Metamodelling and Language Engineering

Étienne ANDRÉ
Hauptseminar Metamodelling
02.02.2006
Agenda

- Concepts
- Motivations
- Meta Modelling Framework
- Example: Small Modelling Language
- Language Composition
- Summary
Agenda

- Concepts
- Motivations
- Meta Modelling Framework
- Example: Small Modelling Language
- Language Composition
- Summary
Concepts: Language

• **Definition (Wikipedia)**
  – A [formal] language can be thought of as a set of formal specifications concerning syntax, vocabulary, and meaning.

• **Definition (Clark, Evan, Kent)**
  – A language consists of models for **concrete syntax**, **abstract syntax** and for the **semantic domain**.
Concepts: Language Engineering

- **Definition (Wikipedia)**
  - Creation of [natural] language processing systems whose cost and outputs are measurable and predictable

- **Goals (Bézivin, Heckel)**
  - Definition of abstract syntax and well-formedness rules, operational and denotational semantics, consistency and refinement relations, and model transformations
**Concepts: Domain Specific Languages**

- **Definition (Wikipedia)**
  - Language designed to be useful for a **specific set of tasks**
  - DSLs tend to be focused on doing one sort of task well

- **Goal**
  - Enhancement of quality, maintainability, portability and reusability

- **Disadvantages**
  - Expensive to design, implement, and maintain!

- **Examples**
  - GraphViz to define directed graphs
  - Csound to create audio files
  - YACC for parsing and compilers
  - SQL for databases
Concepts: Catalysis Method

- **Presentation**
  - Started 1992 by D'Souza D. and Wills A.
  - Object-Oriented method based on UML 1.0

- **Concepts**
  - Action as important as object
  - Modelling frameworks as templates
  - Package import mechanism
  - Refinements: objects and actions at different scales
Agenda

- Concepts
- Motivations
- Meta Modelling Framework
- Example: Small Modelling Language
- Language Composition
- Summary
Motivations

- **Language engineering:**
  - Required to support the design, implementation, and validation of languages with the goal to deliver languages at low cost with high quality
  - Reusability of parts of languages
  - Example: compilers
Agenda

- Concepts
- Motivations
- Meta Modelling Framework
- Example: Small Modelling Language
- Language Composition
- Summary
Meta Modelling Framework: Overview

- **Presented 2002 by Clark, Evans, Kent**

- **Motivations**
  - Deficiencies in the definition of UML 1.3
  - Support MDA

- **Overview**
  - Method for Meta Modelling (MMM)
  - Language for Meta Modelling (MML)
  - Tool for Meta Modelling (MMT)

---

Catalysis → based on MMF

MMF → consistent with influences develops into

UML 1.3 uses UML 2.0

---

Étienne ANDRÉ

Metamodelling and Language Engineering
• **Method for Meta Modelling (MMM)**
  - This approach applies **OO modelling** to the definition of languages, especially OO modelling languages
  - OCL used to define well-formedness **constraints** on the language components
  - “UML needs to become a precisely defined **family of languages**”

---

**Diagram:**

- **Concrete syntax**
- **Abstract syntax**
- **Semantic domain**

- **Display mapping**
- **Semantic mapping**
Meta Modelling Framework: MMM

- MMM Key features
  - Package specialization
    - Partial definitions of model elements in a super package to be consistently specialized in a sub-package
  - Templates
    - Parametric model elements
    - Means of representing reusable modelling patterns
    - Provide a framework for defining language families
Meta Modelling Framework: MML

- Language for Meta Modelling (MML)
  
  - Static OO modelling language:
    - Small
    - Meta-circular, describes itself
  
  - Basic expression language based on OCL
    - Supports sets and sequences
Meta Modelling Framework: MML

- **MML class definition**
  - MML definition = name + expression

```plaintext
class Person
  name : String
  age : Integer
  married : Boolean
  children : Set(Person);
  parents : Set(Person);
  init(s:Seq(Instance)):Person
    self.name := s->at(0)[]
    self.age := s->at(1)[]
    self;
  averageChildAge():Integer
    self.children->iterate(c, a = 0 | a + c.age) / self.children->size;
  inv
    IfMarriedThenOver15
      self.married implies self.age >=16;
    OnlyTwoParents
      self.parents->size = 2
end
```

- **MML association definitions**
  - Only binary associations

```plaintext
association Family
  parents : Person mult: 2
  children : Person mult : *
end
```
Meta Modelling Framework: MML

- **MML package definition**
  - Group definitions of model elements

```plaintext
package People
    class Person
        // as given above...
        end;
    association Family
        // as given above
        end
end
```

- **Powerful package specialization mechanism**

```plaintext
package Employment extends People
    class Person
        yearsInService : Integer
        end;
    class Company
        name : String
        end
    association Works
        company : Company mult : 1
        employees : Person mult : *
        end
end
```
MML templates definition

- Template: parametric model element

```plaintext
package Contains(Container,n1,m1,Contained,n2,m2)
class <<Container>>
   <<n2>>():Set(<<Contained>>)  
   self.<<n2>>
   inv
     UniqueNames
     self.<<n2>>->forAll(c1 c2 |  
                        c1.name = c2.name implies c1 = c2)
end;
association <<Container + Contains>>
   <<n1>> : <<Container>> mult: <<m1>>
   <<n2>> : <<Contained>> mult: <<m2>>
end
end
```

```plaintext
package People
   class Person
     parents():Set(Person)  
     self.parents
     inv
       UniqueNames
       self.parents->forAll(c1 c2 |  
                             c1.name = c2.name implies c1 = c2)
   end;
association PersonContains
   children : Person mult: *
   parents : Person mult: 2
end
end
```

```plaintext
package People extends Container("Person","children",*,"Person","parents",2)  
class Person
   // attribute and method definitions
end
end
```
Tool for Meta-Modelling (MMT)

- Apparently not available!
- Prototype tool written in Java
- Supports MMF Approach
- Several checks
  - Well-formedness
  - OCL constraints
Agenda

- Concepts
- Motivations
- Meta Modelling Framework
- Example: Small Modelling Language
- Language Composition
- Summary
**Example: SML**

- **Example of a Small Modelling Language**
  - Static modelling language consisting of packages and class with attributes
  - Only abstract syntax, semantic domain and mapping between them

![Diagram]

- Concrete syntax
- Abstract syntax
- Semantic domain
- Display mapping
- Semantic mapping
Example: SML

- **Templates libraries definitions**
  - Named model elements

```sml
class <<Model>>
  name : String;
toString():String
    "<" + self.of.name + self.name + ">"
end
end
```

- ...and more templates definitions

```sml
package NameSpace(Container, Contained)
package Contains(Container, Contained)
package Specializable(Model)
package SpecializableContainer(Container, Contained)
  extends Specializable(Container), Specializable(Contained)
package Relation(Name, Domain, Range)
package RelateAtt(R, Domain, Range, DomainAtt, RangeAtt, Pred)
  extends Relation(R, Domain, Range)
package TypeCorrect(R, Domain, Range)
  extends RelateAtt(R, Domain, Range, "type", "value", check)
```
Example: SML

Abstract Syntax

```sml
package AbstractSyntax
extends
    SelfContains("Package"),
    SpecializableContainer("Package","Package"),
    SpecializableContainer("Package","Class"),
    SpecializableContainer("Class","Attribute"),
    Specializable("Attribute"),
    Contains("Package","Class"),
    Contains("Class","Attribute"),
    Clonable("Package","Class"),
    Clonable("Package","Package"),
    Clonable("Class","Attribute"),
    Named("Package"),
    Named("Class"),
    Named("Attribute"),
    NameSpace("Package","Package"),
    NameSpace("Package","Class"),
    NameSpace("Class","Attribute")

class Attribute
    // some definition
end
end
```
Example: SML

**Semantic domain**

```sml
define package SemanticDomain
  extends
    SelfContains("Snapshot"),
    Contains("Snapshot","Object"),
    Contains("Object","Slot"),
    Named("Snapshot"),
    Named("Slot")

class Slot value : Object end
end
```

**Semantic mapping**

```sml
define package SemanticMapping
  extends
    AbstractSyntax,
    SemanticDomain,
    ContainsInstances1(
      "PackXSnap","Package","Class",
      "ClassXObj","Snapshot","Object"),
    ContainsInstances(
      "ClassXObj","Class","Attribute",
      "AttXSlot","Object","Slot"),
    SameName("AttXSlot","Attribute","Slot")
  TypeCorrect("AttXSlot","Attribute","Slot")
end
```
Example: SML

• Conclusion on this example
  – Definition of templates libraries
    • Reusability
  – Simple definition

• MMF applicable to more complex languages
Agenda

- Concepts
- Motivations
- Meta Modelling Framework
- Example: Small Modelling Language
- Language Composition
- Summary
Language Composition

- **Motivation**
  - Reusability
  - Lower costs

- **Structure**
  - Composing model components result in a model that describes a language
Language Composition

- Parallel with **Software Engineering**
  - Invasive language composition: Composition through fragment boxes

- Achieved through **package extension**
  - Not so easy!
  - Metamodel defined by Clark, Evans and Kent
Agenda

- Concepts
- Motivations
- Meta Modelling Framework
- Example: Small Modelling Language
- Language Composition
- Summary
Summary

- Why language engineering?
  - Reusability of parts of languages (packages, templates)
  - Lower costs

- Key features
  - Templates
  - Package extension
  - Language composition
References

http://en.wikipedia.org
http://fr.wikipedia.org
http://de.wikipedia.org
http://www.dcs.kcl.ac.uk/staff/tony/docs/
The end

- Thanks for your attention 😊

- Any questions...?