On Marrying Ontology and Software Technology

Ontological and Metamodelling Technical Spaces

Steffen Staab
Ontologies & Software Tech: Starting Point

Ontologies & Software Tech: Starting Point

IST – Institute for Software Technology @Koblenz

Ontologies

RDF

Well-founded semantics

F-Logic

Model theory

OWL

Fernando Parreiras

Metamodelling

MDA

UML

XMI

Grammar

Andreas Winter
Synthesis!

IST – Institute for Software Technology @Koblenz

Fernando Parreiras

Andreas Winter

It’s the process, stupid!

OWL

Model theory

RDF

Ontology

Well-founded semantics

F-Logic

Metamodelling

MDA

UML

Grammar

RDF

Semantics

It’s the process, stupid!

ISWeb - Information Systems & Semantic Web

Steffen Staab
staab@uni-koblenz.de

ESEC/FSE 2007
4 of 42
Rationale of this Talk

Core Statements:
Ontologies are yet another type of Conceptual Models

Hypotheses:
1 Ontologies and SE modeling may be integrated by MDE!

A Generic Process of MDA + Ontology Technologies

Ontologies are used differently and with different technology than UML usually is

2 New possibilities for SE!
3 New possibilities for ontologies and ontology technologies!

Advantages

(Classification Approach for MDA + Ontologies)
Agenda of this Talk

Part 1: What is an ontology?
- Definition
- Purpose, Usage, Formality, Type, Communities

Part 2: Four cases of MDA & ontology technologies
- Model checking
- Model enrichment
- Ontology engineering
- Hybrid models

Part 3: A radical change
- Ontologies in the Semantic Web
Definition: What is an ontology?
(in computer science)

Based on Gruber 93:

An Ontology is a

formal specification  ➞ Executable, Discussable
of a shared  ➞ Group of persons
conceptualization  ➞ About concepts
of a domain of interest  ➞ Between application
and „unique truth“
Purpose: What is an ontology?

To make domain assumptions **explicit**
- Easier to change domain assumptions
- Easier to understand and update legacy data

To separate **domain knowledge** from operational knowledge
- Re-use domain and operational knowledge separately

A **community reference** for applications

To **share a consistent understanding** of what information means
Usage: What is an ontology?

Thesauri

Topic Maps

Navigation

Information Retrieval

Sharing of Knowledge

Query Expansion

Extended ER-Models

Ontologies

Semantic Networks

Mediation

Consistency Checking

EAI

Reasoning

Queries

Predicate Logic

Taxonomies
Formality: What is an ontology?

Ad-hoc Hierarchies (Yahoo!)
Text Corpora
Lightweight

Glossaries
Folks-onomies

Data Dictionaries (EDI)

Principled Informal Hierarchies
Thesauri

XML DTDs

DB Schema

Informal Hierarchies
Glossaries

Data Models (UML, STEP)

Formal Taxonomies
XML Schemas

Formal Ontologies & Inference

Description Logics

First-order, Higher-order, Modal Logic

F-Logic

Formality: What is an ontology?

Glossaries & Data Dictionaries

Thesauri, Taxonomies

Glossaries & Data Dictionaries

Thesauri, Taxonomies

MetaData, XML Schemas, Data Models

Formal Ontologies & Inference
Example: What is an ontology?

Foundational Model of Anatomy

- Represents structures ranging from macromolecular complexes to body parts
- Contains
  - ~70,000 distinct concepts
  - ~110,000 terms
  - 140 relations
  - Metaclases to define class-level properties
  - Attributed relations
  - Different types of part-whole, location, and other spatial relations
  - Synonyms
Agenda of this Talk

Part 1: What is an ontology?
- Definition
- Purpose, Usage, Formality, Type, Communities

Part 2: Four cases of MDA & ontology technologies
- Model checking
- Model enrichment
- Ontology engineering
- Hybrid models

Part 3: A radical change:
- Ontologies in the Semantic Web
A Standard Process

Model-driven Architecture

UML
PIM

UML
PSM

JAVA
Code

Classical MDA
Reasoning on UML class diagrams

Classical MDA

Translation

OWL

UML

PIM

UML

PSM

JAVA

Code
Reasoning on UML class diagrams allows for checking:

- **Consistency of the diagram:** Can the classes be populated?

- **Classification to** identify the possible omission of an explicit generalization.

- **Equivalence among classes** to discover redundancy.

- **Refinement of properties** to apply stricter multiplicities or typing than the ones explicitly specified in the diagram.
Model Checking: Example

Every WebPortalAccount is used by at most one Researcher.

If Researcher is empty, User and Student will be redundant.

Researcher is disjoint from Student.

Every WebPortalAccount is used by at most one Researcher.
Model Checking: Example

INCONSISTENT

UserAccount

WebPortalAccount

User

Uses

Researcher

Student

{complete, disjoint}

Advantage for SE:
Models with provably higher quality
Platform Independent Model Transformation based on Datalog-style Inferences

Translation

OWL
TRIPLE

UML
PIM
UML
PSM
JAVA
Code

Classical MDA
// Mapping Mab
FORALL Ma @Mb(Ma) {
   // MScThesis2Thesis
   FORALL X MScThesis[typeOf->X]@Ma --> Thesis[typeOf->X]
   // PhDThesis2Thesis
   FORALL X PhDThesis[typeOf->X]@Ma --> Thesis[typeOf->X]
}
Advantage for SE:

Logics-based data and model transformation can be performed at modeling time, at run-time and over relational databases.
MDA & Ontology Case 3: Ontology Engineering

Visual Modeling of OWL DL Ontologies Using UML

Classical MDA

Generation

OWL

TRIPLE

OWL

UML

PIM

UML

PSM

JAVA

Code
Ontology Engineering: Example

Advantages for Ontology Engineering:

Visual models may be more accessible.

UML modeling paradigm is known by many software developers.
Integrating UML Models and OWL Ontologies: **TwoUse**

**TwoUse:** Transforming and Weaving Ontologies and UML for SE

**Previously:**

- Combined Ontology & Software Engineering
- TwoUse: Transforming and Weaving Ontologies and UML for SE

**Diagram:**

- **UML** (PIM) → **UML** (PSM) → **JAVA** (Code)
- **OWL**

**Classical MDA**
Case: Integrating UML Models and OWL Ontologies with TwoUse

M2

M1

PIM

OWL Ontology

UML Class Diagram

Java Code

Code

MDA & Ontology Case 4: Hybrid Models
Hybrid Model: Example UML

```
FreeTradeZone
+memberOfTradeZone
+hasMember

Country
1
+livesIn

Customer
1..n
+hasResident

Purchase
+hasCustomer
+hasOrder
Order
getCharges()

Product
Name: String
```
Hybrid Model: Example OWL

**FreeTradeZone**
- +memberOfTradeZone
- +hasMember
- +hasResident
- +livesIn

**Customer**
- +hasCustomer
- +hasOrder
- +hasResident
- +livesIn

**Purchase Order**
- getCharges()

**Order From EU Customers**
- DutyFreeOrder = OrderFromEUCustomer
- OrdersFromEUCustomers hasCustomer SOME CustomerFromEUCountry

**CountryFromEU**
- <<owlValue>> {hasValue = eu}
- memberOfTradeZone : FreeTradeZone
- +{someValuesFrom = CountryFromEU}
- livesIn

**CustomerFromEUCountry**
- <owlRestriction>>
- DutyFreeOrder
- OrderFromEUCustomers

**OWL (ODM)**
Hybrid Model: Example TwoUse

FreeTradeZone

Customer

Order

Product

Country

DutyFreeOrder

OrderFromEUCustomer

CustomerFromEUCountry

CountryFromEU

Product Name: String

getCharges()
<<oclExpression>>
{context PurchaseOrder::getCharges() : Real
body: if self.isOwllInstanceOf(DutyFreeOrder)
then 0% else 60% endif}
**TwoUse Models (excerpt)**

```
context PurchaseOrder::getCharges() : Real
    body: if self.owlIsInstanceOf(DutyFreeOrder)
    then 0.00  
    else 0.60  
    endif
```

**Diagram Description**
- **M2**
  - **Classifier**
  - **DataType**
  - **Class**
  - **AnyType**
  - **OWLClass**
  - **TUClass**

- **InstanceOf**
  - **Product**
  - **PurchaseOrder**
  - **getCharges()**

- **M1**
  - **Classifier**
  - **DataType**
  - **Class**
  - **AnyType**
  - **OWLClass**
  - **TUClass**

**Code Snippet**
```
context PurchaseOrder::getCharges() : Real
    body: if self.owlIsInstanceOf(DutyFreeOrder)
    then 0.00  
    else 0.60  
    endif
```
TwoUse Models (Excerpt)

M1

Product

M2

DataType

M2

Class

AnyType

OWLClass

InstanceOf

TUClass

OclAny

oclsIsKindOf()
oclsIsTypeOf()

OwlAny

owlIsInstanceOf()

PuchaseOrder

getCharges()

Country

Advantages for SE & Ontology Engineering:

Joint engineering of ontologies and SE models.
Cases seen today (there are more):

1. **Model Checking**, e.g. Reasoning on UML class diagrams (Berardi, 2005)
2. **Model Enrichment**, e.g. Platform independent model transformation based on TRIPLE (Billig et. al)
3. **Ontology Modeling**, e.g. Neon Project (Haase et al., 2007)
Revisiting Hypotheses

Hypotheses:
1 Ontologies and SE modeling integrated by MDE (specifically MDA)!

2 New possibilities for SE!
3 New possibilities for ontologies and ontology technologies!

A Generic Process

Advantages

Models with provably higher quality.

Logics-based data and model transformation.

Visual models more accessible.

Joint engineering of ontologies and SE models.
Open Questions for MDE + Ontologies

- Language(s) and Metamodels:
  - “Best” profiles, metamodels, ontology languages
  - Syntactic and semantic translations between ontology languages

- Other diagram types (beyond class diagrams)

- Even more scalable ontology reasoning technologies

- Case Studies
  - Need for non-trivial case studies

- Best Practices of where and how to use MDE and ontologies
Part 1: What is an ontology?
- Definition
- Purpose, Usage, Formality, Type, Communities

Part 2: Four cases of MDA & ontology technologies
- Model checking
- Model enrichment
- Ontology engineering
- Hybrid models

Part 3: A radical change:
- Ontologies in the Semantic Web
Ontologies are not the Semantic Web

Ontologies ≠ Semantic Web
A Network of Ontologies

\begin{itemize}
\item $O_2$ relatedWith $O_1$
\item $O_1$ priorVersionOf $O_1'$
\item $O_1'$ priorVersionOf $O_1''$
\item $O_1$ extends $O_3$
\item $O_1$ incompatibleWith $M_{1',2}$
\item $M_{1,2}$ target $O_2$
\item $M_{1',2}$ source $O_1'$
\item $O_3$ dependsOn $O_1$
\item $O_4$ extends $O_1$
\end{itemize}
Networked Ontology Metamodel

- OWL Metammodel
  - Rule Metammodel
  - Mapping Metammodel
  - Modularization Metammodel

Ontology Languages / Formalisms

- OWL
- F-Logic
- C-OWL
- Datalog
- E-Connections
- ...

Language Mappings / Groundings

ISWeb - Information Systems & Semantic Web
Steffen Staab
staab@uni-koblenz.de
ESEC/FSE 2007
39 of 42

© P. Haase
Open Questions for Networked Ontologies

- Language(s): for Representation of Networked Ontologies
  - Difficult trade-off: Expressive, but not too complex!
  - Our proposal: Networked RDF Graphs (Schenk & Staab, 2007)

- Semantics: Too many different, often counterintuitive, semantics around now!

- Maintenance and Evolution

- Autonomy of Ontology Owners!

- How to integrate Networked Ontologies into SE?
Now and Near Future

- EU Integrated Project „NeOn“ (2006-2010)
  - Modeling Networked Ontologies

- EU STReP „MOST“ under negotiation (i.e. highly probable to start 2008)
  - Integrating metamodeling, transformations and reasoning
  - Partners
    - SAP (D), Comarch (PL), BOC (A)
    - U Koblenz (D), TU Dresden (D), U Aberdeen (UK)

- We are hiring PhD students and PostDocs!
Thank You!

Steffen Staab  
Andreas Winter  
Fernando Parreiras