EVOLUTION SUPPORT FOR MODEL-BASED DEVELOPMENT AND TESTING SUMMARY

Stephan Bode, Qurat-Ul-Ann Farooq, Matthias Riebisch
{stephan.bode | qurat-ul-ann.farooq | matthias.riebisch}@tu-ilmenau.de

Ilmenau University of Technology, Ilmenau, Germany

1. INTRODUCTION

The First International Workshop on Evolution Support for Model-Based Development and Testing (EMDT2010) was held on September 16, 2010 in Ilmenau, Germany. After a keynote and several paper presentations a workshop discussion was held.

2. GOALS OF THE DISCUSSION

The goal of the workshop discussion was to identify the key challenges, research questions and ideas for the support of evolution in software development and testing. Initiated by keynote and presentations, the participants from industry and academia should exchange their experiences and ideas.

3. DISCUSSION OF THE TERM SOFTWARE EVOLUTION

3.1. Key aspects of the term Evolution

Unfortunately, a clear definition of the term evolution is missing. According to Lehman and Ramil (chapter 1 of [1]), the term evolution reflects "a process of progressive, for example beneficial, change in the attributes of the evolving entity or that of one or more of its constituent elements. What is accepted as progressive must be determined in each context. It is also appropriate to apply the term evolution when long-term change trends are beneficial even though isolated or short sequences of changes may appear degenerative. For example, an entity or collection of entities may be said to be evolving if their value or fitness is increasing over time. Individually or collectively they are becoming more meaningful, more complete or more adapted to a changing environment. "

Our understanding of the term related to the workshop theme covers the following key aspects:

- Modification, change, progress, extension over time
- State of an artefact at different points of time
- Models change: dynamic versus static
- Evolution versus revolution while revolution means the replacement of an existing system by a new one

Two examples for evolution shall illustrate the change of the states:

- A change of natural language requirements leads to a change of the conceptual model, which in turn leads to a change of the class diagram as vertical evolution. This chain has to be traceable backwards.
- A change of the initial requirements (e.g. use cases) leads to a change of the functional specification, e.g. expressed by a visual contract: with pre and post conditions.

3.2. Levels and dimensions of evolution

Evolution of models in a stepwise incremental development in two dimensions:

Horizontal: to add one part after the other, leading to an increased functionality:

- V1 views PDF files,
- V2 views PDF and JPG files

Vertical: to develop parts to detailed level, leading to further refinement:

- From abstract specification to components and to code
- To achieve horizontal evolution, some vertical evolution steps may be necessary.

Evolution results in a traceable sequence of parts.

4. ASPECTS OF SEMANTICS TO BE EXPRESSED IN MODELS

A formal definition of semantics is important for transformability. The following aspects have to be expressed in such a way:

- Structure
- Class diagrams, component diagrams
- Behaviour
- State Charts, Petri Nets
- Conceptual models
- Ontology
- Functional specification
- Visual contracts [2], Java Modeling Language JML: pre and post conditions

5. IDENTIFIED RESEARCH CHALLENGES

The participants identified a set of research challenges

- Mastering complexity: modularization vs. comprehension
- Appropriate level of detail
- Appropriate models (views) for different types of tests
- Appropriate models (notations) for different domains
- Models to cover the relevant aspects of real world
- Decision on separate models for specification, for testing and development as an overhead or necessity
- Expression of semantics of data transformation / functionality
- Dependency relations between models
- Means to bridge the gaps / to overcome the walls between the stages of development
- Usage of ontologies to bridge the gap between informal requirements and design models
- Identification of generalized change types according to their consequences for different development activities
- Tool integration: establishing appropriate meta models and interfaces for:
    - Model creation
    - Code generation
    - Test case generation
- Impact analysis for evolution support: how to identify artefacts affected by changes
- Analysis of the impact of evolution on generated artefacts
- Reuse of the development artefacts during evolution, including test cases
- Software product lines – planned reuse vs. evolution
- Definition of formal criteria for evolution: legal issues for example regarding copyright

From the discussion we can conclude that all mentioned issues are related to the questions:

- Which models are necessary
- How to express the relevant aspects in models
- How to evaluate and to utilize models

6. ACKNOWLEDGEMENTS

We want to thank the contributors of the discussion for their input and their statements: Sven Biegler, Ilmenau University of Technology, Germany; Jirapun Daengdej, Assumption University, Thailand; Stefan Groß, Ilmenau University of Technology Germany; Baris Güldali, University Paderborn, Germany; Christian Kop, University Klagenfurt, Austria; Bernd-Holger Schlingloff, Humboldt University, Germany.

7. REFERENCES

[1] Meir Lehman and Juan C. Fernández-Ramil: Software Evolution. In: Nazim H. Madhavji, Juan C. Fernández-Ramil and Dewayne E. Perry: Software Evolution and Feedback: Theory and Practice, Wiley, 2006, pp 7-40.

[2] Mark Lohmann, Stefan Sauer, Gregor Engels: Executable visual contracts. In: 2005 IEEE Symposium on Visual Languages and Human-Centric Computing, pp. 63-70, IEEE CS Press, 2005.