

Title of Paper

Testing Service-oriented Architectures

Presenter

Dr. Johannes Maria Zaha, Heiko Stallbaum, / University of Duisburg-Essen (D)

Instructional Level

Introductory Intermediate Advanced

Target Group

Test practitioners and engineers, software and test managers, quality assurance managers and development managers as well as other professionals that are facing the challenge of testing service-oriented architectures (or will face this challenge in the future).

Keywords

- Service-Oriented Architecture (SOA)
 - Software Product Line (SPL)
 - SOA Testing
-

Abstract

The service-oriented architecture (SOA) is an architectural paradigm that has moved from theory to practice. The paradigm is the basis of numerous implementations of enterprise application systems and is increasingly adopted by industry. According to IDC's latest survey on service-oriented architectures [IDC 2007], 64% of the interviewed software developers believe that an SOA is already feasible to be applied for several application scenarios and 91% of the interviewed believe that it can be applied to those scenarios within the next 24 months. However, testing applications that adhere to this architectural paradigm remains challenging. The reasons for that are the key principles of service-orientation, which are autonomy, loose coupling, abstraction, and the need for a formal contract (e.g., see [Erl 2005]):

- Autonomy means, that the logic that is governed by a service resides within an explicit boundary. The service has control within this boundary and is not dependent on other services for it to execute its governance.
 - Services are loosely coupled, i.e., they must be designed to interact without the need for tight, cross-service dependencies.
 - Services abstract from underlying logic, meaning that the only part of a service that is visible to the outside world is what is exposed via the service interface. Underlying logic beyond what is expressed in the descriptions of the service interface are invisible and thus should be irrelevant to service requesters.
 - Services share a formal contract. This implies that in order to interact, they only have to share a collection of published meta-data that describes each service and defines the terms of information exchange.
-

Due to these principles of service-orientation, SOA testing faces at least the following two significant challenges:

- How to handle integration complexity? Loose coupling of services enables their composition to a potentially unbound number of service-based applications. In order to assure that a single service will work without a failure in all those compositions, an increased level of testing, including integration testing, should be performed. Due to the unbound number of service-based applications, a service cannot be tested in all possible compositions. Coping with this complexity problem is a considerable challenge in SOA testing.
- How to deal with external services? In contrast to a system which is built from software components, the functionality of a service-based application is achieved by coordinated service invocations. More and more those services are provided by “external” service providers (e.g., Google, Salesforce, ...). Here, a service user is charged for invoking the services of a provider. Thus, a challenge for testing a service-based system is how to reduce the number of costly service invocations that are needed for testing a service-based system.

First publications report on similarities between software product line (SPL) engineering and engineering of service-oriented architectures (SOA) [Helferich 2006]. Both paradigms focus on promoting high levels of reuse to build flexible and cost-effective software systems. Like in the development of an SOA, testing in SPL engineering faces the complexity challenge (cf. [Pohl 2006]).

The focus of this presentation will be on the comparison of SPL engineering and SOA engineering with respect to testing issues. We will show, up to what extent existing SPL testing techniques can be adopted for testing an SOA and how they address the specific challenges of SOA testing.

References

[Erl 2005] T. Erl: Service-Oriented Architecture – Concepts Technology, and Design. Prentice Hall, 2005.

[Helferich 2006] Helferich, A., Jesse, S., Mikusz, M.: Software product lines, service-oriented architecture and frameworks: Worlds apart or ideal partners? In: Draheim, D., Weber, G., eds.: 2nd International Conference on Trends in Enterprise Application Architecture, 29 November - 1 December 2006, Berlin, pp. 143-157.

[IDC 2007] IDC (International Data Corporation): Service Oriented Architecture: The Developer's Perspective. IDC Document Number 207205, June 2007.

[Pohl 2006] K. Pohl, A. Metzger: Software product line testing. Communications of the ACM 49(12), 2006, pp. 78-81.

Biography

Dr. Johannes Maria Zaha currently works as a senior research and teaching assistant at the working group “Software Systems Engineering” at the University of Duisburg-Essen, Germany. Prior to this position, he was a post-doctoral research fellow at the Faculty of Information Technology at the Queensland University of Technology in Brisbane, Australia. His main research areas are component-based software engineering and service-oriented architectures.

Heiko Stallbaum is a research assistant and PhD student at the working group “Software Systems Engineering” at the University of Duisburg-Essen. His current research interests include quality assurance and model-driven software development. Before that, he spent several years in industry, where he participated in development projects in the area of internet technology. He received his diploma in computer science from RWTH Aachen, specializing in software construction.

Dr. Andreas Metzger is a senior research assistant at the “Software Systems Engineering” group at the University of Duisburg-Essen. He received his PhD in 2004 from the University of Kaiserslautern on the topic of model-based quality assurance. His current research interests are software product line engineering, model-driven software development, and quality assurance, which includes requirements-based testing as well as formal reasoning on product line models.

Contact information of Presenter

Dr. Johannes Maria Zaha / Heiko Stallbaum

Software Systems Engineering
Institute for Computer Science and Business Information Systems
University of Duisburg-Essen
Schützenbahn 70
45117 Essen
Germany

Phone: +49 (201) 183 - 4655 / 4657 / 4650

Fax: +49 (201) 183 - 4699

E-Mail: {heiko.stallbaum|johannes.zaha|andreas.metzger}@sse.uni.due.de

WWW: www.sse.uni-due.de
