Traceability Visualization in Metamodel Change Impact Detection

Ludovico Iovino, Davide Di Ruscio, Juri Di Rocco and Alfonso Pierantonio

DISIM - Università degli Studi dell’Aquila – Italy
{name.lastname}@univaq.it
Outline

→ Introduction
→ MDE Ecosystem
→ Metamodel Evolution
→ Coupled Evolution Management Process
→ Change Impact Visualization
→ Dependencies Visualization in TraceVis
→ Proposed Tool Chain
→ Recurrent Patterns
→ Conclusions and Future Works
Introduction

Model Driven Engineering (MDE) aims capturing problems in terms of concepts.

Problems are mapped to solutions by means of model operations defined by transformations.

Problem and solution are described with the help of models.

Model is expressed in terms of concepts and relationships among them given in metamodels.

Metamodel is the central core of an ecosystem containing a wide range of artifacts.
MDE Ecosystem

PetriNet graphical model

Other metamodel in another ecosystem

Textual Editor

PetriNet model

PetriNet2PNML

PNML model

GMLD '13 - 2 JULY 2013, MONTPELLIER, FRANCE
Metamodel Evolution

Classification of changes:

- Do not affect the artefacts
  - No adaptation required

- Affect the artefact:
  - Automatically adapted
  - Not automatically adapted
MDE Ecosystem

Textual Editor

PetriNet metamodel initial version

PetriNet metamodel final version

PetriNet model

PNML model

Petrinet graphical model

Petrinet2PNML

Evolution

Other metamodel in another ecosystem
MDE Ecosystem

PetriNet metamodel initial version

PetriNet metamodel final version

PetriNet model

PNML model

Textual Editor

Evolution

Other metamodel in another ecosystem

PetriNet graphical model

Petrinet2PNML
Trace and visualize the dependencies within the ecosystem in order to enable the modeler to detect those modification compromising the existing artifacts.
Coupled Evolution Management Process

Change Impact Visualization

- TraceVis [1] is a tool able to graphically represent traceability information between structured data.
- TraceVis has been used also to visualize traceability links between source and target models of a transformation [2].
- Our idea: Using TraceVis to graphically represent the impact that the changes being addressed on a metamodel can have on the artefact.

Dependencies Visualization in TraceVis
Dependencies Visualization in TraceVis

Initial Metamodel
Dependencies Visualization in TraceVis

Difference Model
Dependencies Visualization in TraceVis

Existing Transformation
Dependencies Visualization in TraceVis

PetriNet metamodel initial version

PetriNet metamodel final version

Evolution
Dependencies Visualization in TraceVis
Dependencies Visualization in TraceVis
Proposed Tool Chain

MM' -> EMFCompare -> DifferenceModel.emfdiff -> EMFCompare2EcoreDiff

MM -> DifferenceModel.ecorediff

Artefact -> TraceVisGen

DependencyModel

TraceVisModel -> TraceVisMM2XML

TraceVisXML
Proposed Tool Chain

GMLD '13 - 2 JULY 2013, MONTPELLIER, FRANCE
**Proposed Tool Chain**

Differences Derivation

- **MM'**
- **MM**
- **EMFCompare**
- **DifferenceModel.emfdiff**
- **EMFCompare2EcoreDiff**
- **Differences Derivation**

**Artefact**

**DependencyModel**

**TraceVisGen**

**TraceVisModel**

**TraceVisMM2XML**

**TraceVisXML**
Proposed Tool Chain

Model2Model Transformation

Customer Model (MM)

EmfCompare

EMFCompare.emfdiff

EmfCompare2EcoreDiff

DifferenceModel.ecorediff

Artefact

TraceVisGen

Model2Model Transformation

TraceVisModel

TraceVisMM2XML

TraceVisXML

Reference Model (MM')

Dependency Model
Proposed Tool Chain

- MM'
- MM
- DifferenceModel.ecorediff
- Artefact
- DependencyModel
- TraceVisGen
- TraceVisModel
- Model2Code Transformation
- TraceVisMM2XML
- TraceVisXML

Flow:
- MM' to EMFCompare
- EMFCompare to DifferenceModel.emfdiff
- DifferenceModel.emfdiff to EMFCompare2EcoreDiff
- DifferenceModel.ecorediff to TraceVisGen
- TraceVisGen to TraceVisModel
- TraceVisModel to Model2Code Transformation
- Model2Code Transformation to TraceVisXML
Recurrent Patterns

Changes in the affected artefacts are not always directly proportional to the changes in the metamodel

Uniform impact on metamodel and artifact
Recurrent Patterns

Changes in the affected artefacts are not always directly proportional to the changes in the metamodel

Uniform impact example
Heavy metamodel evolution – Light impact on the artifact example

Extract abstract superclass with mandatory field
Light metamodel evolution – Heavy impact on the artifact

Metaclass renaming
Heavy metamodel evolution – Heavy impact on the artifact

- Usually is characterized by a strong change step passing from MM ver 1.0 to vers 3.0
Non breaking changes with no impact

Extract abstract superclass
Conclusion and future works

• The problem of adapting artifacts composing a metamodeling ecosystem when metamodel changes is intrinsically difficult

• Visualize dependencies between metamodel and artifact in term of traceability information can help the modeler to manage the complexity

• A live monitoring of the change impact reflecting on the artifact the proposed change could be an interesting application

• The visualization can be enriched with the categorization of the refactoring
Thank you!