A Diagrammatic Approach to Model Transformations

Adrian Rutle

Department of Computer Engineering
Faculty of Engineering
Bergen University College
Norway

12.09.2008
EATIS 2008, Aracaju, Brazil
Model Driven Engineering (MDE)

- Engineering techniques where models are first-class entities
- Evolved from the popularity of diagrammatic languages such as UML and ER and their usage for specification and documentation of software systems
- Aims to raise the abstraction level of software development from code to models
Model Transformations

- Generation of target models from source models
  
  \[ \text{Src} \xrightarrow{\text{MT}} \text{Trg} \]

- Application areas:
  - Development process: code generation, refinement etc
  - Model management: integration, decomposition etc
  - Migration: from a platform, implementation technology, programming language etc, to another
  - Technology mappings: e.g. JPA where Java classes are mapped to Relational tables, objects to rows in database tables

Adrian Rutle
DPL, MDE and MT (http://gs.hib.no)
Outline

1 Introduction and Motivation
   - Diagrammatic Predicate Logic (DPL)

2 Model Transformations
   - Overview
   - Transformation Rules – Patterns and Input-Output Coordination
   - Transformation Rules – Matches
   - Coordination of rules

3 Possible Properties of Model Transformations
Diagrammatic Predicate Logic (DPL)

- DPL is a diagrammatic approach for the formalization of MDE
- Aims to combine the intuition from graphical modeling languages with the semantic rigor of formal methods
- Based on Sketches/Category theory and first order logic
- Potentials to combine the machinery from category theory with first order logic
1. Introduction and Motivation
   - Diagrammatic Predicate Logic (DPL)

2. Model Transformations
   - Overview
   - Transformation Rules – Patterns and Input-Output Coordination
   - Transformation Rules – Matches
   - Coordination of rules

3. Possible Properties of Model Transformations
MT Definition: Set of transformation rules that describe how instances of the source model are transformed to instances of the target model.

This is demonstrated as an example where an instance of the EMF metamodel, Ecore, is transformed to an instance of Relational DB metamodel, RDB.
Model Transformation

**MT Definition**

- **MT Language**
  - conformsTo

**MT Execution**

- **MT Definition**
  - conformsTo

**Overview**

Transformation Rules – Patterns and Input-Output Coordination

Transformation Rules – Matches

Coordination of rules

Adrian Rutle

DPL, MDE and MT (http://gs.hib.no)
Model Transformation: Ecore to RDB

Source: Ecore

Target: RDB metamodel
Outline

1. Introduction and Motivation
   - Diagrammatic Predicate Logic (DPL)

2. Model Transformations
   - Overview
   - Transformation Rules – Patterns and Input-Output Coordination
     - Transformation Rules – Matches
     - Coordination of rules

3. Possible Properties of Model Transformations
Rule1: EClass to Table + PrimaryKey
Rule 1 with K: EClass to Table + PrimaryKey
Rule 2: EAttribute to Column
Rule2 with K: EAttribute to Column
Outline

1. Introduction and Motivation
   - Diagrammatic Predicate Logic (DPL)

2. Model Transformations
   - Overview
   - Transformation Rules – Patterns and Input-Output Coordination
   - Transformation Rules – Matches
   - Coordination of rules

3. Possible Properties of Model Transformations
An Instance model of Ecore, $I_{EMF}$
Match of pattern $P_1$ in $I_{EMF}$

Input pattern P1

$1$: EClass

Matches of P1

1

upperBound

1

worksFor: EReference

eReferenceType

eReference

eReferenceType

hires: EReference

upperBound

-1

name: EAttribute

eAttribute

eAttributeType

String: EDataType

Person: EClass

Company: EClass
Match of pattern $P_2$ in $I_{EMF}$

Input pattern $P_2$
The Generated Output RDB Model $I_{RDB}$
The Generated Output RDB Model

Input pattern P1

1: EClass

eAttribute

name: EAttribute
eAttributeType

String: EDataType

1

upperBound

eReference

worksFor: EReference
eReferenceType

Person: EClass

Company: EClass

eReference
hires: EReference
eReferenceType

upperBound

-1

EMF

Output pattern P1'

String: DataType

type

name: Column

col

ownedby

1

Pers-id: Column

col

ownedby

[1]

Person: Table

worksFor: Column

P-pk: Primarykey

col

ownedby

[1]

Company: Table

PC-fk: Foreignkey

col

ownedby

[1]

Owner

Comp-id: Column

C-pk: Primarykey

col

ownedby

[1]

Table

[key]

[1]

Ref

RDB
The Generated Output RDB Model

Input pattern P1
1: EClass
1: EClass
1: EAttribute
2: EAttribute
3: EDataType

Output pattern P1'
1: Table
2: Column
3: PrimaryKey

Input pattern P2
1: EClass
1: EAttribute
2: EAttribute
3: EDataType

Output pattern P2'
1: Table
2: Column
3: PrimaryType

EMFI RDBI
Input
pattern P1
Output
pattern P2'

Addrian Rutle
DPL, MDE and MT (http://gs.hib.no)
Transformation Rules – General scheme

\[ [t] : \text{Match}(P) \rightarrow \text{Match}(P') \]
Outline

1. Introduction and Motivation
   - Diagrammatic Predicate Logic (DPL)

2. Model Transformations
   - Overview
   - Transformation Rules – Patterns and Input-Output Coordination
   - Transformation Rules – Matches
   - Coordination of rules

3. Possible Properties of Model Transformations
Coordination of Input patterns $P_1$ and $P_2$
Coordination of Output patterns $P_1'$ and $P_2'$
Transformation Rules – Composition

\[ P_1 \mapsto P_1 + P_2 \leftarrow P_2 \]

\[ \begin{align*}
C_{1,2} & \quad \text{in}_{P_1} \quad \text{in}_{P_2} \\
[PO] & \quad i_{P_1} \quad i_{P_2} \\
[i = P_1] & \quad [i = P_2] \\
[m_1 + m_2] & \quad m_1 \quad m_2 \\
\end{align*} \]

\[ P'_1 \mapsto P'_1 + P'_2 \leftarrow P'_2 \]

\[ \begin{align*}
C'_{1,2} & \quad \text{in}_{P'_1} \quad \text{in}_{P'_2} \\
[PO] & \quad i'_{P'_1} \quad i'_{P'_2} \\
[i = P'_1] & \quad [i = P'_2] \\
[t_1](m_1) & \quad [t_1 | t_2](m_1 + m_2) \quad [t_2](m_2) \\
\end{align*} \]
Properties of Model Transformations

- Tunability: flexibility and more user control
- Traceability: $\text{trace}: I' \rightarrow I$
- Incremental Consistency: preserve changes
- Bidirectionality: $\text{SRC} \leftrightarrow \text{TRG}$
- Satisfaction
- Correctness
Model Transformation: generation of target models from source models

\[ \text{Src} \xrightarrow{\text{MT}} \text{Trg} \]

MT-Definition: set of transformation rules describing how instances of the source model are transformed to instances of the target model

Transformation Rule: \( t : P \rightarrow P' \)

Coordinations between patterns: \( K \) and \( C_{i,j} \)

Rule’s semantics (Execution): \( \llbracket t \rrbracket : \text{Match}(P) \rightarrow \text{Match}(P') \)

Some properties of MT: traceability, bidirectionality etc