is becoming essential for the full exploitation of the field. Custom-built applications obtained by direct merging of software engineering expertise with domain-specific knowledge may be temporary solutions, but they are generally ineffective both in terms of cost and performance. Here we propose a Laboratory Information Management System (LIMS) that enables the domain experts to express laboratory protocols using domain knowledge, free from the incidence and mediation of the software implementation artifacts. In the system that we propose this is made possible by basing the modeling language on an authoritative domain specific ontology.**

***Index Terms*—Model-Driven Engineering; Laboratory Protocols; Ontology; Process Definition Language.**

I. Introduction

A. Motivation and Vision

In the last two decades life sciences and biomedicines have been revolutionized by the introduction of high-throughput procedures and automation methods. Laboratory Information Management Systems (LIMS) are tools used for tracking protocols and samples, in order to reliably cope with such turnout. Unfortunately protocols are still mainly written and exchanged in natural languages which is a serious impediment to quality, efficiency, predictability and repeatability. We claim we should rather strive to represent protocols in a structured and efficient way. In this work, we present a way to bridge the benefits of Ontology onto Model-Driven Engineering (MDE) in order to satisfy this need.

What we ultimately aim to accomplish is to tie in an intimate way ontologies and meta-models. Our aim is to build models that are deeply-rooted on ontologies. We propose a conceptual framework that links a *construct that describes reality* (ontology) to a *construct that prescribes reality* (model). Hence we are trying to bind ontological constraints directly to model elements.

The structure of the paper is as follows. In section II, we give a brief overview on Model-Driven Engineering and Ontology and some relevant literature on how to merge them. In section III, we describe our proposal on how to merge a domain specific ontology with a workflow metamodel.

II. Background

A. Model-Driven Engineering

MDE is an approach to software development which concentrates on designing models that are closer to domain-specific concepts of some particular domain rather than to computing (or algorithmic) ones. MDE's basic concepts are models, meta-models and transformations [1].

A model is a "set of statements about some system under study" [2]. In traditional scientific disciplines, models are usually descriptive. However they are also used as specifications in engineering disciplines, including software design. Therefore a model could equally be descriptive or prescriptive.

A distinctive trait of models is their intended relationship with reality: "A model is an external and explicit representation of a part of reality as seen by the people who wish to use that model to understand, change, manage, and control that part of reality" [3].

Models can represent, describe, and specify things [4]. A *descriptive* model is one that "describes reality, but reality is not constructed from it". A *prescriptive* model is one that "prescribes the structure or behavior of reality and reality is constructed according to the model; that is, the model is a specification for reality" [2]. Since in the realm of software engineering most of the models are used to construct a "reality" from them, in the remainder of this paper we understand a model as prescriptive.

B. Ontologies

The term 'ontology' is currently very controversial controversial because different people have different ideas on the definition of an ontology. However there is a certain consensus in what an ontology is not: it is neither a taxonomy (i.e., a class-subclass hierarchy), nor a dictionary (an ontology does include relationships between terms), nor a knowledge base that includes only individual objects. According to Gruber, an ontology can be defined as "the specification of conceptualizations, used to help programs and humans share knowledge" [5].

 ISBN: 978-1-61208-181-6

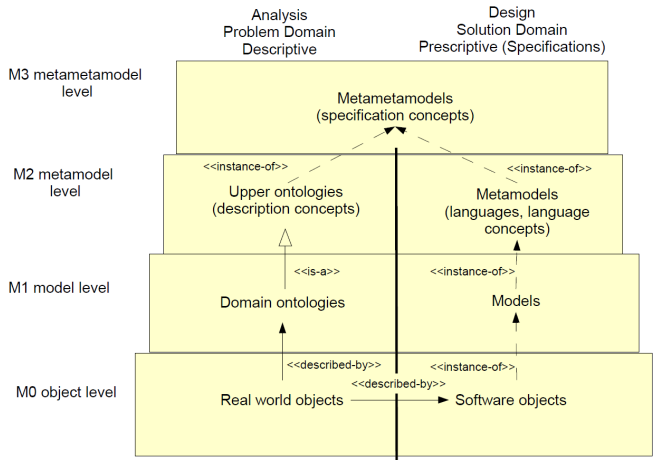

Fig. 1. The ontology-aware meta-pyramid. Domain ontologies live at level M1. Upper ontologies live at level M2. Ontology metalanguages live at level M3. Source: [4].

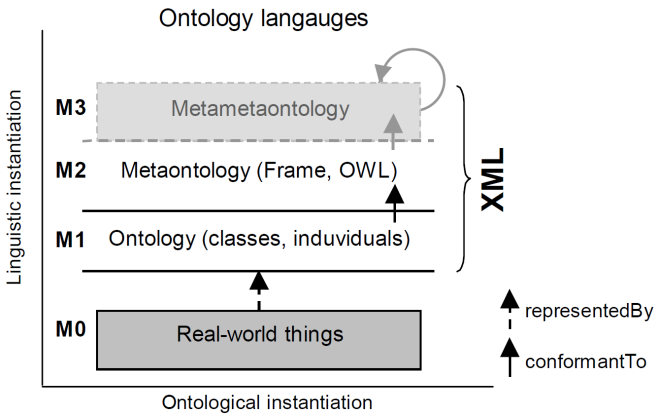

Fig. 2. The four meta-layers in terms of ontological engineering and its orthogonal instance-of relations: linguistic and ontological. Source: [8].

The first formal and explicit approach to ontologies in the technical (not philosophical) sense dates back to 1900, given by Husserl. Later in the 1980's, the ontologies entered the computer science field as a way to provide a simplified and well defined view of a specific area of interest or domain. Semantic web is the modern expression of the field. The Web Ontology Language (OWL) is a modern ontological language endorsed by the World Wide Web Consortium (W3C).

What we really are after is the conceptual relationship between a model and an ontology in the context of knowledge and process management.

An important property of ontologies is the *open-world assumption*, i.e., anything not expressed is unknown [6]. In models the *closed-world assumption* is generally used since what has not been specified is not unknown but true (or false) by default. As we noted earlier, models are usually prescriptive tools. What can we say for ontologies? Ontologies are not specification models since they describe domains and not systems [4].

Ontologies are tools extensively used to express domain knowledge. One serious problem is that differing ontologies may be developed and applied for the representation of one and the same domain. The function of an upper ontology is precisely to "support interoperability between domain ontologies in order to facilitate the shared use of data both within and across disciplinary boundaries" [7]. A domain ontology specializes concepts taken from an upper ontology.

C. MDE and Ontologies

Assmann et al. propose the ontology-aware meta-pyramid [4] (Fig. 1) in order to show how ontologies can be used in MDE. Domain ontologies live at level M1 of the meta-pyramid and correspond to models. An upper ontology, providing a language for ontologies, should live at level M2. One metametamodel language (at level M3) could be used to specify both ontology and metamodels. Both the ontology dimension and the model-driven dimension instantiates from this metametamodel.

Bezivin et al. use a different approach to relate ontology and MDE. The four MDE levels are called in this context linguistic layers [8]. Concepts from the same linguistic layer can be at different ontological layers. Figure 2 depicts the four meta-layers using this important remark. The linguistic instantiation runs on the vertical dimension; conversely the ontological instantiation runs on the horizontal dimension (e.g., like for an upper ontology and a domain ontology built upon it).

Kapsammer et al. propose a mapping between the metametamodel language Ecore (model engineering space) and the ontology definition metamodel (ontology technical space) [9]. Ecore belongs to the Eclipse Modeling Framework (EMF) and it is used to describe models and metamodels. Also the mapping proposed between EMF objects and OWL/RDF resources [10] presents some difficulties (e.g., class membership), because in object-oriented languages the membership of objects is fixed along a derivation hierarchy. In OWL instead, individuals can belong to multiple classes.

Hillairet et al. designed a set of Eclipse plugins that are able to make a round-trip transformation between OWL and Ecore. The project (named EMF4SW) is not yet mature enough to cope with large and complex ontologies. However it is in very active development and it is able to deal with relative small ontologies.

Parreiras et al. describe a vision in which both worlds (MDE and Ontology) co-exist under a common umbrella [11]. The concepts of Metamodeling Technical Spaces (MMTS) as well as Ontological Technical Spaces (OTSs) are introduced, derived from the work of [12]. They suggest some strategies for integrating OTS into MMTS. Furthermore they provide a list of desirable features for the "marriage" MMTS+OTS.

III. Combining Ontologies and Workflows

Our field of application is that of laboratory informatics and scientific experimentation. Relying on our experience and on related literature we observed that laboratory procedures are based on some key elements ([13], [14]). They consist

 ISBN: 978-1-61208-181-6

in workflows that describe some interaction between some objects by means of some actions. As far as the descriptive knowledge of this domain is concerned, the most relevant ontology accepted by the community is the Ontology for Biomedical Investigations (OBI). For the prescriptive part of our effort instead, we opted for on BPMN/XPDL for workflows definition.

A. Ontology for Biomedical Investigations

OBI is an ontology for the description of biological and clinical investigations [15]. OBI describes the design of an investigation, protocols and instrumentation, materials used, data generated and analysis performed on it. The OBI project is developed in the frame of the Ontology for Biomedical Investigations (OBO) Foundry [16], and as such, it adheres to the principles of OBO, as orthogonal coverage and the use of a formal language. OWL was chosen as the OBI language. The ontology is developed to model biomedical investigations, therefore it contains terms for aspects such as:

- Biological material, e.g., DNA
- Instrument, e.g., centrifuge or thermal cycler
- Design and execution of an investigation, e.g., injecting mice with a vaccine to test its efficacy.

OBI relies on the Basic Formal Ontology (BFO) upper ontology. An upper ontology describes concepts of the "Reality" from a high-level of abstraction.

B. XML Process Definition Language

XPDL is a markup language created to ensure interoperability among different workflow management systems. Its main goal is to exchange process definitions, addressing both the graphical and the semantic notations.

The meta-model of XPDL involves the definition of activities, the specification of their order of execution and the involved data. The flow of execution is specified through different constructors: sequence, split, join. An elementary activity is an atomic piece of work [17]. An Activity could modify relevant data declared as DataField. In addition to standard types a user could add external types (by means of an XSD declaration or an external reference).

C. Mapping XPDL and OBI

Finding a method to relate the MDE architecture (its layers of abstraction) to the ontology schema is key to facilitating the systematic use of ontologies inside prescriptive models. Based on the literature we reviewed we built a relation between the classical layers of the MDE architecture, BFO/OBI and XPDL.

Fig. 3 depicts the classic layers of MDE. The workflow components of our formalism are fairly easy to place within this hierarchy. XSD, the XML schema language used to describe XPDL, can be positioned at the M3 level (i.e., meta-meta-model). XPDL conforms to a XSD model and therefore lies at the M2 level (i.e., meta-model). A valid XPDL workflow (i.e., a model for the end user) is at level M1. A specific esecution of a workflow resides at the ground level M0 (not shown in the figure).

In XPDL, the concept of Activity represents the unit of work. An Application is a particular kind of Activity that describes functionalities offered by legacy systems. In XPDL an Application is invoked by means of a Tool Activity. In object-orientation terms, an Application can be seen as an interface for a functionality with a name and a list of parameters. We can think of an interface as a sort of "contract" between a class and the outside world. Every parameter is described with a name, a type, and a mode of passing (input, output, mixed). The Application construct represents the junction point between the workflow world of XPDL and the ontological world of BFO/OBI.

Before defining a mapping between BFO/OBI and XPDL we need to also relate the former to MDE. BFO is written using OWL, hence, in our schema of interpretation, OWL is at level M3 and BFO at M2. OBI is a specialization of BFO in the dimension of the description of the domain. It is not a specialization in the linguistic dimension proper of the MDE [4]. For that reason OBI places at M2 but in a sort of orthogonal dimension to the classic hierarchy (which we show horizontally instead of vertically in Fig. 3). A consequence of this is that instances of BFO/OBI concepts (in OWL called individuals) are at M1. Using this schema of interpretation individuals are tags that have as referent the real objects that we put at M1.

Having said that, it is easier to relate some of the BFO/OBI concepts with the XPDL classes to produce a mapping between elements of the two worlds. Table I presents the resulting mapping.

TABLE I
MAPPING BETWEEN XPDL AND BFO/OBI. THE RELEVANT CONCEPTS OF XPDL ARE MAPPED WITH CONCEPTS FROM BFO AND OBI.

Laboratory	XPDL	BFO	OBI
Protocol	Process	Directive information entity	Plan specification
Sub-protocol	SubFlow	Directive information entity	Plan specification
Unique single step of a protocol	Task/Tool	Directive information entity	Action specification
Real world (e.g., Illumina sample) or theoretical (e.g., Project) items	Data Type	independent continuant	material entity
		generically dependent continuant	information content entity
Objects properties	Data Field	specifically dependent continuant	quality

The main concept of *Protocol* is easily mapped to the workflow model by the notion of *Process*. In the XPDL specification a process is defined as a "combination of various activities with a specified flow of execution". An internal process consists of one or more activities, each comprising a logical, self-contained unit of work". We connected this concept with the OBI concept of *Plan specification*, defined as a "directive information entity that when construct it is realized in a process in which the bearer tries to achieve the objectives, in part by taking the specified actions. Plan speci-

 ISBN: 978-1-61208-181-6

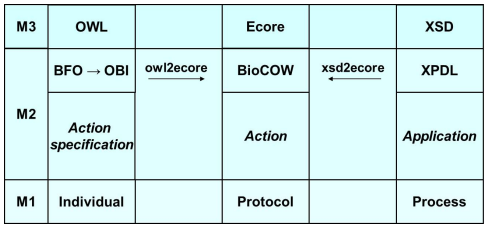

Fig. 3. The BioCOW meta-model is built combining XPDL with BFO/OBI. A standard XSD to Ecore transformation is used for XPDL. For BFO/OBI it has been used an existing tool dealing with OWL to Ecore transformation. As an example we show how the concepts of Action specification (XPDL) and Application (XPDL) are mapped into Action (BioCOW).

fications includes parts such as objective specification, action specifications and conditional specifications. A SubFlow (sub-protocol) is a process itself hence the mapping is the same as for process (i.e, *Plan Specification*).

The second main concept is the notion of unit of work. In XPDL this is backed by the Activity class, which can be of different kinds. One of those is the Task/Tool class, a service or an application required and invoked by the process. In the XPDL metamodel every tool declares a set of Applications. We mapped this XPDL concept with the Action specification in OBI, which defines it as a "directive information entity that describes an action the bearer will take".

Since an Activity is an atomic piece of work that may modify relevant data (declared as DataFields) we mapped on it both the XPDL concept of DataType and DataField. A Datatype in our model could be, aside from standard type, a *OBI:material entity* or an *OBI:information content entity*. We chose to map a DataField with the OBI concept of *quality*.

D. Implementation

To build the described meta-model we used the technology provided by the EMF and Ecore in particular. EMF provides tools to automatically convert heterogeneous formats to Ecore. Specifically there is a standard way to translate an XML Schema Definition (XSD) file in the Ecore format. Since XPDL is formulated in XSD we automatically imported it in EMF.

For the transformation of the OBI ontology into the Ecore format we followed the approach proposed by Hillairet et al. [10]. In particular, we translated the whole BFO ontology and the main classes of OBI from OWL. Using that approach we were able to manipulate both the ontology and the XPDL meta-model in a coherent way inside the EMF framework.

Our meta-model is built using as reference the XPDL meta-model. In order to actually concretize the mapping between XPDL and BFO/OBI we created a new class for every mapped classes. That new class inherits both the XPDL and BFO/OBI class as specified in the mapping shown in table I. For example, the *BioCOW:Action* class has, as a superclass, the *BFO:GenericallyDependentContinuant* class. It is worth noting that we have not specialized directly the XPDL meta-model since it is richer than we need for our purposes. We therefore based our model on XPDL retaining the main concepts and leaving out all the surplus details.

Using the resulting BioCOW (Bio-medicine Combined Ontology [and] Workflow) meta-model, we are now able to describe laboratory protocols in a formal yet intuitive way. By means of the Obeo designer we are able to build a Graphical User Interface (GUI) which associates graphicals symbols with constructs of the BioCOW meta-model. The efforts required to produce a GUI are greatly reduced using Obeo in contrast for example to the Eclipse Graphical Modeling Framework (GMF). Interestingly, however, Obeo still bases on GMF. The graphical editor enables the user to visually specify the desired protocols assembling components from the provided high-level language. In this manner, the designed protocols constructively conform to our meta-model.

E. Assessment

We are currently evaluating the BioCow meta-model in a real-world laboratory environment with the help of domain experts. We are comparing different frameworks analysing protocols widely used in the laboratory under the dimension of the *language* and *mediation* features.

IV. Discussion

The direction of our work relies on the potential of using ontology technologies in a MDE context. Thanks to the OTSs we can, for example, enable automatic reasoning for model consistency checking. Semantically assisted design (SAD) will allow the adoption of "intelligent" editors. Another important prospective is to enable systematic reuse of community-level shared formalized knowledge.

We place ourselves in the framework of the Features Model of Bridging MMTS and OTSs sketched in [12]. In our current work we have focused on some of those desiderata like the *mediation*. In fact, we have built a *mapping* from two specific modeling spaces. (XPDL and OBI) onto two technical spaces (MMTS and OTS). As part of that effort we were able to *integrate* concepts of XPDL with concepts of OBI. Working under the EMF toolbox we were able to incorporate some of the features of MMTS under a common technological and conceptual framework.

References

[1] J.-M. Favre and T. Nguyen, "Towards a megamodel to model software evolution through transformations," in *SETRA Workshop, Elsevier ENCTS*, vol. 127, 2004, pp. 59–74.

[2] E. Seidewitz, "What models mean," *Software, IEEE*, vol. 20, no. 5, pp. 26 – 32, sep. 2003.

[3] M. Pidd, *Tools for Thinking: Modelling in Management Science*, 3rd ed. Wiley, Feb. 2009. [Online]. Available: http://www.worldcat.org/isbn/0470721421

[4] U. Assmann, S. Zschaler, and G. Wagner, "Ontologies, Meta-models, and the Model-Driven Paradigm," *Ontologies for Software Engineering and Software Technology*, pp. 249–273, 2006.

[5] T. R. Gruber, "Towards principles for the design of ontologies used for knowledge sharing," in *Formal Ontology in Conceptual Analysis and Knowledge Representation*, N. Guarino and R. Poli, Eds. Deventer, The Netherlands: Kluwer Academic Publishers, 1993. [Online]. Available: citeseer.ist.psu.edu/gruber93toward.html

 ISBN: 978-1-61208-181-6

[6] Horrocks, I., Schneider, Patel P., and van Harmelen, F., "From shiq and RDF to OWL: The making of a web ontology language," *Journal of Web Semantics*, vol. 1, no. 1, pp. 7–26, 2003. [Online]. Available: {http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.2.7039}

[7] S. Schulz, M. Boeker, and H. Stenzhorn, "How granularity issues concern biomedical ontology integration," *Studies in health technology and informatics*, vol. 136, pp. 863–8, 2008.

[8] J. Bezivin, V. Devedzic, D. Djuric, J. Favreau, D. Gasevic, and F. Jouault, "An m3-neutral infrastructure for bridging model engineering and ontology engineering," in *Int. Conf. on Interoperability of Enterprise Software and Applications (INTEROP-ESA)*, Springer, Ed. Geneva, Switzerland: Springer, 2005, pp. 159–171, 1-84628-151-2. [Online]. Available: http://www.isima.fr/~favreau/files/publications/INTEROP-ESA.Extended.pdf

[9] E. Kapsammer, H. Kargl, G. Kramler, T. Reiter, W. Retschitzegger, and M. Wimmer, "Lifting metamodels to ontologies - a step to the semantic integration of modeling languages," in *In Proceedings of the ACM/IEEE 9th International Conference on Model Driven Engineering Languages and Systems (MoDELS/UML 2006*. Springer, 2006, pp. 528–542.

[10] B. F. Hillairet Guillaume and L. J. Yves, "Bridging emf applications and rdf data sources," in *Proceedings of the 4th international workshop on Semantic Web Enabled Software Engineering (SWESE) at ISWC'08*, 10 2008, pp. 26–40.

[11] F. S. Parreiras, S. Staab, and A. Winter, "On marrying ontological and metamodeling technical spaces," in *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, ser. ESEC-FSE '07. New York, NY, USA: ACM, 2007, pp. 439–448. [Online]. Available: http://doi.acm.org/10.1145/1287624.1287687

[12] I. Kurtev, J. Bézivin, and M. Aksit, "Technological spaces: An initial appraisal," in *CoopIS, DOA 2002 Federated Conferences, Industrial track*, 2002. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.109.332

[13] L. Soldatova, W. Aubrey, R. King, and A. Clare, "The exact description of biomedical protocols," *Bioinformatics (Oxford, England)*, vol. 24, pp. i295–303, Jul 2008, 10.1093/bioinformatics/btn156.

[14] A. Maccagnan, M. Riva, E. Feltrin, B. Simionati, T. Vardanega, G. Valle, and N. Cannata, "Combining ontologies and workflows to design formal protocols for biological laboratories," *Automated Experimentation*, vol. 2, no. 1, p. 3, 2010.

[15] M. Courtot, W. Bug, F. Gibson, A. L. Lister, J. Malone, D. Schober, R. Brinkman, and A. Ruttenberg, "The owl of biomedical investigations," in *OWLED*, ser. CEUR Workshop Proceedings, C. Dolbear, A. Ruttenberg, and U. Sattler, Eds., vol. 432. CEUR-WS.org, 2008, p. xx.

[16] B. Smith, M. Ashburner, C. Rosse, J. Bard, W. Bug, W. Ceusters, L. Goldberg, K. Eilbeck, A. Ireland, C. Mungall, N. Leontis, P. Rocca-Serra, A. Ruttenberg, S.-A. Sansone, R. Scheuermann, N. Shah, P. Whetzel, and S. Lewis, "The obo foundry: coordinated evolution of ontologies to support biomedical data integration," *Nature Biotechnology*, vol. 25, pp. 1251–1255, Nov 2007, 10.1038/nbt1346.

[17] N. Russell, A. H. Hofstede, D. Edmond, and W. M. der Aalst, *Workflow Data Patterns: Identification, Representation and Tool Support*. Berlin/Heidelberg: Springer-Verlag, 2005, vol. 3716, ch. Chapter 23, pp. 353–368.

 ISBN: 978-1-61208-181-6