M. Tichy, E. Bodden, M. Kuhrmann, S. Wagner, J.-P. Steghöfer (Hrsg.): SE 2018,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2018 139

Experiences on Traceability and Consistency Checking across Engineering Tools in an Automation Solution Company

Andreas Demuth[1], Roland Kretschmer[2], Michael Trölss[2], Georgios Kanakis[2], Davy Maes[3], and Alexander Egyed[2]

Abstract: Engineers continuously adapt systems to changing requirements, which is particularly then a challenge when different engineering domains come together. Since engineers of different domains use quite distinct engineering tools, consistent change propagation is essential. This paper discusses experiences with a leading company in the area of production automation in maintaining the consistency between electrical models and the corresponding software controller when both are subject to continuous change. This is complicated by the fact that these engineer use different kinds of tools to capture and maintain models and code.

Keywords: Traceability, Consistency Checking, Experience Report, DesignSpace, Collaboration;

1 Introduction

The engineering of systems is unimaginable without software tools. Engineers use them to capture and analyze engineering problems; specify, implement, test, and maintain engineering solutions, and manage engineering processes. Yet, there is a gap between the capabilities of independently working engineers and the needs of a collaborative engineering team. The existing tool landscape emphasizes the former. Most engineering tools are single-user applications – often of excellent quality but limited in that they support the works of individual engineers and not that of a group of engineers. Herein lies one of the most fundamental problems of software and systems engineering. Engineers know well the engineering tools they use and the engineering knowledge they capture. Yet, engineers lack awareness of the many implications their work has on other engineers and/or other engineering domains. This is a problem because in today's engineering projects, companies continuously adapt their systems to changing customer or market requirements. This requires a flexible, iterative development process in which engineers build and update different parts of the system under construction concurrently.

This paper discusses such an experience in context of the construction of a conveyor belt system with Van Hoecke Automation. This system requires the collaboration among electrical engineers and software engineers. Not only do these engineers work on

[1] Dynatrace Austria GmbH, Freistaedter Strasse 313/2, 4040 Linz, Austria, andreas.demuth@jku.at
[2] Johannes Kepler Universität, ISSE, Altenbergerstraße 69, 4040 Linz, Austria, [firstName].[lastName]@jku.at
[3] Flanders Make, Oude Diestersebaan 133 Lommel, 3920 Belgium, davy.maes@flandersmake.be

different perspectives of that system but they use quite distinct tools to capture their engineering knowledge. As such, the electrical engineer captures electrical circuit diagrams in EPlan Electric P8 and the software engineering captures source code in Eclipse. Neither tool knows about the existence of the other. The key question that we asked was how these engineers could be made aware of the respective implications of their changes on each other's domain.

To address this problem, we developed the DesignSpace cloud [De15] to let engineers define relationships among arbitrary development artifacts. This way engineers can connect e.g. a motor element from the circuit diagram to its respective piece of code. Engineers define these relationships through explicit links in a wizard-style interface. These links (i.e., traceability [An01]) then provide the basis for consistency checking among the artifacts of these two tools. Changes engineers make in their tools are instantly synchronized with the DesignSpace and engineers receive instant error feedback if such changes violate defined consistency rules. For scalability, the DesignSpace cloud utilizes the Model/Analyzer approach to fast, incremental consistency checking [Eg11]. Further details about the case study are provided in an experience report [De16] we published previously at the International Conference on Software Maintenance and Evolution.

2 Acknowledgement

This research was supported by the Austrian Science Fund (FWF): P25289 N15, the Austrian COMET-K2 Programme of the Linz Center of Mechatronics (LCM), and a grant of the JKU Linz Institute ofTechnology (LIT)

3 References

[An01] Giuliano Antoniol, Bruno Caprile, Alessandra Potrich, and Paolo Tonella. Design-code traceability recovery: selecting the basic linkage properties. *Sci. Comput. Program.*, 40(2-3):213–234, 2001.

[De16] Andreas Demuth, Roland Kretschmer, Alexander Egyed, and Davy Maes. Introducing traceability and consistency checking for change impact analysis across engineering tools in an automation solution company: An experience report. In *International Conference on Software Maintenance and Evolution (ICSME), USA, 2016*.

[De15] Andreas Demuth, Markus Riedl-Ehrenleitner, Alexander Nöhrer, Peter Hehenberger, Klaus Zeman, and Alexander Egyed. Designspace: an infrastructure for multi-user/multi-tool engineering. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, April 13-17, 2015*, pages 1486–1491, 2015.

[Eg11] Alexander Egyed. Automatically detecting and tracking inconsistencies in software design models. *IEEE Transactions on Software Engineering*, 37(2):188–204, 2011.