Mashups: An Approach to Overcoming the Business/IT Gap in Service-Oriented Architectures

Stefan Bitzer and Matthias Schumann

University of Goettingen,
Institute for Information Systems,
Platz der Goettinger Sieben 5
D-37073 Goettingen, Germany
sbitzer@uni-goettingen.de,
mschuma1@uni-goettingen.de

Abstract. For quite a long time already, great importance has been attached to the concept of Service-Oriented Architectures for future IT-architectures. However, a major challenge in implementing this concept lies in the gap between the functional department and IT department. Mashups, an architecture also based on services, try to avoid this gap by letting the user himself integrate the services. The following article analyzes similarities and differences between both architecture approaches, and explains to what extent and in which cases Mashups could complement a Service-Oriented Architecture.

Keywords: Mashups, Service-Oriented Architecture, Business/IT Gap.

1 Introduction

Since the midst 1990's, science and practice have been dealing with the concept of Service-Oriented Architecture (SOA) (Natis, 2003). However, even though great importance has been attached to this concept for IT-architectures, effective usage in practice has been rather low (Legner and Heutschi, 2007). One of the main problems for this is the recurring gap between functional knowledge and technology. It is also referred to as the Business/IT Gap (Josuttis, 2007). This divergence is generated by the fact that the people involved, the future users and the IT-staff, have a different understanding of terms and work at cross-purposes.

One solution of bridging this gap is the Business Process Execution Language (BPEL). BPEL allows for the orchestration of services that run fully automatic afterwards (Farahbod et al., 2004). In order to enable human process interaction in BPEL, two more specifications have been added. WS-Human Task describes general functions that embed humans. WS-Bpel4People specifies the application of WS-Human Task directly in BPEL (Russell and van der Aalst, 2007). In order to present the business process graphically, the Business Process Modelling Notation (BPMN) acts as specification language for the graphical description. Since BPMN is graph-oriented while BPEL is primary block-structured, the translation between the languages is quite difficult (Ouyang et al., 2006) and comprises several problems (Recker et al., 2006). Furthermore, the use of BPMN is complicated due to large range of functions (Recker, 2008).

Mashups are another architecture specializing in the integration of services (Jhingran, 2006). They are applications of the Web 2.0 concept (O'Reilly, 2005) and deal with the simple integration of services and content by the user. Both architectures are service-oriented, but Mashups place particular emphasis on the participation of users - a typical feature of Web 2.0 (Cañas et al., 2007). Given the problem described in the beginning, the question arises whether the service orientation of SOAs is sufficient in practice or whether the user should be integrated more actively into the process design. One way of overcoming the Business/IT Gap could be the use of Mashups. First approaches have already been discussed in the literature (Hierro et al., 2008; Schroth and Janner, 2007), focussing on a global, user-centred SOA. This article argues for a different approach, namely to use Mashups complementarily or rather as a part of SOA. In order to be able to identify the potentials of usage, a structured comparison of Mashups and SOAs is required. Contact points and areas of differences have to be examined. The final aim of this article is to find out whether Mashups can be used as a part of SOA in order to integrate users actively into the exercise.

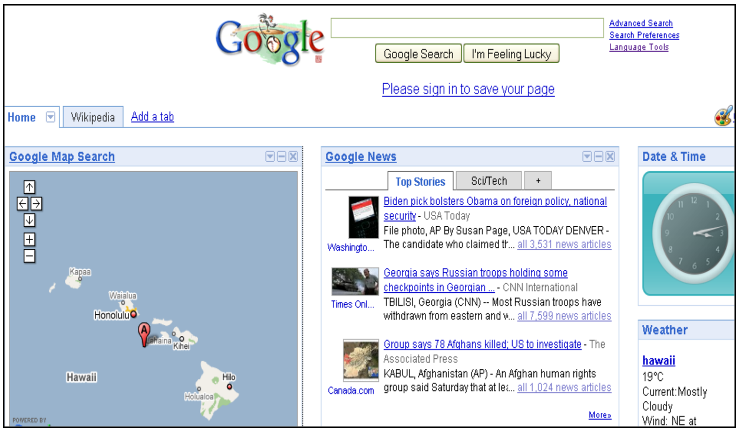
For this purpose, Mashups and possible categories of Mashups will be introduced in the second section. Afterwards, a definition of Service-Oriented Architectures will be given. Section three and four analyze the similarities of and differences between the two types of architectures and provide a summary of the results. The article ends with a conclusion and a short outlook.

2 Mashups and Service-Oriented Architectures

2.1 Mashups

In connection with the Web 2.0, Mashups integrate Web services, data and other content (Floyd et al., 2007). These elements of a Mashup can frequently be obtained free of charge from the internet. Thus, Web services and data formerly separated from each other are integrated into the user's own applications. A simple example for a Mashup is iGoogle (http://www.google.com/ig), illustrated in figure 1, where components from different sources can be aggregated into a personally customized Web site (Guo et al., 2008). Furthermore, the integration of several Web services presents us with the possibility of combining applications and data originally separated, which leads to new benefits for users (Jhingran, 2006).

Mashups are Web 2.0 applications and therefore characterized by typical features of the Web 2.0 (Cho, 2007). Mashups use "the web as a platform"; that is to say they are a Web application to gain access to services over the internet (Jhingran, 2006). Furthermore, Mashups are a typical example of the Lightweight Programming Model (LPM). In order to guarantee an efficient distribution in the Web, the applications and services should avoid complex designs and concentrate on simplicity and loosely linked systems. With regard to Mashups this indicates hat the administration and creation of Mashups should be designed simply, so the users can easily access to technology. To achieve this, standardized interfaces and intuitive programming languages are used. Despite their simplicity, Mashups as Web 2.0 applications have to offer the user-friendliness of desktop applications and have to undergo continuous development.



Biden pick bolsters Obama on foreign policy, national security - USA Today
File photo, AP By Susan Page, USA TODAY DENVER - The candidate who claimed th... all 3,531 news articles

Georgia says Russian troops holding some checkpoints in Georgian ... - CNN International
TBILISI, Georgia (CNN) -- Most Russian troops have withdrawn from eastern and w... all 7,599 news articles

Group says 78 Afghans killed; US to investigate - The Associated Press
KABUL, Afghanistan (AP) - An Afghan human rights group said Saturday that at le... all 1,024 news articles

hawaii
19°C
Current:Mostly Cloudy
Wind: NE at

Fig. 1. Extract of an iGoogle Web site

According to the definition given in the beginning of this section, Mashups integrate and combine services, data and other content. This description is very general as these tasks can also be performed by other technologies. It is therefore reasonable to specify this definition even further. The user is supposed to be actively integrated into the application, as it is a characteristic of Web 2.0. In this case, it implies that the user creates and assembles the Mashups on his own. This form of user participation is described as user-driven or “user-centred micro-orchestration“ (Cañas et al., 2007; Grumann, 2006).

In this process, the user should be able to integrate and combine the desired data and services according to his own wishes. Until recently, these possibilities rarely existed, as earlier applications mostly had to be implemented and administered by IT-experts. Furthermore, these experts were needed for the adaptation of applications to the user’s demands. Now, due to the concrete integration of users into this process, Mashups enable the applications and information to be developed by users for users. However, in this context it is important that the necessary data and services are available in the required granularity (Grumann, 2006).

2.2 Mashup Categories

Regardless of the differing opinions about the specifications of Mashups in the literature (Guo et al., 2008; Kulkarni, 2007), it is possible to deduce a classification scheme. In this paper we distinguish Mashups on the basis of the functional range, the target group and the location of the technical implementation.

The *functions of Mashups* can be divided in *presentation level, data level* and *logic based Mashups* (Dornan, 2007; Kulkarni, 2007). *Presentation level Mashups* provide layout and information in various ways as Web services. In enterprises, these Mashups resemble customized portals; comparable to iGoogle. *Data level Mashups* concentrate on the extraction and combination of data from different sources and integrate them into the user's own internet site. Well-known examples are Web pages combining online maps with further data, e. g. Healthmap.org. *Logic based Mashups* are the most complex type and contain services combined with application logic. They are currently used in price-comparison websites like Kayak.com, which use Web services to send inquiries to online travel agencies and book flights afterwards.

The *user groups* are differentiated into *consumer* and *enterprise Mashups*. Up to now, Mashups are mainly used for private applications. However, more and more businesses use Mashups as internal software architecture. These enterprise Mashups focus on the individualization of software by staff members (Grumann, 2006; Proto, 2007). Ideally the employees are able to integrate internal and external resources into their Mashups. Therefore, safety aspects during data processing play an important role in this area (Vikram and Steiner, 2007). In 2007 IBM released the IBM Mashup Center, a software that claims to meet the requirements described above. Among other things, employees with limited IT-skills shall be enabled to mix information from different data sources (IBM, 2008).

The technical realization can be done in two places: directly on the client or on a server. *Client-side Mashups* integrate services and content on the client, mostly in a Web browser (Ort, Brydon and Basler, 2007b). In contrast, *server-side Mashups* are created on a server. This internal server also functions as a proxy between the client and the respective providers, so the actual work is relocated from the client's web browser to the server. This version is advantageous because it offers an improved handling of safety requirements (Ort, Brydon and Basler, 2007a).

In order to be able to make a meaningful comparison between Mashups and SOAs, the same requirements should be applied to functional range and target groups in both architectures. Consequently, Mashups of the following categories are used: functional range *data* or *logic based*, target group *enterprise* and technical conversion *server-side*.

2.3 Service-Oriented Architectures

Even though SOAs have been under discussion for a long time, there is still no generally accepted definition in the literature. On the one hand, SOA can be regarded as a method (Jardim-Goncalves et al., 2006), and on the other hand, as a system architecture (Krafzig et al., 2008). Moreover, an SOA is generally characterized by its ability to enable different applications to exchange and process data independent of the underlying system software and the chosen programming language. Therefore, complete applications or parts are offered as services that can be used without coding efforts. This form of integration is described as a loose linkage of services. The enclosure in services creates independence from platforms and programming language, and enables the integration of legacy-applications in services. The integration of individual components into an SOA is done through standardized interfaces.

At present, using Web services is the most common way of implementing an SOA because the standardization of certain aspects (addressing, security, transactions, and policy) is well advanced. However, Web services are only one possible solution for the implementation of an SOA (Santillo, 2007).

There are approaches how to integrate the functional departments into the SOA's process design. The Business Process Execution Language (BPEL), also known as BPEL for Web Services (BPEL4WS), is the basic element. It offers various possibilities of orchestrating services (Farahbod et al., 2004). However, this combination has to run fully automatically, meaning that in practice all parameters needed for a process have to be calculated beforehand. If important information is missing, the user cannot enter it while the process is still running. Unfortunately, this is a great limitation, as automatic processes rarely exist in practice (Zimmermann et al. 2005).

In July 2005, IBM and SAP published a white paper addressing this problem entitled "BPEL4People". The paper reveals that "Human Interaction" should be seen as a major part of running processes. Suggestions are given on how to implement these human interactions in processes in BPEL. In the meantime, two more specifications have evolved from these: *WS-Human Task* describes functions to integrate people. *WS-BPEL4People* describes the application of WS-Human Task directly with BPEL (Russell and van der Aalst, 2007). This alteration enables people to take part in processes supported by Web services or even activate them. However, it is not the intention of BPEL4People to involve staff members more strongly in the development of processes or applications as such. Currently, nearly all products on the market are able to execute proprietary implementations of the BPEL4People idea. In the near future, it will probably offer a standardized implementation based on the specifications available at the present time.

BPEL is a purely XML-based description of Web service cycles and cannot be depicted graphically. Accordingly, it is complex and difficult to apply (Zimmermann et al., 2005). Since 2005 the specification language BPMN – Business Process Modelling Notation – is being used to graphically model business processes. The BPMN standard defines how a BPMN diagram should be translated into BPEL. The problem is that the ability of expression is not the same in BPMN and BPEL. BPMN models are usually under-specified and details that are relevant for the execution are ignored. Furthermore, the translation of a BPMN model into a BPEL scheme will in some cases lead to semantic deviation. BPMN, for example, is graph-oriented while BPEL is primarily block-structured. This makes translations between the languages quite difficult (Ouyang et al., 2006) and leads to several problems (Recker et al., 2006). In addition, the application of BPMN is not easy, as it has a large number of functions (Recker, 2008). A further limitation is the fact that translations between BPMN and BPEL can only go in one direction, even when using the WSBPEL version 2.0. This means that BPEL definitions can be obtained out of the BPMN, but due to missing language elements it is not possible to create a BPMN model from a BPEL definition (Giner, Torres and Pelechano, 2007).

3 A Comparison of Mashups and Service-Oriented Architectures

3.1 Similarities between Mashups and Service-Oriented Architectures

SOAs and Mashups are both service-orientated and are often associated with each other in this context (Cañas et al., 2007). The following similarities can be derived from this basic idea.

Both architectures are based on the encapsulation of services and data. Therefore, they are independent of system software and programming languages. Unfortunately, this independence only works in theory, as both Mashups (Palfrey and Gasser, 2007) and SOAs (IBM, 2006) show a dependency in practice. Furthermore, in both cases encapsulation relies on the usage of loosely connected and widely distributed services. The result is a high degree of agility, so that services can be renewed and completely replaced during running requests in a simple way. This enables a quick adaptation to new circumstances (Schroth and Janner, 2007).

The foundations for this linkage of dispersed services and data are well-defined standardized interfaces. These interfaces do not only guarantee the integration of components into Mashups and SOAs. They also lead to a reduction of programming work, because components are developed separately from the architectures. It is therefore easy to reuse the services, e. g. in another composition for other purposes. It is also possible to combine separate services into one new service and make it available separately. In the case of SOAs, this is done by so called complex services (Yu and Lin 2005). Mashups use widgets for the bundling of services and data. These widgets are reusable fragments of Mashups and small applications for a specific task (Hoyer et al., 2008). Both SOAs and Mashups allow an additional level of abstraction by combining services.

Even though neither Mashups nor SOAs require standards in definition, a lot of the services used are Web services (Santillo, 2007; Jhingran, 2006). Whereas SOAs mainly use WS*-Web services with SOAP as transmitting standard, the present development of Mashups tends towards RESTful Web services (see also the next section).

3.2 Differences between Mashups and Service-Oriented Architectures

The main differences between Mashups and SOAs result from the fact that Mashups originate from the area of the Web 2.0 while SOAs have its source in the business process management and refer to the realisation of the similarities described above.

The design of both architectures is based on the encapsulation of services and their combination. However, Mashups and SOAs are realised differently. Mashups, as a Social Software (Cho, 2007), involve the user more strongly in the development process. Thus, people are an essential element of value proposition. In order for the user to be able to create and adapt Mashup applications according to his needs, the services have to provide the required degree of freedom with regard to structure and content. This process is carried out by user-friendly interfaces, most of which rely on human-oriented documentation (Pautasso, 2008). However, SOA applications are used to connect separated business functionalities and their applications. Accordingly, emphasis is not laid on the user, but on the connection of applications; e. g. realized via the BPEL (Farahbod et al., 2004). In this professional application, the composition of static control and governance processes is determined in advance (Weill and Ross, 2004). Thus, the approach to implement an SOA or a Mashup is different. Additionally, the concrete realisation of both architectures also varies.

Figure 2 shows an abstracted process, where the functional department is integrated into the design process of an SOA. At the beginning, the functional department analyses their processes. Afterwards, the functional department, in cooperation with the IT-department, describes these processes with Business Process Modelling, e. g. graphically with the BPMN. In doing so, BPM is supposed to be the bridge between IT and Business (White, 2004). Due to the rich and partially complicated BPMN language, communication difficulties between the IT and the functional department can occur during this step. The next step is the translation into

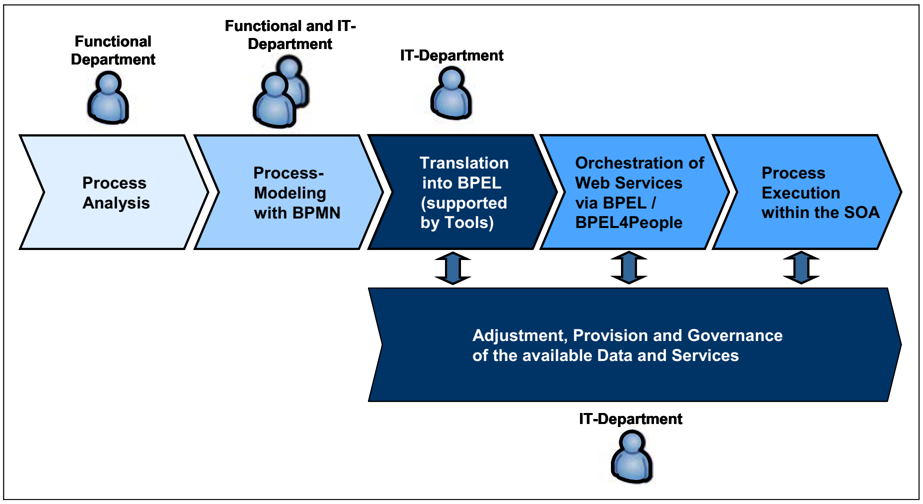

Functional Department
Functional and IT-Department
IT-Department
Process Analysis
Process-Modeling with BPMN
Translation into BPEL (supported by Tools)
Orchestration of Web Services via BPEL / BPEL4People
Process Execution within the SOA
Adjustment, Provision and Governance of the available Data and Services
IT-Department

Fig. 2. Integration of the functional department into the process design within an SOA

BPEL, or BPEL4People. Even though it is presently supported by several tools (Ouyang et al., 2006), it is still subject to the restrictions described in the section *"Service-Oriented Architectures"*. After the successful translation of the processes, the services are orchestrated via BPEL in the fourth step. Finally, the services are executed in the SOA. To safeguard an unobstructed procedure of the SOA, the services have to be provided and controlled by the IT department at least starting from the step of translation. It can be summarised that the IT department is deeply integrated into the process design within an SOA in this scenario.

The process design within a Mashup architecture, shown in figure 3, is fundamentally different from the process in an SOA. The functional department combines services and data on its own with the help of a Mashup Building Platform. This solves the problem of communication between specialists and IT-department, as well as the problem of translating from BPMN into BPEL and the respective time-consuming process. On the other hand, process analysis and modelling with BPMN enable a structured examination of processes and correlations. In the end, this leads to an improved understanding of the business processes (Melão and Pidd, 2000). This approach is not included in the process presented in figure 3. As described in the section before, in Mashups the services chosen can be aggregated through widgets. Other than with the complex services in SOAs, this can be done using drag-and-drop in a graphic development environment (Proto, 2007). Widgets thus convert the used interfaces into a graphical visualisation.

The IT-department's main function is therefore the reprocessing of data and services for the specialist department. It is also responsible for its adaptation, control and maintenance. Once they are reprocessed, the workload of the IT-department is reduced.

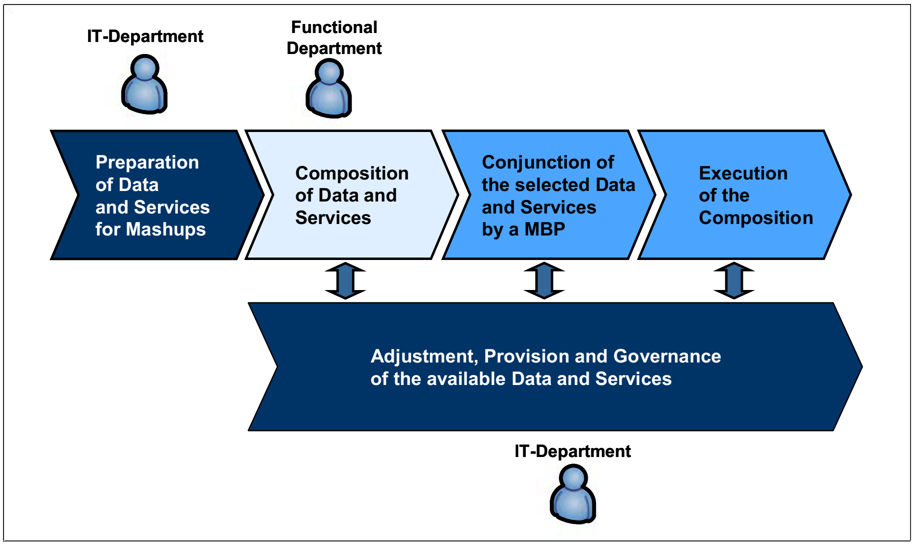

IT-Department
Functional Department
Preparation of Data and Services for Mashups
Composition of Data and Services
Conjunction of the selected Data and Services by a MBP
Execution of the Composition
Adjustment, Provision and Governance of the available Data and Services
IT-Department

Fig. 3. The process design within a Mashup architecture

The processes shown in figures 2 and 3 lead to further differences between the two architectures. A semantic interoperability is needed in the process of selecting services with SOAs (Vetere and Lenzerin, 2005). In the interaction, machines are not able to react to mistakes occurring in the semantics between the services and correct them. Consequently, it has to be guaranteed that all services involved in an application interpret data semantically in the same way. Mashups, on the other hand, design contents of functions with the goal of integrating people directly via the user-interface. The user, as the developer of the application, interprets the data and is thus able to correct possible semantic mistakes between services (Schroth and Janner, 2007).

Another technical difference between SOAs and Mashups can be found in the transmission standards used. The World Wide Web Consortium (W3C) has established standards for SOAs, such as SOAP for message exchange, WSDL for description and UDDI as a directory service for Web services. Web services based on SOAP as a message protocol are action-oriented, so-called WS-* or SOAP Web services (Snell, 2007). In contrast, Mashups are presently developing in the direction of a resource-oriented architecture, so-called Representational State Transfer (REST) Web services (Richardson and Ruby, 2007). In order to approach Web services as a resource, the REST-architecture uses simple HTTP-method calls such as GET, PUT, DELETE or POST. Both architectures, resource-oriented and action-oriented, revert to HTTP. However, SOAP Web services merely use HTTP for the transport of remote procedure calls (RPC). On the other hand, REST Web services directly apply the transmission protocol's methods. Because simple HTTP-methods are used for the integration of services into the Mashup, there is no room for misinterpretation. The exchange of messages via SOAP allows for more freedom within an SOA. However, this makes it more complex and renders the maintenance of interoperability more difficult (Schroth and Janner, 2007).

4 Summary

The similarities and differences found in previous sections are presented in the following table.

Table 1. Similarities and Differences between Mashups and SOAs

Feature	Mashup	SOA
Design of Architecture	Use of loosely linked and encapsulated services for the connection of dispersed applications	
	Well defined, standardized interfaces	
	Creation of assembled services	
	High degree of agility	
	User as the essential element of the value proposition	Connection of separate business functions
	Freedom of structure and content	Ex-ante restriction via control and governance processes
	Internal and external services	Mostly internal services
	For situational applications	For complex and standardized business processes
Design of Technology	Independence of system software and programming language	
	Integration through standardized interfaces	
	Reuse of components, little programming work	
	Web services as mainly used services	
	User (functional department) integrated directly into the developing process	Speciality department only integrated into the developing process in cooperation with the IT-department
	Orchestration of data and services by Mashup Building Platforms	Orchestration of data and services by BPMN and BPEL
	Semantic interpretation by the user	Semantic interoperability has to be guaranteed
	Trend towards REST Web services	SOAP Web services as de facto standard
	Connection of services and data in widgets	Connection of services as complex services

It can generally be stated that BPEL shows its strengths where long-lasting processes are to be implemented. BPEL, or rather BPEL4People, can thus be viewed as the instrument for automating stable and persistent business processes in an SOA. It is not BPEL's goal to write conventional applications. The aim is rather to line up a certain amount of services, which can be either inside or outside of a company in order to simulate a business process that is to be implemented (Emmerich et al., 2005). In an SOA the implemented business processes are standardized and more or less persistent. Mashups, on the other hand, are used to individualise and adapt applications as well as data (Hoyer et al., 2008). The focus here lies on the ad hoc combination and integration of services and data; hence one speaks in this context of situational applications (Jhingran, 2006; IBM 2008). This explains why Mashups are heavily discussed in the area of Business Intelligence (BI) and knowledge workers (Soriano et al. 2007, Proto 2007). In this regard Microsoft focuses on Mashups as BI tools not only for financial analysts, but also for engineers, geoscientists and operations. The Mashup aggregates and visualizes multiple data streams, e. g. in Exploration & Production (oil industry) it collects data from the Geographic Information System (GIS), the BI- and HR-system for tracking hurricanes to ensure personnel safety (Brulé and Hodges, 2007).

5 Conclusion and Outlook

As shown in the previous two sections, Mashups and SOAs both focus on the use of services. Differences are due to the architectures' diverging intentions and to the concrete realisation of the technology; therefore the two architectures are used in different application scenarios.

A main difference lies in the respective process design. The interactions between functional department and IT department are an essential part of an SOA if the business process is to be optimally reflected in an SOA. Because of this complex process, it is difficult to meet the high demand of small but individual solutions. This is where Mashups show their strengths: the user can combine and create small applications and data according to his needs. This is especially convenient for small, partial routine and individual tasks. In addition to higher demands on required interfaces and freedom of structure and content, Mashup Building Platforms that are easy to handle will become necessary. If the use of these platforms is still too complicated for the functional department, at least the IT department can create Mashups up to ten times faster than normal (Brulé and Hodes, 2007).

The user-driven integration and combination of services via Mashups could be a step towards involving the functional departments into the process design of applications and thus help to meet the problem of the Business/IT Gap. However, complex and standardized business processes will still have to be implemented through the cooperation of IT and functional departments, for example using BPEL4People. This is due to the fact that a staff member of the functional department will not be able to cope with simulating a complex business process with the help of a Mashup Building Platform. Additionally, it is often not desirable that every staff members is able to create and change a standardized business process according to his wishes. Even though Mashups offer a lot of freedom for the user, in a professional environment it is necessary to control the use with a governance structure. Only then Mashups as situational applications and SOA for standardized business processes can complement one another and lead to a better relationship between functional and IT department. The different intentions of SOA and Mashup clearly show that both architectures are not mutually exclusive, but complement one another.

References

1. Brulé, M., Hodges, C.: Improving E&P performance with right-time business intelligence. World Oil 11 (2007)
2. Cañas, M.A., Hierro, J.J., Hoyerl, V., Janner, T., Lizcano, D., Reyes, M., Schroth, C., Soriano, J.: Enterprise Mashup Putting a face on the next generation global SOA. In: Proceedings of WISE 2007 (2007)
3. Cho, A.: An introduction to Mashups for health librarians. Journal of the Canadian Health Libraries Association 28, 19–22 (2007)
4. Dornan, A.: Enterprise Mashups: Mashed Or Half-Baked? (2007), `http://www.networkcomputing.com/immersion/soa-ws/showArticle.jhtml?articleID=201804242` (retrieved 2009-01-30)

5. Emmerich, W., Butchart, B., Chen, L., Wassermann, B., Price, S.L.: Grid Service Orchestration Using the Business Process Execution Language (BPEL). Journal of Grid Computing 3, 283–304 (2005)
6. Farahbod, R., Glaesser, U., Vajihollahi, M.: Specification and Validation of the Business Process Execution Language for Web Services. In: Zimmermann, W., Thalheim, B. (eds.) ASM 2004. LNCS, vol. 3052, pp. 78–94. Springer, Heidelberg (2004)
7. Floyd, I.R., Jones, M.C., Rathi, D., Twidale, M.B.: Web Mash-ups and Patchwork Prototyping: User-driven technological innovation with Web 2.0 and Open Source Software. In: Proceedings of HICCS-40 (2007)
8. Giner, P., Torres, V., Pelechano, V.: Bridging the Gap between BPMN and WS-BPEL. M2M Transformations in Practice. In: Proceedings of MDWE 2007 (2007)
9. Grumann, G.: Enterprise Mashups. InfoWorld 28, 31 (2006)
10. Guo, R., Zhu, B.B., Feng, M., Pan, A., Zhou, B.: Compoweb: a component-oriented web architecture. In: Proceeding of WWW 2008 (2008)
11. Hierro, J.J., Janner, T., Lizcano, D., Reyes, M., Schroth, C., Soriano, J.: Enhancing User-Service Interaction Through a Global User-Centric Approach to SOA. In: Proceedings of ICNS 2008 (2008)
12. Hoyer, V., Stanoeysk-Slabeya, K., Janner, T., Schroth, C.: Enterprise Mashups: Design Principles towards the Long Tail of User Needs. In: Proceedings of IEEE SCC 2009 (2009)
13. IBM. IBM Workplace Collaboration Services V2.6 (2006), `http://www-306.ibm.com/common/ssi/rep_ca/7/897/ENUS206-00/ENUS206-007.PDF` (retrieved 2009-01-30)
14. IBM. IBM Mashup Starter Kit (2008), `http://www.alphaworks.ibm.com/tech/ibmmsk` (retrieved 2009-01-30)
15. Jardim-Goncalves, R., Grilo, A., Steiger-Garcao, A.: Challenging the interoperability between computers in industry with MDA and SOA. Computers in Industry 57(8), 679–689 (2006)
16. Jhingran, A.: Enterprise Information Mashups: Integrating Information, Simply. In: Proceedings of VLDB 2006 (2006)
17. Josuttis, N.: SOA in Practice – The Art of Distributed System Design. O'Reilly, Sebastopol (2007)
18. Krafzig, D., Banke, K., Slama, D.: Enterprise SOA: Service-Oriented Architecture Best Practices, 7th edn. Prentice-Hall, NJ (2008)
19. Kulkarni, S.: Enterprise Mashup. In: Proceedings of Indic Threads. com Conference on Java Technology (2007)
20. Legner, C., Heutschi, R.: SOA Adoption in Practice-Findings from Early SOA Implementations. In: Proceedings of ECIS 2007 (2007)
21. Melão, N., Pidd, M.: A conceptual framework for understanding business processes and business process modelling. Information Systems Journal 10(2), 105–129 (2000)
22. Natis, Y.V.: Service-Oriented Architecture Scenario, Gartner Research Note AV-19-6751, Stamford (2003)
23. O'Reilly, T.: What Is Web 2.0 (2005), `http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html` (retrieved 2009-01-30)
24. Ort, E., Brydon, S., Basler, M.: Mashup Styles, Part 1: Server-Side Mashups (2007a), `http://java.sun.com/developer/technicalArticles/J2EE/mashup_1/` (retrieved 2009-01-30)

25. Ort, E., Brydon, S., Basler, M.: Mashup Styles, Part 2: Client-Side Mashups (2007b), http://java.sun.com/developer/technicalArticles/J2EE/mashup_2/ (retrieved 2009-01-30)
26. Ouyang, C., van der Aalst, W.M.P., Dumas, M., ter Hofstede, A.H.M.: Translating BPMN to BPEL. In: Proceedings of CAiSE 2006 (2006)
27. Palfrey, J.G., Gasser, U.: Mashups Interoperability and eInnovation. Berkman Publication Series, 11/07 (2007)
28. Pautasso, C.: BPEL for REST. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) BPM 2008. LNCS, vol. 5240, pp. 278–293. Springer, Heidelberg (2008)
29. Proto. Mashups - Understanding Mashup Building Platforms for Business Applications, Whitepaper (2007), http://www.zdnet.de/itmanager/whitepapers/0,39026294,88011773p-39002281q,00.htm (retrieved 2009-01-30)
30. Recker, J.: BPMN Modeling – Who, Where, How and Why. BPTrends 5(3), 1–8 (2008)
31. Recker, J., Indulska, M., Rosemann, M., Green, P.: How Good is BPMN Really? Insights from Theory and Practice. In: Proceedings of ECIS 2006 (2006)
32. Richardson, L., Ruby, S.: RESTful Web Services. O'Reilly, Beijing (2007)
33. Russell, N., van der Aalst, W.M.P.: Evaluation of the BPEL4People and WS-HumanTask Extensions to WS-BPEL 2.0 using the Workflow Resource Patterns. BPM Center Report BPM-07-10 (2007)
34. Santillo, L.: Seizing and Sizing SOA Applications with COSMIC Function Points. In: Proceedings of SMEF 2007 (2007)
35. Schroth, C., Janner, T.: Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services. IEEE IT Professional 9(3), 36–41 (2007)
36. Snell, J.: Resource-oriented vs. activity-oriented Web services (2007)
37. Soriano, J., Lizcano, D., Cañas, M.A., Reyes, M., Hierro, J.J.: Fostering Innovation in a Mashup-oriented Enterprise 2.0. Collaboration Environment, SISN 1(1), 62–68 (2007)
38. Vetere, G., Lenzerini, M.: Models for semantic interoperability in service-oriented architectures. IBM Systems Journal 44(4), 887–903 (2005)
39. Vikram, K., Steiner, M.: Mashup Component Isolation via Server-Side Analysis and Instrumentation. In: Proceedings of Web 2.0 Security and Privacy Workshop (2007)
40. Weill, P., Ross, J.W.: IT Governance: How Top Performers Manage IT Decision Rights for Superior Results. Harvard Business School Press, Boston (2004)
41. White, S.A.: Introduction to BPMN, BPTrends, 07/04 (2004)
42. Yu, T., Lin, K.Y.: Service Selection Algorithms for Composing Complex Services with Multiple QoS Constraints. In: Benatallah, B., Casati, F., Traverso, P. (eds.) ICSOC 2005. LNCS, vol. 3826, pp. 130–143. Springer, Heidelberg (2005)
43. Zimmermann, O., Doubrovski, V., Grundler, J., Hogg, K.: Service-oriented architecture and business process choreography in an order management scenario: rationale, concepts, lessons learned. In: Proceedings of OOPSLA 2005 (2005)