Computer Science and Engineering 2012, 2(5): 68-76
DOI: 10.5923/j.computer.20120205.05

Requirements for Operations to Take Advantage of the Flexibility of Soa

Liane Will[1,2]

[1]Institut für Technische und Betriebliche Informationssysteme, Otto-von-Guericke-Universität, Magdeburg, 39106, Germany
[2]Active Global Support, SAP AG, Berlin, 10178, Germany

Abstract Efficient, controlled, and managed IT operations are strongly influenced by the architecture of a solution. A service-oriented architecture has a high level of flexibility and can respond to environmental changes and requirements. In this paper, we show, within the change management process, that simple reuse of operation tools, typically developed for client/server-based architectures, do not work with the requirements of SOA. To take advantage of the flexibility of SOA, the traditional change management system-oriented approach has to be changed into a business process-oriented approach, which results in fundamental new requirements for change management tools and services. Furthermore similar conclusions can be derived for other typical operation tasks. Consequently, the common understanding of the necessary central elements of SOA has to be extended. Another fundamental element is required for process-oriented operations: a central operations cockpit. We show what such a tool looks like and which requirements services have to fulfil. We use the example of change management. For service design and development, a SOA-compatible service has not only to fulfil requirements of the common basic elements, but also has to provide a central operations cockpit.

Keywords SOA, Change Management, ITIL, ITSM, Operations, Flexibility, Service Design

1.Introduction

The ongoing development and implementation of solutions based on service-oriented architecture (SOA) raise new issues for the governance and the operation of such solutions. How important and challenging it is to take advantage of the flexibility is investigated in[1]. But they try to solve it by adapting organizational structures, processes and employees. Without doubt this is necessary but not sufficient. A similiar approach follows[2] by investigation of the lifecycle of SOA based solutions. They found as well an increasing complexity and a high alignment effort for developers. We follow the proposals that and how organizations and its processes have to be adapted. Nevertheless, we assume that the technical way of governance of SOA hast to be changed. Client/server-based solutions are more or less fixed in their technical component structure. They are system-oriented. In the following article, using the example of change management, we conclude that service orientation requires a new type of tool for change execution as well as for the operations within IT Service and Support Management in general.[3] investigated as well in the change management in SOA environment. They found that the management of service portfolio is an essential task of IT operations and requires the carful maintenance of service repository. Service repository needs to contain all relevant information to find and manage appropriate services. We will show that this is necessary but also not sufficient. We use an example of a simplified sales order process to demonstrate typical new challenges that need to be addressed if you want to take advantage of the flexibility of SOA based solutions. We describe the properties required in a tool fit for business process-oriented change management, and conclude that such a tool must be considered to be an essential element within SOA. As a result a service has to provide such a tool standardized, as is common for the known basic elements of SOA.

This paper is structured as follows: In Chapter 2, we summarize the basics of SOA and also what is understood by the term flexibility. In Chapter 3 we show, using our example of processing sales orders, differences in change management of a client/server and a SOA-based solution and requirements for operations and services which arise for the latter. In Chapter 4 we discuss the main quality criteria and best practices in IT Service Management (ITSM). We have chosen the execution of changes to show the new challenges in more detail and how such a central operations cockpit needs to work and what a service has to provide. We summarize our findings in Chapter 5.

* Corresponding author:
liane.will@sap.com (Liane Will)
Published online at http://journal.sapub.org/computer

2. Fundamentals

The IT Governance Institute published the Control Objectives for Information and Technology (COBIT)[4] and good practices within IT Infrastructure Library (ITIL)[5] which are established as common guidelines in the area of IT governance and management.

[6] discusses how to build an agile enterprise. They focus on processes, roles and responsibilities, and how technology, e.g., SOA could be used. The following typical questions arise when we discuss SOA in this context:

- Who is the real data owner and are there agreements for the reuse of data?
- Who uses a service and how is cost distributed?
- Who is responsible for error analysis and solution providing for a commonly used service in SOA?
- Who decides whether or not a service is allowed to be used by other business departments?

Assuming that governance processes are established and the above mentioned questions are answered; what happens to the operations and the technical implementation if you exhaust the capability of flexibility of SOA?

To analyze this, it is important to understand SOA and its basics which form the basis of the flexibility capability.

2.1. Definition of SOA

In contrast to the introduction of client/server architecture, the development of SOA is an evolution. Accordingly, the definition of SOA also develops over time.

There is a consensus that one key element is the service itself. The Organization for the Advancement of Structured Information Standards (OASIS) defines the key element service as: “a service is a mechanism to enable access to one or more capabilities, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description”[7].

Based on this[8] defines SOA as follows: „SOA is a system architecture which allows the usage of diverse, different and maybe incompatible methods or applications as reusable and free accessible services and therefore provides a platform, technology and language independent usage and recycling”[7]. This definition is open and focuses on the properties of SOA and its services to determine SOA. An alternative definition is given by[9], where SOA is defined in terms of its elements:”a Service-Oriented Architecture (SOA) is a software architecture that is based on the key concepts of an application frontend, service, service repository, and service bus. A service consists of a contract, one or more interfaces, and an implementation”[9]. The authors define SOA using its own components to describe the necessary properties. In contrast,[8] ignores the specification of the implementation. The technical realization is not described. However, there is a common understanding of the components described by[8] and both definitions are reasonable without contradicting one another. We will show that the definition by[9] needs to be extended by another component for the operations of a SOA-based solution.

To enable service exchanges, further information (or meta information) on the services has to be made available. Therefore, the service repository is introduced to manage all available services in a landscape. This repository provides information about services, e.g., on interfaces, process logic, input and output, as well as technical information. The service repository includes all information used when exchanging services, whether it is necessary for finding comparable substitutes or adapting a solution to new requirements. As a quasi-standard description language for this information the Universal Description Discovery and Integration (UDDI) is used as defined by OASIS[10]. UDDI is platform independent and extensible. A service bus is used for exchanging data between services.

The Service Bus can operate heterogeneously according to different interface technologies or communication methods. This enablement is an advantage compared to direct interfaces between services. Data is processed and distributed in a service specific sense. This reduces double processing by a large amount. A unified graphical interface (GUI) is used as the central user access point to SOA-based solutions.

Nevertheless, what a service needs to fulfill in detail, and which properties it has to possess is not provided in the definitions.[10] defines a service as "a mechanism to enable access to one or more capabilities, where the access is provided using a prescribed interface and is exercised consistently with constraints and policies as specified by the service description."[11] gives a number of principles for developing services. However, a service could consist of one single, fairly atomic activity or a sequence of activities. A process is built as an order of linked services which could also split and run in parallel, depending on intermediate outcomes. A process has a start state and an end state. The crucial criteria -- which activities are combined to an orchestrated service -- are finally decided by the recycling and reusability approach described in the service definition. The level of detail within a service is called granularity[8]. The granularity of services within a process is not stringent. If a service definition is too small, the number of services increases dramatically within a SOA based solution. On the other hand, if the service definition includes too much, you reduce the exchangeability of a service. The granularity should be chosen according to the functional context of the business processes, they are supporting.

However, a service has to provide and cooperate with the basic elements: Service Bus and Service Repository. Therefore a service has to fulfill – in addition to the business functional requirements -- non-functional requirements resulting from the integration with Service Bus and Service Repository, see figure 1. We will show that these non-functional properties have to be extended by functionalities resulting from the change management related to the operation task to run a service. Certainly, SOA based solutions can be very stable in their service composition. But one of the main purposes of SOA design was to increase the capacity for flexibility and to take advantage of. So let us first clarify what flexibility means in the context of SOA.

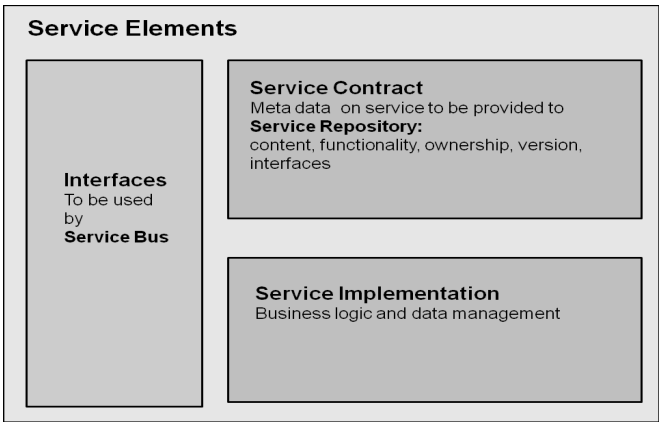

Figure 1. Service Elements

2.2. Flexibility

The concept of flexibility in SOA solutions is the opportunity to exchange and adapt services at runtime. A key requirement is the full encapsulation of services and standardized interfaces. Nevertheless, there is not one common valid definition of the term flexibility.[12] analyzes different definitions of flexibility and their dimensions. Furthermore the authors summarize these dimensions in the context of a company's structure and challenges.[13] distinguishes between flexibility of usage and flexibility to change and adapt an information system.[14] summarizes these different understandings into one unified matrix. As this encompasses all three dimensions, we use the definition of flexibility according to[14]:

- Yielding to pressure, capacity to react, and adapt to external pressure
- Capacity for new situations, summary of opportunities to adapt and react on new requirements as well as customers demands
- Susceptibility, difficulty to modify

SOA-based solutions can potentially excel in all three dimensions because of the opportunity to exchange services during runtime. Software providers focus their products on this capability; see[15] und[16]. Users benefit because of the ability to quickly adapt to new situations and the capability to reuse services[6].

In the following, we analyze what it means for the operations of such a solution if you want to take advantage of flexibility.

3. System Operations Versus Solution Operations

If you exchange a service in a complex SOA environment, certain prerequisites have to be taken into account. Two services can only be exchanged without changing the business process if they have identical business logic, functionality, data, and interfaces. This requirement does not mean that the service needs to come with the same technology or operational requirements. However, input data for the service is not altered with a replacement of the old service. Instead, the new service has to create the same data format. The business logic has to be carefully considered in the new service and requirements on functional (and non-functional) properties have to be fulfilled. In order to substitute a service the other service has to fit into the interface environment as well. It is a prerequisite that the output of the calling service can be transformed by the service bus into the required input for the new service. The output of the new service needs to be transformable by the service bus to create the required input for the follow-up service.

Today, operation tasks are normally processed with the focus on technical components, systems, or databases. Each provider has its own

- system-specific tools
- system-specific maintenance actions and quality criteria
- system-specific knowledge and experience required to use tools, execute actions, and understand quality criteria

In the following, we show that this strategy and the associated methods are not sufficient anymore. A solution includes a number of user scenarios. These scenarios are assigned to certain user requirements. From the technical point of view a solution contains all the technical components such as systems or services and user interfaces which are necessary to fulfill the user requirements. With the evolution from client/server-based solutions to SOA-based solutions, new needs arise to develop a SOA solution including service-oriented administration. As a descriptive example, we present an example in the next section.

3.1. Example: Sales Order Process

We use a virtual, simplified solution of a sales order process to demonstrate the difficulties and challenges inherent in SOA. Event-driven Process Chains (EPCs) are a typical method for graphically modeling business processes[17]. An EPC consists of a sequential order of events, activities or functions, and flows. An EPC is a directed graph that can be interpreted as a colored Petri Net for describing business processes. There exist other graph based models for business processes, e.g., UML activity diagrams. We use EPCs due to the model-driven development at SAP. Note, transformations from different diagrams are possible, e.g., UML activity diagrams and EPCs[18].

The order within an EPC is interpreted as follows: An event is a passive element that describes under which circumstances an activity is activated. A function transforms a business process state into a resulting state. In the case of multiple possible results of a function, a decision node (flow operator) inside the flow is used. Furthermore, the flow can proceed in parallel, that is to say, different functions can be executed at the same time. This method is integrated into the Architecture of Integrated Information Systems (ARIS)[17] which offers a software-provided business process modeling with EPCs as the core element. Figure 2 shows our sales order process as an EPC notation.

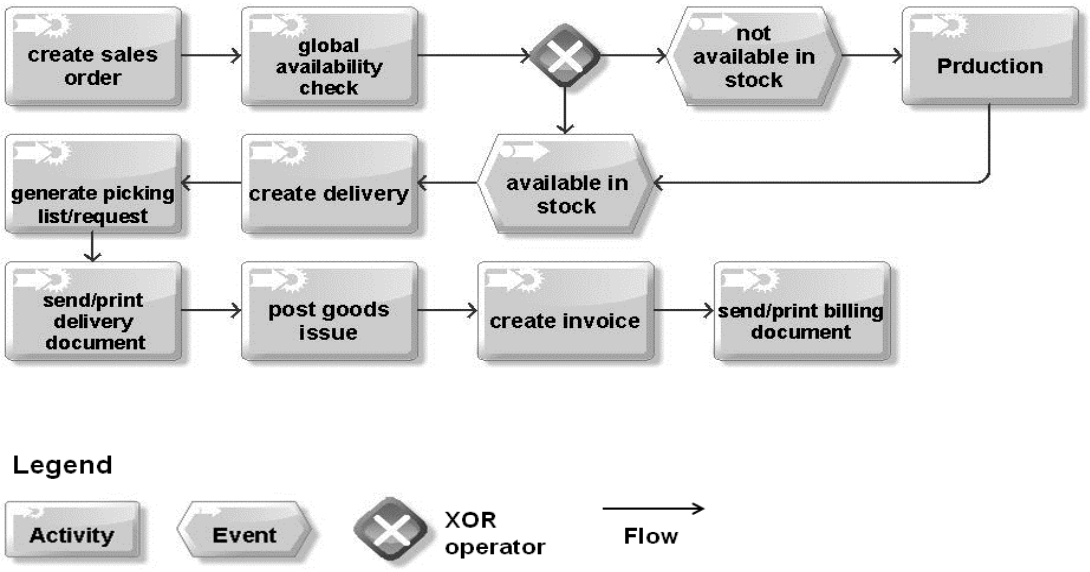

Figure 2. Business Process Flow of Sales Order Process

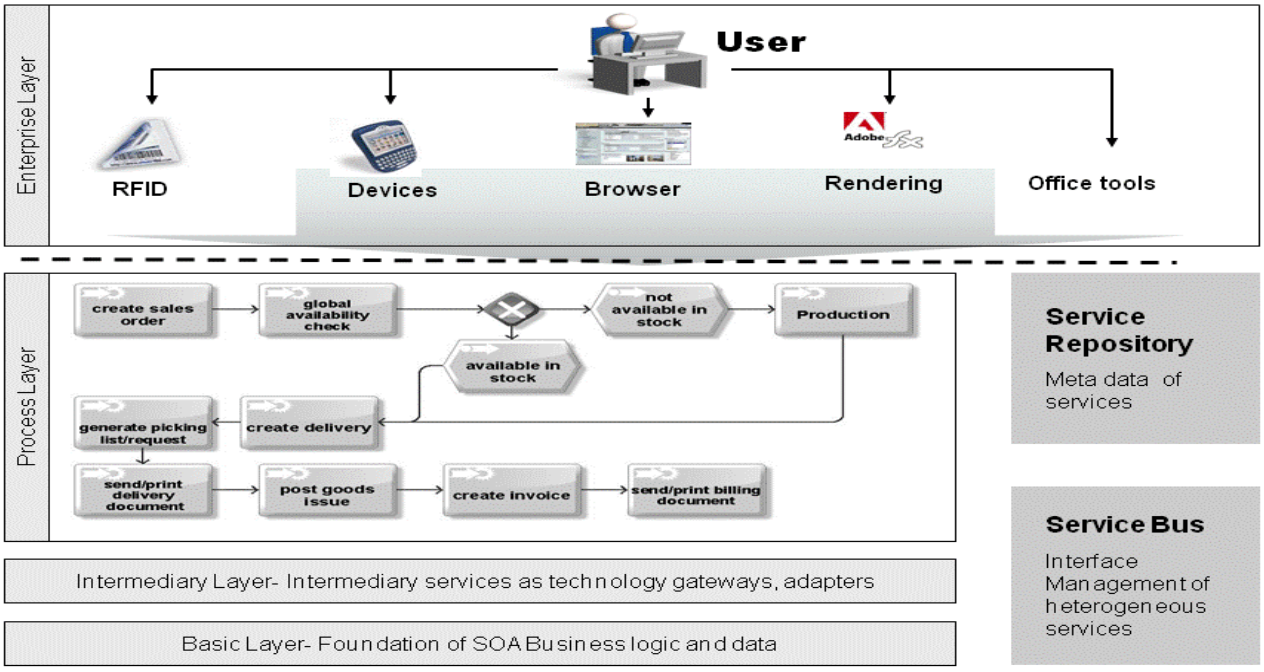

Figure 3. Sales Order Process in a SOA-based solution

Our sales order process starts with the creation of the sales order. The next activity is to check the global availability of the order, whether different stocks have all the order elements available or new production is necessary. If the ordered products are not available they have to be produced. To keep it simple we assume that production is one single service, even though this will not be common in practice. The delivery order can be created either after production or as a result of stock availability. The next activity is the generation of a picking list or a request. In order to complete the delivery, the necessary delivery documents have to be printed. Afterwards, the ordered goods can be sent to the customer. An invoice is created to trigger the payment of the ordered goods. The bill has to be printed and sent to the customer. In the real world, the process is more complex. However, for our purposes, this basic example is sufficient to underline our approach.

3.2. Sales Order Process As A SOA-Based Solution

To keep it simple, in Figure 3 each activity corresponds to a service in the process layer of SOA. The user has in general a unified access tool which is part of the enterprise layer. The service repository includes metadata from all the services involved, information on services which are additionally available and ready to be integrated instead or in addition to the SOA-based solution. Note, this is not of relevance for the users. The service bus is needed for com-

munication and data transfer between the services. Both the service repository and the service bus are services themselves. The intermediary layer consists of the services providing technology adapters, gateways etc. The basic layer bundles all the services which are fundamental for the business logic such as data management or general security services. However, there is no need for end users to interact with services other than the enterprise layer. The user sees the solution as a unit and is not interested in -- or even aware of -- any technical components or services. This decoupling of technical implementation and business functionality is an essential advantage for the user of SOA-based solution.

The users of the sales order process have certain expectations and requirements regarding functionality and non-functional properties such as stability or performance of the business process. The IT organization responsible for the solution landscape is charged with operating the solution in this expected manner. In the next chapter, we show the essential differences to consider if the environment is based on client/server architecture or SOA.

3.3. Differences between Client/Server and SOA-Based Operations Related to Change Management

In client/server architecture there is almost a direct link between technical systems and activities necessary for successful operations, e.g., our sales order process. As a result you can map activities of the sales order process to technical systems as shown in figure 4. In most cases it is sufficient to operate each individual system according to technical indicators and therefore, this is commonly called system administration or system operations. For a SOA-based operations this system-oriented architecture has to develop to a service- and solution-oriented approach. Let us take the example of change management. A change can be required due to two main reasons.

1. planned adaption of solution
2. unplanned modification is necessary due to a former incident

Let us see what happens if we want to take advantage of the flexibility of SOA and want to extend the sales order process. This would be a planned change. There should be an additional option if an ordered product is not available. If the ordered product B is not available in stock it should be bought by another vendor instead. In the former client-server architecture, one has to design and implement etc. this new functionality preferred in the existing systems. Due to the SOA approach let us assume there is a service available by this vendor which can be integrated. Of course the real world is not black and white and it is an evolution and not a revolution to SOA. As a result we could have a mixture of client-server backbone and service-oriented parts in a solution.

Let us focus on the integration of a new service into the existing solution. In addition we assume this new service uses another type of technology which was not used before in the current solution components, either systems or services. Today there are no rules for a common deployment tool technology independent. So each service provider has their own deployment tool, own logging mechanism, own version handling etc. So in our example the change can be deployed, but because of new technology it will come with its own deployment tool. Change management is to be seen in close connection to the day-to-day operations. Under the circumstances described, you will struggle to answer typical questions in SOA operations such as:

- An incident occurs and affects a business process. Was something changed in the solution that could be the cause?
- How to deploy related changes affecting more than one service with different technology?
- How to monitor the overall change history of a SOA-based solution?
- How to integrate a new service technology into the day-to-day operations like solution-based performance, data or throughput management?

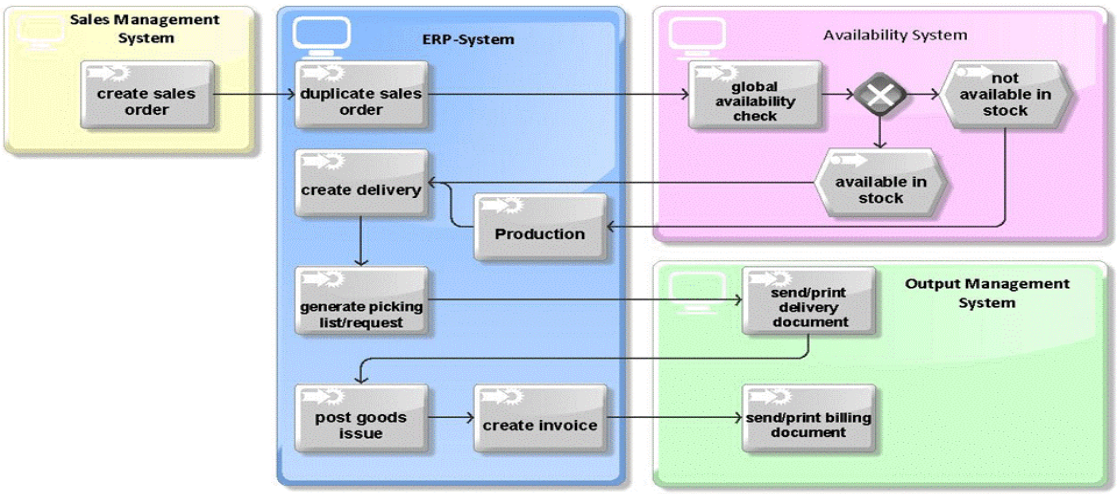

Figure 4. Client/server oriented view on flow of Sales Order Process

To answer these questions a new type of tool is necessary in SOA. In summary for the change management there are three new, main requirements in SOA:

• central control, management, tracking and reporting of change management process (solution oriented approach instead of system oriented)

• solution-oriented deployment of changes (synchronization of changes to different SOA-services)

• SOA-service technology independent deployment of changes

Neither the technical component nor the technical system is the unit to be operated. By the investigation in other ITSM processes we found similar issues and requirements. In addition the ITSM processes are linked and not independent of each other, so that one process can reuse information from another. This is another argument for centralizing the management and execution of ITSM processes. This requires another new type of tool: a process-oriented, central operations cockpit.

3.4. Analysis of the Current Situation

The need for a new type of tool has been recognized by the industry in recent years. Some software providers have developed their own concepts for the operations of SOA-based solutions such as HP OpenView or Quality Center, IBM Tivoli or Rational, or SAP Solution Manager 7.x. Every tool supports monitoring and administration. However, at first, each tool requires an interface. Thereby, each tool has its strengths, especially in the management of provider-owned products. Thus, a reactive integration strategy is common: Even if a service has reached or is at least expected a certain distribution, a necessary interface is provided to integrate service specific techniques, indicators, and their measurements into the operations tools.

It is in the responsibility of the service developer to provide the elements needed. Which methods are utilized for a given service in the operations tool is decided by the provider of a tool. Often the result is the loss of information and control. Furthermore, the interpretation of figures shown or activities on offers usually requires specific knowledge and experience by users of that particular operations tool.

For example, we show the SAP Solution Manager[19] and its advantages and disadvantages of the stated procedure. SAP Solution Manager follows a solution- and lifecycle-based approach. In particular, all available SAP components based on ABAP or Java can be monitored using this tool. This is done by specially developed interfaces that are provided in this environment. However, for each new solution that is integrated, these interfaces have to be provided and consistently developed. In theory, this procedure is also open for non-SAP solutions. As a result of this strategy, for each new service integrated into the SOA environment, a corresponding effort during the development phase is necessary. A consolidated procedure within the SAP Solution Manager, where all data flow together, is challenging. Furthermore, this again requires solution- and provider-specific knowledge.

In Table 1 we present the advantages and challenges currently open within the change management of SOA-based environments with the SAP Solution Manager. Note, this tool-specific example can be applied to other tools as well. Accordingly, SAP Solution Manager provides a suitable improvement in the tasks and activities within Change Management and the other ITIL processes[20] for the SOA environment. However, it still requires a provider-independent, universally usable, central operations tool based on unified rules to operate SOA-based solutions. Regarding the change management process, SOA mainly affects the execution of changes.

Table 1. Advantages and Challenges of SAP Solution Manager according to Change Management

Advantages	Open Challenges
System independent	SAP centric
Support of different SAP deployment techniques (ABAP, Java) via reuse of ABAP-based Correction and Transport Management	No direct integration within the service repository or service bus
Integration opportunity for other deployment	For each service, specific orchestration is required which means effort. Missing linkage to service-specific non-SAP development tools No automatic version control system for non-SAP services
Synchronization of system overlapping changes	Only possible for supported and integrated deployment techniques
Central access control to change protocols of integrated deployment techniques	No standard for protocols. No normalization for content evaluation

In the next section we discuss typical requirements and quality criteria of IT Service Management.

4. It Service Management

Which requirements are necessary come generally from IT service and support management as described in the IT Infrastructure Library (ITIL)[5]. ITIL was first developed by the Central Computer and Telecommunications Agency (CCTA, UK) in 1989. The last update in 2007 is called ITIL III. ITIL describes provider and customer independent "good practices" during the application lifecycle, introducing a concept of a fixed number of IT service management processes necessary for successful IT operations in any organization. In addition, ITIL is used as a specification how the described controls and techniques by COBIT can be established. Note, "service" in the sense of ITIL is different to "service" as a basis for SOA. In the following, we call this type of service 'provider service' as opposed to a 'SOA service' as a part of SOA. ITIL includes all the tasks from an organizational perspective with respect to operation and maintenance of an IT system as a service.

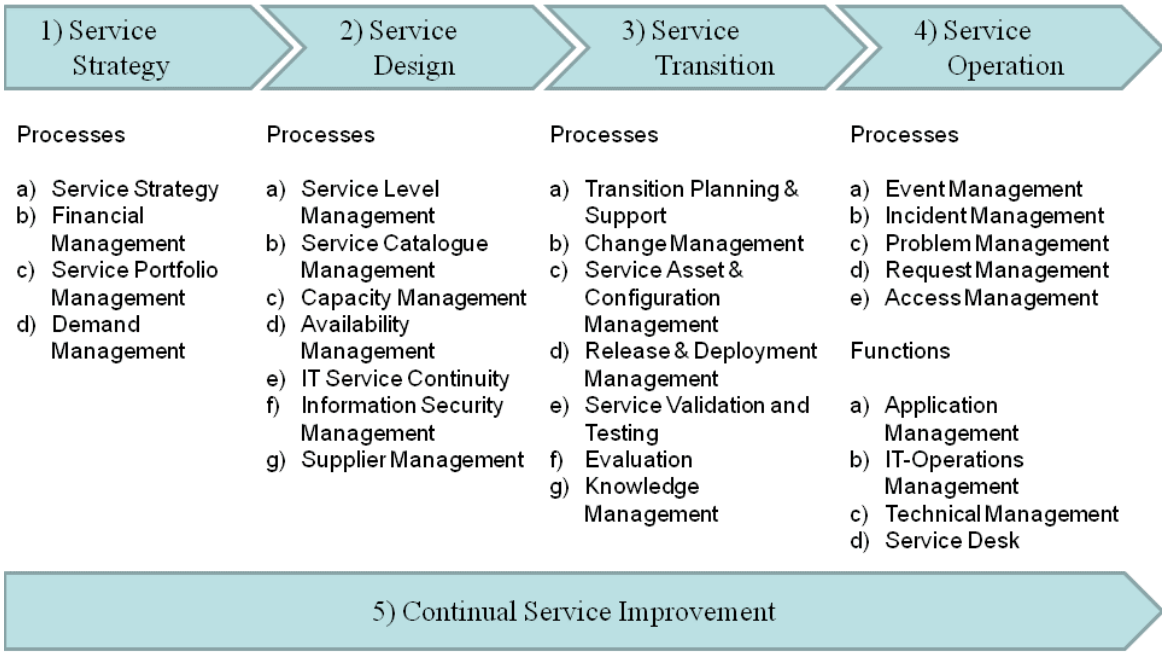

Figure 5. ITIL Overview

"A service is a means of delivering value to customers by facilitating outcomes customers want to achieve without ownership of specific costs and risks"[5]. ITIL considers provider services in their life cycle. The entry point is the strategy of building provider-services, which define target and balance cost and risks. Figure 5 presents the overview on ITIL with the phases approach and the related processes and functions.

The execution of this strategy is planned and designed during the Service Design phase. The Service Transition phase covers all activities as part of the related processes which are necessary to hand over a service to production. The day-to-day operation is part of the Service Operation phase. After the Service Operation phase follows the Continual Service Improvement phase.

4.1. Main Differences in Operations of Client Server and SOA Based Solution

In this paper we analyzed the change management process within the phase Service Transition and how it is affected by specifics of SOA. From the ITIL point of view, a change is every type of change affecting a service by a provider, which includes technical and functional changes to programs, components, configuration, and SOA services. The good practices for the change management process can also be applied to SOA-based solutions. The process still stays the same including its quality criteria. The new challenge is the execution of a change which can consist of more than one single changes affecting more than one SOA services at the same time. This requires considerable synchronization and logging of all related changes that are part of the main change.

By the investigation in the other processes directly or partly related to the operations of a solution we found similar issues.[21] gives an overview and discusses general challenges in operations in the event that you want to take advantage of SOA flexibility. In this paper we demonstrated changes and new requirements for the change management process in SOA.

Table 2 shows the main differences between client/server and SOA-based solution operations.

Table 2. Comparison of the client/server and SOA-based approaches

System administration in client/server-based solutions	Solution administration in SOA-based solutions
Technology- and system-oriented tools	Unified central tools with solution-wide approach
System-oriented activities and terms	Service- and solution-oriented activities and terms
System- and technology-specific knowledge required for usage of tools and execution of activities	Unification of solution and service-oriented administration allows operations without deep knowledge of service-technology specifics

The conclusion is that you need a new type of tool which is independent of service technology and is supported by each service. As shown for the change management if you try to operate a SOA-based solution with the same approach as is common in client/server-based solutions you are:

- discounting the advantages of SOA
- experiencing a rise in the cost of operation
- not fulfilling the common quality criteria of solution operations such as stability, availability, and performance.

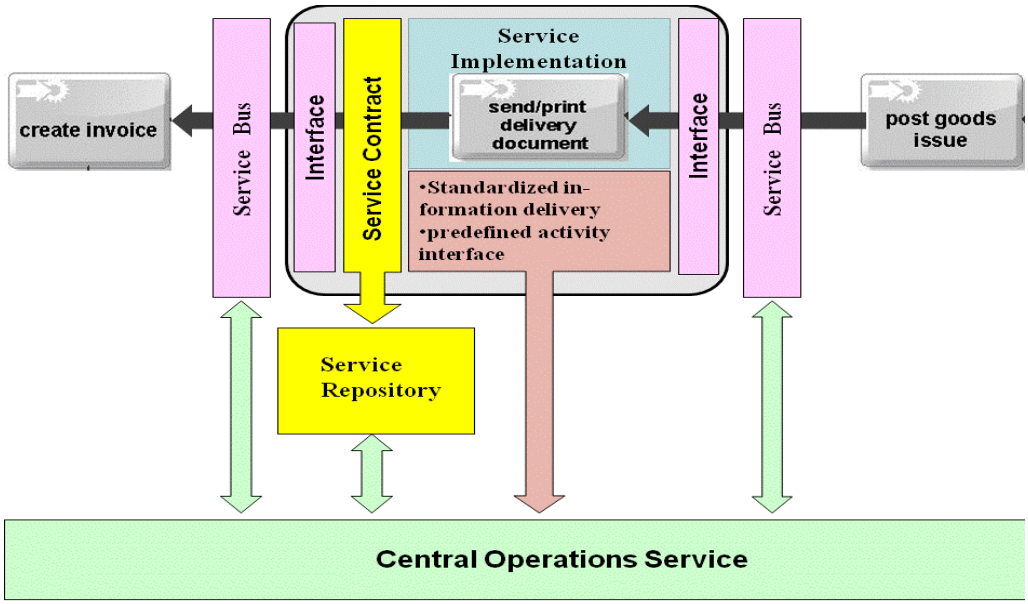

Figure 6. Service Elements to support of Central Operations Service

The change management process is a very important process in the life cycle of a solution. The quality of this process is key for the flexibility of a solution. We demonstrated which new challenges can be expected through implementation of SOA for this process. We also demonstrated that it is insufficient to reuse client/server-based tools and methodologies.

Specification of the central operations cockpit

We investigated for each process of IT Service and Support Management and found similar changes and challenges as we described in this paper for the Change Management. ITIL provides reusable, general valid descriptions of optimized processes in governance. The general approach and the quality criteria stay the same, however, the execution must change from a system- and technology-oriented to a solution- and service-oriented approach. ITIL can be used for SOA[21], however, a adaptation within SOA is necessary. For each solution there have to be defined quality criteria based on services in operations. The loose coupling of services is the most important difference compared to fixed compositions of client/server-based solutions. For using the capability of flexibility in SOA, you need an additional, central component in SOA: a central operations cockpit, e.g., for the management, controlling, and logging of changes. This requires on the one hand the standardization of non-functional properties which need to be provided by each service to be integrated into SOA and to be made operational. This leads to the extension of the service content as shown in figure 6 by non-functional functionality for the fulfillment of operation requirements. This is to support a central operations service. A first approach was already given for the performance management by[22]. The Open Group defined an Application Response Measurement (ARM) 4.0 API which has to be taken into account already during service design. ARM provides predefined API for performance measurement but is does not solve the measurement unit. There is no unification of the performance measurement metric so that the outcome can only be compared with difficulty. On the one hand you need a tool which integrates such services and provides aligned functionalities for the execution of IT Service and Support processes as well as for the operations. In case of ARM a central service description for monitoring and reporting on the measured performance is missing as well. ARM could be seen as a step in the right direction but it is not sufficient neither for the necessary performance management in SOA nor for the operations in general. In summary you need a central operations service in a SOA landscape which runs as a central operations cockpit providing the execution of the ITSM processes. Figure 6 shows how such a central operation service interacts. As a conclusion the service elements as shown in figure 1 have to be extended. Beside the common elements interfaces, service contract and service implementation and the interaction between service repository and service bus each service has to provide standardizes information and an interface for operations activity and monitoring. A standardization and unification is necessary, to reduce the necessary service technology-specific knowledge and experience to a minimum. It is to be expected that for some of the new requirements general adapters can be developed which support every service. Other requirements can only be fulfilled if there are established rules or a type of standard for non-functional properties. In this case, each service has to support these standards, and these properties must already have been factored into the Service Design Phase. To accomplish such a standard within a single company might be easy compared to the challenge of establishing an industry-wide standard incorporating providers like Oracle, IBM, SAP, or Microsoft. New challenges arise if one integrates services from a Cloud into a solution that was

not covered in our investigations.

5. Conclusions

In this paper we demonstrated the challenges and requirements in change management if you take advantage of the flexibility of SOA-based solution. If you do so the reuse of client/server operation tool is not sufficient. Instead of system oriented operations you need a solution oriented operations and a new type of central, solution oriented operations service as central cockpit. As a result the common understanding of SOA elements as well as service elements has to be adapted which means extended.

ACKNOWLEDGMENTS

I would like to thank Gunter Saake for his generous support in preparing this paper. This work is partly supported by the Bundesministerium für Bildung und Forschung (BMBF) within the framework of the project ViERforES-II (Nr. 01IM10002B).

REFERENCES

[1] Beimborn D, Weitzel T. What Are Important Governance and Management Mechanisms to Achieve IT Flexibility in Service-Oriented Architectures (SOA)?: An Empirical Exploration. In Proceeding of 2011 44th Hawaii international conference on system sciences: (HICSS 2011) Kauai, Hawaii, 4-7 January 2011.

[2] Kleiner C, Dunkel J. Establishing Service Management in SOA. International Journal of E-Entrepreneurship and Innovation 2012;3(1):1–17.

[3] Schepers T, Iacob M, van Ecl P. A lifecycle approach to SOA governance. In Proceedings of the 2008 ACM symposium on Applied computing (SAC '08), New York, USA. pp. 1055-1061, 2008.

[4] IT Governance Institute, COBIT Framework (Control Objectives for Information and related Technology), Release 4.1, 2007. Online Available: http://www.isaca.org/cobit.htm, 20.2.2012.

[5] Office of Governance Commerce, IT Infrastructure Library V3. Information Technology Infrastructure Library Version 3 Core, The Stationary Office, London, Great Britain, 2007.

[6] F.A. Cummins, Building the Agile Enterprise: With SOA, BPM and MBM. OMG Press, Burlington, MA, USA, 2009.

[7] Oasis, Reference Model for Service Oriented Architecture. Online Available: http://docs.oasis-open.org/soa-rm/v1.0/, 21.2.2012.

[8] I. Melzer, S. Eberhard, Service-oriented architectures with Web Services. Spektrum Akad. Verl., Heidelberg, Germany, 2010.

[9] D. Krafzig, K. Banke, D. Slama, Enterprise SOA. Service Oriented Architecture Best Practices, Prentice Hall International, Upper Saddle River, NJ, USA, 2005.

[10] Oasis, UDDI Version 3.0.2. UDDI Spec Technical Committee Draft, Online Available: http://uddi.org/pubs/uddi-v3.0.2-20041019.pdf, 16.02.2012.

[11] T. Erl, Service-Oriented Architecture. Concepts, Technology, and Design, 4th edition. Prentice Hall, NJ, USA, 2005.

[12] S. Eicker, A. Nagel, P. M. Schuler, Operationalization of flexibility in business process management, Proceedings of the IADIS International Conference Information Systems 2010. Porto, Portugal, 2012.

[13] J. Gebauer, F. Schober, Information System Flexibility and the Performance of Business processes, 2005. Online Available: http://www.business.illinois.edu/working_papers/papers/05-0112.pdf, 20.2.2012.

[14] J. S. Evans, Strategic Flexibility for high technology manoeuvres. In Journal of Management Studies 28(1); pp. 69-, 1991.

[15] Software AG, Five Steps for Building a Business Case for SOA Governance, 2011. Online Available: http://ebookbrowse.com/buscase-soa-wp-nov08-web-tcm17-45795-pdf-d41781850, 21.2.2012.

[16] D. Sprott, Business Flexibility through SOA, 2011. Online Available: Error! Hyperlink reference not valid.ftp://ftp.software.ibm.com/software/soa/pdf/CBDIWhitepaperBusinessFlexibiliyThroughSOA.pdf, 20.2.2012.

[17] A.-W. Scheer, ARIS - business process modeling. Springer, Berlin, Germany, 1998.

[18] M. Nuettgens, T. Feld, V. Zimmermann, Business Process Modeling with EPC and UML: Transformation or Integration? In: Schader M, Korthaus A, editors. The Unified modeling language: Technical aspects and applications. Heidelburg, New York: Physical-Verlag;, p. 250–61, 1998.

[19] M.-O. Schaefer, M. Melich, SAP Solution Manager, Galileo-Press, Bonn, Germany, 2011.

[20] S. Schoeler, L. Will, SAP IT Service &Application Management: The ITIL-Guide for SAP Operations, Galileo-Press, Bonn, Germany, 2006.

[21] L. Will, Operations requirements in SOA based solutions. In Proceedings of the Fifth IEEE International Conference on Research Challenges in Information Science (RCIS), Gossier, Guadeloupe, France. pp. 1-10, 2011.

[22] The Open Group. Application Response Measurement: ARM 4.0 version 2, Online Available: https://collaboration.opengroup.org/tech/management/arm, 17.4. 2012.