An Approach to Requirement Analysis in Automated Production Systems

Timo Maier[1], Kiana Busch [2], Ralf Reussner [2]
Karlsruhe Institute of Technology (KIT),
[1] timo.maier@student.kit.edu, [2]{kiana.busch, reussner}@kit.edu

Abstract Automated production systems (aPS) involve different disciplines, like mechanical and software engineering. Evolution has to be seen as a repetitive activity in these systems. Complexity of hardware and especially software is constantly rising and demands for automated solutions, as change propagation analysis by hand is slow and error-prone. In this paper, we present an approach to automatically calculate change propagation based on requirement changes in aPS.

1 Introduction

Automation has become a crucial success factor for manufacturing industries. aPS are software-controlled mechanical systems, which are under operation for several decades [8]. During their lifetime aPS are subject of evolution due to new technological developments or changing requirements [7]. aPS involve multiple disciplines, as they comprise both hardware and software. The hardware further includes mechanical (e.g. fixtures) and electrical parts (e.g. sensors). Mutual dependencies between these disciplines cause evolution and change management to be challenging [7]. But, changes are often implemented ad hoc by responsible employees. Thus, change management in aPS is not well documented [7].

This paper presents an approach to automate change propagation analysis by extending an existing maintainability framework to support requirements in aPS. Automating change analysis allows profound documentation. Furthermore, the consideration of requirements allows to work on a higher level of abstraction and increases usability.

2 Foundation

Our approach extends the existing Karlsruhe Architectural Maintainability Prediction (KAMP) [6]. KAMP is an architecture-based approach to change propagation and maintenance effort estimation, using models as primary artifacts. KAMP can be applied to architectural elements of component-based software (e.g., interfaces). Propagation is then automatically calculated based on these changes. The result is a task list containing all necessary steps to implement a change. KAMP4aPS [8] and KAMP4IEC [5] extend KAMP to hardware and control software in aPS, respectively.

To formalize and trace requirements and design decisions we used an existing metamodel persisting requirements and design decisions [2, 4]. An extension to this metamodel [3] also considers options (see Fig. 1), which represent alternatives to resolve requirements and design decisions. For example, if a new requirement demands to consider the color of workpieces, and a design decision is made to use an optical sensor, the corresponding option is "Introduce new optical sensor".

3 Related Work

There already exist approaches to automated change propagation analysis in aPS (e.g., [1]). However, they often do not consider requirements. The evolution of aPS is discussed in [7]. It provides a comprehensive analysis of the topic. But, no practical approach is presented.

4 Approach

In order to consider requirements in the change propagation analysis of aPS we developed KAMP4aPS4Req. The metamodels for requirements and design decisions [2, 4, 3] are included into the change propagation analysis of KAMP for the aPS domain. This allows for specifying changes on a higher level of abstraction by changing requirements and design decisions instead of architectural elements. To

support a modular architecture, the approach is separated in three modules (cf. Fig. 1): the common module which applies to both hardware and software, and a separate module for each of those. This enables us to use the approach for hardware without needing the software module and vice versa. The change propagation first deals with the requirement level. It applies to both hardware and software, as requirements, design decisions, and options are specified for the aPS as a whole. Afterwards, change propagation on the architecture level is calculated separately. The following subsections present the change propagation analysis in more detail.

4.1 Requirement level

The starting point of the change propagation analysis is a set of changed requirements. The metamodels provide references from requirements to design decisions and options. A user can specify design decisions for each requirement and the corresponding options for requirements and design decisions. These references are then used to calculate the change propagation from requirements to design decisions and options, as seen in Fig. 1. The change calculation algorithm is implemented in a set of change propagation rules along the references. As change propagation on the architecture level is not involved yet, this step is independent of the underlying architecture, allowing it to be exchangeable.

4.2 Architecture level

The options, that the user created in the first step in their turn reference specific elements in the architecture. Using these references, the propagation to the architecture level is calculated. This step is performed separately for hardware and software, which manifests in separate implementations. If an option references an architectural element, this architectural element is subject of change and, thus, is marked as changed. However, this changed element can lead to a number of other architectural elements which have to be changed in their turn. Tracking of this further propagation within the hardware and software model is then taken care of by KAMP4aPS and KAMP4IEC, respectively. The resulting task list then contains all needed changes within the hardware and software models, as well as affected requirements and design decisions.

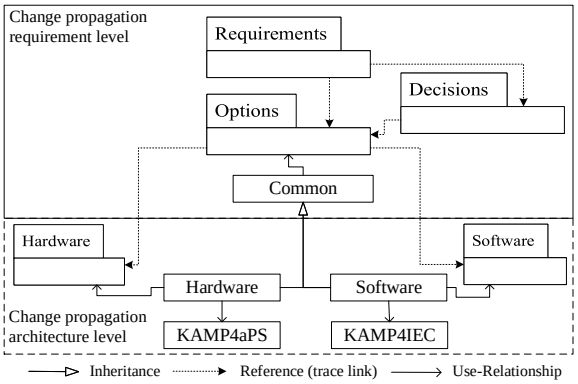

Figure 1: Overview of the KAMP4aPS4Req approach

5 Conclusion

We presented KAMP4aPS4Req, which enables support of automatic derivation of change propagation based on requirement changes in aPS. Our approach can be used to automatically estimate the effort of change implementation in advance. Knowledge about the change analysis process is made explicitly available, instead of relying only on the experience of responsible employees.

References

[1] T. Kehrer et al. “Propagation of Software Model Changes in the Context of Industrial Plant Automation”. In: *Automatisierungstechnik* 62 (2014).

[2] Zoya Durdik and Ralf Reussner. “On the Appropriate Rationale for Using Design Patterns and Pattern Documentation”. In: *QoSA*. 2013.

[3] Rene Hahn and Peter Schuller. “Architecture as Connection between Requirements and Quality Prediction to Support Design Decisions”. In: *Praxis der Forschung, KIT*. 2015.

[4] Martin Küster. “Architecture-Centric Modeling of Design Decisions for Validation and Traceability”. In: *ECSA’13*. Vol. 7957. Springer, 2013.

[5] Jannis Rätz. “Erweiterung eines Wartbarkeits-Frameworks für die Programmiersprache IEC 61131-3”. Bachelor’s Thesis. KIT, 2017.

[6] K. Rostami et al. “Architecture-based assessment and planning of change requests”. In: *QoSA*. 2015.

[7] Birgit Vogel-Heuser et al. “Evolution of software in automated production systems: Challenges and research directions”. In: *JSS* 110 (2015).

[8] Birgit Vogel-Heuser et al. “Maintenance effort estimation with KAMP4aPS for cross-disciplinary automated PLC-based Production Systems - a collaborative approach”. In: IFAC (2017).