A Framework for Generating Query Language Code from OCL Invariants

Florian Heidenreich, Christian Wende, Birgit Demuth
Software Technology Group
Starting Point

Department of Computer Science, Software Technology Group

Model-Driven Integrity Engineering (MDIE)

- Motivation
- MDIE and the MDA Stack
- Framework Architecture
- Applications
- Open Issues and Challenges
Motivation

- **Data-intensive applications** (business applications/information systems)

- **Database rules** versus Application rules
  - most of the integrity constraints should be an integral part of the database
  - role of integrity constraints is often underestimated in practice

- **Many proprietary dialects** of the (relational) SQL standard

- **Query/Declarative/SQL-like languages** in several technical spaces
  - Relational
  - Object-relational
  - Object-oriented
  - Navigational (XML and others)
  - ...

Motivation
Business Rules Approach

MDIE and the MDA Stack
The Big Picture

Our Framework Context

Department of Computer Science, Software Technology Group

CIM (Business rules) --- is-instance-of --- CIM metamodel (SBVR)

PIM (Pivot model, PIM rules) --- is-instance-of --- PIM metamodel (Pivot MM, PIM rule MM)

PIM/PSM (Data model, PIM/PSM rules) --- is-instance-of --- PIM/PSM metamodel (Data MM, PIM/PSM rule MM)

Coded rules --- is-instance-of --- Programming/Query language

M2M transformation

M2M transformation specification (e.g. by QVT)

M2M transformation specification (e.g. by QVT)

M2C transformation

M2M transformation specification (e.g. by template language)
Query Code Transformation by the Dresden OCL2 Toolkit
Model Transformation Framework

TransformationEngine
+InvokeTransformation()

- strategy
1

context

strategy

concrete strategy

Uml 2 ISchemaFacade
+invoke()

Uml 2 DBSchema
+invoke()

«interface»
ITransformation
+invoke()
+setConfigurationParameter()
+getResult()
+getTrace()

M2MTransformation
+setConfigurationParameter()
+getResult()
+getTrace()
Data schema access:

```
<<interface>>
ISchemaFacade

<<interface>>
ISchemaElement

Guide
```

String-Template for SQL92 Code Generation:

```plaintext
constraint_body(
    constraint_name, context, context_alias, expression)
::= <<
    create view $constraint_name$ as
    (select * from $context$ as $context_alias$
     where not ($expression$))
>>
```
MDIE and the MDA Stack
Our Framework Approach

Department of Computer Science, Software Technology Group

- **PIM**
  - (UML1.5 model/OCL2.0 invariants)

- **PIM metamodel**
  - (UML1.5/OCL2.0 MM)

- **PIM/PSM**
  - (Abstract Syntax Model (ASM) ISchemaFacade object)

- **PIM/PSM metamodel**
  - (AS metamodel/ISchemaFacade)

- **Coded OCL 2.0 invariants**

- **Query language**
OCL2SQL Application

Schema Propagation

UML/OCL

Constraint Propagation

CWM (Relational)

SQL database schema

ISchema Facade

SQL integrity views

Model Transformation

Code Generation
MDIE and the MDA Stack

Application Example

Department of Computer Science, Software Technology Group

PIM
(Abstract syntax model (ASM) of OCL invariants)

PIM metamodel
(OCL2.0/UML1.5 MM)

OR schema mapping

PIM/PSM
(ASM/ISchemaFacade object)

PIM/PSM metamodel
(OCL2.0 MM/ISchemaFacade)

Templates (in template language)

Code
(SQL integrity views)

SQL language

M2M transformation

is-instance-of

M2C transformation

is-instance-of
OCL2XQuery Application

Schema Propagation → CWM (XML) → XML Schema

UML/OCL

Constraint Propagation → ISchema Facade → XQuery

→ Model Transformation
→ Code Generation
Open Issues

From the theoretical/conceptual viewpoint:

- No proof that our approach works for all declarative/SQL-like languages
- Declarative/SQL-like languages not formalized
- No pattern for arbitrarily complex iterate expressions (OCL-to-SQL)
- Are data models/rules PIM or PSM models/rules?

From the implementation-technical viewpoint:

- Current model transformation approaches (QVT) not considered
- OCL2XQuery not yet implemented
- Not yet migrated to the new architecture of the Dresden OCL2 Toolkit
1. Increase the role of database rules in every day applications such as Web applications or information systems!

2. Is OCL resp. OCL as core language of production rules (PRR) the adequate language for automating of business rules?
   - power of the language
   - paradigm of the language
Results

1. Idea of model- and database driven development of business rules that we call **Model-Driven Integrity Engineering**. Sharing the database by all applications, it
   - reduces development costs
   - improves the quality of data bases

2. **Query Code Generation Framework** to transform OCL-based rules to a manifold of platforms and their corresponding query language code

3. Running **OCL22SQL application**
Questions/ Comments?
context Person
inv inv1: self.grade.value > 0

select VALUE
from GRADE
where PKG in (select GRADE_PK from PERSON where PKP = SELF.PKP)
Template definition

\[
\text{arithmetic\_expression\_plus(operand1,operand2)} ::= \ll
($\text{operand1}\$ + $\text{operand2}\$)
\rr
\]

Use in code generation

```java
StringTemplate template =
templateEngine.getInstanceOf("arithmetic_expression_plus");
template.setAttribute("operand1", "1");
template.setAttribute("operand2", "2");
```