Methods of Model Quality in the Automotive Area

Robert Reicherdt and Sabine Glesner

Software Engineering of Embedded Systems
Technische Universität Berlin
Ernst-Reuter-Platz 7
10587 Berlin
robert.reicherdt@tu-berlin.de
sabine.glesner@tu-berlin.de

Abstract: MATLAB Simulink is the most widely used industrial tool for developing complex embedded systems in the automotive industry. These models often consist of multiple thousands of blocks and a large number of hierarchy levels. The increasing complexity leads to challenges for quality assurance as well as for maintenance of the models. In this article, we present an novel approach for slicing Simulink Models using dependence graphs and demonstrate its efı̈ciency using case studies from the automotive and avionics domain. With slicing we can reduce the complexity of the models removing unrelated elements, thus paving the way for subsequent static quality assurance methods. Moreover, slicing enables the use of techniques like change impact analysis on the model level.

Model-based development (MBD), especially based on MATLAB/Simulink, is a well established technique in automotive area and widely used in industry. However, despite the increasing significance of MDB, quality assurance and maintenance techniques are quite immature compared to to the techniques present for classical software development. In our Project *MeMo - Methods of Model Quality* [HWS+11], we have aimed to develop new analysis approaches and techniques to increase the quality of MATLAB/Simulink models. This was joint work together with two industrial partners from the area of model and software quality assurance.

One major result of the MeMo project is our novel approach [RG12] for the slicing of MATLAB/Simulink models. This approach consists of two parts: (1) a dependence analysis for MATLAB/Simulink models and (2) the slicing based on a reachability analysis. The dependence analysis in our approach is based on the sequential simulation semantics of MATLAB/Simulink models. While we can derive data dependences directly from the lines representing signal flow in the model, we derive control dependences from the *conditional execution contexts* of the models. MATLAB/Simulink uses conditional execution contexts for the conditional execution of blocks in the model. However, these contexts cannot be extracted directly, neither from the model file nor via the MATLAB/Simulink API. Hence, we have reimplemented the calculation using safe overapproximations for cases in which the informal and incomplete MATLAB/Simulink documentation does not contain sufficient information. Subsequent to the dependence analysis, we create a dependence graph and calculate the slices using a forward or backward reachability analysis.

We have implemented our approach into a tool which is able to parse a model, to store it in a database and to slice the model. To get the necessary information, we have implemented our parser in two phases. In the first phase we parse the models from the model files to build a skeleton. In the second phase we then use the MATLAB/Simulink API to extract additional run-time information (e.g., port widths and signal dimensions) from the models, which are not available directly from the model file.

In the evaluation on a number of cases studies we were able to show an average reduction of 45% in size of the models using our original approach. By now, we have extended our approach with a more precise analysis of data dependences in bus systems, which led to average slice sizes around 37%. This is another reduction of around 12% compared to our original approach.

In the last decade, only few approaches for the slicing of modeling notations for reactive systems have been published. However, most of these approaches have aimed at state-based notations such as extended finite state machines or state charts. A comprehensive overview for these approaches is given in [ACH+10]. To the best of our knowledge, our approach is the first approach published about the slicing of MATLAB/Simulink models.

Besides the ability to reduce the complexity of a model for a specific point of interest (i.e. a block or a signal) this approach offers new possibilities in the development and maintenance of models in MBD. Like in classical software development, in MBD it is important to track changes and especially their impact on the modeled systems. With our slicing technique, we are now able to lift slicing-based *change impact analysis* from the level of classical software development to the model level.

To do so, we have started a new project, the CISMo project. Together with an industrial partner, we aim to develop a novel slicing-based change impact analysis for MATLAB/Simulink models. Besides the development of this analysis, the project also aims to enhance this analysis with heuristics and a parametrization to increase scalability and tailor it to the needs of industrial applications. Moreover, we still plan to extend our slicing approach to *Stateflow* which is often used in MATLAB/Simulink models and is a state-based notation. With this extension, we aim to gain even more precision.

References

[ACH+10] K. Androutsopoulos, D. Clark, M. Harman, J. Krinke, and L. Tratt. Survey of slicing finite state machine models. Technical report, Technical Report RN/10/07, University College London, 2010.

[HWS+11] Wei Hu, Joachim Wegener, Ingo Stürmer, Robert Reicherdt, Elke Salecker, and Sabine Glesner. MeMo - Methods of Model Quality. In *Dagstuhl-Workshop MBEES: Modellbasierte Entwicklung eingebetteter Systeme VII*, pages 127–132, 2011.

[RG12] Robert Reicherdt and Sabine Glesner. Slicing MATLAB Simulink models. In *34th International Conference on Software Engineering (ICSE)*, pages 551–561, 2012.