BPM, OPEN SOURCE AND SOA -- MISSION IMPOSSIBLE?

Martin Schöpple, Philipp Brune and Heiko Gewald
University of Applied Sciences Neu-Ulm - Wileystraße 1, D-89231 Neu-Ulm, Germany

ABSTRACT

Business Process Management (BPM) promises to improve alignment between IT and business processes. In recent years, web services and service-oriented architectures (SOA) have been intensively discussed as paradigm for IS architectures designed to enable BPM. However, since SOA is not precisely defined, little is known about its actual real-world adoption and how it can be successfully implemented. Since many SOA concepts are based on open standards, open source software (OSS) seems a natural choice for SOA implementation. To foster the discussion whether a SOA may be purely based on OSS tools, this paper presents a proof of concept implementation of a realistic core business process. The feasibility of using OSS for SOA and of commonly accepted general SOA principles are analyzed and critically discussed. The results demonstrate that many SOA concepts lead to a more efficient IS architecture only if they are used carefully and well-tailored to the actual requirements.

KEYWORDS

Web Services, Service-Oriented Architecture, Business Process Management, Open Source Software, Business Process Implementation

1. INTRODUCTION

In recent years, service-oriented architectures (SOA) are amongst the most intensively discussed topics in business information systems engineering research (Hirschheim et al., 2010, Welke et al., 2011). Usually, SOA is understood as an enterprise architecture style designed to enable business process management (BPM). SOA are based on lose coupling of distributed, independent, encapsulated functional units referred to as services (denoting software components as well as organizational or business units), flexibly combined and orchestrated by a workflow engine (enterprise service bus, ESB) to form the business processes (Josuttis, 2008). The common vision is that service orchestration may be performed by graphical process modeling without the classical development tasks (i.e. writing program code) thus allowing for closer and more agile business-IT alignment (Varadan et al., 2008). In general, SOA are not a pure IT but also an organizational or managerial concept. Thus, considerable amount of research examines SOA and specifically its organizational implications (Joachim, 2011). Despite SOA are in principle not related to any actual implementation technologies for the services and the ESB, it is commonly assumed that services and service orchestration are based on popular Web-Service standards and technologies (like e.g. SOAP, REST, WSDL, WS-BPEL, BPMN) (Josuttis, 2008). The latter, more narrow under-standing of a SOA, frequently adopted by software vendors and application developers, is referred to as Web-Service-oriented Architecture (WSOA) in this paper.

While WSOA concepts have been successfully used for integrating heterogeneous enterprise applications and modularizing existing (legacy) applications for many years (Schmidtmann, 2005), it still remains an open question if and under which conditions WSOA are really feasible as an architectural style for building entire enterprise IT architectures, enabling businesses to successfully replace the traditional complex monolithic enterprise applications (like ERP and CRM systems) with a more flexible solution. Despite that it has been claimed that this kind of SOA is dead (Manes, 2009), a systematic study of the implications of actually constructing an enterprise-wide IT infrastructure by means of a WSOA is still lacking. In addition, for lowering the entrance barrier to WSOA adoption, the possibility to implement it using open standards and stable and efficient Open Source Software (OSS) tools and components is considered crucial, since both are well-established and commonly used in tradition-al software development.

This paper analyses whether it is possible to implement a pure WSOA using only Open Source Software (OSS). A commonly known real life business process (online ordering) was chosen to serve as proof of concept-implementation. The feasibility such a WSOA and its impacts on information systems development regarding development tools, maintainability, security and performance are analyzed and discussed and critical issues explicated. The paper closes with guidance for further research and the conclusion.

2. REVIEW OF THE LITERATURE

The information systems literature provides a vast variety of topics in SOA related articles. Academic research covers the whole spectrum from the management perspective (cost/benefit analysis etc.) to technical implementation details. The same holds true for Open Source Software, which received a great deal of attention in the last years. It seems like an obvious case to combine these two immensely rich re-search streams to identify the opportunities OSS offers for implementing SOA. Contrary to the authors' expectations, the publicly available body of literature is surprisingly small when it comes to merging these two closely related areas.

2.1 Business Value of SOA

SOA is a topic which has long been discussed as the new hot thing in information technology and systems development. Up to today the discussion is ongoing. Welke et al. (Welke et al., 2011) and Hirschheim et al. (Hirschheim et al., 2010) provide contemporary and comprehensive overviews on the business and IT related drivers for implementing SOA. They name "reusability" as the number one factor for adopting SOA. To assess the organizational readiness for implementing SOA and address the issue of differences in readiness perception between business and IT they provide a CMMI-like maturity model. In conclusion they support the overall benefits of SOA and support the case for further research in this area.

2.2 Business Process Management (BPM) and SOA

"Business Process Management and Service Oriented Architectures are considered a powerful combination for supporting enterprise’s success" (Adam and Doerr, 2008). Even though this claim stands widely undisputed in current information systems research, we hardly see any published material on how this combination is actually being set to work in practice. We see a lot of research supporting the different advantages SOA offers in combination with BPM and we also see authors proposing frameworks how to tackle the most common problems in SOA implementation.

Varadan et al. (Varadan et al., 2008) for example argue for the strong advantages of SOA to increase business process flexibility. Joachim et al. (Joachim, 2011) argue in the same vain and provide an empirically tested alternative research model for SOA governance.

The convergence of BPM and SOA is also supported by several authors e.g. Leyking et al. (Leyking et al., 2007) or Draheim (Draheim, 2010). Draheim provides an in-depth evolutionary overview of SOA and demonstrates how BPM and SOA con-verged in the course of latter years. He also points out the difficulties and limitations for actually implementing such a converged environment in real life situations.

2.3 Open Source SOA

The majority of publications address the opportunities to build an SOA with OSS either from a conceptual point of view or report case studies. Wieland (Wieland, 2007) gives an overview on OSS tools for SOA development and discusses different business models on OSS environments including the respective implications of differ-ent licensing agreements.

Viering et al. (Viering et al., 2009) conduct a literature review on SOA and identify 175 papers for further examination. Although they created a specific category on implementation patterns ("How are SOA and web services adopted in practice?") they found a number on case studies on SOA implementation (mainly from the financial services sector (see e.g. (Baskerville et al., 2010), pp. 30–53) but did not report findings on SOA with OSS.

One of the rare cases of developing a software that comprises real life business cases in a SOA concept using OSS is documented by Siebenhaar et al. (Siebenhaar et al., 2008). As this is the actual documentation of a study project he emphasis is more on comparing different approaches the respective teams used as to the question whether OSS tools are mature enough to replace commercial software for such developments.

Altogether, authors unanimously acknowledge the value of SOA and as such foster research on this topic. Various authors also underline the demand for OSS tools to be used to develop SOA/BPM (e.g. (Wieland, 2007), pp.1–5). However, an actual documented proof of concept, to analyze whether it is possible to build an SOA based exclusively on OSS tools is still outstanding.

2.4 Research Gap

In conclusion the vast majority of authors expect superior business value to be gained through service oriented architectures, especially when intelligently combined with a business process management layer. However there are still some issues unsolved mainly on organizational level and specifically on SOA governance.

Although we find rich streams of publication in different areas of SOA and BPM implementation we identify a huge gap when it comes to actual implementation of these concepts in practice. Apart from some cases predominantly in the financial services industry we hardly find documented experiences examining how actual SOA/BPM implementations look like and what insights can be drawn from these implementations. Furthermore another big trend in information systems -namely the use of open source software- has widely been neglected in this context.

As OSS increasingly finds its way into professional environments the question "Is it possible to implement a real life business process based on SOA principles with just using OSS tools?" becomes ever more important to researchers and practitioners alike. This paper provides an answer to this question by examining a proof of concept for a real life business process which is implemented using only OSS tools.

3. BASIC SOA AND WSOA CONCEPTS AND TOOLS

The main architectural concepts of SOA and WSOA are loose coupling, the use of an ESB and the use of (web) services. Loose coupling is a fundamental concept of software architecture and goes far beyond the different elements of a SOA. It aims at the minimization of dependencies and cohesion between different software components. The ESB realizes this principle consequently at the infrastructure level and enforces the independence as well as the interoperability of processes and services. It usually offers a service registry to manage existing services in the enterprise, security support functionality and a workflow engine to execute business processes by calling services. A service in this sense is as a software component with a standardized, open interface offering a well defined business functionality. It enables users as well as other applications and services to use this business functionality in a technology-neutral and platform-independent way, locally or via network. Services may be flexibly combined to form business processes, are reusable and support loose coupling of the different components (Josuttis, 2008, Melzer, 2008).

In context of a WSOA, services are implemented as web services. Their biggest advantage is the use of standardized web protocols like HTTP for invoking service operations. Therefore, they can be easily integrated in existing IT infrastructures and can be invoked by every web capable device (Daum, 2005). The four most important standards regarding web services are the Web-Service Description Language (WSDL), a XML format used to describe the external interface of a service, the XML message format SOAP used to invoke service operations, the directory service UDDI and the Web-Service Business Process Execution Language (WS-BPEL), an XML format used to model and execute business processes based on web services (Josuttis, 2008).

When the proof of concept implementation was started, the available open source tools and components were evaluated and the most promising tools were selected based on criteria such as offered functionality, stability, performance, development activity and project and community support. Some components have been updated in the meantime or were being replaced by alternative technologies (such as exporting stateless session beans as web services in EJB 3) (Goncalves, 2009). However, these updates do not significantly affect the results of this study.

3.1 Apache Axis2

Apache Axis2 (http://axis.apache.org) is a Java framework for implementing web services using the SOAP protocol (versions 1.1 and 1.2) as defined by the W3C (http://www.w3c.org) as well as the popular REST (REpresentational State Transfer) pattern and serves as a runtime environment for such web services. Thereby, the interaction with the web service implementation takes place via an object model which primarily mimics the structure of the corresponding WSDL 2.0 file. For XML processing, Axis2 uses its own object model called AXIOM (Axis Object Model) which is based on the streaming API for XML (SAX, http://jcp.org/aboutJava/communityprocess/final/jsr173/index.html) and increases the memory-efficiency notably compared to other document models like Document Object Model (DOM). The functionality of Axis2 can be ex-tended by so called modules. A module can mount itself at any point in the message handling process inside the engine and conduct changes to the in- and outgoing mes-sages. This makes it possible to later extend Axis2 by any WS-* standard like e.g. WS-Security.

3.2 Enterprise Service Bus (ESB)

At the heart of any WSOA an ESB is required for executing the business processes. Therefore, the selection of an appropriate open source ESB forms the basis the proof-of-concept implementation. For building up a WSOA the selected components were required to support all the web service related WS-* standards, in particular WS-BPEL. This lead to the exclusion of the popular Mule ESB (http://www.mulesoft.org), which does not support BPEL processes by default. Due to unclear perspectives regarding development activity and community support, also OpenESB (aka GlassfishESB) was discarded (http://wiki.open-esb.java.net). Finally, the well supported Apache ODE (Orchestration Director Engine) in combination with the Apache ServiceMix ESB was finally selected.

Apache ODE is an open source, BPEL-capable workflow engine developed by the Apache Software Foundation (http://ode.apache.org). It enables the execution of business processes in a WSOA.

3.3 Development Tools

Eclipse (http://www.eclipse.org) is one of the most popular integrated development environments (IDE). Starting with version 3.0, Eclipse itself forms only the core framework, while the actual functionality is provided by numerous plug-ins. Today, Eclipse offers plug-ins for different programming languages as well as nearly every other purpose. Also for WSOA development various useful extensions are available. In the present study, Eclipse IDE 3.5 (Galileo) for Java EE Developers was used. The most important extension in combination with Axis2 are the Apache Axis2 Tools for Eclipse.

NetBeans IDE (http://www.netbeans.org) is a Java-based, open source integrated development environment originally developed by Sun Microsystems for Java developers. Like Eclipse, also NetBeans today supports further programming languages and its functionality may be extended using plug-ins and so-called packs. In the present work, NetBeans IDE 6.7 was used. NetBeans was used in addition to Eclipse due to its powerful and well designed BPEL designer. The latter is part of the Service-oriented Architecture Project (http://soa.netbeans.org/soa/), which provides different modules for NetBeans supporting SOA development. However, this project seems not to be continued further and NetBeans 6.5 was the last release explicitly supported by it.

Only recently, the new open source BPEL Designer plugin V1.0 for the Eclipse IDE has been released (http://www.eclipse.org/bpel/). While the BPEL processes as shown in this paper originally were designed using the NetBeans BPEL designer, the authors evaluated the new Eclipse BPEL Designer by re-modeling these processes partially. The functionality and the graphical WS-BPEL editor of the new Eclipse plugin were found to be comparable to the former NetBeans BPEL designer. Thus, the conclusions drawn below from regarding the BPEL modeling remain valid when using the new Eclipse plugin instead.

soapUI (http://www.soapui.org) is an open source software tool for testing web services. The software is written in Java. soapUI i.e. allows the invocation and inspection of web services using the imported WSDL file. Moreover, it offers additional functionality for performing different performance and functional tests of web services. In the present work soapUI version 3.0.1 was used.

4. PROOF OF CONCEPT STUDY

4.1 Example Process

In order to validate the feasibility of a WSOA architecture, first a typical and realistic business process had to be selected. We decided for a simplified customer order process like it is commonly implemented in Enterprise Resource Planning systems. Fig. 1 depicts the process as event-driven process chain diagram.

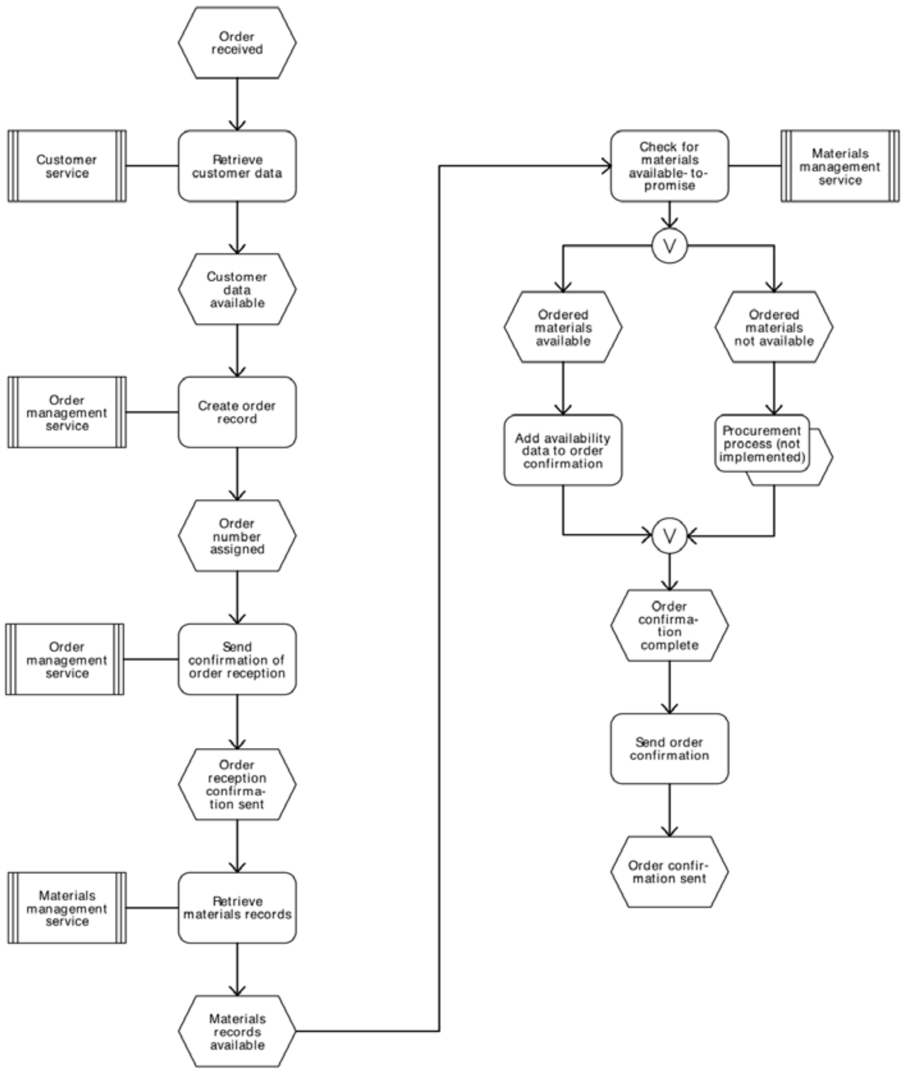
Order received
Customer service
Retrieve customer data
Customer data available
Order management service
Create order record
Order number assigned
Order management service
Send confirmation of order reception
Order reception confirma-tion sent
Materials management service
Retrieve materials records
Materials records available
Check for materials available- to-promise
Materials management service
V
Ordered materials available
Ordered materials not available
Add availability data to order confirmation
Procurement process (not implemented)
V
Order confirma-tion complete
Send order confirmation
Order confir-mation sent

Figure 1. Simplified customer order process.

To execute its business functions, this process requires different basic services implementing the core tasks of customer, materials and order management. In the examined WSOA these services have been implemented as SOAP-based web-services in Java using the Apache Axis2 engine. In the following, German names of certain components are provided in brackets as far as necessary, since the actual proof of concept implementation was done in German language.

4.2 Basic Web Services

Basically there are two possible approaches of developing web services, namely the top-down and the bottom-up approach. In the former, the interface of the web service is developed first in form of a WSDL document, from which the required service Java classes are generated afterwards. The latter starts by first implementing all web service functionality e.g. as Java classes and generating the corresponding WSDL file afterwards from these classes. This is especially useful for converting legacy code into web-services. The Axis2 tools for Eclipse support both variants and allow the generation of the respective missing parts. For the present study, the customer service (KundeService), order management service (BestellungenService) and materials management service (LagerService) have been developed using the top-down approach, each storing its respective data independently in its own relational database and providing at least the respective CRUD (create, retrieve, update, delete) operations in the service interface.

4.3 WS-BPEL Processes

Combining the above web services, the business process illustrated in Fig. 1 was implemented by two executable WS-BPEL processes and deployed to the Apache ODE workflow engine. In the present study, the two processes are used, namely the ordering service (BestellService) and the materials processing service (LagerProzessService). The ordering service forms the main part of the process implementation and orchestrates the other web services and processes. It is intended to be called by a client application to perform a customer order.

The ordering service process invokes the materials processing service to obtain the material records for the ordered materials. The materials processing service executes a WS-BPEL process that basically encapsulates the materials management service and allows its asynchronous invocation. Its purpose is to examine also asynchronous process invocation and so to improve the overall performance and load balancing.

The realized processes are based on the BPEL Version 2.0 as specified by OASIS. The NetBeans IDE 6.7 with the BPEL-Designer Plug-In was used as the development environment. The processes were run in Apache ODE 1.3.3 deployed in Apache Tomcat 6.0.24. The Java Runtime Environment was Java Version $1.6.0_17$ under Mac OS X Version 10.6.2.

4.4 Results

The evaluation of the described realistic proof of concept implementation of a WSOA allowed to study in detail all implications of such an IS architecture, especially regarding development and performance. In addition, it revealed important insights regarding the feasibility of WSOA in general and especially its realization using OSS.

First, it can be clearly stated that it is in principle possible to build a working WSOA purely from OSS components. Also for implementing common security standards or a web service SSO and the integration with enterprise-wide identity management solutions (like i.e. directory services) suitable OSS solutions exist.

However, regarding the feasibility and the actual benefits of a WSOA in general, it became obvious that the overall performance and the computer resource consumption of a WSOA implementation is orders of magnitudes higher compared to the tradition-al architectural pattern of monolithic applications. Within the proof of concept implementation, even a single workflow instance of the order process runs about 5s on standard hardware (Intel Core 2 Duo, 2.1 GHz, 2.5 GB RAM) without further workload on the machine. An alternative implementation of the same process inside a monolithic application using i.e. Java EE (Backschat et al., 2007) on the same hardware would need less than 1s. This time lag is due to the WSOA protocol overheads (SOAP, http, WS-Security, etc.) and the necessary XML generation and parsing and in agreement with results from other studies (Tertilt, 2011). Thus, the functional granularity of the web service operations is crucial for the overall performance. Web service operations corresponding to individual business process functions (or tasks) like in the presented proof of concept in general proved to be too small to achieve a reasonable performance compared to monolithic applications. Thus, it became obvious that specifying the right-sized service operations is the key success factor.

In addition, the loose coupling approach of WSOA requires implementation of distributed transactions in WS-BPEL by means of optimistic transaction processing using compensation handlers in case of a rollback. This significantly increases the modeling complexity as well as further affects the performance compared to transaction monitors like CICS or Java JTA-compliant application servers in traditional monolith-ic applications (Backschat et al., 2007, Goncalves, 2009).

Regarding software development in a WSOA, the approach basically replaces the programming of processes and business rules in high-level programming languages (like i.e. Cobol or Java) with their well established and elaborated development and debugging tool support (like i.e. Eclipse IDE) by graphical process models. The latter are represented as XML documents with still comparably poor OSS development tool support (regarding i.e. debugging and testing). In many cases, additional handcoding of XML was necessary. Commercially available tools (like i.e. Intalio BPM suite) seem to provide a significant improvement, however are not licensed as OSS. Thus, the two main challenges regarding the use of SOA as a means of BPM turned out to be first the transition from the graphical process models to the executable WS-BPEL and WSDL documents, which still requires manual effort and is poorly supported by the development tools, especially those from the OSS domain. Second, regarding the functional granularity of the service operations, the desired flexibility by using small functional units to support BPM is contradicted by their high performance impact. So designing adequate services is the most crucial issue, requiring a suitable compromise regarding their functional granularity.

5. CONCLUSION

In conclusion, this paper presented a proof of concept implementation of a realistic business process to validate the concepts of a web service-oriented architecture (WSOA) for enterprise IS implementation and the feasibility of its realization with open source software (OSS). While the realization with OSS proved to be in principle feasible, the evaluation of the described proof of concept implementation revealed some critical key issues that need to be taken seriously when implementing a WSOA to support BPM. Namely, the functional granularity of the service operations showed to be crucial for the performance and efficiency of the implementation, while it is also important for the flexibility regarding BPM.

Further research is needed to empirically study service development and determine best practices and methodologies to specify, design and implement services of adequate functionality regarding performance and flexibility. In addition, at least as open source software is concerned, a powerful yet easy to use development environment for graphically modeling, testing and deploying business processes is still missing, thus smoothing the transition from BPM to workflow execution. Therefore, we suggest that a respective development effort should be started in the future.

REFERENCES

Adam, S. and Doerr, J. How to better align BPM and SOA –Ideas on improving the transition between process design and deployment. 9th Workshop on Business Process Modeling, 2008.

Backschat, M., Ruecker, B., Scheidt, S. and Hammerstein, O. 2007. *Enterprise JavaBeans 3.0. Grundlagen - Konzepte – Praxis,* Heidelberg, München, Elsevier Spektrum Akademischer Verlag.

Baskerville, R., Cavallari, M., Hjort-Madsen, K., Pries-Heje, J., Sorrentino, M. and Virili, F. 2010. The Strategic Value of Service-Oriented Architecture in Banking. *Int. J. Information Technology and Management,* 9**,** pp. 30-53.

Daum, B., Franke, S., Tilly, M. 2005. *Webentwicklung mit Eclipse. J2EE, Web-Services, Rich und Thin Clients,* Heidelberg, dpunkt-Verlag.

Draheim, D. 2010. *Business Process Technology. A Unified View on Business Processes, Workflows and Enterprise Applications,* Berlin, Springer-Verlag.

Goncalves, A. 2009. *Beginning Java EE 6 Platform with GlassFish 3 – From Novice to Professional,* New York, Apress.

Hirschheim, R., Welke, R. and Schwarz, A. 2010. Service- Oriented Architecture: Myths, Realities, and a Maturity Model. *MISQExecutive,* 9**,** pp. 37-48.

Joachim, N., Beimborn, D., Weitzel, T. Eine empirische Untersuchung des Wertbeitrages von serviceorientierten Architekturen (SOA). Internationale Tagung Wirtschaftsinformatik, 2011. pp. 861-870.

Josuttis, N. 2008. *SOA in der Praxis. System-Design für verteilte Geschäftsprozesse,* Heidelberg, dpunkt-Verlag.

Leyking, K., Dreifus, F. and Loos, P. 2007. Serviceorientierte Architekturen. *Wirtsch. Inform,* 49**,** pp. 394–401.

Manes, A. T. 2009. *SOA is Dead; Long Live Services* [Online]. Available: http://apsblog.burtongroup.com/2009/01/soa-is-dead-long-live-services.html [Accessed 01.11.2011 2011].

Melzer, I. 2008. *Service-orientierte Architekturen mit Web Services. Konzepte - Standards – Praxis,* Heidelberg, Spektrum Akademischer Verlag.

Schmidtmann, V. 2005. *Web Services-basierte Referenzarchitektur für Enterprise Application Integration,* Berlin, wvb Wissenschaftlicher Verlag.

Siebenhaar, M., Lehrig, T., Braun, J. and Görge, T. 2008. Entwicklung einer SOA-basierten Webanwendung zur Buchung und Verwaltung von Segeltouren: Propriet\"are Software vs. Open Source. *Wirtsch. Inform,* 50**,** pp. 325–329.

Tertilt, D., Krcmar, H. 2011. Generic performance prediction for ERP and SOA applications. *Proc. 19th European Conference on Information Systems.* Helsinki: AIS.

Varadan, R., Channabasavaiah, K., Simpson, S., Holley, K. and Allam, A. 2008. Increasing business flexibility and SOA adoption through effective SOA governance. *IBM Syst. J,* 47**,** pp. 473–488.

Viering, G., Legner, C. and Ahlemann, F. The (Lacking) Business Perspective on SOA - Critical Themes in SOA Research. Internationale Tagung Wirtschaftsinformatik, 2009. pp. 45-54.

Welke, R., Hirschheim, R. and Schwarz, A. 2011. Service-Oriented Architecture Maturity. *IEEE Computer***,** pp. 61–67.

Wieland, T. 2007. SOA auf Basis von Open-Source Software - Freie Architekturen mit freier Software. *OBJEKTspektrum***,** pp. 1–5.