Towards an Eclipse Ontology Framework: Integrating OWL and the Eclipse Modeling Framework

Carsten Schneider[1]

WeST — Institute for Web Science and Technologies,
University of Koblenz-Landau
Universitaetsstrasse 1, Koblenz 56070, Germany
caschneider@uni-koblenz.de

Abstract. Model driven software development and semantic web technologies are spreading increasingly. But today semantic web technologies are rarely used in the software development, even though the additional use of these technologies could improve the software development process. One reason for the rare use of these technologies in the software development is that semantic web technologies are barely integrated with the established software development tools. This paper proposes an approach for integrating semantic web technologies with model driven software development.

1 Introduction

In today's software development models have an important role. The developers create several models, which they write down in modeling languages like UML [1] or Ecore [2]. These models are used for the specification and documentation of software. Therefore the relevance of the models in the software development already is comparable to the relevance of an architect's plan in putting up a building.

In model driven software development a further step is taken. In addition the developers use the models as input for a code generator. The generator implements a skeletal structure for the software on the basis of the models. The generated code can be modified and extended by the developer. The Eclipse Modeling Framework (EMF) [2] is a framework for model driven software development, which is frequently used. The Eclipse Modeling Framework uses its own metamodel Ecore and provides mappings from some other modeling languages like UML to Ecore. The Ecore models are the basis for the generation of the Java code.

During the last years semantic web technologies, which are based on ontologies, have spread increasingly. With the spread of these technologies the desire for using these technologies also for the software development arised. It was begun to deal with ontology driven software development. In the ontology driven software development ontologies are used as basis for the code generation. An

ontology is a formal representation of knowledge. The knowledge is described by concepts and the relations between these concepts. The Web Ontology Language (OWL) [3] is a language for the definition of ontologies. OWL allows the definition of classes by constraints on the properties of their members. It is possible to execute queries on ontologies by using a query language like SPARQL [4]. Although ontologies also are models, any mention of models in this paper refers to classical UML-like models, if there is no other comment.

Both models and ontologies have different advantages and disadvantages. Ontologies define classes with logical expressions. The classes, which are defined by an ontology, have a very flexible, dynamic class membership. But ontologies do not provide an opportunity to specify the dynamic behaviour. In contrast models enable the developer to specify the dynamic behaviour. But models use a static class membership.

For future development it is desirable to combine model driven software development and ontology driven software development with each other to use the advantages of both and to avoid their restrictions. Such a combination must take the differences of models and ontologies into account. These differences and approaches how they can be bridged are handled by [5, 6]. The question is how model driven software development and ontology driven software development can be combined so that the functionality of the model driven development is maintained, the model driven developers can use a familiar environment and some advantages of the ontology driven development are additionally available.

This paper proposes an approach for such a combination. This approach extends the functionality of the Eclipse Modeling Framework by the possibility to use semantic web technologies. It provides additional features to developers of the model driven community like defining OWL property restrictions for attributes and references or attaching queries to operations. The generated code prohibits the violation of the property restrictions and executes the attached query during an execution of the operation. Developers of the ontology driven community get a framework, which provides them the opportunity to develop software ontology driven, to model the dynamic behaviour and to attach queries to operations. The framework could also be used to create an ontology and to populate the created ontology with instances.

2 Running Example

This section introduces a simple purchase order example as running example. Figure 1 shows an Ecore diagram for the example. The example includes three classes: `Store`, `Customer` and `Order`. There are containment references from `Store` to `Customer`, from `Store` to `Order` and from `Customer` to `Store`. In addition there is an reference `previousOrders` from `Order` to `Order`. An order has one operation called `getOrders`. We want that a SPARQL query is executed during a call of `getOrders` and we want that `previousOrders` is transitive.

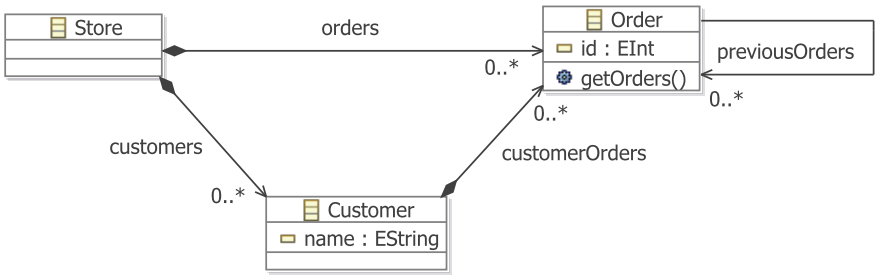

Fig. 1. Purchase Order Ecore Diagram

2.1 Challenges

The challenge is to generate the whole code of this example. With model driven development frameworks we had to implement both the code for the transitivity of `previousOrders` and the code for the query execution by hand. With ontology driven development frameworks we could generate the code for `previousOrders`, but we had to write the whole code of `getOrders` by hand. With the approach of this paper it would be possible to generate the code for both `getOrders` and `previousOrders`.

2.2 Advantages

The proposed approach has several advantages for developers. Developers of the ontology driven community get the opportunity to model the behaviour. This makes it possible to generate more code and reduces the programming effort. By describing the behaviour of an operation with a SPARQL query the opportunity to query a generated ontology is provided. This could simplify the implementation of operations for developers of the ontology driven community and the model driven community.

Developers of model driven software get the opportunity to use property restrictions for attributes and references. For instance this allows the definition of transitive references. The maintenance of the EMF functionalities improves the acceptance in the EMF community and the use of UML-like models simplifies the entry in the approach for software developers.

The approach provides the opportunity to create ontologies based on a class diagram to developers, who are not from the semantic web area. Editors for the creation of ontologies and the population of these ontologies are provided. These editors could simplify the creation of ontologies and support the avoidance of inconsistencies.

3 Integrating OWL and EMF

3.1 Objectives

The proposed work should be an approach for combining model driven software development and ontology driven software development. A tool should be implemented that fulfills the following requirements:

- It generates Java code based on models and semantic web technologies.
- It uses an Ecore model with annotations of the semantic web area as input for the code generator.
- It should be possible to annotate property restrictions to attributes and references and to annotate queries to operations.
- The generated code extensively implements the semantics of the Ecore model and its annotations.
- Its code generator extends the EMF Generator and supports code regeneration like the EMF Generator does.
- It is integrated into the TwoUse Toolkit [7].

3.2 Annotated Ecore Model

The tool uses an Ecore model with annotations of the semantic web area as basis for the code generation. For instance it should be possible to annotate a SPARQL query to an operation of an Ecore class to describe the behaviour of this operation. The generated code of this operation contains code that executes this SPARQL query. The Ecore annotation connects the SPARQL query file with the operation. The path of the SPARQL query is derived from the source property of the annotation and inserted into the generated code during the code generation.

In addition references and attributes of an Ecore class can be annotated by OWL property restrictions. For instance it should be possible to define that a reference or an attribute should be transitive. Therefore Ecore annotations are attached to the reference respectively the attribute. The value of the source properties of these annotations define the types of the restrictions. The references properties of the annotations can be used to define other elements of the Ecore models as parameters for the restrictions.

For the purchase order example we would create the SPARQL query for `getOrders` and add an EAnnotation to `getOrders` in the Ecore model, whose source property we set to the path of the SPARQL query. In addition we would add an EAnnotation with source `#ObjectProperty` to `previousOrders` and an EAnnotation with source `#TransitiveObjectProperty` to this Annotation.

3.3 Extension of the EMF Generator

The tool extends the EMF generator with the facility of generating code corresponding to the annotations of the semantic web area. In addition the tool

should provide the possibility to use the original functionality of the EMF Generator. For the implementation of the code generator the JET templates [2] of the EMF generator are extended. The code generator of the tool should use code merging like the EMF generator to support the regeneration of the code so that developers can mark hand-modified code to prevent overwriting of the hand-modified code.

3.4 Generated Code

The generated code extensively implements the semantics of the Ecore model and its annotations of the semantic web area. The generated code is based on the code, which the EMF generator would generate for the Ecore model. An ontology that models the semantics of the Ecore model and its annotations is generated. The Ecore model is transformed into the TBox of the ontology. The ABox of the ontology includes the instances of the Ecore model. For the purchase order example the TBox would include the `Store`, `Customer` and `Order` classes and their properties and the ABox would include the `Store`, `Customer` and `Order` instances and their properties.

The generated code uses the generated ontology for implementing the semantics of the annotated Ecore model. The generated code for an operation, that is associated with a SPARQL query, includes code that executes the annotated query on the generated ontology. The path of the query is derived from the source property of the annotation and stored into a variable of the generated code.

The generated ontology is also used to ensure that the property restrictions, which are associated with a reference or an attribute are not violated. According to the EMF code generation the generated code for multiplicity-many attributes and references includes only a get method, which retrieves an `EList`, and no set method. For multiplicity-many attributes and references, which are associated with property restrictions, the retrieved list is a special list, whose methods prohibit the violation of these restrictions. For the purchase order example the get method for `previousOrders` would retrieve a `TransEList`, whose methods use the generated ontology to ensure, that the transitivity is maintained.

3.5 Integration into the TwoUse Toolkit

The tool will be implemented as an Eclipse plugin. Therefore it can be integrated into the TwoUse Toolkit without much effort. The TwoUse Toolkit is a toolkit that already provides many features that support the model driven development and the ontology engineering. Because of the integration into the TwoUse Toolkit the tool can use these features. For instance the OWLizer of the TwoUse Toolkit can be used for the generation of an ontology from an annotated Ecore model. By the integration of the tool into the TwoUse Toolkit a software development environment is provided that supports model driven software development as well as a combination of model driven and ontology driven software development.

4 Related Work

There are several works that deal with the use of semantic web technologies in the software development. In this section some of these works are presented and briefly summarized.

The work of Andreas Eberhart [8] deals with the automatic generation of inference systems from RDF-Schema and RuleML sources. The generation is performed by a compiler, which was especially developed for this purpose. One can choose between the OntoJava compiler and the OntoSQL compiler. The OntoJava compiler generates Java code for an object database with a built-in system of rules. The OntoSQL compiler generates SQL code for an IBM DB2 database server. Within this work several mappings from RDF-Schema and RuleML into Java code respectively SQL code are described. The work of Andreas Eberhart focusses on the ontology driven software development. Therefore it does not provide support for the model driven software development.

In the work of Kalyanpur et al. [9] a mapping from OWL sources to Java code is presented. In this approach OWL classes are mapped to Java classes and Java interfaces. Moreover it is described how OWL properties can be embedded into the Java classes and Java interfaces. The restrictions, which are defined by the OWL properties, are modeled by several listener implementations. The mapping of this work was used in the Ontology Creator module of the HarmonIA Framework. This work focusses on the ontology driven software development. It does not provide the advantages of the model driven software development like the possibility to model the dynamic behaviour.

Holger Knublauch [10] treats the question how a semantic web application can be wisely developed in an ontology driven way. Therefore an example of the tourism domain is used to introduce the developed software architecture. Knublauch divides the application into a semantic web layer and an internal layer. The semantic web layer contains the ontologies and provides them to other applications. In addition the ontologies serve as control for the internal behaviour and can be used for the generation of some internal components. The internal layer contains control and reasoning mechanisms. It is recommended that Protégé with the OWL Plugin is used for the development. Therewith it is possible to create ontologies, check their consistency and generate the corresponding Java classes. Holger Knublauch focusses on the development of semantic web applications and does not treat the question how the semantic web technologies could be used for the development of other applications.

Puleston et al. [11] are concerned with the integration of objectoriented and ontological representations. They consider three possible approaches for the combination of OWL and Java: a direct approach, an indirect approach and a hybrid approach. The direct approach uses the ontology as draft for the program classes. The OWL model is statically transformed into a corresponding Java representation. In the indirect approach the java classes do not model the concepts of the domain. Instead of that the Java classes access an external OWL model. The third approach is a hybrid of the direct and indirect ones. In the hybrid model the Java classes model a limited part of the ontology. This part is dynamically

extended by an OWL model. The work of Puleston et al. focuses on the hybrid approach and illustrates the hybrid approach with the help of a case study of a medical records system. Puleston et al. treat the integration of ontological representations with programs but do not use them for the development process of the programs.

Hillairet et al. [5] treat an approach how EMF applications and RDF data sources can cooperate with each other. They present a system, which allows to instantiate an EMF object from a RDF data source and to serialize EMF objects into RDF data sources. Therefore a domain specific mapping language is offered that allows to define mappings between elements in the EMF domain model and elements in an OWL/RDF ontology. This mapping language can also be used to define how an ontology should be built from a domain model. Because of this it is possible to create a RDF data source from a domain model and store model objects in the created RDF data source. Hillairet et al. do only treat the cooperation between EMF applications and RDF data source and not the question how the ontologies could be used during the creation of applications.

The work Ontology Definition Metamodel of the OMG [12] defines metamodels and UML profiles for OWL, RDF and Topic Maps. In addition this work describes several mappings like mappings from an UML model into an OWL model. This work focusses on making the concepts of the model driven development applicable to the engineering of ontologies, but does not cover the question how ontologies could improve the model driven development.

5 Conclusion

This paper proposes an approach for combining model driven and ontology driven development. The approach is based on the Eclipse Modeling Framework and its metamodel Ecore. It is intended to extend the EMF generator by adding the possibility to use semantic web technologies for the code generation and within the generated code. Therefore the extended code generator gets Ecore models with annotations of the semantic web area as input.

The integration of the extended generator into the TwoUse Toolkit achieves that a rich software development environment is provided, which could improve the software development process. In addition it could be used to create an ontology and to fill the created ontology with instances. The proposed approach provides additional features to developers of the model driven community and of the ontology driven community. Because developers of the model driven community can use the semantic web technologies in a familiar environment, it could support a broader adoption of these technologies in this community.

References

1. OMG: Unified Modeling Language: Superstructure, version 2.2. Object Modeling Group. (2009)

2. Steinberg, D., Budinsky, F., Paternostro, M., Merks, E.: EMF: Eclipse Modeling Framework 2.0. Addison-Wesley Professional (2009)
3. Knublauch, H., Oberle, D., Tetlow, P., Wallace, E.: A Semantic Web Primer for Object-Oriented Software Developers. W3c working group note, W3C (2006)
4. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF (Working Draft). Technical report, W3C (2008)
5. Hillairet, G., Bertrand, F., Lafaye, J.: Bridging EMF applications and RDF data sources. In: SWESE 2008, 4th Workshop on Semantic Web Enabled Software Engineering, workshop at ISWC 2008, the International Semantic Web Conference 2008, Karslruhe, Germany (2008) 26–40
6. Parreiras, F.S., Staab, S., Winter, A.: On Marrying Ontological and Metamodeling Technical Spaces. In Crnkovic, I., Bertolino, A., eds.: ESEC/SIGSOFT FSE. (2007) 439–448
7. WeST Web Science and Technologies: TwoUse Project. Website `http://code.google.com/p/twouse/`.
8. Eberhart, A.: Automatic Generation of Java/SQL Based Inference Engines from RDF Schema and RuleML. In Horrocks, I., Hendler, J.A., eds.: International Semantic Web Conference. Volume 2342 of Lecture Notes in Computer Science., Springer (2002) 102–116
9. Kalyanpur, A., Pastor, D.J., Battle, S., Padget, J.A.: Automatic Mapping of OWL Ontologies into Java. In Maurer, F., Ruhe, G., eds.: SEKE. (2004) 98–103
10. Knublauch, H.: Ontology-Driven Software Development in the Context of the Semantic Web: An Example Scenario with Protege/OWL. In Frankel, D.S., Kendall, E.F., McGuinness, D.L., eds.: 1st International Workshop on the Model-Driven Semantic Web (MDSW2004). (2004)
11. Puleston, C., Parsia, B., Cunningham, J., Rector, A.L.: Integrating Object-Oriented and Ontological Representations: A Case Study in Java and OWL. In Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T.W., Thirunarayan, K., eds.: International Semantic Web Conference. Volume 5318 of Lecture Notes in Computer Science., Springer (2008) 130–145
12. OMG: Ontology Definition Metamodel. Object Modeling Group. (2009)