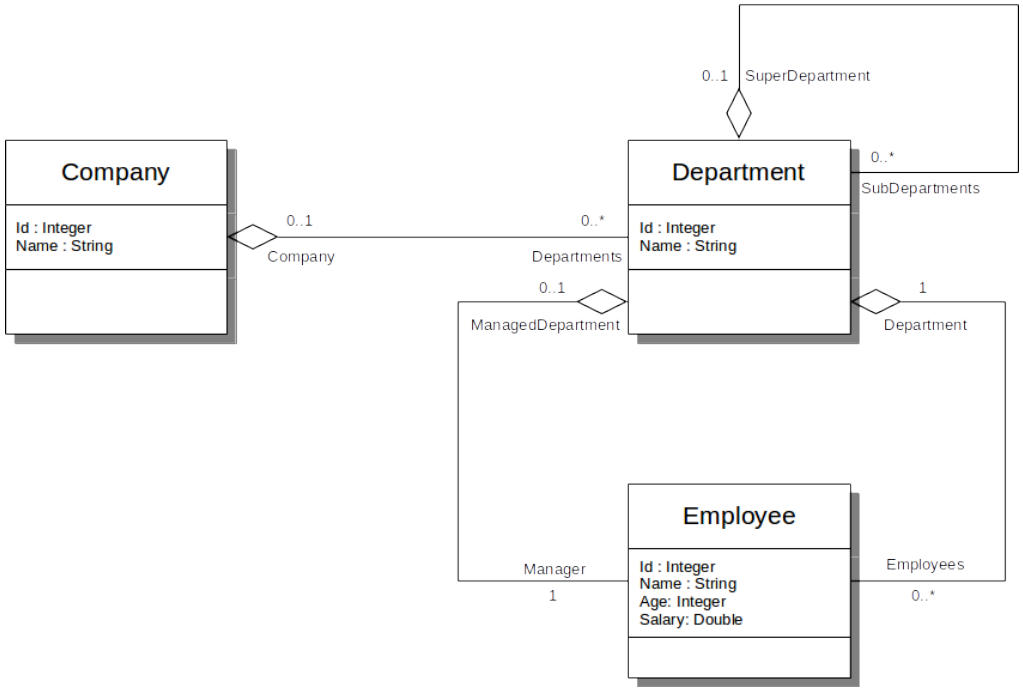
B.Sc. Thesis Exposé

Megamodel-driven Traceability Recovery & Exploration of Correspondence & Conformance Links

Maximilian Meffert
(210 101 205)

University of Koblenz-Landau

Motivation I

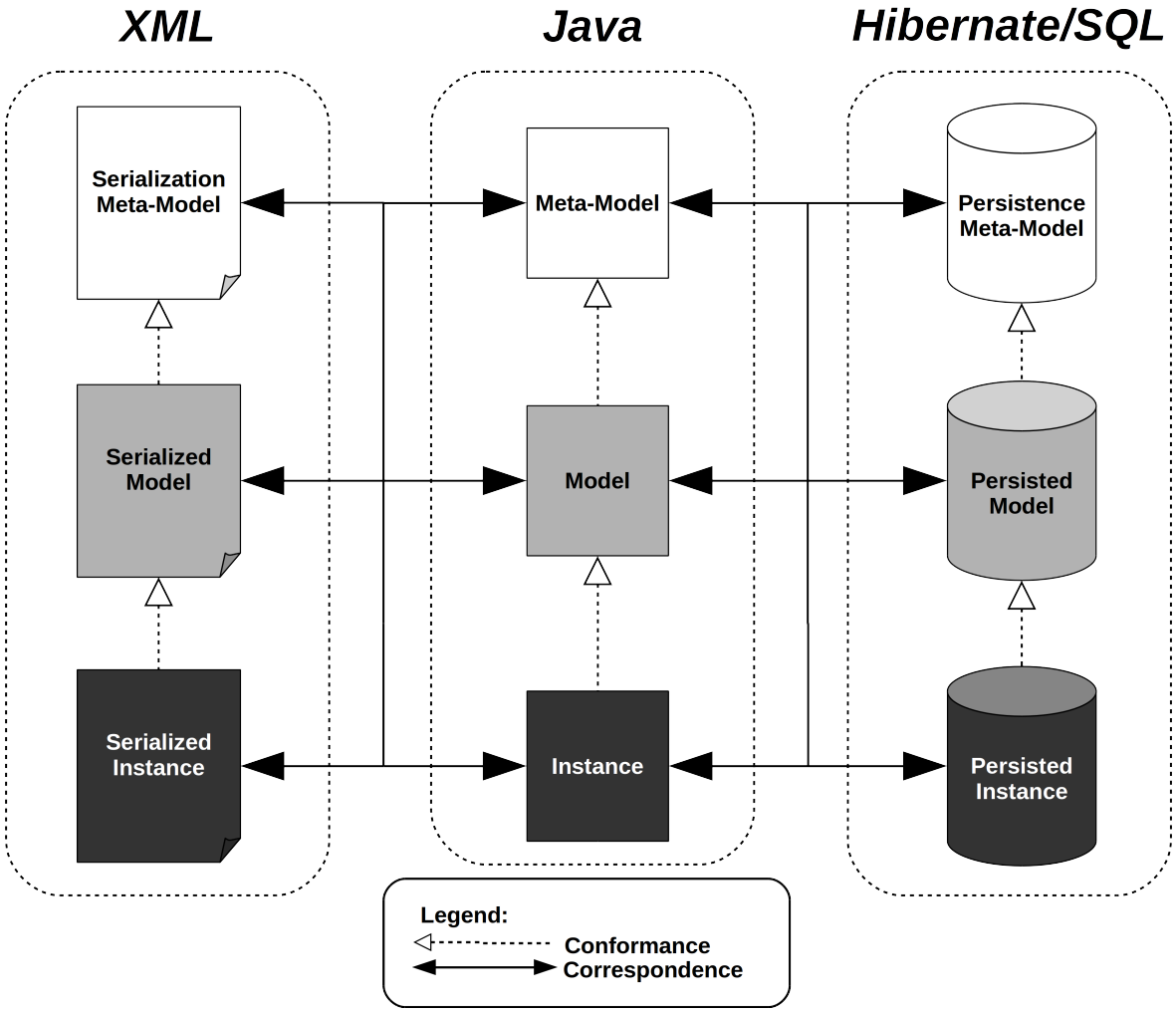


The *101companies* Human Resources Management System

Given an application-/domain-model, it can be ...

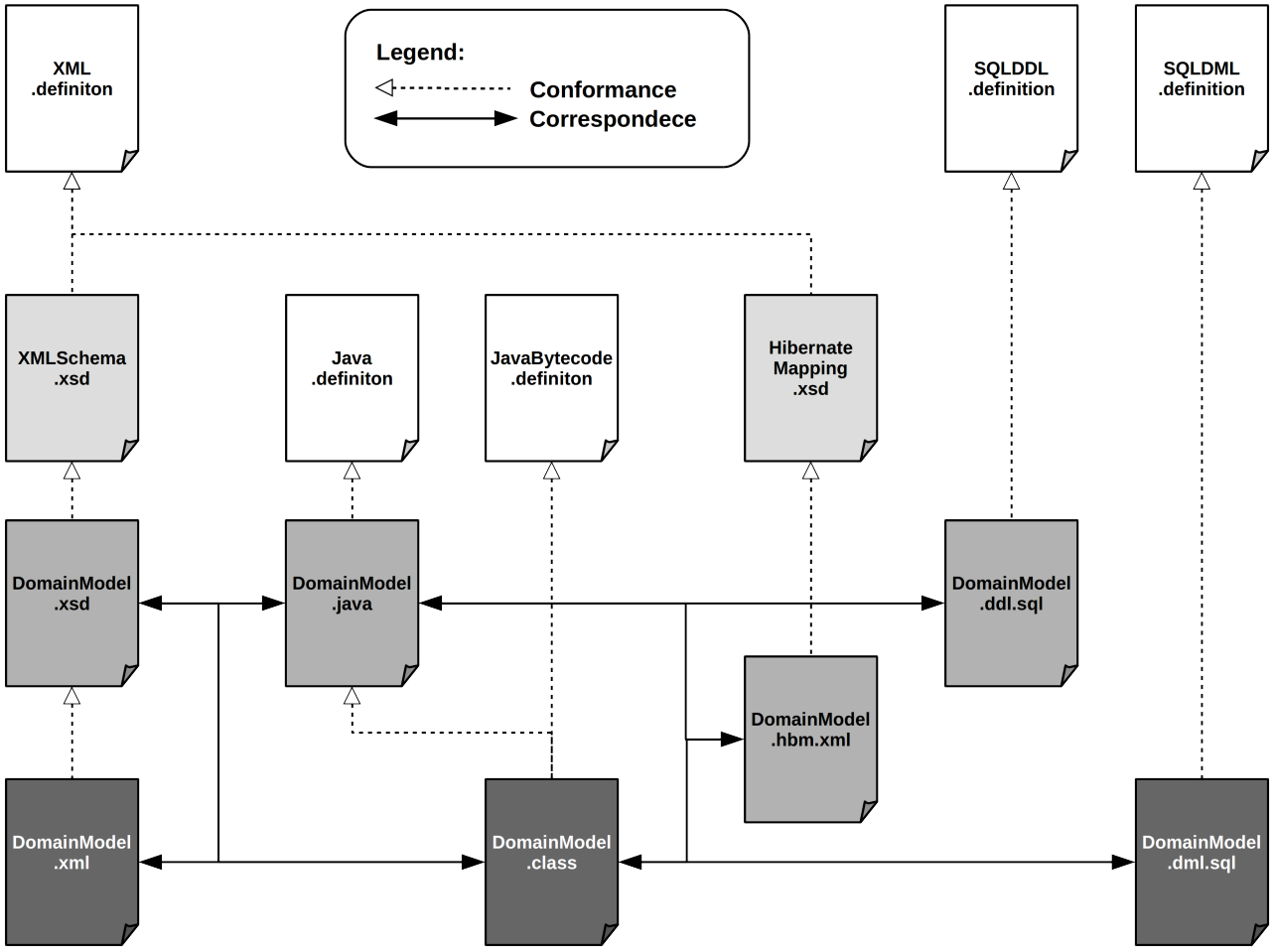
- ... **serialized**, e.g. to XML
- ... **persisted**, e.g. into a relational database

Motivation II



O/R/X Correspondence & Conformance Scenario

Motivation III



O/R/X Correspondence & Conformance Artifact Links

Example

Listing 1: Company.xsd

```
<xs:complexType name="company">
        ...
         <xs:attribute name="id" type="xs:int" use="required"/>
        ...
</xs:complexType>
```

Listing 2: Company.java

```
@XmlRootElement(name="company")
@XmlAccessorType(XmlAccessType.FIELD)
public class Company {
        @XmlAttribute
        private int id;
        ...
}
```

XSD-Java Correspondences

Traceability

- **Traceability**
  The ability to interrelate artifacts of a software development process.[6][1]
- **Traceability Link**
  The element of a relationship between software artifacts.[6]
- **Traceability Recovery**
  The (automatic) analysis of software artifacts for traceability links. (References Needed!)
- **Traceability Exploration**
  The process of navigating and illustrating traceability links. (References Needed!)

Formal Background

- **Parthood/Mereology**[4][5]

$$x \text{ partOf } x$$
$$x \text{ partOf } y \wedge y \text{ partOf } x \Rightarrow x = y$$
$$x \text{ partOf } y \wedge y \text{ partOf } z \Rightarrow x \text{ partOf } z$$
$$x \text{ properPartOf } y \Leftrightarrow x \text{ partOf } y \wedge \neg(y \text{ partOf } x)$$

- **Correspondence**[4]

$$(a_1, a_2) \in R \subseteq L_1 \times L_2$$
$$\wedge \forall b_1 \in L_1 : b_1 \text{ partOf } a_1 \Rightarrow (\exists! b_2 \in L_2 : b_2 \text{ partOf } a_2 \wedge b_1 \text{ correspondsTo}_R \ b_2)$$
$$\wedge \forall b_2 \in L_2 : b_2 \text{ partOf } a_2 \Rightarrow (\exists! b_1 \in L_1 : b_1 \text{ partOf } a_2 \wedge b_2 \text{ correspondsTo}_R \ b_1)$$
$$\Rightarrow a_1 \text{ correspondsTo}_R \ a_2$$

- **Conformance**[4]

$$\forall x \in \text{ Any } : x \in L \subseteq \text{ Any } \Leftrightarrow \exists d \in D \subseteq \text{ Any } : x \text{ conformsTo } d$$

Research Hypotheses

RH1 **Fragment Correspondence Hypothesis**

$\forall a_1 \in L_1, a_2 \in L_2 \exists b_1 \in L_1, b_2 \in L_2 :$

$a_1 \text{ correspondsTo}_R \ a_2 \Rightarrow b_1 \text{ partOf } a_2 \wedge b_2 \text{ partOf } a_2 \wedge b_1 \text{ correspondsTo}_R \ b_2$

RH2 **Fragment Conformance Hypothesis**

$\forall a_1 \in L, a_2 \in D \exists b_1 \in L, b_2 \in D :$

$a_1 \text{ conformsTo } a_2 \Rightarrow b_1 \text{ partOf } a_2 \wedge b_2 \text{ partOf } a_2 \wedge b_1 \text{ conformsTo } b_2$

Note, these hypotheses may be problematic / to weak.
Since parthood is reflexive they are inherently true.

Research Questions

RQ1 **Is correspondence to some extend strictly mereologically induced?**

$$\forall a_1 \in L_1, a_2 \in L_2 \exists b_1 \in L_1, b_2 \in L_2 :$$
$$a_1 \text{ correspondsTo}_R \ a_2$$
$$\Rightarrow b_1 \text{ properPartOf } a_2 \wedge b_2 \text{ properPartOf } a_2 \wedge b_1 \text{ correspondsTo}_R \ b_2$$

RQ2 **Is conformance to some extend strictly mereologically induced?**

$$\forall a_1 \in L_1, a_2 \in L_2 \exists b_1 \in L_1, b_2 \in L_2 :$$
$$a_1 \text{ conformsTo } a_2$$
$$\Rightarrow b_1 \text{ properPartOf } a_2 \wedge b_2 \text{ properPartOf } a_2 \wedge b_1 \text{ conformsTo } b_2$$

Thesis Objectives

TO1 Implementation of a MegaL/Xtext-extension[3] capable of recovering traceability links representing parthood, correspondence and conformance relationships between code fragments.

TO2 Implementation of a MegaL/Xtext-extension[3] allowing for an user to visually explore traceability links, i.e. parthood, correspondence and conformance relationships between code fragments

TO3 Providing an extensive discussion comparing MegaL[2] with related approaches on traceability recovery.

TO4 Providing an extensive discussion comparing MegaL[2] with related approaches on ontologies for software artifacts or software engineering in general.

TO5 Providing answers for the research questions.

References

[1] Ieee standard glossary of software engineering terminology. IEEE Std 610.12-1990 pp. 1–84 (Dec 1990), http://ieeexplore.ieee.org/servlet/opac?punumber=2238

[2] 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

[3] Härtel, L.: Linguistic architecture on the workbench. bachelor thesis, University of Koblenz-Landau

[4] 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

[5] 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

[6] Winkler, S., Pilgrim, J.: A survey of traceability in requirements engineering and model-driven development. Softw. Syst. Model. 9(4), 529–565 (Sep 2010), http://dx.doi.org/10.1007/s10270-009-0145-0