Menthor Editor: an ontology-driven conceptual modeling platform

João MOREIRA[a,1], Tiago Prince SALES[b,c,d], John GUERSON[c,d], Bernardo Ferreira Bastos BRAGA[c,d], Freddy BRASILEIRO[c,d], Vinicius SOBRAL[c]
[a] *University of Twente, Enschede, The Netherlands*
[b] *Institute of Cognitive Sciences and Technologies ISTC CNR*
Laboratory for Applied Ontology (LOA), Trento, Italy
[c] *Federal University of Espírito Santo,*
Ontology & Conceptual Modeling Research Group (NEMO), Vitória (ES), Brazil
[d] *Menthor company© , Vitória (ES), Brazil*

Abstract. The lack of well-founded constructs in ontology tools can lead to the construction of non-intended models. In this demonstration we present the Menthor Editor, an ontology-driven conceptual modelling platform which incorporates the theories of the Unified Foundational Ontology (UFO). We illustrate how UFO categories can improve the design of domain ontologies. Moreover, the verification and validation approaches are demonstrated with ontologies of our catalogue. The complete execution of the model-driven engineering is exemplified, including situation modelling.

Keywords. Ontology-driven Conceptual Modelling, UFO, OntoUML, Menthor.

1. Introduction

A challenge to the modelling of ontologies is the lack of well-founded structural and temporal constructs of the conventional design techniques. Ontology-driven conceptual modelling has been successfully applied to overcome this issue, where ontological analysis based on a foundational ontology supports the development of well-founded ontologies. In this demo we cover an ontologically well founded language named OntoUML, which is based on The Unified Foundational Ontology (UFO) [1]. Also, we present a model-driven engineering (MDE) platform that supports OntoUML modelling. This modelling tool has been developed as an academic effort for several years under the name of OntoUML Lightweight Editor (OLED) [2-5]. Recently, OLED has been entirely refactored and transformed into a commercial tool, named Menthor Editor[2]. Our goal is to demonstrate how theories behind UFO research can be used in practice by exemplifying Menthor Editor's features with diverse domain ontologies from our catalogue. These features include the use of UFO stereotypes and inherited rules, the model verification and validation approach and situation modelling. This paper is structured as follows: Section 2 presents the ontology-driven conceptual

[1] Corresponding Author.
[2] http://www.menthor.net/

modelling process and some capabilities of Menthor Editor. Section 3 describes what will be demonstrated and how the contribution will be illustrated interactively. Finally, we conclude the paper with the expected contributions with this demo.

2. Menthor platform

In this section we describe the main features of Menthor Editor, following the full description presented in [2]. First, we describe the MDE approach to ontology engineering used in the editor. Second, we present some of the main features for model verification and validation, such as syntactic rules with OCL support and visual simulation with Alloy Analyzer. Then, we introduce on going work with the editor for the Situation Modelling Language (SML) in EA, which is integrated to OntoUML and used in the visual simulation process. Finally, we discuss how OntoUML models can be transformed to OWL and SWRL following some design criteria. Figure 1 illustrates the ontological MDE approach, where greyed activities are supported by the Menthor Editor. In this approach, the cyclical process of modelling, verifying and validating is performed until the domain ontology achieves the intended quality.

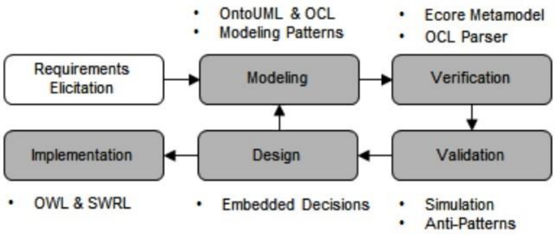

Figure 1. Ontological model-driven engineering supported by Menthor.

There are two ways to model domain ontologies with the Menthor Editor. First, the tool provides a class diagram interface with OntoUML stereotypes (Figure 2). Second, Sparx's Enterprise Architecture[3] (EA) tool may be used for modelling, where the models may be exported to Menthor Editor using an OntoUML plug-in for EA, i.e. a UML profile that reflects OntoUML meta-model, implemented with the MDG technology[4]. Domain ontologies are modelled in OntoUML, having constraints formalized with OCL. Menthor Editor provides an OCL editor with syntax verification (parsing) to textual constraints, as well as syntax highlight and code-completion. Moreover, Menthor supports the representation of dynamic invariants through temporal OCL, a method depict in [6].

The OntoUML meta-model in Menthor Editor is defined in ECore and incorporates the syntactical rules of the OntoUML language. Automatic verification of these rules is supported by the Menthor Editor, assuring that the domain ontology respects the syntactical rules of OntoUML.

Validation can be performed to rule out unintended state of affairs through visual simulation and anti-pattern detection. Ontological anti-patterns "are configurations that when used in a model will typically cause the set of valid (possible) instances of that

[3] http://www.sparxsystems.eu/enterprisearchitect/

[4] http://www.sparxsystems.com.au/resources/mdg_tech/

model to differ from the set of instances representing intended state of affairs in that domain" [2]. Menthor Editor has a catalogue of anti-patterns, described in [7] and an anti-pattern management process with automatic detection, guided analysis and automatic refactoring. Visual simulation is provided through Alloy Analyzer, which automatically generates object diagrams of instances of the model that the user may inspect to find if the model can represent intended or unintended state of affairs.

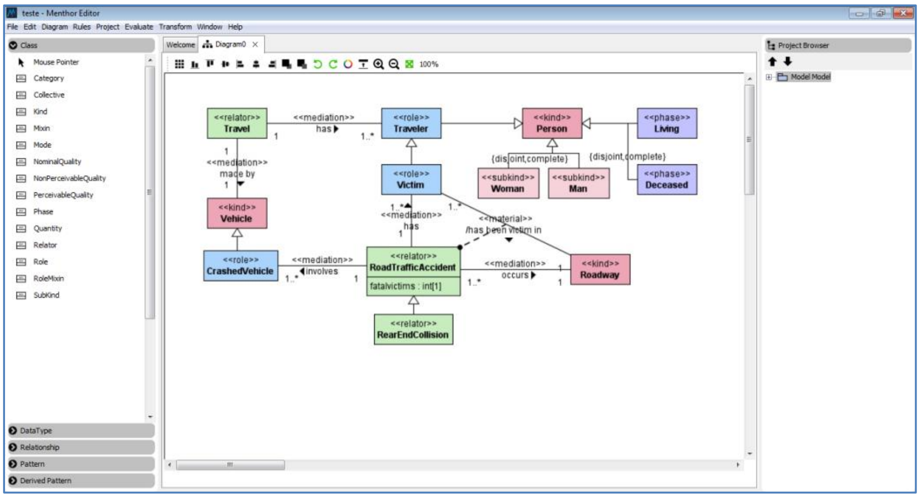

Figure 2. A domain ontology in Menthor interface.

Ongoing work in SML is described in [8] and includes the creation of a SML editor in EA, following a similar approach of the OntoUML plug-in. Therefore, situation types can be modelled in EA, having structural aspects defined with OntoUML. The designer can validate the situation models by exporting them (via XMI) to the Menthor Editor.SML provides the specification of the notion of a situation, i.e. a configuration of part of reality that can be understood as a whole, and is described in [8]. Through SML the designer can focus on the high-level patterns that emerge in time by specifying the events that trigger a situation type. The designer defines a set of rules among structural properties (from OntoUML) in a visual way. In Figure 3 we illustrate a SML model of the situation type "fever", which is triggered when the temperature of a patient is greater than 37. The integration of SML with OntoUML was introduced in [9] and takes advantage of the Alloy visual validation approach within Menthor Editor [10].

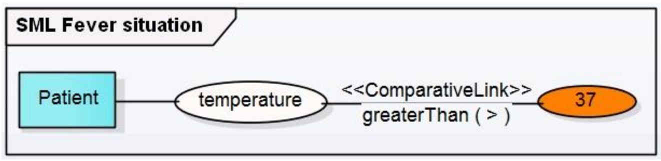

Figure 3. A SML model with the situation type "fever".

Finally, the domain ontology implementation can be automatically generated in OWL and SWRL through model transformations, taking design decisions into account. Menthor Editor presents a set of settings to configure the transformation approach, including filters, axioms and data types' selection. While requirements elicitation is not covered by Menthor, the integration with EA enables software developers to use EA capabilities for requirements management along with OntoUML models.

3. Demo: modelling well-founded domain ontology

We aim on demonstrating how a software developer can take advantage of the main features of Menthor Editor's platform, which are the result of years of research involving UFO [1]. A number of ontologies have been developed with the Menthor Editor (or in last versions of OLED) in diverse domains, which were assessed and described in [7] and are available in Menthor's model repository[5].We plan on exemplifying each capability of Menthor in the different domains of our catalogue. For example, the OntoUML syntactic checker and OCL constraint editor will be illustrated in the road traffic accident ontology, as described in [2]. In addition, the genealogical ontology will be used to exemplify the representation of dynamic invariants with the temporal OCL approach [6]. The validation approach with visual simulation and anti-patterns detection will use each example of [7] to exemplify the application of semantic anti-patterns. For example, the association cycle anti-pattern and Alloy visual validation will be illustrated with the organizational ontology O3. The binary relation between overlapping types anti-pattern will be illustrated with the transportation regulation ontology (MGIC). The imprecise abstraction anti-pattern will be illustrated with the Electrocardiogram (ECG) ontology. The relation specialization anti-pattern will be illustrated with the OntoBio ontology. The relator mediating overlapping types anti-pattern will be illustrated with the service ontology (UFO-S). The repeatable relator instances anti-pattern will be illustrated with the configuration management task ontology (CMTO). All these examples will be shown in an interactive way with the audience, which will be able to participate and experiment with the tool.

Moreover, we plan to show the execution of the entire process in the development of a domain ontology for the construction of a software. In particular, we plan on illustrating how an early warning system for the detection of disease outbreaks can be designed with Menthor Editor platform by following the example of [9]. Situation types within this application domain, e.g. possible contagion and epidemics spread, are specified with SML. OntoEmerge, a disaster core ontology [8], will be used as the domain ontology representing the structural aspects and containing healthcare elements, such as patient, hospital and exam. We will introduce common design errors in this ontology and use Menthor Editor's capabilities to illustrate how to address them. Furthermore, we intend to illustrate how the specification generated in Menthor platform with the situation type definitions can support the implementation with the rule-based approach described in [11]. Clearly, the contribution of using Menthor Editor to improve ontology construction by taking advantage of the inherited theories, is better described in a demonstration session.

4. Conclusion

In this demo we intend to illustrate UFO theories in a practical way through the Menthor Editor, the commercial tool built based on the OLED. Each feature will be

[5] http://www.menthor.net/browse-models.html

illustrated with the support of domain ontologies from our catalogue. Moreover, a complete execution of the ontological model-driven engineering approach will be illustrated in a specific case. We expect to leverage the research in ontology-driven conceptual modelling as a result from this demo.

References

[1] G. Guizzardi, G. Wagner, and R. D. A. Falbo, "Towards Ontological Foundations for Conceptual Modeling: The Unified Foundational Ontology (UFO) Story," 2015.

[2] J. Guerson, T. P. Sales, G. Guizzardi *et al.*, "OntoUML Lightweight Editor: A Model-Based Environment to Build, Evaluate and Implement Reference Ontologies," *2015 IEEE 19th International Enterprise Distributed Object Computing Workshop*, pp. 144-147, 2015.

[3] G. Guizzardi, and V. Zamborlini, "Using a trope-based foundational ontology for bridging different areas of concern in ontology-driven conceptual modeling," *Science of Computer Programming,* vol. 1, pp. 1-27, 2014.

[4] P. P. F. Barcelos, C. C. Reginato, M. E. Monteiro *et al.*, "On the importance of truly ontological distinctions for standardizations: A case study in the domain of telecommunications," *Computer Standards and Interfaces,* vol. 44, pp. 28-41, 2016.

[5] M. R. Blackburn, and P. O. Denno, "Using Semantic Web Technologies for Integrating Domain Specific Modeling and Analytical Tools," *Procedia Computer Science,* vol. 61, pp. 141-146, 2015.

[6] J. Guerson, and J. P. A. Almeida, "Representing Dynamic Invariants in Ontologically Well-Founded Conceptual Models," 2015.

[7] T. P. Sales, and G. Guizzardi, "Ontological anti-patterns: Empirically uncovered error-prone structures in ontology-driven conceptual models," *Data and Knowledge Engineering,* vol. 99, pp. 72-104, 2015.

[8] J. L. R. Moreira, L. Ferreira, M. V. Sinderen *et al.*, "Towards ontology-driven situation-aware disaster management," *Applied Ontology,* vol. 10, pp. 339-353, 2015.

[9] V. M. Sobral, J. P. A. Almeida, and P. D. Costa, "Assessing Situation Models with a Lightweight Formal Method," *IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, 2015.

[10] V. M. Sobral, "A model-driven approach to the conceptual modeling of situations: from specification to validation," 2015.

[11] P. D. Costa, J. P. A. Almeida, I. S. A. Pereira *et al.*, "Rule-Based Support for Situation Management," *Fusion Methodologies in Crisis Management,* 2016.