Semi-automated decision making support for undocumented evolutionary changes

Jan Ladiges, Alexander Fay
Automation Technology Institute
Helmut Schmidt University
Holstenhofweg 85, 22043 Hamburg, Germany
Email: {ladiges, fay}@hsu-hh.de

Christopher Haubeck, Winfried Lamersdorf
Distributed Systems and Information Systems
University of Hamburg
Vogt-Kölln-Straße 30, 22527 Hamburg, Germany
Email: {haubeck, lamersdorf}@informatik.uni-hamburg.de

1 Introduction

Long-living systems evolve under boundary conditions as diverse as the systems themselves. In the industrial practice of the production automation domain, for example, adaptations and even the initial engineering of control software are often performed without a formalized requirement specification [1].

Nevertheless, operators must decide during operation if such an undocumented change is consistent to the (informal) specification since changed behavior can also occur due to unintended side effects of changes or due to other influences (like wear and tear). In addition, the system behavior is strongly dependent on both, the software and the physical plant. Accordingly, approaches are needed to extract requirements out of the interdisciplinary system behavior and present it to the operator in a suitable format.

The FYPA²C project (Forever Young Production Automation with Active Components) tries to realize an extraction of behavior related to non-functional requirements (NFR) by monitoring and analyzing signal traces of production systems. In doing so, the specific boundary conditions of the production automation domain should be considered.

2 The Evolution Support Process

The assumption of this approach is that the externally measured signal traces of programmable logic controllers (PLCs) provide a basis to capture the NFRs on the system. Fig. 1 shows how low-level data (the signals) can be lifted to high-level NFR-related information. First, the signal traces which are created during (simulated) usage scenarios are used to automatically generate and adapt dynamic knowledge models. Such models are e.g. timed automata learned by the algorithm described by Schneider et al. in [2]. Each model expresses specific aspects of the system and serves as a documentation of the underlying process. An analysis of these models can provide NFR-related properties of the system in order to evaluate the influences of changes. Such properties are e.g. the throughput rate or the routing flexibility (see [3]).

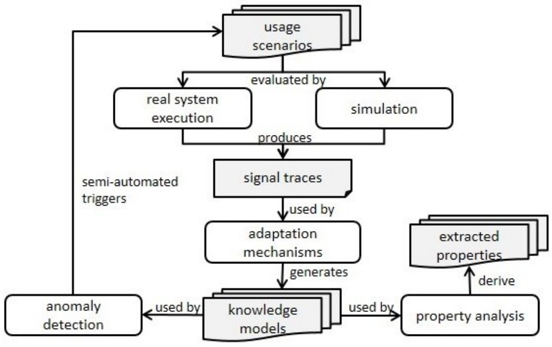

Figure 1: Process of extracting system properties

Similar work has been done in [4]. Here, automata are generated out of test cases which are e.g. derived from design models. An invariant analysis allows for extracting functional requirements which can be monitored. However, the FYPA²C approach assumes that no formal models or test-cases are present and it aims at the extraction of NFRs.

Since not every I/O-signal of a PLC includes information about needed aspects, a selection of the signals has to be done. Therefore, signals get enriched by semantics. The semantics include which kind of information is given by the signal. A signal stemming from a sensor which identifies the material of a workpiece (e.g. a capacitive sensor distinguishing workpieces) would get the semantic *workpieceIdentification*. Note that enriching signals is a rather simple step compared to creating e.g. design models.

Since a monitoring system cannot decide if a performed change and its influences on the NFRs is intended (or at least acceptable), a practical semi-automated evolution support process with a “user in the loop” is used. At first an anomaly detection engine detects whenever a behavior is observed that contradicts the knowledge models and, therefore, can indicate an evolutionary change. In case of timed automata the anomaly detection method presented in [2] is used. This anomaly is, in a first step, reported to the user. At this point only the actual anomaly, the context it occurs in, and a limited amount of current properties and probable influences can be reported since only influences on the already observed scenarios can be considered. Deductions on the overall properties are very restricted at this point. If a decision cannot be made here, the changed behavior is added to the concerned knowledge models in order to evaluate the effects on the system properties in detail. This is done by an analysis based on the extracted scenarios that are applied on the plant or a simulation. The advantage of these steps is that the operator can be informed based on the overall NFR-related properties of the system. As a reaction the change can be reverted if unintended or, if it is intended, adapted scenarios and models can be treated as valid.

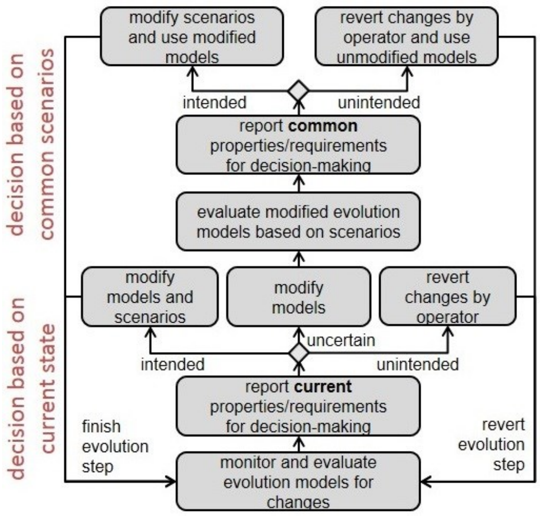

Figure 2: Semi-automated evolution support process

If there is no possibility for a proactive determination of the system properties (missing of simulation and no availability of the system for tests), an adaptation of the models during operation is the only remaining option and just the already observed changes can be evaluated. When an unacceptable influence is observed the operator can react accordingly. However, the scenarios observed after the occurring change can be compared to the stored ones in order to estimate the completeness of the adapted knowledge models.

To be more precisely, consider the following simple example: A conveyor system is responsible for transporting workpieces to a machine located at the end of the conveyor system. Workpieces are detected by lightbarriers at both ends of all conveyors. A requirement on the throughput rate demands that the transport does not take longer than 60 seconds. A PLC collects the signals stemming from the lightbarriers and starts the transport when a workpiece reaches the first conveyor and stops it, when the workpiece reaches the machine. Conveyor speed can be parameterized within the PLC-program. A timed automaton (as a knowledge model) represents the transportation and is learned based on the observed signal traces by the learning algorithm in [2].

The automaton should just include signals related to the transportation. Therefore all I/O signals of the PLC are enriched by simple semantics and the learning algorithm is applied only on signals with the given semantic *workpieceDetection*. These are all signals stemming from lightbarriers. Accordingly, an analysis on the automaton enables deducing the transporting times by aggregating the transition times. Due to maintenance the motors of the conveyors are exchanged by motors with a higher slip resulting in a slower transportation. Unfortunately, the operator did not adapt the parameters in the PLC. During the first run of the plant the slower transportation is detected as a time-anomaly and reported to the operator after the workpiece passed the first conveyor. The operator can now decide if the anomaly is intended (or at least acceptable) or not. If he is not able to do this decision, for example due to a high complexity of the conveyor system, he can declare the anomaly as uncertain and the knowledge model gets further adapted during the transportation until a deduction about the fulfillment or violation of the throughput requirement can be done. If the requirement is violated the operator can react accordingly by changing the parameters in the PLC code.

References

[1] G. Frey, L. Litz, “Formal methods in PLC programming,” in *Intl Conf on : Systems, Man, and Cybernetics,* vol.4, 2000.

[2] S. Schneider, L. Litz, and M. Danancher, “Timed residuals for fault detection and isolation in discrete event systems,” in *Workshop on : Dependable Control of Discrete Systems*, 2011.

[3] J. Ladiges, C. Haubeck, A. Fay, and W. Lamersdorf, “Operationalized Definitions of Non-Functional Requirements on Automated Production Facilities to Measure Evolution Effects with an Automation System,” in *Intl. Conf. on Emerging Technologies and Factory Automation*, 2013.

[4] C. Ackermann, R. Cleaveland, S. Huang, A. Ray, C. Shelton, E. Latronico, „Automatic requirement extraction from test cases,“ in *Intl. Conf. on Runtime Verification*, 2010.