Challenges for Functional Testing of reconfigurable Production Systems

Andreas Zeller
Institute of Industrial Automation
and Software Engineering
University of Stuttgart
Email: andreas.zeller@ias.uni-stuttgart.de

Michael Weyrich
Institute of Industrial Automation
and Software Engineering
University of Stuttgart
Email: michael.weyrich@ias.uni-stuttgart.de

***Abstract*—The increasing complexity of software in production systems leads to a growing relevance of testing. This trend will continue due to the flexibilization, and IT networking of production systems. This paper proposes a discussion about the requirements of future production systems which will be reconfigurable, decentralized controlled and will consist of autonomous components. From these predicted requirements, new challenges for functional testing are derived. To elucidate these challenges, three use cases illustrate when testing measures will be necessary within the operation phase of production systems.**

I. Introduction

Due to high demands on reliability and availability, functional testing plays an important role in the development phase and commissioning phase of production machines. The necessary testing effort has significantly increased within the last decades because of the growing amount of software within production and automation systems. This process will continue in the next years. According to a study, the importance of IT and automation in the field of mechanical engineering will increase around 23 % between 2015 and 2018 in Germany [1]. For this reason, there is still one principle for plant operators - "never change a running system".

This is in strong contrast to the growing customer demands for individually designed products and the resulting vision of a smart factory. This vision demands a highly flexible production, where frequent changes of the production systems are not only unavoidable but also desired.

These changeable production systems impose novel challenges on the verification process. In this paper the effects on functional testing are analyzed and concrete use cases are derived. The paper focuses on the verification if the functionality performs correctly according to given design specification within the operation phase. Aspects of non-functional testing like performance testing, stress testing and security testing as well as the testing of an asynchronous communication are also relevant but not considered.

II. State of the Art

Nowadays, functional testing is an essential part of the engineering process of a production system. Typically this engineering process follows the V-Model, whereby the unit-test, integration-test, system-test and acceptance-test are defined. The testing process starts with unit-testing. That is mostly done with white-box-testing techniques by evaluating the code [2]. Integration-tests validate the correct interaction between different units. The system-test verifies the whole system and is normally performed with black-box-testing techniques and should be done by engineers that weren't involved in the engineering process. According to a worldwide survey, the system-test is by far the most budget consuming test level [3]. Regression tests have to be performed, whenever the systems have changed to ensure that the changes didn't affect a given requirement. This can even concern unchanged parts of the system [4]. For these test cases, which have to be performed frequently, test automation is used to reduce testing time and increase product quality by a better test coverage [5]. The acceptance test is the final step before the system goes into operation and is often done with real process data. Thereby the interaction between the product, the process and the resources (PPR) is validated. Changes of the product requirements, process parameters and resource configurations can influence each other and have to be ensured together.

Once in operation, the production systems in mass production are seldom modified to reduce the risk of a fall out. The focus is on maintenance and quality management, where the functional testing plays a subordinated role. If reconstruction measures are necessary, these have to the validated by tests before the system becomes operational again. Caused by the proportion of software in modern production, these reconstruction measures are being increasingly replaced by software reconfigurations. Due to demand for a higher product diversity, the period of time between reconstruction- / reconfiguration-measures is decreasing. This increases the importance of testing in the operation phase of production systems.

III. How a flexible Production affects Testing

According to a survey [6] where 16 testing experts have been interviewed the importance of testing in the operation- and maintenance-phase will grow significantly within the next ten years. In this chapter the new challenges for testing within the operation phase are derived by the changing of production systems and their components.

A. Requirements for IT-Systems of a reconfigurable Production System

To preserve competitiveness in manufacturing industry, many production facilities have to be made more flexible. The flexibility of the IT-system which coordinates the production is a key element for a flexible production facility. The IT-System has to provide the opportunity for ad hoc networking, interoperability, software updates on the fly and fulfill different performance requirements like latency, data rate and a robust operation. For this reason the requirement for these IT-systems are being addressed in the following.

Compared to centralized IT-Infrastructures, decentralized IT-Infrastructures have advantages regarding flexibility [7]. These distributed systems don't differentiate from a centralized system to the user, but haven't a common storage which firstly leads to a higher complexity in the communication architecture [8] and secondly causes a distribution of system knowledge on the autonomous components of the system. There are different approaches to realize distributed systems with Software-Agents [9] [10] or based on a service-oriented Architecture like OPC-UA [11]. The norm DIN SPEC 91345 describes the communication of a "Industry-4.0-component" SOA based [12]. Both approaches can support following characteristics [13]:

- **Encapsulation:** Separation between the implementation of the functionality and the call mechanism. It enables implementation independence and protects the core functionality from disturbances in the network [12].
- **Semantic described Interfaces:** The components within a network communicate through a common semantic [12]. This standardized semantic allows interoperability between components of different manufacturers.
- **Visibility:** To realize ad hoc networking the components have to be visible for the other participants of the network. This is often realized by a server where new components register (i.e. service registry [14] or directory facilitator [15]).
- **Scalability:** The decentralized architecture allows to add and remove components. There is no central node which has to handle all the coordination in the network. The semantic described interfaces and the visibility simplify the scaling in size.
- **Loose Coupling:** The components aren't connected in a static way, but couple on demand. This coupling on demand allows the reuse of a service by different clients.
- **Orchestration:** Process services orchestrate basis services to higher-value services. This enables the customer to order higher-value services without the necessity to know the individual process steps.

These characteristics pave the way for a flexible coordination of the production process. Thereby an easy reconfiguration is enabled by encapsulating the functionality and a common semantic described interface. Thus [16] describes a distributed, service-oriented architecture, a communication infrastructure and a semantic bases as the preconditions to realize an Industry 4.0 use case. This helps to handle the growing complexity. Nevertheless, the reconfigurations lead to new challenges for the testing process.

B. Derived Challenges for Functional Testing

Often occurring reconfigurations and changing production tasks lead to an increasing volatility of production environments. Due to the fact that changes have to be validated, this constitutes new challenges for the test process to validate the correct functionality of the production system within the operation phase. The high range of functions with a high degree of freedom and the high autonomy of the components results in a huge amount of test cases which are necessary to validate the full functionality of the component. When updating or reconfiguring the component, it can be very time consuming to select and run necessary test cases. Not just the testing of a single component is getting more complex but also the validation of the cooperation between different components. Due to the encapsulation, the functionality of a component is protected by changes from the outside, but the interaction with other components which were modified could be disturbed. Knowing the functional dependencies is necessary to estimate the effects of a reconfiguration on other parts of the product system. The distributed design of IT-systems causes a distribution of the knowledge about the system model to the autonomous components. Furthermore the dependencies of the system model change by reconfigurations and ad hoc networking and different production tasks frequently. [13] describes a method to collect this distributed knowledge about the dependencies within the production system on run-time.

On the one hand a semantic described interface enables ad hoc networking and interoperability, on the other hand, due to the gained flexibility in communication, it impedes the validation process of integration testing because the amount of possible cooperation partners increases.

As described in chapter II the product, process and resource have to be tested together within a system boundary. Thus a change of the input products can lead to the necessity for testing even if the process parameters and the resource configuration didn't change. The challenges for functional testing of reconfigurable production systems are summarized:

- changing production environment
  - reconfigurations
  - software updates
  - ad hoc networking
- different production tasks
- decentralized system
  - high degree of autonomy of individual components
  - fractal system knowledge

To demonstrate the new challenges for testing within the operation phase, the characteristics of a flexible production described above are transferred in concrete use cases. This is done by means of a small and simplified reconfigurable production system as illustrated in fig. 1. It illustrates that even a down-to-earth scenario causes different challenges to validate changes.

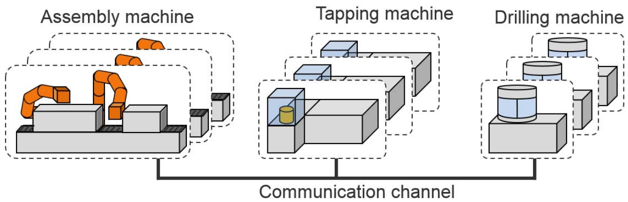

Fig. 1. Reconfigurable production system described in the use case

IV. Testing Use Cases

The way changes affect neighboring components doesn't differ from conventional production systems. The effects are rather much more extensive and less transparent because of the higher amount of reconfigurations and the high flexibility of the system.

The reconfigurable production system of this use case consists of three different component types: a drilling machine, a tapping machine as well a an assembly machine. The components are connected via a service-oriented architecture, were every component offers a service like drilling, tapping and assembling. Due to the loose coupling between the components, mostly it isn't possible to locate the dependencies on the basis of the physical build up of the production system. Assuming that there is only one component per type and one component is just offering one service, there are already theoretically 15 possible constellations how a workpiece can be guided through the production system. In many cases some constellations can be excluded for example tapping before drilling, but when scaling the scenario it's not obvious anymore which process steps can follow in sequence.

As one can't validate all possible constellation it's reasonable just to validate the functionality which is demanded by the next production order. To that it's mandatory to know which resources will be used to perform the process steps. If there's only one component per type this is quite easy, but when including redundancy, there are several ways to process an order through the production system.

The process of the use case is illustrated in fig. 2 with help of the formalized process description (VDI/VDE 3682). The input product **P1** is a metal block (see fig. 3). A hole is drilled into the metal block by the drilling machine illustrated in fig. 1. In the next process step a tap is cut in the drilled hole. In the last process step a screw (**P4**) is screwed in the processed metal block (**P3**). The following use cases describe the effects of changes in the process, in the software, and caused by scaling and redundancy.

A. Use Case 1: Changing Product Type

This use case regards the effects of small changes of product requirements on the production process and the resources.

Assuming the production system is currently producing the product described above. Due to changing customer wishes the requirements for the manufactured product changes a bit. As input product **P1** a plastic block shall be used instead of a

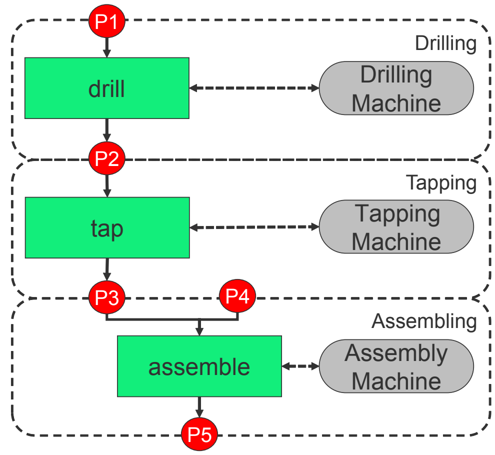

Fig. 2. Product-Process-Ressource of the use case where a hole is drilled, a thread tapped and a screw tightened

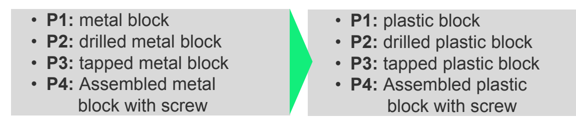

Fig. 3. Changing input-, intermediate- and end-products caused by changing customer requirements

metal block. As shown in fig. 3 this has effects on the other intermediate products **P2**, **P3** as well as on the end product **P5**. The changes of the product requirement changes the process, the products as well as the resource in the system boundary "drilling". Due to the changed material of **P1**, the process parameter of drilling have to be adapted to drill into plastic. This leads to a reconfiguration of the control unit as well as the hardware of the resource. Due to the reconfigurations of the drilling machine a unit- / component-test has to be performed to validate its correct functionality.

Within the system boundaries "tapping" and "assembling", the processes as well as the resources remain unchanged but the input-products **P2**, **P3** and output-products **P3**, **P4** are different. Due to the changed input-products, the process step should be validated, although not reconfiguration measures were necessary within the system boundaries. When scaling this scenario a small change of the product requirement can cause high validation efforts to ensure the correct functionality of the system. The more such changes come up, the more relevant it gets to estimate the effects of changes.

B. Use Case 2: Software-Updates on the fly

This use case regards the changes of a resource on neighboring components.

The encapsulation of the functionality of components facilitates the updating of software because it can be regarded as isolated, just connected to the network by a semantic defined interface. The resource affected by the software updates has to be validated. Due to a higher functionality the validation effort of this unit will increase but it's rather the validation of the interaction with other components that rises to a challenge.

Updating the tapping machine can affect the process tap e.g. by influencing the characteristic torque curve of the thread tap. This may affect the output-product **P3** which in turn affects the system boundary "assembling" what can lead to the necessity to validate further process steps that are dependent.

This only regards the validation of the system for one specific product type order, if several different product types are produced. The dependencies between the different process steps and different components are getting increasingly complex.

C. Use Case 3: Resource Redundancy

The use cases didn't regard resource redundancy yet. Assuming an additional drilling-, an additional assembling- as well as an additional tapping-machine is included in the network. Due to the ad hoc capability of the devices this can easily be done when there is a demand regardless if the components are the same type or even by the same manufacturer.

The production system still shall produce the product type described in fig. 2. Instead of one possible production path, there are 8 different constellations. If a validation of every path is necessary, because it's a critical process and a full flexibility of the system is required, the validation effort can take up to a factor of 8. When scaling this scenario with 5 process steps and 4 redundant components per process steps, there are 1024 possible paths.

V. CONCLUSION

Small changes can have big effects on the neighboring components. The more complex and networked a system gets, the harder it is to estimate which components are affected by the changes and should be validated. The paper derived the new challenges for functional testing from the requirements of future reconfigurable production systems. The new challenges are elucidated by three use cases. This concerns the validation of:

- software- / hardware-reconfigurations of a component
- often changing production tasks
- ad hoc networking causes changing system dependencies
- several production paths, made possible by resource redundancy

It is shown that not only reconfigurations of the components result in the necessity of validating, but also changes of the product or the production process. This is due to the fact, that the correct interaction between the process, the product and the resource has to be validated. In that regard the process steps can't be regarded as isolated because they are connected by the input-/ output-products. The more flexible process steps can be combined, the more complex these dependencies get. Due to the distribution of the system knowledge on the participants of the network, these dependencies aren't available at a central spot. This hampers the estimation of changes on other parts of the production system.

REFERENCES

[1] VDMA, *IT and automation in mechanical engineering - VDMA trend study 2015 (orig:IT und Automation im Maschinenbau - VDMA-Trendstudie 2015) Zusammenfassung*, 2015.

[2] F. Witte, *Testmanagement and Softwaretest (orig: Testmanagement und Softwaretest)*, Springer Fachmedien Wiesbaden, ISBN: 978-3-658-09964-0, 2016.

[3] International Software Testing Qualifications Board (ISTQB), *ISTQB Worldwide Software Testing Practices Report 2015-2016*, link called on 12. May 2016: http://www.istqb.org/references/surveys/istqb-worldwide-software-testing-practices-report-2015-2016.html

[4] International Software Testing Qualifications Board (ISTQB), *Standard Glossary of Terms used in Software Testing*, Version 3.1, link called on: 12. May 2016: http://www.istqb.org/downloads/category/20-istqb-glossary.html.

[5] D. Rafi and K Moses, *Benefits and Limitations of Automated Software Testing: Systematic Literature Review and Practitioner Survey*, The 7th IEEE/ACM International Workshop on Automation of Software Test (AST), Switzerland, 2012.

[6] A. Zeller and M. Weyrich, *A networked and flexible Industry 4.0 Production needs new test methodologies (orig:Industrie 4.0 mit vernetzter und flexibler Produktion erfordert neue Testmethodiken)*, atp edition - Automatisierungstechnische Praxis,10/2015 p.22-25.

[7] A. Tanenbaum and M. Steen, *Distributed Systems - Principles and Systems*, 2nd ed., Pearson Prentice Hall, New Jersey, ISBN: 0-13-239227-5, 2007.

[8] J. Schmidt and A. Zeller, *Approach for model driven development of flexible automation systems by service orientation (orig: Ansatz zur modellgetriebenen Entwicklung flexibler Automatisierungssysteme durch Serviceorientierung)*, 14. Fachtagung EKA Entwurf komplexer Automatisierungssysteme, Beschreibungsmittel, Methoden, Werkzeuge und Anwendungen, 24.-25. May 2016.

[9] S. Ulewicz and D. Schuetz and B. Vogel-Heuser, *Integration of Distributed Hybrid Multi-Agent*, Systems into an Industrial IT Environment, 12th IEEE International Conference on Industrial Informatics (INDIN), Porto Alegre, Brasil, 27.-30. July 2014.

[10] P. Goehner, *Software agents for a flexible coupling of automation systems (orig: Softwareagenten fuer die flexible Kopplung von Automatisierungssystemen)*, 6. VDI-Expertenforum "Agenten im Umfeld Industrie 4.0", Muenchen, Germany, 2014.

[11] F. Palm and S. Gruener et al., *Open Source as Enabler for OPC UA in Industrial Automation*, 2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA), Louxemburg, 08.-11. September 2015.

[12] Deutsches Institut fuer Normung e. V., *DIN SPEC 91345 - Reference Architecture Model Industrie 4.0 (RAMI4.0)*, April 2016.

[13] A. Zeller and M. Weyrich, *Validating reconfigurations of production systems within the operating phase*, Kongress Automation 2016, 07.-08. June 2016, Baden-Baden, Germany.

[14] W. Mahnke, S.-H. Leitner, M. Damm, *OPC-Unified Architecture*, Springer-Verlag Berlin Heidelberg, ISBN: 978-3-540-68898-3, 2009.

[15] *FIPA Agent Management Specification. FIPA (Foundation for Intelligent Physical Agents)*, 2004, link called on 11. May 2016: http://fipa.org/specs/fipa00023/SC00023K.html.

[16] A. Fay and C. Diedrich et al., *Semantic definitions for Industry 4.0 - Standards as a semantic basis*, atp edition - Automatisierungstechnische Praxis, 7-8/2015 p.30-41.