Exposé for B.Sc. Thesis

Megamodel-driven Traceability Recovery & Exploration in an O/R/X-Mapping scenario along the mereological aspects of software artifacts

Maximilian Meffert
210101205

University of Koblenz-Landau

1 Introduction

This exposé[1] outlines the thesis:

Megamodel-driven Traceability Recovery & Exploration in an O/R/X-Mapping scenario along the mereological aspects of software artifacts

for acquiring the degree Bachelor Science (B.Sc.) in Computer Science.

The central topic of the thesis will be the study of MegaL's traceability capabilities.

1.1 Road-map

Section 2 motivates the topic of the thesis. 3 gives a short overview over the necessary background information. 4 defines a preliminary hypothesis and formulates the research questions. 5 specifies the important objectives for the thesis. 6 describes the approach and methodology of the thesis. Eventually 7 outlines the interim structure of the thesis.

2 Motivation

A common task for during the development of software systems is to persist and serialize a domain model. Consider a simple web-service where data is stored in a database and served via HTTP in serialized form, e.g. XML or JSON.

3 Background

The thesis will be based but is not limited to aspects of the following topics:

[1] `http://www.softlang.org/info:expose`

3.1 Ontologies

3.2 Traceability

3.3 Mereology

Mereology is the systematic study of the relationships between wholes and part [6]. The core of its discourse is the Parthood- or partOf -Relationship, which is axiomatized as follows:

Definition 1 (partOf). *We denote:*

$$x \ partOf \ y$$

and mean that x is constituent for y. We assume that:

P1 $x \ partOf \ x$ *(Reflexiveness)*
P2 $x \ partOf \ y \wedge y \ partOf \ x \Rightarrow x = y$ *(Antisymmetry)*
P3 $x \ partOf \ y \wedge y \ partOf \ z \Rightarrow x \ partOf \ z$ *(Transitivity)*

3.4 Megamodelinng (with MegaL)

MegaL[2] is a Domain Specific Language (DSL) and technology designed for modeling the linguistic architecture of software systems. It is actively developed by the Software Language Team[3] at the University of Koblenz-Landau and currently has two implementations:

- MegaL/Xtextan eclipsebased Integrated Development Environment (IDE)
- and MegaL/Texta text-file and console based implementation with an

3.5 Program Analysis

Program Analysis is the process of automatically analyzing programs. This can be done (a) statically by analyzing the source code of a program or (b) dynamically through monitoring and intercepting the runtime of a program, analyzing transient artifacts.

3.6 XML Data Binding (with JAXB)

XML Data Binding is the process of automatically mapping instance- and model-level objects into a XML-serialized form. That is, for the Java Architecture for XML Binding (JAXB)[4], a Java class itself is mapped via annotations to an XML Schema, both model-level objects. With this mapping Java class-instances can easily be de-/serialized into XML Documents, a process called un-/marshaling. XML Data Binding allows to directly work with a domain model rather than working with direct XML access.

[2] `http://www.softlang.org/megal/`
[3] `http://www.softlang.org/`
[4] `https://docs.oracle.com/javase/tutorial/jaxb/intro/arch.html`

3.7 Object-Relational Mapping (with JPA/Hibernate)

Object-Relational Mapping is the process of automatically mapping instance- and model-level objects into a relational data-storage. In case of SQL-database technologies, model-level objects are mapped to SQL/DDL (Data Definition Language) statements and instance-level objects are mapped to SQL/DML (Data Manipulation Language) statements. Usually, these statements are then sent to and executed by a SQL-database server. This allows to directly work with a domain model when persisting data.

The Java Persistence API (JPA)[5] is a Java standard interface for working with Object-Relational Mapping. Hibernate[6] is one of many JPA implementations. It provides annotations and a XML Schema for defining Object-Relational Mappings between Java classes and database tables.

4 Research Hypotheses & Questions

4.1 Research Hypotheses

$$
\begin{aligned}
x \text{ properPartOf } y &:\Leftrightarrow x \text{ partOf } y \wedge \neg(y \text{ partOf } x) \\
x \text{ atomOf } y &:\Leftrightarrow x \text{ properPartOf } y \wedge \neg\exists z(z \text{ properPartOf } x) \\
x \text{ fragmentOf } y &:\Leftrightarrow x, y \in L \wedge x \text{ properPartOf } y
\end{aligned}
$$

Fragment Correspondence Hypothesis If two artifacts are in a Correspondence-Relationship, they contain constituent parts in the same relation to each other, i.e.:

$$
\begin{aligned}
&A_1 \text{ correspondsTo}_R \ A_2 \\
&\Rightarrow \exists a_1, a_2 : a_1 \text{ partOf } A_1 \wedge a_2 \text{ partOf } A_2 \wedge a_1 \text{ correspondsTo } a_2
\end{aligned}
$$

Fragment Conformance Hypothesis If two artifacts are in a Conformance-Relationship, they contain constituent parts in the same relation to each other, i.e.:

$$
\begin{aligned}
&A_1 \text{ conformsTo } A_2 \\
&\Rightarrow \exists a_1, a_2 : a_1 \text{ partOf } A_1 \wedge a_2 \text{ partOf } A_2 \wedge a_1 \text{ conformsTo } a_2
\end{aligned}
$$

4.2 Research Questions

Research Questions are:

RQ1 description

[5] `http://www.oracle.com/technetwork/articles/javaee/jpa-137156.html`
[6] `http://hibernate.org/`

5 Objectives

Objectives for the thesis are:

O1 Implementation of a MegaL/Xtext-extension capable of recovering traceability links representing PartOf-, Correspondence- and Conformance-Relationships between code fragments.
O2 Implementation of a MegaL/Xtext-extension allowing for an user to visually explore traceability links, i.e. PartOf-, Correspondence- and Conformance-Relationships between code fragments.
O3 Providing an extensive discussion comparing MegaLwith related approaches on traceability recovery.
O4 Providing an extensive discussion comparing MegaLwith related approaches on ontologies for software artifacts or software engineering in general.

6 Methodology

Thesis and research will utilize an example-driven approach inspired by the 101system[7]. For this, the system to study will be an imaginary Human Resource Management System (HRMS). Figure 1 shows an UML-Class-Diagram depicting the model of this system.

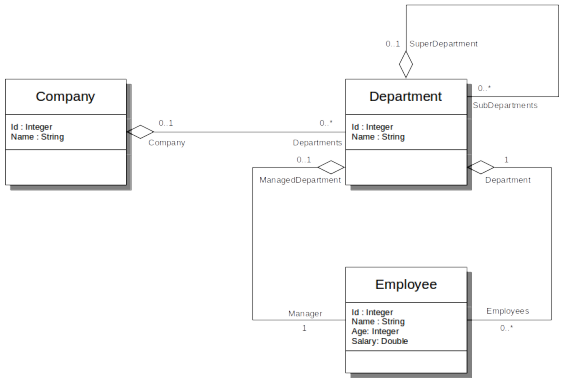

Fig. 1. The Human Resource Management System Model

This HRMS will be implemented in plain Java with two scenarios in mind:

1. XML-Binding with JAXB
2. Persitence with JPA/Hibernate

Both scenarios will be studied with the concrete instance of the HRMS model depicted in figure 2.

[7] `https://101wiki.softlang.org/101system`

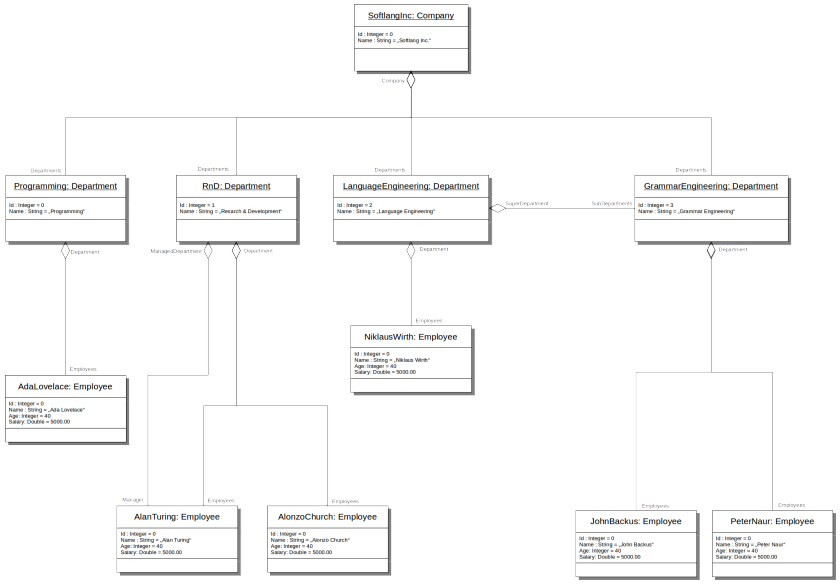

Fig. 2. The Human Resource Management System Instance called *Softlang Inc.*

7 Structure of the Thesis

The interim structure[8] of the thesis is depicted in figure 3. 1

1. **Introduction**
2. **Background**
3. **Related Work**
4. **Methodology**
5. **Requirements**
6. **Design**
7. **Implementation**
8. **Case Study**
9. **Analysis/Results**
10. **Conclusion**

Fig. 3. Structure of the Thesis

[4] [1] [6] [2] [5] [3]

[8] `http://softlang.wikidot.com/info:thesis-structure`

References

1. Favre, J., Lämmel, R., Varanovich, A.: Modeling the linguistic architecture of software products. In: Model Driven Engineering Languages and Systems - 15th International Conference, MODELS 2012, Innsbruck, Austria, September 30-October 5, 2012. Proceedings. pp. 151–167 (2012), http://dx.doi.org/10.1007/978-3-642-33666-9_11
2. Favre, J., Nguyen, T.: Towards a megamodel to model software evolution through transformations. Electr. Notes Theor. Comput. Sci. 127(3), 59–74 (2005), http://dx.doi.org/10.1016/j.entcs.2004.08.034
3. Lämmel, R.: Coupled software transformations revisited. In: Proceedings of the 2016 ACM SIGPLAN International Conference on Software Language Engineering, Amsterdam, The Netherlands, October 31 - November 1, 2016. pp. 239–252 (2016), http://dl.acm.org/citation.cfm?id=2997366
4. Lämmel, R., Varanovich, A.: Interpretation of linguistic architecture. In: Modelling Foundations and Applications - 10th European Conference, ECMFA 2014, Held as Part of STAF 2014, York, UK, July 21-25, 2014. Proceedings. pp. 67–82 (2014), http://dx.doi.org/10.1007/978-3-319-09195-2_5
5. Lämmel, R.: Technology modeling with megal in software development (2015), http://userpages.uni-koblenz.de/~laemmel/pttcourse/slides/technology.pdf
6. Varzi, A.C.: Parts, wholes, and part-whole relations: The prospects of mereotopology. Data Knowl. Eng. 20(3), 259–286 (1996), http://dx.doi.org/10.1016/S0169-023X(96)00017-1