

**09021 Abstracts Collection**  
**Software Service Engineering**  
— Dagstuhl Seminar —

Frank Leymann<sup>1</sup>, Tony Shan<sup>2</sup>, Olaf Zimmermann<sup>3</sup> and Willem-Jan van den Heuvel<sup>4</sup>

<sup>1</sup> Universität Stuttgart, D

<sup>2</sup> University of Phoenix, USA

<sup>3</sup> IBM Research - Zürich, CH  
olz@zurich.ibm.com

<sup>4</sup> Tilburg University, NL  
wjheuvel@uvt.nl

**Abstract.** From 04.01. to 07.01.2009, the Dagstuhl Seminar 09021 “Software Service Engineering” was held in Schloss Dagstuhl – Leibniz Center for Informatics. During the seminar, several participants presented their current research, and ongoing work and open problems were discussed. Abstracts of the presentations given during the seminar as well as abstracts of seminar results and ideas are put together in this paper. The first section describes the seminar topics and goals in general. Links to extended abstracts or full papers are provided, if available.

**Keywords.** Service engineering, software engineering, service-oriented computing, service-oriented analysis and design, SOA, systems engineering, Web engineering

## 09021 Executive Summary – Software Service Engineering

Service-Oriented Architecture (SOA) constitutes an important, standards-based and technology-independent distributed computing paradigm and architectural style for discovering, binding, assembling, and publishing loosely-coupled and network-available software services. With SOA-enabled applications operating in highly complex, distributed, and heterogeneous execution environments, SOA practitioners encounter the limits of traditional software engineering. In this Dagstuhl seminar, we have discussed and explored the fundamental tenets underpinning the development and maintenance of SOA systems. As a result of our discussions, we position software service engineering as an evolving and converging discipline that embraces the open world assumption. Software service engineering entails a departure from traditional software engineering disciplines such as component-based development, supplementing them with techniques and patterns tailored to service enablement, composition, and management.

2 Frank Leymann, Tony Shan, Olaf Zimmermann and Willem-Jan van den Heuvel

*Keywords:* Service engineering, software engineering, service-oriented computing, service-oriented analysis and design, SOA, systems engineering, Web engineering

*Joint work of:* van den Heuvel, Willem-Jan; Zimmermann, Olaf; Leymann, Frank; Shan, Tony

*Full Paper:* <http://drops.dagstuhl.de/opus/volltexte/2009/2041>

## **Next Generation SOA and Service Engineering**

*Thomas Erl (SOA Systems Inc. - Vancouver, CA)*

A brief overview of how established practices, principles, and patterns have formed the basis for next generation SOA and service-oriented computing and how these practices, principles, and patterns pertain to and are being formalized via service engineering.

*Keywords:* Soa, service engineering, service-orientation, principles, patterns

## **Understanding SOA Design Patterns and Principles**

*Thomas Erl (SOA Systems Inc. - Vancouver, CA)*

This session begins with a review of service-oriented computing fundamentals, including the service-orientation design paradigm and associated principles. After establishing the method of service-orientation and the target state as represented by a set of strategic goals associated with service-oriented computing, a tour of the soa design patterns catalog is provided during which each of the 85 patterns is briefly described.

*Keywords:* Soa, service engineering, service-orientation, principles, patterns

## **Integrating Business and Technology Views With a Model-based Product Line Approach**

*Paul Grünbacher (University of Linz, AT)*

A product line approach needs to provide support for different roles such as product and project managers, business domain experts, or software architects. Using three examples from the domains of component-based development, plugin architectures, and services we show that linking business and technology views is success-critical.

We have developed the tool-supported approach DOPLER for modeling product lines that provides flexibility to be used in different environments such as

component-based development, IEC 61499 function blocks, or plugin architectures. More recently, we started using DOPLER to support the definition and adaptation of service-oriented software systems and to provide different views for business people and engineers.

The service-oriented computing paradigm offers capabilities for designing flexible and evolvable systems. Despite many benefits adapting a service-oriented system to different environments and evolving requirements remains challenging. Several research groups have thus proposed the use of product line variability models to support the evolution of service-oriented systems. Together with UPC Barcelona we are currently developing extensions to the DOPLER tools that allow to define product line models of service-oriented system, to use the models to describe possible adaptations, to support service monitoring based on metrics, and to perform system adaptation based on monitoring information. Our approach supports different views for engineering and business domain experts.

*Keywords:* Service orientation, software product line

## Software Service Engineering - Architect's Dream or Developer's Nightmare?

*Gregor Hohpe (Google Inc. - Mountain View, US)*

Architectural principles such as loose coupling are the key drivers behind the adoption of service-oriented architectures. Service-oriented architectures promote concepts such as composition, process modeling, protocol design, declarative programming, event-based programming, and object-document mapping. These architectural ideals can be fraught with challenges for developers who are faced with unfamiliar programming models and immature tools. This paper briefly reviews the service-oriented architecture concepts and highlights the most pressing challenges for developers. These challenges suggest several focus areas for tool builders and software service engineering researchers.

*Keywords:* Event-based programming, declarative programming, object-document mapping, patterns, process modeling, protocol design, service composition, software engineering, SOA

*Full Paper:* <http://drops.dagstuhl.de/opus/volltexte/2009/2042>

## Interoperability and other desired emerging software properties

*Manfred Jeusfeld (Tilburg University, NL)*

The current SOA software stack is an answer to the lack of information about remote service providers and lack of control over them.

The future of SOA is in our view dependent on the ability to limit the complexity, either by simplifying the languages and protocols, e.g. by falling back to simple HTTP instead SOAP. Or one has to be able to recognize patterns from which the WS-\* code can be largely generated.

We have developed a classification framework for patterns from the Ishikawa diagrams, which relate contextual properties (observed variables) with emerging properties, such as the presence of an interoperability problem. We propose that patterns should be classified by such a structured scheme rather than by simple keywords. The classification framework allows to formulate causal queries like "Which patterns are applicable when negotiating (=process) a business model (=product) between multiple business partners (=producers).

More details are in the paper:

*Keywords:* Interoperability, classification, meta modeling, situational method engineering

*Full Paper:*

[http://www.sciencedirect.com/science?\\_ob=ArticleURL&\\_udi=B6V0G-4RVG3S6-1&\\_user=522558&\\_rdoc=1&\\_fmt=&\\_orig=search&\\_sort=d&view=c&\\_acct=C000026138&\\_version=1&\\_urlVersion=0&\\_userid=522558&md5=a51604c2351416187257860bb5d03f49](http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B6V0G-4RVG3S6-1&_user=522558&_rdoc=1&_fmt=&_orig=search&_sort=d&view=c&_acct=C000026138&_version=1&_urlVersion=0&_userid=522558&md5=a51604c2351416187257860bb5d03f49)

*See also:* Jolita Ralyté, Manfred A. Jeusfeld, Per Backlund, Harald Kühn, and Nicolas Arni-Bloch (2008): A Knowledge-based Approach to Manage Information Systems Interoperability. *Information Systems*, 33(7-8), pp. 754-784. Elsevier.

## SOA - Context and Lessons Learned

*Nicolai Josuttis (IT Communication - Braunschweig, DE)*

The 21st century is the century of globalization. This also applies to IT. More and more, system development is only a task in the broader approach of system landscape maintenance.

Here, SOA comes into play. To realize business processes over distributed systems you need concepts that deal with globalization. You might have heard about loose coupling, high interoperability, and service-orientation as key elements of SOA. But the real power SOA comes from dealing with the fact that systems landscapes are heterogeneous and have different owners.

Based on his experience of bringing SOA in operation at major world-wide companies Nicolai Josuttis explains what SOA really is and why its concepts are so important for the next decades.

*Keywords:* SOA, globalization, heterogeneity, distribution

*Full Paper:*

<http://www.soa-in-practice.com/>

*See also:* Nicolai Josuttis: SOA in Practice, O'Reilly, 2007

## **Enterprise Mashups**

*Anna-Lena Lamprecht (TU Dortmund, DE)*

A location-based service Mashup using Google, Amazon & Co. is realized with the jABC.

*Keywords:* Service Composition, Mashups, WSDL, REST

*Joint work of:* Margaria, Tiziana; Kubczak, Christian; Steffen, Bernhard

## **Modeling, Design, and Analysis for Service-oriented Architecture Workshop**

*Einar Landre (StatoilHydro ASA - Stavanger, NO)*

Reference to workshop relevant for those interested in Service Engineering. Read more here: <http://events.deri.at/mda4soa2008/index.html#6>

*Keywords:* Modeling, Service Oriented, Architecture

*Full Paper:*

<http://events.deri.at/mda4soa2008/index.html#6>

## **Service design in time and safety critical domains**

*Einar Landre (StatoilHydro ASA - Stavanger, NO)*

The slides attached to the seminar homepage illustrate key issues adopting a service oriented approach in time constrained and safety critical domains. The slides were not presented, but served as background for discussion.

*Keywords:* Time constrained, safety critical

## **Cases of Software Services Design in Practice**

*Susanne Patig (Universität Bern, CH)*

During the last years, several approaches for the design of software services in service-oriented architectures (SOA) have been proposed. Often these approaches are too rough or too academic to provide guidance for real world SOA projects. Moreover, since the existing SOA design approaches are often not sufficiently validated, their successfulness in practice can be doubted. The research presented here aims at learning from successful SOA projects. Two cases of such projects are described. In the cases similarities show up that are distinct from existing SOA design approaches (mainly the purely academic ones) and, thus, point to necessary enhancements of these approaches.

*Keywords:* SOA Design Approaches, Service Design, Case Study

*Full Paper:* <http://drops.dagstuhl.de/opus/volltexte/2009/2047>

## Composing RESTful Services

*Cesare Pautasso (Universität Lugano, CH)*

Recent technology trends in Web Services (WS) indicate that a solution eliminating the perceived complexity of the WS-\* stack may be in sight: advocates of REpresentational State Transfer (REST) have come to believe that their ideas explaining why the World Wide Web works are just as applicable to solve enterprise application integration problems and to radically simplify the plumbing required to build service-oriented architectures. In this talk we focus on the problem of composing both RESTful Web services with WS-\* Web services. We discuss how the assumptions made by current standards for Web service composition are challenged by the new paradigm. We demonstrate how Web service composition languages and tools can be evolved to cope with REST using the JOpera for Eclipse process modeling environment as an example.

*Keywords:* Service Composition, RESTful Service API, Mashup, BPEL for REST

*Full Paper:*

<http://www.jopera.org/docs/publications/2008/bpel4rest>

*See also:* C. Pautasso, BPEL for REST, Proc. of the 7th International Conference on Business Process Management (BPM 2008), Milano, Italy, September 2008

## On RESTful Service Composition

*Cesare Pautasso (Universität Lugano, CH)*

Composition is one of the central tenets of service oriented computing. This paper discusses how composition can be applied to RESTful services in order to foster their reuse. Given the specific constraints of the REST architectural style, a number of challenges for current service composition languages and technologies are identified to point out future research directions.

*Keywords:* REST, Web Service Composition, RESTful Service Composition

*Full Paper:* <http://drops.dagstuhl.de/opus/volltexte/2009/2043>

## Service Modelling and Testing - Selected Considerations and Thoughts

*Ina Schieferdecker (TU Berlin, DE)*

This talk discusses model-driven engineering of services. It considers principles of services in telecommunication, business IT, Internet, and software development. The specific nature of services such as the uncertainty of their usage in an open world, their dynamics along loose coupling and late binding principles, or their service requirements put forward in contracts requires a dedicated engineering approach for services and specific quality assurance methods. First approaches in service modelling use local and global views on services and/or consider their collaborations. As services are always-on, highly dynamic, contract-based and service-quality oriented, new methods for service testing are needed as well. These include online tests, build-in tests and run-time auditors and supervisors.

Software service engineering has to address semantic service specifications, which are not just structural interfaces; should provide 'clean' and 'handy' service modeling methods provided by both general-purpose and domain-specific languages; and should offer sound quality assurance methods for service-based systems, in particular testing. Overall, model-driven engineering (MDE) and software service engineering (SSE) can benefit from each other and should be combined.

*Keywords:* Services in Telecommunication, Service Modelling, Service Testing

## Designing Software Services for Business Agility

*Harald Wesenberg (StatoilHydro - Trondheim, NO)*

This presentation focuses on identifying the right services for a domain. The work is based on 10+ years of experience from StatoilHydro identifying and building services that are stable and useful for many years, even across technology shifts.

By focusing on workflows with their activities, events and information usage, a set of reusable services can be identified through little effort (especially if the workflows already exists) and built/composed from existing services. This approach is especially suitable for agile development practices, as it significantly reduces the amount of up-front analysis compared to traditional service design practices.

*Keywords:* Enterprise Architecture, SOA, Domain Driven Design, Business Architecture, Software Service

*Full Paper:* <http://drops.dagstuhl.de/opus/volltexte/2009/2044>

## Pattern-based Modeling of Process-Driven SOAs

*Uwe Zdun (TU Wien, AT)*

Service-oriented architectures are increasingly used in the context of business processes. However, the proven practices for process-oriented integration of services are not well documented yet. In addition, modeling approaches for the integration of processes and services are neither mature nor do they exactly reflect the proven practices. We propose a pattern language for process-oriented integration of services to describe the proven practices. We also propose a modeling concept based on pattern primitives for these patterns. A pattern primitive is a fundamental, precisely specified modeling element that represents a pattern. We present a catalog of pattern primitives that are precisely modeled using OCL constraints and map these primitives to the patterns in the pattern language of process-oriented integration of services.

*Keywords:* Patterns, SOA

## SOA decision modeling

*Olaf Zimmermann (IBM Zürich Research Lab. - Rüschlikon, CH)*

On Service-Oriented Architecture (SOA) construction projects, software architects are concerned with the characteristics of high quality services and how such services can be designed. For instance, they require advice regarding service interface design and guidance how to ensure that the constructed services meet the non-functional requirements that have been stated for them. To give such advice and guidance, service identification, specification and realization methods and techniques are required. Unlike service identification and specification, existing methods and techniques do not cover service realization well.

In our work, we investigate whether reusable architectural decision models can support service realization. Architectural decision models capture the knowledge (rationale) justifying certain designs. In the current state of the art, architectural decisions are captured ad hoc and retrospectively on each project, if at all; this is a labor-intensive undertaking without immediate benefits. On the contrary, we investigate the role reusable architectural decision models can play during SOA design: We treat recurring architectural decisions as first-class method elements and propose an architectural decision modeling framework and a reusable architectural decision model for SOA which guide the architect through the SOA design. Our approach is tool supported.

In the framework, we provide a technique to systematically identify recurring decisions. Our reusable architectural decision model for SOA conforms to a metamodel supporting reuse and collaboration. The model organization follows Model-Driven Architecture (MDA) principles and separates long lasting platform-independent decisions from rapidly changing platform-specific ones.

The alternatives in a conceptual model level reference SOA patterns. This simplifies the initial population and ongoing maintenance of the decision model. Decision dependency management allows knowledge engineers and software architects to check model consistency and prune irrelevant decisions. Moreover, a managed issue list guides through the decision making process. To update design artifacts according to decisions made, we inject decision outcome information into design model transformations. Finally, a Web-based collaboration system provides tool support for the framework steps and concepts.

SOAD project news and results are available at: <http://soadecisions.org/soad.htm>

*Keywords:* SOA design, architectural decisions

*Full Paper:*

<http://dx.doi.org/10.1016/j.jss.2009.01.039>

## **Engineering Document Services & Mashups for Situational Collaboration**

*Christian Zirpins (Universität Karlsruhe (TH), DE)*

In the MoSaiC project (Mashups for Situational Collaboration), we propose to investigate on a novel service-oriented approach for composite enterprise documents supporting ad-hoc collaboration and weakly structured innovative processes within and across enterprises. As technical foundation, we propose to integrate electronic documents with software services and processes underlying SOA. We are developing a Web-based architecture and prototype platform of a lightweight document service bus that shall represent document fragments as stateless software services and introduce a simple composition model to coordinate document content, editing and publishing services. Our document service model will allow us to conduct exciting research on an intuitive mashup language for document-centered situational collaboration. Collaborative mashups will regulate and enforce flexible and dynamic interaction patterns for document communication supporting conventional and virtual teams.

*Keywords:* Document-Driven Situational Collaboration, Collaborative Interaction Patterns, Document Service Architecture, Document Service Mashups

*Joint work of:* Zirpins, Christian; Tai, Stefan; Schuster, Nelly

## **Issues and Challenges of Leveraging Legacy Systems in SOA**

*Ying Zou (Queen's University - Kingston, CA)*

A business application automates a collection of business processes. A business process describes how a set of logically related tasks are executed, ordered and managed by following business rules to achieve business objectives.

An "online book purchase" business process contains several tasks such as buying a book, ordering a book, and sending out promotions. In this ever changing business environment, both of business applications and business processes are modified to accommodate changed business requirements and improve the performance of the organization. These continuous modifications introduce problems in the following two aspects: 1) Business process definitions are rarely updated to reflect the current business processes deployed in business applications. 2) Business processes may be cloned (e.g., copied and slightly modified) to handle special circumstances or promotions. Identifying these clones and removing them help improve the efficiency of an organization. However, business processes are defined with textual languages that cannot be automatically understood.

To maintain business process definitions up to date, we present our techniques that automatically recover business processes from business applications and identify clones in the recovered business processes. To refine the recovered business processes and mark the functional equivalent tasks, we use existing code clone detection tools, such as CCFinder and CloneDR, to detect clones in business applications, and lift clones from code level to business process level. We also propose a change impact metric which estimates the code to be modified as a result of a business process change. The effectiveness of our techniques is demonstrated through a case study on 15 large open source business applications.

*Keywords:* Business process, process recovery