Abstracting Complex Languages through Transformation and Composition

Jendrik Johannes, Steffen Zschaler, Miguel A. Fernández, Antonio Castillo, Dimitrios S. Kolovos and Richard F. Paige

08/10/2009, MoDELS ‘09, Denver, Colorado
Stakeholders:

Domain Expert 1: Telecommunications Device Configuration

Domain Expert 2: Telecommunications Network Configuration

Language Engineer

Mr. Concrete

Mrs. Abstract
A Complex Problem

Mr. Concrete

Johannes, Zschaler, Fernandez, Castillo, Kolovos, Paige: Abstracting Complex Languages through Transformation and Composition
Building the **Concrete** DSL

**DMTF**

*Standard Document*

**Domain Meta Model**

**Defining Syntax**

**EuGENia**

**emftext**

**minimal input**

---

Johannes, Zschaler, Fernandez, Castillo, Kolovos, Paige: Abstracting Complex Languages through Transformation and Composition
Building the **Concrete DSL**

---

**Defining Semantics**

ETL, QVT, MOF2Text, etc.

---

**much more input**

---

Model Transformations
Some Time Later…

Johannes, Zschaler, Fernandez, Castillo, Kolovos, Paige: Abstracting Complex Languages through Transformation and Composition

complex
simple

Mr. Concrete
Suddenly…

Abstracting Complex Languages through Transformation and Composition

Mrs. Abstract
Building the Abstract DSL

Abstract Domain Concepts

Domain Meta Model 2

Defining Syntax

minimal input

EuGENia

emftex
Building the Abstract DSL

Defining Semantics

ETL, QVT, MOF2Text, etc.

much more input

Model Transformations

Johannes, Zschaler, Fernandez, Castillo, Kolovos, Paige: Abstracting Complex Languages through Transformation and Composition
Our Assets

- Languages in the same domain
  - One is an abstraction of the other

- Experts of the domain
  - Can use tooling for the concrete language

- Specific class of model transformations
  - Reoccuring mapping patterns can be reused
Abstract Language Building

Abstract Concept  --->  Set_Of_Concrete_Concepts

- Model each abstract concept in the concrete DSL as *model fragment*
- Annotate abstract DSL metamodel with mapping using *mapping patterns*

```java
@classmap("Router.cim")
```

Router
Implementation

Annotated Metamodel
Abstract DSL

instance of

Abstract DSL Instance

Concrete DSL Fragments

meta level

model level

Simple!

Concrete DSL Instance

Simple!

Johannes, Zschaler, Fernandez, Castillo, Kolovos, Paige: Abstracting Complex Languages through Transformation and Composition
Conclusion

- Case study based on Common Information Model (CIM) – DMTF Standard

- Simplified construction of abstract languages
  - Using Mapping Patterns
    1. Element Mapping
    2. Element Mapping with Variability
    3. Attribute Mapping
    4. Link Mapping
  - Separation of concerns in model transformations
    - Mapping separated from definition of concrete model fragment
  - Use of concrete language tooling to define concrete model fragments
    - Better integration of domain experts

- Prototype based on Epsilon and Reuseware
  - http://www.eclipse.org/gmt/epsilon/
  - http://www.reuseware.org/

- See also: http://reuseware.org/index.php/Abstract_CIM_DSLs
The stakeholders in this presentation were modeled with the Reasonably Clever Mini-Mizer (www.reasonablyclever.com/mm2/)