

Marrying Ontologies and Model Driven Engineering Technical Spaces: The TwoUse Approach

Fernando Silva Parreiras

ISWeb — Information Systems and Semantic Web,
Institute for Computer Science, University of Koblenz-Landau
Universitaetsstrasse 1, Koblenz 56070, Germany
parreiras@uni-koblenz.de

Abstract. The Ph.D. proposal addresses challenges in composing metamodel-based modeling and ontology technologies. Although the two paradigms, UML-like models and OWL ontologies, and their related technical spaces seem closely related, in the state-of-the-art research and practice the two technologies are just beginning to converge and the relationship between the two is still under exploration. The proposal comprehends different facets of the composition challenge, namely: (1) domain analysis of the different technical spaces; (2) the specification of a coherent framework for integrated use of both modeling approaches, comprising the benefits of UML models and OWL ontologies; Applications of the proposed framework to improve (3) Model Driven Engineering and (4) Semantic Web technologies. Additionally, this paper enlightens the research methods to be used as well as the correspondent validation procedures.

1 Introduction

The Unified Modeling Language (UML) [1] and Web Ontology Language (OWL) [2] constitute modeling approaches with different strength and weaknesses that make them appropriate for use of specifying different aspects of software systems. In particular, OWL ontologies are well suited to specify classes using an expressive logical language with highly flexible, dynamic and polymorphic class membership, while UML diagrams are much more suitable for specifying not only static models including classes and associations, but also dynamic behavior.

Though MOF based metamodels [3] and UML profiles for OWL ontologies have been proposed in the past, an integrated use of both modeling approaches in a coherent framework has been lacking so far. This paper unveils research problems involving the composition of these two paradigms and present research methods and validation techniques to assess the application of a novel framework integrating UML-like models and OWL ontologies and technologies.

Firstly, a characterization of different metamodeling technical spaces and ontological technical spaces is required to recognize the features of the different

paradigms and to elicit the requirements of an integrated framework (Problem 2.1). Consequently, the method of integration, the TwoUse approach, is specified (Problem 2.2) and conceptually verified against the requirements resulted from the first investigation.

Thereafter, shortcomings of canonical approaches in Model Driven Engineering (MDE), i.e., software design patterns and model transformation design patterns are addressed by applying the proposed framework with the aim at reducing complexity and improving reusability (Problem 2.3).

Concerning Semantic Web technologies, issues addressing the gap in clarity and accessibility of languages that operate ontologies, e.g., Ontology translation languages and semantic web service specification languages, are undertaken (Problem 2.4). The TwoUse framework is used to support the development of platform independent models aiming at improve maintainability and comprehensibility. The application of TwoUse is endorsed through experimental validations relying on case studies as observational method.

2 Problems and Goals

2.1 Requirements for Integrating Ontological and Metamodeling Technical Spaces

Problem. Over the last decade, the Web, AI and database communities have successfully investigated and promoted the use of *ontologies* as modeling and reasoning frameworks for the management of models and corresponding (Web) data. Ontologies and MDA technologies exhibit different foci. OMG MOF targets automating the management and interchange of metadata whereas knowledge representation focuses on semantics of the content and on automated reasoning over that content [4].

While the focus of these communities is somewhat different, the question still arises *how the scientific and technical results around ontologies, ontology languages and their corresponding reasoning technologies can be used fruitfully in model-driven software engineering and vice versa.*

Objectives. While investigating this problem, our goal is to analyze different integration approaches available in the literature. The result of such analysis is a feature model, preliminarily discussed in [5]. A feature model comprises a feature diagram, depicted in Fig. 1, description of the features and examples.

The feature model reveals the different possible choices for an integrated approach of Metamodeling and Ontologies technical spaces and can also be used as a taxonomy to categorize the existing approaches published in the literature, as discussed in [5]. Furthermore, the classification allows for eliciting requirements for an composed approach, used to validated the results of Problem 2.2.

Initial patterns we find in our own work as well as in related works shows that next to existing technical spaces of established meta modeling frameworks, new technical spaces are positioned that either enrich or exploit the software

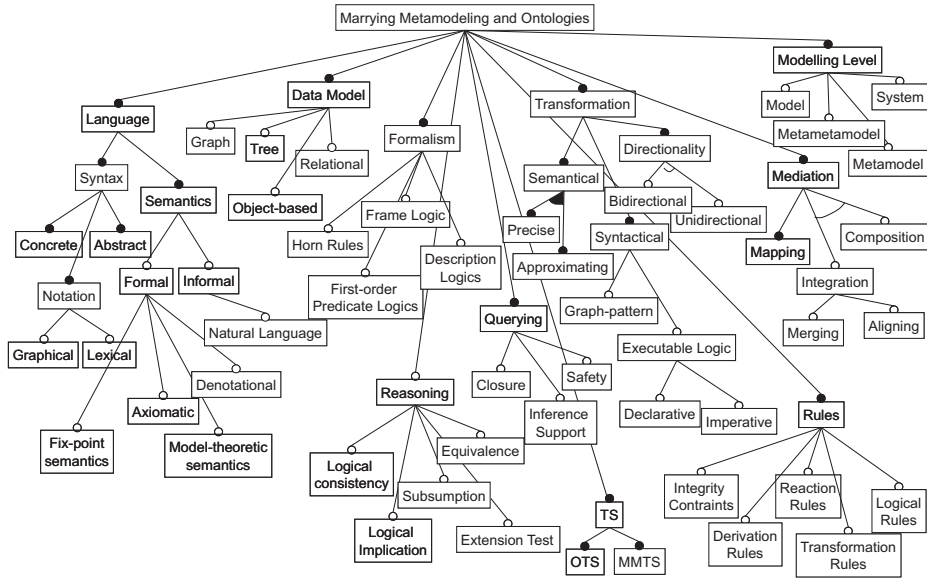


Fig. 1. First version of the Feature Model of Integrating Metamodeling and Ontologies Technical Spaces.

engineering capabilities by or for ontology technologies. We aim at identifying the main characteristics of such approaches and at designing a feature model to enlighten the possible conceptual choices.

Motivation. Model Driven Engineering (MDE) trends to be based on the *Meta-modeling Technical Space* as well as on *Ontological Technical Spaces*. A technical space can then be understood as a body of knowledge comprising modeling languages and transformation facilities.

Further transformation into Ontological Technical Spaces provides additional analysis and implementation support, not as efficiently available in metamodeling technical spaces. Thus, delineating the attributes of integrated approaches leads to improve the understanding of the field and provide requirements for novel techniques.

2.2 Transforming and Weaving Ontologies and Model Driven Engineering

Problem. UML-like models and OWL ontologies comprise constituents that are similar in many respects, like: classes, associations, properties, packages, types, generalization and instances [6]. However, both approaches have their advantages and disadvantages. UML modeling provides means to express dynamic behavior, whereas OWL does not. OWL is capable of inferring generalization

and specialization between classes as well as class membership of objects based on the constraints imposed on the properties of class definitions, whereas UML class diagrams do not allow for dynamic specialization/generalization of classes and class memberships or any other kind of inference *per se*.

Contemporary software development should make use of the benefits of both approaches to overcome their restrictions. This consideration led us to the following challenge: *How can we develop and denote models that benefit from the advantages of the two modeling paradigms?*

Objectives. To overcome such a challenge, our aim is to provide a framework comprising the following building blocks: (i), an integration of the MOF-based metamodels for UML, EMOF, OCL and OWL, (ii), a new MOF-based metamodel for entities that jointly belong to the different paradigms, (iii), the specification of dynamic behavior referring to OWL reasoning (using OCL-like expressions), (iv), the definition of a joint profile for denoting hybrid models, and, (v), closure operations that determine whether entities must diffuse from one paradigm into the other and, hence, how they are to be mapped onto metamodels. Together, these building blocks constitute our original approach to Transform and Weave Ontologies and UML in Software Engineering — *TwoUse*¹.

An outline of such framework is explored in [7]. Figure 4 presents the package organization of the integrated metamodel. Since OCL is used for different languages (EMOF [3], QVT [8], ATL [9], EMF [10]) with different purposes (constraining, querying, initializing, validating), the OCL extension is a pivot metamodel, allowing to plug new metamodels that use OCL into our approach, extending the range of application.

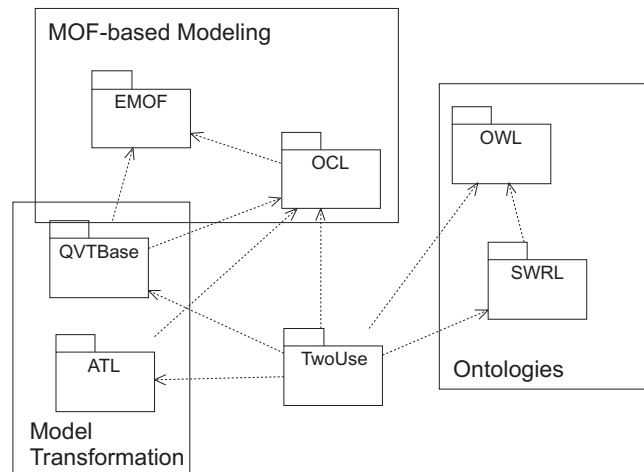


Fig. 2. TwoUse package diagram.

¹ <http://isweb.uni-koblenz.de/Projects/twouse>

Motivation. While mappings between modeling paradigms have been established a while ago (e.g., [6]), the task of an integrated language for UML and OWL models that could be denoted in one hybrid diagram has not been dealt with before. The challenge of this task arises from the large number of differing properties germane to each of the two modeling paradigms (cf. [5] for an analysis) making the integration of UML models and OWL models difficult.

An approach which is able to capture structural and behavioral features of problem domains and which allows modelers to describe the semantics of the domain at the level of an OWL ontology yields improvements on the maintainability and complexity even for development of nonlogical systems.

2.3 Fusing Ontologies into Model Driven Engineering

Problem. Design patterns provide elaborated, best practice solutions for commonly occurring problems in software development. During the last years, design patterns were established as general means to ensure quality of software systems by applying reference templates containing software models and their appropriate implementation to describe and realize software systems. With the arise of MDE, these ideas have been extended to model transformation design patterns [11].

However, in [12] we have observed that, in software design patterns managing variants, the decision of what variant to choose is delegated task to client classes. In model transformation design patterns, the transformation input pattern is tightly bound to the metamodel, causing the review of the transformation model when the source metamodel changes.

Hence, the question arises of *how the selection of specific classes could be determined using only their descriptions rather than by weaving the descriptions into constructs like client classes or transformation rules.*

Objectives. To remedy this problem, we work towards identifying patterns able to decouple class selection from class definition (in UML-like) or from transformation rule (in MDE) by exploiting OWL-DL modeling and reasoning. Thus, we explore Design Patterns that includes OWL-DL modeling and that leads us to a minor, but powerful variation of existing practices.

One of these new design patterns was previously described in [12]. Figure 3 shows the Selector Design Pattern, applied to overcome the drawback of delegation presented by the Strategy Pattern or by the Abstract Factory Pattern, when both Strategy and Abstract Factory are used in conjunction.

For the dissertation, we aspire to recognize slots in software design patterns or model transformation design patterns where an ontology-based approach improves the software quality. To realize the new design patterns, we apply our TwoUse approach in order to allow for joint Metamodeling and OWL-DL modeling.

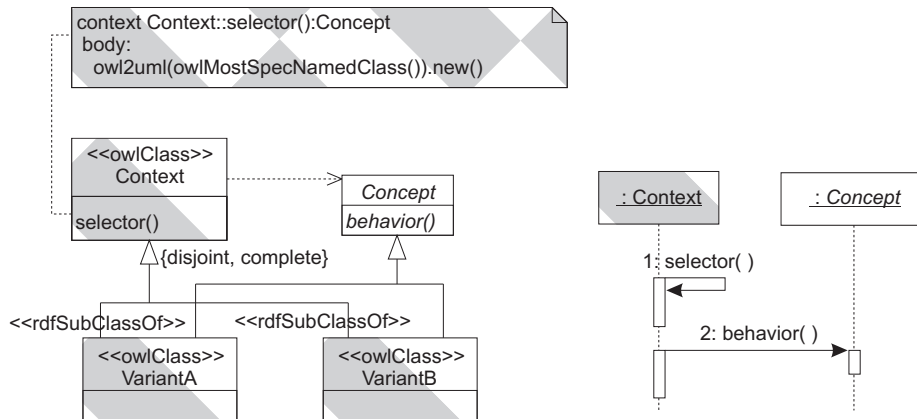


Fig. 3. The Selector Design Pattern of integrating UML and OWL.

Motivation. Solutions based on patterns like Strategy embed the treatment of variants into clients' code at various locations, leading to an unnecessary tight coupling of classes. This issue has already been identified by [13] as a drawback of pattern-based solutions e.g. when discussing the Strategy Pattern and its combination with the Abstract Factory Pattern.

In MDE, approaches dealing with automatic generation of model transformation specification already use upper ontologies and subsumption reasoning to automatically generate transformation rules. Nevertheless, changes in the metamodel still require the regeneration of the transformation model. Using the TwoUse approach in model transformation will enable to avoid the regeneration, allowing the decision of which class to use to be handled at runtime.

2.4 Fusing Model Driven Engineering into Semantic Web

Web Service Specification with MDE and Ontologies: TwoUse Application. Using other UML diagrams, like the sequence diagram.

Problem. Much attention has been given to ontology mapping to Semantic Web Services Specification. However, as these tasks get more complex, current approaches fail to provide clarity and accessibility to the modelers who need to see and understand the semantic as well as the lexical/syntactic part of the specification. The modelers usually can modify them only in a very intricate and disintegrated manner, drawing his attention away from the core task proper down into the diverging technical details of the solution used..

From this scenario, *the problem of supporting generative techniques in ontology field, like ontology mapping or semantic web services specification, emerges, adding expressiveness without going into platform specifics, i.e. how to fill the abstraction gap between specification languages and programming languages.*

Objectives. We propose a representation approach for generative specification of ontology tasks based on model-driven engineering (MDE). In order to reconcile semantic reasoning with idiosyncratic lexical and syntactic translations, we integrate the different layers into a representation based on a joint meta model. The joint meta model comprises description logics to specify, represent and execute semantic operations as well as OCL constraints for specifying lexical and syntactic translations.

A preliminary effort into this direction is the language for Model Driven Specification of Ontology Translations presented in [14]. Figure ?? The paper presents an ongoing solution for ontology translation specification that intends to be more expressive than ontology mapping languages and less complex and granular than programming languages. The solution comprises a concrete syntax supported by the TwoUse integrated metamodel.

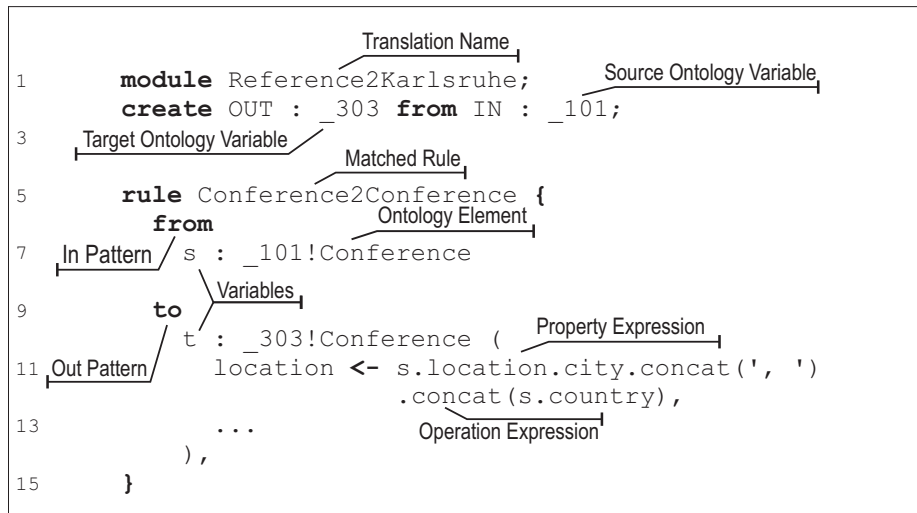


Fig. 4. Snippet of a rule in the specification of ontology translations

Motivation. Filling the gap in ontology translation domain between ontology mapping languages and general purpose programming languages helps to improve productivity, since modelers will not have to be aware of platform-specific details. Moreover, maintenance and traceability would be facilitated because mapping knowledge is not longer embedded in source code of programming languages anymore.

3 Methods and Validation

Firstly, we investigate Problem 2.1 by carrying out a literature survey. A domain analysis is undertaken over approaches integrating metamodeling technical spaces and ontological technical spaces found in literature. Domain analysis is concerned with analyzing and modeling the variabilities and commonalities of systems or concepts in a domain [15].

The research result is a descriptive model, characterized by a feature model for the problem area. The feature model is then validated using formal semantics and reasoning support of OWL as describe in [16].

The feature model is used as requirements to undertake Problem 2.2. A case study will help to improve the framework for further applications. The research result is technique involving the integration of different metamodels — The TwoUse approach. The TwoUse approach will be firstly validated against the requirements resulting from Problem 2.1, following by case study requirements. Such validation is helpful to detect, develop, refine frames of reference.

In the case study, we apply reverse engineering to the software developed internally in the study group, x-cosimo [17], in order to have UML models. Since TwoUse aims at improve reuse and maintainability, these UML models will be compared with TwoUse models using software quality metrics for such attributes.

Using the TwoUse framework, we examine the literature of design pattern identifying patterns that could benefit from an integrated approach and propose new patterns. Since it is hard to measure quality improvements of design patterns, we intend to show that ontology-based design patterns make it possible to do something that was impossible heretofore.

For the semantic web technologies, we apply twouse to propose generative approaches, i.e., ontology translation specification or semantic web service specification. Since we claim that TwoUse improves comprehensibility and productivity, we assess the size of the produced TwoUse models from the perspective of the user against current approaches. As domain for ontology translation we use two ontologies of bibliographic references from the test library of the Ontology Alignment Evaluation Initiative (OAEI) [18]. For the semantic web service specification...

References

- [1] OMG: Unified Modeling Language: Superstructure, version 2.1.1. (February 2007)
- [2] McGuinness, D.L., van Harmelen, F.: OWL Web Ontology Language overview (February 2004) Available at <http://www.w3.org/TR/2004/REC-owl-features-20040210/>.
- [3] OMG: Meta Object Facility (MOF) Specification. (November 2005) Available at: <http://www.omg.org/cgi-bin/doc?formal/2005-11-01.pdf>.
- [4] Frankel, D., Hayes, P., Kendall, E., McGuinness, D.: The model driven semantic web. In: 1st International Workshop on the Model-Driven Semantic Web (MDSW2004), Monterey, California, USA (2004)

- [5] Silva Parreiras, F., Staab, S., Winter, A.: On marrying ontological and metamodelling technical spaces. In: ESEC/FSE'07, Cavtat near Dubrovnik, Croatia, ACM Press (September 2007)
- [6] OMG: Ontology Definition Metamodel. (October 2006) Available at <http://www.omg.org/cgi-bin/doc?ptc/07-09-09.pdf>.
- [7] Silva Parreiras, F., Staab, S., Winter, A.: TwoUse: Integrating UML models and OWL ontologies. Technical Report 16/2007, Universität Koblenz-Landau (2007) Available at http://www.uni-koblenz.de/~aggrimm/arbeitsberichte/arbeitsberichte_16_2007.pdf.
- [8] OMG: MOF QVT Final Adopted Specification. (July 2007) Available at <http://www.omg.org/cgi-bin/apps/doc?ptc/07-07-07.pdf>.
- [9] Jouault, F., Kurtev, I.: Transforming models with ATL. In: Satellite Events at the MoDELS 2005 Conference. Volume 3844 of LNCS., Jamaica, Springer (2005)
- [10] Budinsky, F., Brodsky, S.A., Merks, E.: Eclipse Modeling Framework. Pearson Education (2003)
- [11] Bézivin, J., Jouault, F., Paliès, J.: Towards model transformation design patterns (2005)
- [12] Silva Parreiras, F., Staab, S., Winter, A.: Improving design patterns by description logics: A use case with abstract factory and strategy. In: Modellierung 2008. LNI, GI (March 2008)
- [13] Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design patterns: elements of reusable object-oriented software. Addison-Wesley Professional (1995)
- [14] Silva Parreiras, F., Staab, S., Winter, A.: Model driven specification of ontology translations (2008) Available at <http://www.uni-koblenz.de/Projects/twouse/mdsot.pdf>.
- [15] Czarnecki, K.: Domain engineering. In: Encyclopedia of Software Engineering. John Wiley & Sons, Inc. (2002)
- [16] Wang, H., Li, Y.F., Sun, J., Zhang, H., Pan, J.: Verifying Feature Models using OWL. *Journal of Web Semantics* **5** (June 2007) 117–129
- [17] Franz, T., Staab, S., Arndt, R.: The x-cosim integration framework for a seamless semantic desktop. In: K-CAP 2007 – Proceedings of the Fourth International ACM Conference on Knowledge Capture, Whistler, BC (10 2007)
- [18] Euzenat, J.: Ontology Alignment Evaluation Initiative (Mai 2007) available at: <http://oaei.ontologymatching.org/>.

Appendix - Structure of the Dissertation

1. Introduction and Overview. Motivation, contribution, problem description, overview, reader's guide.
2. Part I Foundations
 - (a) Technical Spaces
 - i. Ontology Technical Spaces
 - A. Ontology Modeling
 - B. Ontology Languages. Ontology languages and ontology markup languages
 - ii. Model Driven Engineering Technical Spaces
 - iii. Grammar Technical Spaces

- (b) Ontology Technical Spaces vs. Grammar Technical Spaces. OWL and object orientation.
 - (c) Ontology Technical Spaces vs. MDE. OWL and MDE
 - (d) Requirements for integrating Ontology Technical Spaces and MDE
 - (e) Related Work
3. Part II - The TwoUse Approach
- (a) TwoUse Conceptual Architecture
 - i. UML Profiles
 - ii. Metamodels
 - iii. Model Library
 - iv. OCL to query Ontologies
 - v. Semantics
 - (b) The Role of TwoUse in MDE
 - i. TwoUse Process
 - ii. Model Transformations
4. Part III - Realization
- (a) Implementation. Tools
 - (b) Application
 - i. Supporting Model Transformation with Ontologies
 - ii. Improving Design Patterns by Description Logics
 - iii. Model Driven Specification of Ontology Translations
 - iv. Web Service Specification with MDE and Ontologies
 - (c) Evaluation
5. Part IV - Next Generation of MDE
- (a) Conclusion
 - (b) Next Steps