Towards Model History Analysis Using Modeling Deltas

Dilshodbek Kuryazov and Andreas Winter
Carl von Ossietzky Universität, Oldenburg, Germany
{kuryazov,winter}@se.uni-oldenburg.de

1 Motivation

The evolving complex software models are designed and maintained by a team of designers using *collaborative modeling tools* with a support of *version control*. Collaborative modeling tools provide a teamwork of several designers on a shared modeling artifact, whereas model version control is used to store, manage and handle the histories of that model. During the evolution and maintenance process of models, model designers feel a need for *history analysis* feature for tracing and comprehending the change history of a complete model or its particular artifacts.

In order to analyze the histories or trace a particular element of an evolving model, designers need to determine answers to several questions such as (1) How often does an element change? (2) When is an element created? (3) When is an element deleted? (4) Which elements are constantly changing? (5) How does the history of an element look like? (6) How was the state of a whole model in earlier versions? (7) What are the differences between any two versions of a model? etc. These analysis questionnaires are also partly defined in [3]. For answering these questions, the change histories of modeling artifacts have to be identified and stored in appropriate ways for further analysis and manipulation. To this end, this paper presents early status of history visualization to model history analysis using modeling deltas.

The differences between subsequent model versions are represented in difference documents, also referred to as *Modeling Deltas* [2]. Modeling Deltas are executable sequence of modification operations which transform a model from one state to another. Modeling deltas represent information about the whole history of a model. Thus, modeling deltas are essential for building and developing various services and components for version control, history analysis and collaborative applications on top of them. It is quite essential to reuse and exploit the model differences in further analysis and manipulations i.e. only difference representation is useless if difference information is not reusable.

The general *Delta Operations Language (DOL)*, meta-model generic and operation-based approach is introduced in [2] to model difference representations. Conceptually, DOL is a set of domain-specific languages for model difference representation in terms of operations. A specific DOL for a specific modeling language is derived from the meta-model of a modeling language. A specific DOL is fully capable of representing model differences conforming the given meta-model in terms of DOL operations. Only changed elements between model versions are identified and represented in Modeling Deltas. Each modeling delta consists of the semantic differences between subsequent model versions. DOL-based modeling delta representation is applied to model history analysis in this paper.

The remainder of the paper is structured as follows: Section 2 gives a motivating example of DOL-based difference representation. Model history analysis using modeling deltas is discussed in Section 3. Section 4 draws some conclusions.

2 Modeling Delta Representation

In order to present the idea behind the DOL-based approach to model history analysis, this section explains a simplified example of model difference representation in terms of DOL operations.

Figure 1 depicts three subsequent versions of the same UML activity diagram. The example model illustrates the case of *ordering system*. Each concept of the model is assigned to a *persistent identifier*. The first model version has one *Receive* action. In the second version, a new action *Fill Order* and control flow `g7` are created, the name of the existing action is changed to *Receive Order*, and the target of the control flow `g4` is also changed to the new action. Then, the target of control flow `g5` is reconnected back to the final node and the created action `g6` and the control flow `g7` are deleted in the third version.

Each of the modeling concepts can be *created*, *changed* or *deleted* during the evolution process. Thus, the DOL-based approach considers only these three basic operations for representing all kind of model changes ([2], [1]).

The differences between subsequent versions of that model are represented in terms of delta operations. In order to be independent from the underlying implementation technique, the most recent version (*version 3*) of the model is also represented by DOL operations. Eventually, there are two *modeling deltas* for representing the difference between three subsequent versions

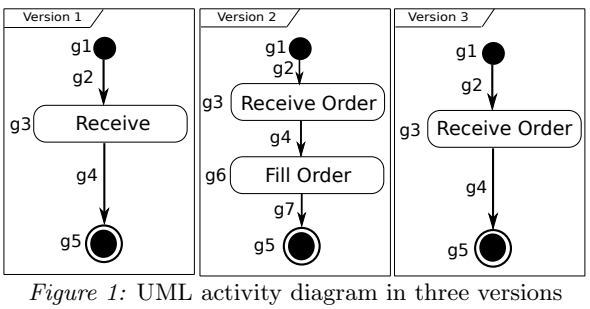

Figure 1: UML activity diagram in three versions

and one so-called *active modeling delta* for representing the recent model version. The active delta only consists of *creation* operations (Figure 2) which results in most recent version of the model.

```
1  g1=createInitialNode();
2  g3=createOpaqueAction("Receive Order");
3  g5=createActivityFinalNode();
4  g2=createControlFlow(g1,g3);
5  g4=createControlFlow(g3,g5);
```

Figure 2: Active delta

The differences delta between the third and the second versions consists of three DOL operations for creating one action (`g6=createOpaqueAction("Fill Order");`), one control flow (`g7=createControlFlow(g6,g5);`) and changing the target of `g4` (`g4.changeTarget(g6);`). In the same vein, the difference delta between the second and the first versions contains three operations changing the target of `g4` (`g4.changeTarget(g3);`), deleting `g6` (g6.delete();) and deleting `g7` (`g7.delete();`).

The approach represents differences in directed modeling deltas (*backward delta*) which are precisely executable descriptions of differences i.e. deltas are applicable to models and applying results in other version (older version in this example) of the model.

3 Model History Analysis

Analyzing model histories is the best aid in comprehending and understanding what changes are made by designers or to know how a model evolves. Also, observing the model history and its evolution process assists the users in making important decisions in further steps.

The entire set of modeling deltas in a repository represents the complete history of the model. The DOL approach also provides several DOL-services for reusing and manipulating the DOL-based modeling deltas [2]. One of these services is the *change tracer* which allows to trace the change history of a specific modeling artifact and gather required information about it. The model history analysis is built on top of the change tracer DOL-service.

The change tracer receives a list of modeling deltas and looks through a chain of modeling deltas. It seeks change information of a requested model element based on its persistent identifier by concatenating the given set of modeling deltas. The outcome of the change tracer service is a report about change history in an appropriate form. For example, the change tracer service is employed for the example in Section 2. It receives three modeling deltas (one active and two differences deltas) as input and it is requested to return history reports for the control flow `g4`. The resulting list of changes is depicted in Figure 3.

```
1  g4=createControlFlow(g3,g7);
2  g4.changeTarget(g6);
3  g4.changeTarget(g5);
```

Figure 3: History information of Control Flow `g4`.

Finally, the detected change history information can be used in further analysis by different visualizations. This approach uses a tabular view to visualize change information, but difference information can be visualized in any other forms, like model, tree, graph or even textual. Importantly, most of the questions stated in Section 1 can be answered in the current status of the visualization.

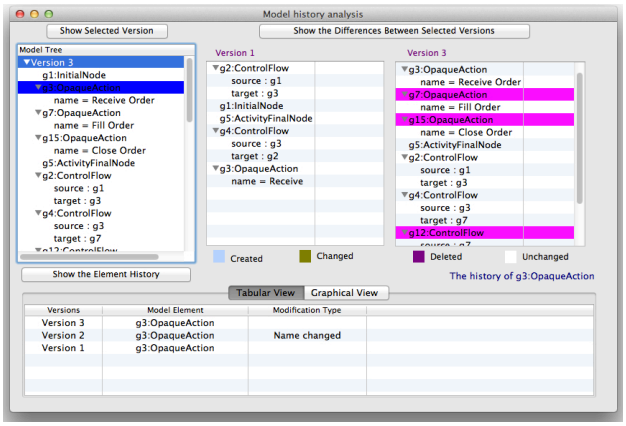

Figure 4: History Analysis User Interface

The screen shot in Figure 4 displays the example model in Section 2. The user interface shows the model tree on the left, including all versions with their elements. One model version can be selected, and a whole version can be seen by clicking *Show Selected Version* button or any two versions can be compared in the tabular view on the right side highlighting different kinds of changes with different colors. To see the history information of any model element, it can be selected from the model tree. The change history information is listed on the table on bottom.

4 Conclusion

This paper has addressed model history analysis using the DOL-based modeling deltas to model difference representations. The DOL representation is an appropriate approach for difference representation as, it makes the data representation efficient and allows suitable data structure for data processing. Modeling deltas embody all necessary information about the complete change history of a model. These information can easily be extracted and reused by the DOL-services in further analysis.

Implementation of the analysis tool is planned to be finalized in near future and the whole set of analysis questions will be covered by this tool.

References

[1] Dilshodbek Kuryazov. Delta operations language for model difference representation. In Plödereder et. al., editor, *44. Jahrestagung der Gesellschaft für Informatik*, volume 232, Stuttgart, 2014. GI.

[2] Dilshodbek Kuryazov and Andreas Winter. Representing model differences by delta operations. In Reichert et al., editor, *18th International Enterprise Distributed Object Oriented Computing Conference, Worshops and Demonstrations (EDOCW)*, pages 211–220, Ulm, 2014. IEEE.

[3] Sven Wenzel. How to trace model elements? In *9. Workshop Software-Reengineering (WSR'07)*, Bad Honnef, May 2007.