Control Engineering Practice 73 (2018) 171–185

Contents lists available at ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

Increasing system test coverage in production automation systems

Sebastian Ulewicz *, Birgit Vogel-Heuser

Technical University Munich, Institute of Automation and Information Systems, Boltzmannstr. 15, 85748 Garching near Munich, Germany

ARTICLE INFO

Keywords:
Automated production system
System testing
Programmable logic controller
Test coverage assessment
Code instrumentation

ABSTRACT

An approach is introduced, which supports a testing technician in the identification of possibly untested behavior of control software of fully integrated automated production systems (aPS). Based on an approach for guided semi-automatic system testing, execution traces are recorded during testing, allowing for a subsequent coverage assessment. As the behavior of an aPS is highly dependent on the software, omitted system behavior can be identified and assessed for criticality. Through close cooperation with industry, this approach represents the first coverage assessment approach for system testing in production automation to be applied on real industrial objects and evaluated by industrial experts.

1. Introduction

Automated production systems (aPS) in factory automation have high requirements regarding availability and reliability (Vogel-Heuser, Fay, Schaefer, & Tichy, 2015), as these systems typically run over long periods of time (decades) and system failures or incorrect behavior can increase costs. The volume and complexity of aPS's software has risen substantially over the last decade (Vyatkin, 2013), exacerbating the problem of ensuring reasonable system quality. This quality is typically investigated and assured by testing. Apart from unit tests performed on single software modules in an early design phase, system tests of the integrated functionality of software and hardware are defined and performed in late phases of development, often as late as during on-site plant commissioning.

From the authors' experience, test plans for system testing exist in most companies in the field of automated production systems engineering, yet the definition of the individual test cases is abstract and generic. On the one hand, large parts of these test plans can be reused between projects, on the other hand, the individual test cases leave a lot of room for interpretation during the testing process. Additionally, tests are performed manually, as many functions are not related to the software alone but also to the integrated system comprised of mechanical and electrical hardware as well as software. Thus, many actions performed during these tests, such as placing intermediate products into the machine and visually verifying the correct product quality, cannot be performed fully automatically: Sensors and actuators that would enable automated testing are not available due to their cost. Instead, the test operator is required to perform these actions manually. This testing process is often performed under high time pressure in an uncomfortable on-site environment and based on the mentioned vaguely specified requirements on the system. This results in uncertainty of the adequacy of the performed tests: The adequacy of the performed tests to ensure the abstractly defined required functionality is often based on the experience and intuition of the test operator. Subsequently, the possibility of not testing critical behavior and thus overlooking critical faults in the system represents a realistic problem.

Code coverage is a possibility to assess test adequacy (Zhu, Hall, & May, 1997). As the behavior of the integrated automated system is largely dependent on the software, a coverage assessment of implemented behavior can be performed: by identifying uncovered (untested) code, unintended omissions of testing system behavior can be revealed. Based on this finding, an approach was developed consisting of an instrumentation of the control software to allow for recording of execution traces and an analysis of these traces for coverage assessment and identification of untested code. The approach was implemented in a prototypical tool and evaluated using a real industrial case study and a subsequent expert evaluation yielding promising results.

The main contribution of the presented approach is the ability to identify untested behavior during system testing of fully-integrated industrial production automation systems controlling discrete processes without the need for formalized requirements or simulations. Thus, for the first time, the approach provides valuable support in quantitatively assessing and increasing testing quality in fully-integrated industrial aPS in industrial quality assurance scenarios.

The structure of the paper is as follows: In Section 2, an overview of requirements gathered from industrial experts is presented in order to rate existing approaches and to guide the development of the presented

* Corresponding author.
E-mail address: sebastian.ulewicz@tum.de (S. Ulewicz).

https://doi.org/10.1016/j.conengprac.2018.01.010
Received 19 March 2017; Received in revised form 11 November 2017; Accepted 22 January 2018
0967-0661/© 2018 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

approach. Section 3 presents related work in the field of production automation and adjacent domains. These works are analyzed and a research gap is identified. In the concept section (Section 4), the approach is described in detail regarding the choice of the coverage metric, the code instrumentation and coverage assessment. Information about the implementation of the approach is detailed in Section 5 followed by an application on a case study and an expert evaluation in Section 6 which allows for a qualitative impression of the concept's performance and applicability. In Sections 7 and 8, a conclusion and an outlook on future work are discussed.

2. Industrial requirements regarding system testing in production automation

The aim of the approach presented in this work is to be closely aligned with industrial requirements in the field of production automation engineering. For this reason, multiple workshops with up to seven highly experienced experts from three internationally renowned companies active in this field were conducted to infer these requirements. Six main requirements regarding applicability of a possible approach were identified:

Requirement 1 (R1): support of industrial software properties

Programming standards relevant for the industry, i.e. the programming standard IEC 61131-3 for Programmable Logic Controllers (PLC) dominant in the engineering of automated production machines, has to be supported. Of the five defined programming languages, Structured Text (ST) and Sequential Function Chart (SFC) in particular need to be supported as these are the most commonly used programming languages within the companies questioned.

Requirement 2 (R2): real time capability and memory size

The approach should not influence the real time properties of the tested system in a way that would not permit needed real time capabilities of the system to hold. The needed real time capabilities are seen as unaffected if a possible increase in execution time of modified code does not lead to the PLC scan cycle time to be exceeded. In addition, possibly increased size of compiled control code software should not lead to exceeded memory on the execution hardware (PLC).

Requirement 3 (R3): inclusion of hardware and process behavior

Testing a system integrates all parts of the system, meaning software, hardware and the controlled process. To be able to assess a system's conformance to its specification, all parts should be as similar to the final system as possible, i.e. the software running on the final execution hardware, controlling the final version or the hardware setup and technical process. For this reason, using a simulation rather than the real hardware is often not sufficient for final system tests, as the validity of the described behavior is a simplification of real hardware behavior. In addition, simulations are costly to produce – in particular for aPS produced in small lot sizes – and automatic generation of simulations with available documents as proposed by Barth and Fay (2013) and Puntel-Schmidt et al. (2014) are not available in many cases and for the participating industry partners in particular. This problem especially applies to medium and smaller sized companies, where an approach which is independent of simulations is required, as these are often no option for system testing in production automation for economic reasons.

Requirement 4 (R4): manipulation of hardware and process behavior

The approach needs to be applicable on real industrial testing use cases, as defined by the currently performed system test cases in the company. System tests, as described in this approach, are defined as black box tests (test derived from a specification rather than the code itself) of a fully integrated system comprised of software, controlled hardware and the technical process. The tests include manual manipulations of the hardware or technical process that cannot be performed by the software. As an example, manually opening and closing doors or putting intermediate products in the machine can be typical operations during system testing.

Requirement 5 (R5): no need for formalized functional requirements

The problems stated in the introduction could be mitigated using more detailed and formalized functional requirements. Using a connection between requirements, test cases and models of different engineering views of the system could enable validation of the involved models (Estevez & Marcos, 2012) and more detailed relation between requirements and a system's software code itself could be created using static feature location techniques (Dit, Revelle, Gethers, & Poshyvanyk, 2013). Yet in practice, this would require adequate software tools, substantial effort regarding training and additional resources for specification for each new engineering project. As this tradeoff between an initial investment and its outcome is highly speculative, according to the participating industrial experts, the approach must be independent from formalized functional requirements.

Requirement 6 (R6): support the assessment of test adequacy (finding untested behavior)

Here, the approach is to increase efficiency and quality during the quality assurance process of special purpose machinery by supporting the tester, who might be experienced software engineers or inexperienced technicians, when assessing the test adequacy. A generic coverage assessment, i.e. "100% of behavior has been tested", is seen as questionable because a resource for completely testing a system is not feasible and specific numbers may have little meaning. Therefore, rather than assessing how complete the system behavior was tested, the requirement was set to finding untested behavior and assessing its need for specifying tests.

3. Related work in the field of test coverage assessment

Coverage metrics in the field of computer science have been an active research topic for many years. They can be used for test case generation (Anand et al., 2013), change impact analysis (Bohner & Arnold, 1996; De Lucia, Fasano, & Oliveto, 2008), regression test selection and prioritization (Engström & Runeson, 2010; Yoo & Harman, 2012) or for assessing test suite adequacy (M. C. K. Yang & Chao, 1995; Zhu et al., 1997). While some approaches have already been incorporated into the production automation domain, coverage metrics have rarely been used for assessing test suite adequacy in this field. In the following, a closer look into work related to the presented approach will be taken.

3.1. Requirement based test coverage

These coverage metrics are based on the relation of requirements and test cases in which test cases check whether the system under test fulfills a set of requirements. In reverse, if an approach uses functional requirements or specifications for test generation, it is assumed that the generated test case is adequate for these requirements.

A basic realization of this approach is commercially available in multiple requirements management tools, such as IBM Rational DOORS

دریافت فوری ← متن کامل مقاله

ISIArticles
مرجع مقالات تخصصی ایران

- ✓ امکان دانلود نسخه تمام متن مقالات انگلیسی
- ✓ امکان دانلود نسخه ترجمه شده مقالات
- ✓ پذیرش سفارش ترجمه تخصصی
- ✓ امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
- ✓ امکان دانلود رایگان ۲ صفحه اول هر مقاله
- ✓ امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
- ✓ دانلود فوری مقاله پس از پرداخت آنلاین
- ✓ پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات