Supporting Semi-Automatic Co-Evolution of Architecture and Fault Tree Models

Sinem Getir[1], André van Hoorn[2], Timo Kehrer[3], Yannic Noller[4], Matthias Tichy[5]

Abstract: In this work, we report about recent research results on "Supporting Semi-Automatic Co-Evolution of Architecture and Fault Tree Models", published in [Ge18]. During the whole life-cycle of software-intensive systems in safety-critical domains, system models must consistently co-evolve with quality evaluation models. However, performing the necessary synchronization steps is a cumbersome and often manual task prone to errors. To understand this problem in detail, we have analyzed the evolution of two representatives of system models and quality evaluation models, namely architecture and fault tree models, for a set of evolution scenarios of a factory automation system called Pick and Place Unit. We designed a set of intra- and inter-model transformation rules which fully cover the evolution scenarios of the case study and which offer the potential to semi-automate the co-evolution process. In particular, we validated these rules with respect to completeness and evaluated them by a comparison to typical visual editor operations. Our results show a significant reduction of the amount of required user interactions in order to realize the co-evolution.

Keywords: System architecture, fault trees, safety, model co-evolution, model transformation

Summary

A rigorous quality evaluation is among the key methods for the dependable engineering of software-intensive systems in safety-critical domains. To that end, model-based approaches have been proposed which use quality evaluation and system models to gain knowledge about the quality of a system. In model-based quality evaluation, the consistency of the involved models is of utmost importance. For example, the failures of an architectural component must be adequately considered in an associated fault tree model. While this consistency requirement can be reasonably met for a particular snapshot of a system, quality evaluation models typically become outdated when the system evolves, i.e., quality evaluation models and system models evolve in an inconsistent way. As a consequence, quality evaluation leads to highly improper results. Hence, loosely inter-related models such as architectural models and quality evaluation models like fault trees should consistently evolve in parallel, a phenomenon to which we refer to as (consistent) *model co-evolution*.

Since loosely inter-related models are typically changed in isolation of each other, one adequate approach to support developers is *model synchronization*, i.e., the task of adapting a model in response to changes in one of its inter-related counterparts in order to achieve consistent co-evolution. In general, however, achieving this kind of model co-evolution cannot

[1] Humboldt-Universität zu Berlin. getir@informatik.hu-berlin.de
[2] University of Stuttgart. van.hoorn@informatik.uni-stuttgart.de
[3] Humboldt-Universität zu Berlin. timo.kehrer@informatik.hu-berlin.de
[4] Humboldt-Universität zu Berlin. noller@informatik.hu-berlin.de
[5] University of Ulm. matthias.tichy@uni-ulm.de

doi:10.18420/se2019-13

be fully automated as usually assumed by existing approaches to model synchronization. At best, developers may be supported by *recommending* possible synchronization actions, as, e.g., in the model-based (co-)evolution framework known as CoWolf [Ge15]. To achieve consistent co-evolution, CoWolf follows a rule-based approach where incremental model transformations are used to recommend both intra- and inter-model change actions. However, since the adequacy of the recommendations strongly depends on the transformation rules being used by the tool, the evolution problem is shifted to the engineering of proper transformation rules.

We tackle this problem of engineering proper transformation rule sets for an important class of models in the context of model-based hazard analysis, namely system architecture models and fault tree models. We extend our previous work [Ge13] on the evolution of the so-called Pick and Place Unit (PPU) [LFV13], a case study from the automation engineering domain. To study co-evolution in terms of the PPU, we created consistent software architecture and fault tree models for all safety-relevant evolution scenarios. Thereupon, we conducted a thorough quantitative analysis of the evolution scenarios with respect to the co-evolution of the models, i.e., how changes in one model affect changes in the other model. We show that the models do not co-evolve in a systematic, automatable way and instead expertise of the developer is required to achieve co-evolution. Moreover, we developed a set of model transformation rules for 1) the independent evolution of software architecture and fault tree models and 2) the synchronization of one model based on changes in another model ensuring a correct co-evolution of both models. We show that the presented set of model transformations is *complete*, i.e., it supports performing all co-evolutions of the case study scenarios, and improves the *task efficiency* by reducing the amount of required model transformation applications to realize the co-evolution by, on average, 52% compared to visual editing operations and 85% compared to atomic model changes. Finally, we implemented these rules in the tool CoWolf to enable the co-evolution of fault trees and software architecture models.

References

[Ge13] Getir, S.; Van Hoorn, A.; Grunske, L.; Tichy, M.: Co-Evolution of Software Architecture and Fault Tree models: An Explorative Case Study on a Pick and Place Factory Automation System. In: NiM-ALP @ MoDELS'13. Pp. 32–40, 2013.

[Ge15] Getir, S.; Grunske, L.; Bernasko, C. K.; Käfer, V.; Sanwald, T.; Tichy, M.: CoWolf–A generic framework for multi-view co-evolution and evaluation of models. In: ICMT'15. Springer, pp. 34–40, 2015.

[Ge18] Getir, S.; Grunske, L.; van Hoorn, A.; Kehrer, T.; Noller, Y.; Tichy, M.: Supporting semi-automatic co-evolution of architecture and fault tree models. Journal of Systems and Software 142/, pp. 115–135, 2018.

[LFV13] Legat, C.; Folmer, J.; Vogel-Heuser, B.: Evolution in Industrial Plant Automation: A case study. In: IECON'13. IEEE, pp. 4386–4391, 2013.