7-1-2013

Enhancing Soa With Service Lifecycle Management - Towards A Functional Reference Model

Michael Max Fischbach
University of St. Gallen, St. Gallen, SG, Switzerland, michael.fischbach@unisg.ch

Thomas Puschmann
University of St. Gallen, St. Gallen, SG, Switzerland, thomas.puschmann@bei-sg.ch

Rainer Alt
University of Leipzig, Leipzig, Sachsen, Germany, rainer.alt@uni-leipzig.de

Enhancing SOA with Service Lifecycle Management – Towards a Functional Reference Model

Fischbach, Michael, University of St. Gallen, Müller-Friedberg Str. 8, 9000 St. Gallen, Switzerland, michael.fischbach@unisg.ch

Puschmann, Thomas, Business Engineering Institute St. Gallen AG, Holzstr. 34, 9004 St. Gallen, Switzerland, thomas.puschmann@bei-sg.ch

Alt, Rainer, University of Leipzig, Grimmaische Strasse 12, 04109 Leipzig, Germany, rainer.alt@uni-leipzig.de

Abstract

Service-orientation aims at decomposing monolithic application systems into services, i.e. functional units that adhere to certain criteria such as standardized interfaces and the ability to be flexibly combined with each other. With the growing diffusion of this paradigm, the management of an increasing number of services along their lifecycle (Service Lifecycle Management - SLM) is becoming a success factor. Although comprehensive IT support is not the only ingredient for successfully managing services, it is a contribution to compensate for the growing complexity inherent in any SOA. Surprisingly, the topic of application support in SLM has not yet received systematic coverage in the literature from a functional perspective. This paper proposes a functional reference model for SLM and describes the underlying design process. The model supports practitioners in analyzing, designing and implementing software support for SLM. It further enables to compare and evaluate software solutions and as such it supports IT investment decisions. Scientifically the model represents an approach towards designing information systems in the area of SLM. The paper argues that companies should pursue a best-of-breed approach, as there is no single solution available that comprehensively supports the entire lifecycle. Further, the lack of application support of SLM in practice mainly stems from the absence of integrated solutions and missing knowledge on how to evaluate potential applications.

Keywords: Service Management, Service Science, Service-oriented Architecture, SOA, Service Lifecycle Management, SLM Applications

1 Introduction

1.1 IT-support for Service Lifecycle Management

Service-oriented architectures (SOA) have been discussed as a major architectural pattern in literature and practice since many years. Grounding in software development, SOA primarily aims at decomposing monolithic application systems into modular services with standardized interfaces. These are expected to enhance the efficiency of (re)configuring business processes when business requirements are changing. For example, the multinational bank Credit Suisse (CS) maintains thousands of software modules or technical services and achieves an average re-use factor of four. It is reported that only by decomposing their former monolithic mainframe application landscape into services, CS was able to achieve a Net Present Value of 9mn Swiss Francs over five years. With a growing maturity and diffusion of SOA, companies are faced with the coordination of an increasing number of services. Some challenges in this respect include heterogeneous service specifications, development processes and lifecycle lengths. This paper argues that a comprehensive Service Lifecycle Management (SLM) approach consisting of strategic guidelines, processes, role models and supporting IT applications is needed to meet this challenge.

Strategic and process considerations have already been discussed in practice and academic literature, but applications that aim at supporting the entire service lifecycle have not yet received attention in research. In the first place a model would be required that structures the functionalities of such an SLM application environment. Although application support is not the sole ingredient for successfully managing services (others are e.g. service definitions, organizational embedment, etc.), it may nevertheless be regarded as a key enabler for dealing with the underlying complexity[1]. For example, CS operates a custom-built SLM application that supports SLM tasks, such as release management, re-use identification and service versioning. This application compensates the additional coordination effort that emerges from SOA, e.g. the identification of re-use opportunities, which is an information-intensive task and is often performed manually.

On the other hand the market for SLM software applications is diverse and fragmented. Many different application types may be utilized within SLM, e.g. *Enterprise Service Buses* and *Message Brokers*. They may be separated into two general classes: business-focused and technical applications. The former are rather concerned with business issues, such as costing and pricing. The latter comprise solutions, such as service repositories and monitoring applications and focus on the management of technical services. The vast diversity of potentially relevant applications and the varying functional scopes of similar applications from different vendors pose great challenges on both, (potential) user companies and SLM software vendors, with respect to application architecture- design and evaluation for future SLM architectures.

1.2 Research question and potentials

This research pursues the question "*which software-based support possibilities exist in the field of SLM and how can they be structured*?" The result is a functional reference model for the domain of SLM applications. Several basic requirements emerge from prior research on reference models. Becker et al. (2004) and Fettke & Loos (2007) postulate its applicability in multiple contexts. Specifically, a reference model should be generally applicable within a certain domain (in this case SLM) and should offer a predefined structure that helps in solving problems (Rosemann & Schütte 1999). To construct and evaluate the model, the paper follows the principles of Design Science Research (DSR) (Peffers et al. 2008; Hevner et al. 2004), which aim at designing scientifically rigorous artifacts that help solving practical problems (March & Smith 1995). The DSR approach embeds real-world cases as a means to inductively derive the artifact and to deductively show its applicability. The reference model provides the following benefits:

[1] For example, (Bullinger et al. 2003) state empirical evidence to have shown that information technology fosters the establishment of new procedures and methods in the area of service development. At the same time they find that IT support in this regard is still largely underdeveloped and hence regard the topic as one of future's outstanding challenges.

- It supports *practitioners* in analyzing their company's SLM application architecture, evaluating and comparing available software solutions and defining the scope for proprietary application development efforts. Further, it helps software providers to define the functions of their solutions and to align SLM solutions that are part of the same software line.
- For *researchers* the generic and abstracting nature of the model serves as a "theory for designing" information systems (Gregor 2006). For example, it might be the basis for deriving a service-oriented reference architecture for SLM. As the first comprehensive, lifecycle-wide model for software support in SLM it constitutes as a map for aligning future research activities

After a review of relevant literature in section 2 the functional reference model is introduced with an emphasis on both the design process and the model itself (sections 3.1 and 3.2 and 3.3). A case study in a Swiss universal bank illustrates one possible form of artifact application (section 3.4). Section 3.5 discussed the findings. The paper concludes with a short summary and an outlook on future research (section 4).

2 Approaches to Service Lifecycle Management

SOA is "a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains" (MacKenzie et al. 2006). These capabilities are encapsulated in services. A multitude of different service definitions emerged (Bardhan et al. 2010), ranging from services as a standardized software artifacts (e.g. in the form of webservices (Erl 2007; Baida 2006; Lawler & Howell-Barber 2007)), to the notion of services as encapsulated business functionalities exchanged between humans agents (Bardhan et al. 2010). Unfortunately, the term SLM is also not clearly defined. Subsequently, it is understood as the collective label for the theoretical as well as practical models and methods to manage service elements in a competitive environment along their complete lifecycle (Gummesson 1994). In this manner the anticipated approach focuses on the management of technical as well as semi-technical services from both, a business-oriented and a technical perspective. A clear dichotomy of existing approaches between technical and business-oriented approaches can be identified.

Existing SLM approaches

Technical approaches mostly focus on the management of web services and necessary technical infrastructure. In this notion, Behara & Inaganti (2007) define „SOA management" as "the Management and Monitoring of applications, services, processes, middleware, infrastructure, and software tools in accordance with the business goals". Berbner et al. (2005) propose a management approach for SOA-based application systems, while they essentially restrict to the description of a corresponding software architecture. In contrast, Behara & Inaganti (2007) focus on the identification and explanation of different SOA management layers, namely "overall SOA management", "process management", "service management", "middleware management" and "infrastructure management". Another class of approaches belongs to the area of ITSM (IT Service Management), which refers to the management of IT services such as workplaces and application management. An example is the *IT Infrastructure Library* (ITIL) (Ogc 2007). Contrary to the aforementioned SOA management field, ITSM does not focus on fine-granular, purely technical services, but rather on IT-intensive, though usually not fully automated services.

In the past years, more *business-oriented* approaches emerged from scientific disciplines, such as marketing, product management and engineering. Especially in the latter the term Service Engineering was coined (Bullinger & Scheer 2005) which aims at transferring methods and models from industrial product development and management to the services area. Similarly, IBM introduced the term *Service Science, Management and Engineering* (Maglio & Spohrer 2008), which aims at achieving interdisciplinarity. For instance, Kohlborn et al. (2009) analyze different portfolio management techniques and transfer their findings to the field of Service Portfolio Management. Further, several results from the area of marketing (Lusch & Vargo 2006) also belong to the business-oriented class, because they frequently aim at a systematization of sales efforts and have no focus on IT-related aspects.

In summary, the mentioned contributions tend to a) either focus on technical or on non-technical services, b) either cover technical (e.g. interface definition, technical composition) or business-related tasks (e.g. pricing, costing) activities. An SLM approach, as to the scope of this paper, should target on the management of different service types (see Kohlmann & Alt (2007)) from both, a technical and a business oriented perspective. The functional reference model for SLM developed in this paper has to consider these criteria.

Literature on Software Support in SLM

All mentioned approaches have in common that there is no comprehensive guidance as to how they might be supported by software applications. ITIL recently launched the so-called *Software Scheme*, an assessment procedure to certify software solutions ITIL compliant. Two problems remain with this: first, it is commercial and thus not completely accessible; second, it heavily relies on metrics like *number of current users in practice* rather than focusing on a detailed functional level. Berbner et al. (2005) outline the architecture of a management system for SOA-based applications. Apart from descriptions of the different components, they do not provide comprehensive functional assessments of these components. Several works focus on small aspects of software support in SLM (e.g. related to requirements engineering, see chapter 3). Apart from these, no systematic scientific or practical model for application support in SLM is available that provides a detailed overview on a functional level. This paper proposes a functional reference model for application support in SLM. The proceeding follows Hevner et al. (2004) who suggest analyzing and generalizing existing instantiations.

3 Designing a Reference Model for SLM

3.1 Reference model design

Reference models in the IS domain may be distinguished with respect to *what* and *how* they describe. ARIS, one representative of a reference model for the *Architecture for Integrated Information Systems*, distinguishes four descriptive views (organizational, control, functional, data) and three descriptive layers (business design, technical design, implementation) (Scheer 1992; Scheer 1997; Scheer et al. 2005). The functional reference model in this article focuses on the business design of the functional layer, i.e. it derives and describes SLM supporting functionalities from a business point of view rather than providing e.g. technical implementations of certain functionalities.

As stated in section 1.2, the construction and evaluation of the reference model developed in this paper follows the principles of the *Design Science Research Methodology* proposed by Peffers et al. (2008) and the corresponding guidelines of Hevner et al. (2004)[2]. Additionally, the DSR approach used for the development of this reference model embedded several case studies in order to design the artifact, show its practical applicability and validate it. As such, the case studies address the “design as a search process” and the “design evaluation” guidelines of (Hevner et al. 2004). In this paper, the case study at CS has been conducted by two researchers in collaboration with two CS employees. Thereby, several telephone interviews and further publicly available sources (presentations, books) have served as information sources for the determination of CS’s SLM application functionality. Building on this, the researchers leveraged the functional reference model to analyze the obtained results. In order to bring DSR into action, Peffers suggests a design process with multiple iterations between design and evaluation. It is implemented by means of the consortium research approach described in Österle & Otto (2010), which differentiates between analysis, design, evaluation and diffusion. This process has been instantiated within the consortium research project “Sourcing in the Financial Industry” (CC Sourcing**)** at the Universities of Leipzig, St. Gallen and Zurich, where researchers collaborate with currently 18 partner companies, and follows the four phases from Österle & Otto (2010). The DSR approach embeds several case studies in order to a) design the artifact and b) show its practical applicability and validate it. As such, the case studies address a) the “design as a search process” and b) the “design evaluation” guidelines of Hevner et al. (2004).

In a *first step*, the *analysis* phase, problem identification and subsequent definition of objectives took place. Between November 2010 and January 2011 the problem has been derived from practitioners’ needs. The

[2] Hevner et. al. state seven guidelines: design as an artifact, problem relevance, design evaluation, research contributions, research rigor, design as a search and communication of research.

SLM application market is rather fragmented. Many different solutions exist, each of which may considerably vary in functionality. Practitioners from the CC Sourcing community reportedly are not able to identify and evaluate suitable SLM solutions, due to a lack of means to compare these and to estimate the likely benefits each solution would bring in terms of process support. Hence, the objective was to construct an applicable artifact that addresses these problems.

The *second step*, lasting from February to July 2012 encompassed reference model *design*. This phase follows generally accepted design principles stated by Becker et al. (2002); Fettke & Loos (2004). The models' structure is based on the ARIS convention, which postulates a hierarchical structuring of functions (Scheer 2001). Specifically, the model clusters function sub-groups into function groups and functions into function sub-groups. The function groups and function sub-groups have been derived in four sub-steps:

- *Application and practice case identification*: more than 80 solutions have been identified by web searching, analysts' talks with Gartner and by Focus Groups C and H (see Table 3 in the appendix). Applications were selected if they a) have self-reported focus on SLM or b) were suggested by a consortium member or c) if they were regarded as having connections to SLM (based on literature links). Indicative functional analyses led to seven application categories with 26 sub-categories. Following the suggestion of Kawaguchi et al. (2004), the main criterion for categorization was functional coherence within the categories and as much orthogonality as possible between the categories. The application grouping allows selecting one application from each sub-category for an in-depth analysis. This in turn ensures broader function coverage compared to a random selection without prior grouping. Table 1 provides an overview.
- *Functional analysis and function derivation*: a detailed functional analysis of each application by means of demo versions, white papers, presentations, telephone interviews with a Gartner analyst[3] and six further bilateral interviews (at Entris Banking and Credit Suisse) yielded 35 function sub-groups with 150 functions. The function list has been cross-checked with literature contributions, especially (Raverdi 2008; Treiber et al. 2008; Yelmo et al. 2007; Weiss Ferreira Chaves et al. 2006; Papazoglou & Heuvel 2006; Casati et al. 2003) to ensure bottom-up consistency to existing literature.
- *Model structuring*: the functions are categorized into seven different function groups. Strong functional coherence within groups and weak coupling between groups was the main criterion for categorization (Kawaguchi et al. 2004).
- *Process mapping (section 3.3)*: each function is linked to the SLM process phases it supports. Links have been derived from three sources: first, the applications' documentations have been screened. Second, group discussions at focus group meetings were utilized to discuss the support potentials of each function. Third, corresponding information from the talks to Gartner and further bilateral interviews in the consortium members' organizations was extracted.

The *third step* comprised demonstrating the artifacts' applicability and *evaluating* it. The model has been applied to real-world case studies with the Swiss bank Credit Suisse (see section 3.4), Entris Banking and Bank Julius Baer in Switzerland as well as Degussa Bank in Germany. Additionally, functional analyses of further software solutions were performed using the reference model. These cases were presented and discussed in focus groups E and F. As suggested by Simon (1998) and Hevner et al. (2004), steps two and three have been iteratively performed in order to improve and refine the model.

The *fourth step* comprised *diffusion* activities in science and practice (Peffers et al. 2008). A working paper and several presentations served to communicate this model to practitioners, while the paper at hand describes the construction process in order to make it accessible for repeated use and verification.

Application category	Application subcategory and examined solution
Technical SOA	*ESBs*: Oracle Service Bus for Financial Services; *Registries*: Membrane SOA Registry; *Repositories*: IBM Websphere SR and Repository; *Monitors*: Predic8 Membrane SOAP Monitor; *SOA Governance*: TIBCO Active Matrix Governance Framework; *Wrappers*: WSO2 DataServices Server; *Mashups*: WSO2 Mashup Server; *Message Brokers*: WSO2 Message Broker; *Architecture Management*: ARIS SOA Architect
Application Management	*A. Lifecycle Management*: SmartBear ALM Complete; *A. Performance Management*: Compuware Gomez
IT Service Management	*ITSM*: BMC Software IT Service Management Suite; *IT Infrastructure Monitoring*: Nagios XI

[3] Michael Dunne, former analyst at Gartner.

Business Logic	*Business Rules Engines*: IBM Websphere Operational Decision Management; *Rules Repositories*: OpenRules Rules Repository; *Ontology Modeling*: Protégé; *Reasoners*: Clark & Parsia Pellet
Processes	*Process Repository*: WSO2 Business Process Server; *Business Process Management*: IBM Business Process Manager
Business Information	*Datawarehouse*: Information Data Warehousing; *Business Intelligence*: Information Builder WebFOCUS; *Business Activity Monitoring*: WSO2 Business Activity Monitor; *Complex Event Processing*: TIBCO Business Events
Service Configuration	*Service Configurators*: jCatalog Product and Service Configurator; *Configuration/Pricing/Quoting*: Camilion Product Authority; *Pricing Engines*: Zafin Labs miPricing

Table 1: Application (Sub-) categories according to first functional analysis and analyzed solutions.

3.2 Elements of a functional SLM reference model

The functional reference model for SLM comprises 150 functions, 35 function sub-groups and seven function groups. Figure 1 illustrates the model's function groups and the respective sub-function groups. The following list introduces each function group, while Table 4 in the appendix provides a complete function listing.

- *Service Description*: Service descriptions are a necessary prerequisite for SLM, as they contain important functional and non-functional information about services. Functions in this group help creating, maintaining and analyzing service descriptions.
- *Service Modeling and Design*: one of the most important stages in a services' lifecycle, as most features are determined here (Krug et al. 2010). These functions support activities ranging from business- and technical requirements management to implementation and testing of services or parts thereof.
- *Service Valuation*: These functions support in taking valuation-related decisions throughout the whole lifecycle. Example include pricing and costing functions that help determining the (anticipated) service cost and deriving pricing schemes and expected revenues.
- *Service Exchange and Integration*: Especially in inter-organizational service networks the provision of standardized means to integrate partners in order to exchange services is an important task of SLM. Functions in this group cover service discovery, -contracting and -exchange.
- *Service Monitoring, Analysis and Reporting*: Monitoring and analysis of services throughout their lifecycle is the basis for corrective actions or other dispositive management decisions. Examples are policy conformance monitoring and Service Level Agreement (SLA) monitoring.
- *Process Management*: Services support processes and are themselves the result of processes. Consequently process management plays an important role in SLM. Functions address the complete process lifecycle.
- *Cross Functions*: Functions that would either fit into more than one or none of the other groups. SLM integration for example concerns with the integration of SLM-related applications. Thus it does not address a certain aspect of SLM itself but rather can be seen as an enabler for SLM in general.

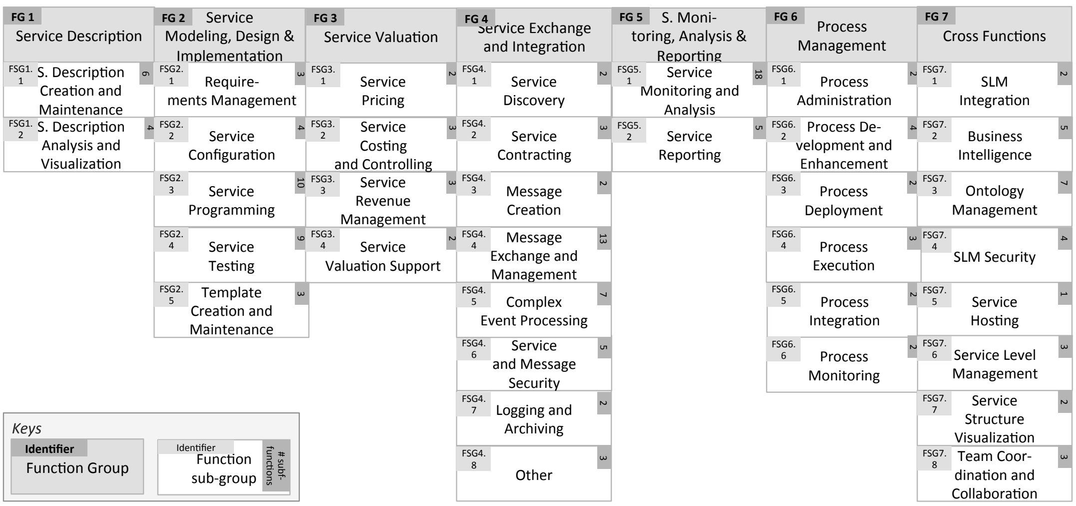

Figure 1: Function groups (gray) and contained function sub-groups.

3.3 Process mapping

Functions support different activities within SLM. In order to specify this link, a two-step procedure is followed. In a first step a generic SLM process is derived from existing approaches SLM approaches, e.g. ITIL, Behara & Inaganti (2007), Brown et al. (2006) and the CS case (see section 3.4). Consolidation and mapping led to the seven process phases 'Identification', 'Requirements analysis', 'Conception', 'Development', 'Implementation', 'Operation' and 'Enhancement'. [4] Step two mapped the identified functions to the process's phases.

Due to space constraints this article simplifies in two respects: first, rather than indicating the SLM process support capabilities for each function, an overall indication of the function sub-groups' support is depicted; second, rather than mapping the functions to activities within the SLM process, an indicative aggregated mapping between functions and process phases is depicted.

Function Group	Function sub-group	Identification	Req. Analysis	Conception	Development	Implementation	Operation	Enhancement
1 Service Description	1.1 S. Description Creation and Maintenance	■	■	■	■	■	■	■
	1.2 S. Description Analysis and Visualization	■	■	■	■	■	■	■
2 Service Modeling, Design & Implementation	2.1 Requirements Management		■					■
	2.2 Service Configuration			■	■	■		■
	2.3 Service Programming					■		■
	2.4 Service Testing				■	■		■
	2.5 Template Creation and Maintenance				■	■		
3 Service Valuation	3.1 Service Pricing			■	■		■	■
	3.2 Service Costing and Controlling			■	■		■	■
	3.3 Service Revenue Management						■	
	3.4 Service Valuation Support			■	■		■	■
4 Service Exchange and Integration	4.1 Service Discovery			■	■		■	
	4.2 Service Contracting						■	
	4.3 Message Creation						■	
	4.4 Message Exchange and Management						■	
	4.5 Complex Event Processing						■	
	4.6 Service and Message Security			■	■		■	
	4.7 Logging and Archiving						■	
	4.8 Other						■	
5 Service Mon., An. & Rep.	5.1 Service Monitoring & Analysis						■	
	5.2 Service Reporting						■	
6 Process Management	6.1 Process Administration			■	■	■	■	■
	6.2 Process Development and Enahncement				■			■
	6.3 Process Deployment					■		
	6.4 Process Execution						■	
	6.5 Process Integration				■	■		
	6.6 Process Monitoring						■	
7 Cross Functions	7.1 SLM Integration	■	■	■	■	■	■	■
	7.2 Business Intelligence	■	■	■	■		■	■
	7.3 Ontology Management			■	■	■	■	■
	7.4 SLM Security	■	■	■	■	■	■	■
	7.5 Service Hosting						■	
	7.6 Service Level Management	■	■	■	■	■	■	■
	7.7 Service Structure Visualization	■		■	■	■	■	■
	7.8 Team Coordination and Collaboration	■	■	■	■	■	■	■

Table 2: Mapping between functions and SLM process phases.

Subsequently, each process phase is explained and exemplary details on how the specific functions support the activities within each process phase are provided. Table 2 shows the complete mapping between the model's functions and supported SLM phases.

- *Identification*: The identification of potential services. Activities include market analyses, idea generation & prioritization and feasibility assessments. For example, service structure visualization functions help identifying dependencies between services (service users and suppliers, both technical and non technical). Standard solutions in this area are scarce. Service repositories, e.g. *IBM Websphere SR&R*, often provide flat-table visualizations.
- *Requirements analysis*: Derivation, analysis, filtering and evaluation of business and technical requirements. It specifies the ideas from the previous phase and is the foundation for service conception. The function 'Requirements Management' supports the process by storing, tracing (i.e. change tracking) and valuing requirements. The *Open Source Requirements Management Tool* is an exemplary application focusing on this function.
- *Conception*: Provision of a concept for the anticipated service, including business case, operational design, functional design, indicative *Service-Level Agreement* with pilot customers, application mappings and an assessment of re-use potentials and make-or-buy decisions. The concept is the basis for service development. For example, business case generation is supported by revenue simulation- and valuation criteria management functionality. Service Discovery and Service Configuration functions assist in service re-use identification. Especially in the area of pricing standard solutions exist, e.g. *miPricing* by *ZafinLabs*.

[4] The process has been discussed and refined in Focus Groups A, B and G. Activities have been detailed in Focus Group D.

- *Development*: Creation of the service, including programming of code artifacts, process implementations, organizational integration of network partners, test planning and functional testing. Service template functions accelerate the development process and foster standardization. For example, Credit Suisse built proprietary templating functions (see 3.4). Various *Webservice Development Kit*s provide a development environment for webservice creation, e.g. *JWSDP*.
- *Implementation*: Integration into runtime systems and setting live. Includes activities such as user documentation, user trainings, infrastructure setup, integration tests, data migrations, portfolio integration and finally roll-out of the new service. Different *Extraction, Transformation and Loading* (ETL) tools, e.g. by *TRANSOFT*, provide data migration functions. The popular tool *soapUI* supports webservice testing, such as test case specification and test result analysis.
- *Operation*: Ensuring smooth service operation, including incident-, security-, continuity-, capacity-, configuration- and availability management. Further comprises activities with regard to service-controlling, -billing, -demand planning and -competition analyses. The operation phase is targeted by many of the examined solutions. An important function sub-group is *Monitoring & Analysis*, which includes functions like SLA compliance monitoring and application monitoring. For example *SmartBear ALM Complete*, an application lifecycle management solution, includes the latter.
- *Enhancement*: Further development of existing services. Activities comprise continual process improvements, regulatory adaptations, releasing, technological changes and service disposal. This phase is supported by functions of almost all groups, except *Service Exchange and Integration*, as it comprises activities from all other process phases. Especially important is service versioning (function of function subgroup 1.1), as it keeps track of different service versions and related compatibility issues. Service versioning includes versioning of descriptions, interfaces, code and endpoints.

3.4 Application of reference model at Credit Suisse

CS is a Swiss-based multi-national universal bank. Since more than 10 years, CS is operating and enhancing a technical SOA. Services in their understanding are fully automated, technical services. SOA is instantiated by CORBA- and increasingly webservices. Consequently, SLM at CS is situated within the IT-department. CS operates over thousand services with an average re-use factor of four. In order to deal with the resulting complexity, a systematic SLM process is in place, comprising the phases 'Requirements management', 'Specification & Design', 'Implementation' and 'Use'. Each phase is terminated by a comprehensive review process called *Stage Gate*. To support the process (and enable SOA in general), CS operates two main applications: the Credit Suisse Exchange Bus (CEXB) and the Interface Management System (IFMS), which will subsequently be analyzed. While the former mainly offers functionality for communication between services, i.e. run-time support, the latter also includes design-time aspects like data type maintenance, SLA templates and code generators. The case study has been conducted by two researchers in collaboration with two CS employees. Thereby, several telephone interviews and further publicly available sources (presentations, books) have served as information sources for the determination of CS's application functionality (and providing software solutions) in the area of SLM. Building on this, the researchers leveraged the functional reference model to analyze the obtained results. Due to space restrictions, only the two main findings will be discussed:

- CS' SLM solutions do not yet offer service valuation support. On the pricing and revenue side this may be attributed to the internal orientation of CS' SOA. Calculation of competitive market prices and resulting revenue plans and simulations are not necessary, as there is no competition or external customer. Regarding costing, the IFMS system already allows to attach basic cost-related information to a service description. However, this feature is not yet utilized and there is no further integration with costing systems. In summary, valuation support at CS is scarce yet and might be – at least on the costing side – an opportunity for future investment.
- 2) CS is forced to stick to a *Best-of-Breed* strategy or to develop in-house. The IFMS system is completely developed in-house. CS states that there is no single standard software available that offers the required set of functions without containing too many unneeded functions. A reverse analysis using the presented reference model confirms this notion. In order to resemble the functionality of the IFMS application, CS

would have had to buy several of the solutions mentioned in section 3.1, which would have left CS with many unneeded and redundant functions.

3.5 Findings during model construction, evaluation and application

An analysis of the findings of this research shows five major results:

- *Lack of an integrated service management application architecture:* Although approaches, such as e.g. ITIL provides indications on adequate software support of its processes, no systematic treatment, e.g. in the form of a functional reference model, has yet been developed in academia or practice.
- *Dichotomy of technical and business services resp. aspects of services:* Service management applications exhibit a dichotomy with respect to the addressed service types. Most of them solely focus on technical services, e.g. *Enterprise Service Buses*, while none provides an integrative view on both business- and IT-services (resp. business- and IT-*aspects* of services). Hence it is not surprising that a similar dichotomy prevails in companies; especially in the area of service description companies still fall short of covering both business- and IT-aspects.
- *Low service management application usage*: The inductive case studies showed that software support adoption is underdeveloped, mainly due to a lack of knowledge about possibilities. Many tasks are still manually performed, such as SLA management in the case of Entris Banking.
- *Focus on best-of-breed SLM application architecture*: Currently no single software solution covers all relevant functions throughout a service's lifecycle. Rather, the landscape is fragmented with solutions concentrating on a certain set of functionalities, such as message exchange functionalities for runtime support (e.g. *WSO2 Message Broker*) and process management suites (e.g. *Bonita Open Solution*). Companies are thus forced to use best-of-breed architectures for their service SLM, which is not unproblematic. For example, during service design activities business process management reports might provide valuable input. However, today users must either procure the adequate reports from the respective systems or build a custom-tailored application that integrates service design and process management functions.
- *Unequal functional support along the service lifecycle*: Assuming that a) function granularity is homogeneous and b) the examined applications constitute an unbiased sample from the population (both ensured in the Design phase, see section 3.1), Table 2 reveals that the operation phase is more supported by contemporary applications than any other phase in the lifecycle. This may be attributed to the initially technical SOA management understanding that focused on exchanging services during runtime rather than addressing the entire service lifecycle (see e.g. Erl (2005)).

4 Conclusion and Outlook

SLM has been presented as an approach that is critical for compensating for the additional complexity and coordination requirements in any SOA. From a management point of view, five requirements can be identified (Fischbach et al. 2013): 1) *Service description*: The description of is the basis of many management-related activities along the service lifecycle. 2) *Value orientation*: SLM decisions should account for its likely effects on service costs and revenues. This necessitates a) a suitable approach for service-centric costing and knowledge on the exact dependencies between services. 3) *Inter-organizational perspective*: increasing decentralization of service provision requires an inter-organizational focus of the SLM approach. This includes clear definition of the roles each player assumes and how the SLM activities are distributed among these roles. 4) *Portfolio view*: the SLM process as presented before addresses single services (and their sub-services). A portfolio view manages the entirety of services and their relationships. This includes the avoidance of redundancies, among others.

The fifth requirement, the application support, helps to manage services according to these requirements. This paper presents a functional reference model for the domain of SLM, aiming to elaborate software application support possibilities in SLM. The reference model contributes to research in being the first comprehensive, lifecycle-wide means for analyzing IT support in service management from a functional viewpoint. As such, it can serve as a basis for the development of more specialized models, e.g. models that help designing appropriate IT support for different roles within SLM (e.g. service developers, see below). The findings from section 3.5 have several implications for companies and software vendors. First, companies aiming at lifecycle-wide, comprehensive application support in SLM presently need to either stick to a *Best-of-Breed* approach or develop their own application. Software vendors should focus on integrated solutions. The requirement of a lifecycle-wide management of services, both from a technical and business viewpoint, is not yet reflected by today's solutions.

The research process and the resulting artifact feature limitations that ought to be considered during future research activities. First, part of the artifact's designing had to deal with extracting tacit practitioner's knowledge and converting it into explicit, reproducible knowledge (Österle & Otto 2010). This conversion leaves room for subjectivity, which can be reduced by more cases. Second, it is not possible to implement a totally objective rating mechanism during the evaluation of functional coverage in the CS case study. The employed strategy, i.e. to take the ratio of covered to total functions relies on the assumption that all functions are equally important and exhibit the exact same granularity, which is hard to prove. And third, inherent to the reference model is a logical cut as to which functionality to include and which to exclude. Although the selection criteria ensured inclusion of all main functionalities, the reference model's completeness may be insufficient in certain, specialized scenarios.

Partially drawing from these limitations, research should focus in two directions. First, additional cases may further validate the model. The second research need aims at embedding the model into an architecture that considers strategic and organizational aspects in addition to the currently incorporated process mapping. Specifically, mapping the SLM process activities to roles within a generic (intra- and inter-organizational) role model (see e.g. Puschmann & Alt (2011)) would enable discussions on how different business models in the area of SLM could best be supported by software applications. This in turn would allow for decisions on suitable software investments dependent on the pursued business model.

Appendix

<table>
<tr><th>Focus Group</th><th>Duration (min)</th><th>Date</th><th>Main objectives</th><th>Partici-pants</th></tr>
<tr><td>A</td><td>135</td><td>8th Oct. 10</td><td>Problem identification SLM process derivation</td><td>30</td></tr>
<tr><td>B</td><td>285</td><td>24th Feb 11</td><td>SLM process evaluation</td><td>32</td></tr>
<tr><td>C</td><td>150</td><td>26th May 11</td><td>Application identification and indicative analyses</td><td>33</td></tr>
<tr><td>D</td><td>150</td><td>22nd Sep 11</td><td>SLM role model and activities</td><td>31</td></tr>
<tr><td>E</td><td>360</td><td>16th Feb 12</td><td>Functions derivation and discussion; discussion of first case study results</td><td>24</td></tr>
<tr><td>F</td><td>150</td><td>1st Jun 12</td><td>Presentation of the model, case studies and final refinements</td><td>32</td></tr>
<tr><td>G</td><td>180</td><td>7th Feb 11 (interim)</td><td>SLM process refinement</td><td>6</td></tr>
<tr><td>H</td><td>180</td><td>10th May 11 (interim)</td><td>Application identification</td><td>8</td></tr>
<tr><td colspan="5">Participants: management-level staff from the German-speaking financial industry. Average composition: 39% banks, 39% providers, 16% software vendors, 6% consultants. Interview dates and topics are available upon request.</td></tr>
<tr><td colspan="5">Further information: the consortium research project CC Sourcing is in its 9th year now. Every two years the partner base changes slightly. Generally, the consortium researchers select the participating companies so that the whole financial value chain is covered. Specifically, the following company types participate: interbank (e.g. stock exchanges, interbank communications providers), back-office providers (e.g. transaction banks, service integrators), front-office organizations (e.g. sales banks). The participating individuals are almost all executive-level with more than ten years of professional experience. All focus group meetings followed a similar schedule: a) presentation of the artifact's current state, b) enclosing plenary discussion, c) separation into three sub-groups to work on certain aspects of the artifact. In case of contradictory opinions a democratic solution was sought, whereas partners have veto rights.</td></tr>
</table>

Table 3: Focus groups and objectives.

<table>
<tr><th>Group & sub-groups</th><th>Functions</th></tr>
<tr><td colspan="2">Service Description</td></tr>
<tr><td>S. Description Creation and Maintenance</td><td>service metadata management, description (e.g. WSDL) storage, non-functional service description, operational service description, functional service description, (e.g. WSD)L import</td></tr>
<tr><td>S. Description Analysis and Visualization</td><td>description (e.g. WSDL) validation, WSDL dependency tracking (services, schema, policies), description (e.g. WSDL) change preview, description (e.g. WSDL) visualization</td></tr>
<tr><td colspan="2">Service Modeling, Design & Implementation</td></tr>
<tr><td>Requirements Management</td><td>requirements tracing, requirements repository, requirements evaluation support</td></tr>
<tr><td>S. Configuration</td><td>end user service configuration support, expert service configuration support, configuration rules defini-</td></tr>
</table>

	tion and maintenance, infrastructure integration
S. Programming	proxy service creation, rule service creation, code (e.g. wsdl2java) conversion, data type definition (e.g. XML schema), data type integration, stub generation (e.g. Javascript), data service creation, webservice creation, automated code generation, source control management integration (e.g. GIT, SVN)
S. Testing	service test invocation, test specification, test implementation, test execution (remote/on-site), test result analysis, test automation / scheduling, test case repository, cross browser testing, webload testing
Template Creation and Maintenance	registry sequency template creation, endpoint template creation, SLA template creation
Service Valuation	
S. Pricing	service pricing engine, MIS pricing data integration
S. Costing and Controlling	service costing engine, cost planning, cost analysis
S. Revenue Management	revenue simulation, scenario comparison, revenue planning
S. Valuation Support	supporting sub-functions for the other valuation sub-functions, e.g. costing and pricing data gathering/integration
Service Exchange and Integration	
S. Discovery	semantic repository search, plain-text repository search
S. Contracting	service contract versioning, service contract cloning, service contract deprecation
Message Creation	message format library, message configuration (rules, transformation, messages)
Message Exchange and Management	message visualization, publish topics via WS standards, subscribe to topics via WS standards, m. box management, add/remove new topic, browse topic, add/remove m. box, add/remove queue, m. sending/receival, m. storage and forwarding, m. priority management, m. tracing (request and response), service-based business events publication
Complex Event Processing (CEP)	XML event processing, XML event production, event stream definition, event stream registry store, XML event filtering, CEP specification, CEP execution
S. and Message Security	role based topic authorization, topics management and permission queuing, user based topic authorization, service access rights control, message transport encryption
Logging and Archiving	message collection and archival, repository activity logging and filtering
Other	endpoint metadata collection, service registration, mediation policy authoring
Service Monitoring, Analysis & Reporting	
S. Monitoring and Analysis	proxy service, endpoint and sequence monitoring, real-time service operation monitoring, service invocation analysis (trend identification), activity correlation analysis, SOA policy monitoring, repository activity analysis, SLA (compliance) monitoring, real-time service exchange monitoring, routing rules monitoring, synthetic monitoring and analysis, real user monitoring and analysis, internet analysis, web site analysis, application monitoring, server monitoring, database monitoring, network monitoring
S. Reporting	service monitoring results layouting, service-related reporting, repository event notifications (email, webservice forwarding), governance reporting, automated outage user notification
Process Management	
Process Administration	process artifact management, process repository
Process Dev. and Enh.	business process modeling, process notation conversion, process versioning, hot update (e.g. for BPEL)
Process Deployment	process service deployment, process description export (e.g. BPEL)
Process Execution	process instance data cleanup, business process caching and throttling, workflow management
Process Integration	process import, user integration for process tasks
Process Monitoring	business process monitoring, further supportive sub-functions
Cross Functions	
SLM Integration	SLM interface maintenance, SLM adapter maintenance
Business Intelligence (BI)	Data warehouse (DWH) runtime integration, DWH webservice and metadata integration, DWH results visualization, browser-based mobile BI, BI data distribution
Ontology Management	ontology visualization, ontology editing, reasoning, ontology consistency check, concept satisfiability check, ontology classification and realization, ontology debugging
SLM Security	single-sign on (e.g. SAML), user profile creation and maintenance, multifactor authentication (e.g. XMPP-based), identification key storage
S. Hosting	service hosting (and further supporting sub-functions)
S. Level Management	SLA management, service level creation support, service level maintenance
S. Structure Visualization	service provider and user visualization, technical service structure visualization
Team Coord. and Coll.	project management support, team collaboration tools, change management support

Table 4: Function listing.

References

Baida, Z., 2006. Software-aided Service Bundling.

Bardhan, I.R. et al., 2010. An Interdisciplinary Perspective on IT Services Management and Service Science. *Journal of Management Information Systems*, 26(4), pp.13–64.

Becker, J. et al., 2004. Epistemologische Positionierungen in der Wirtschaftsinformatik am Beispiel einer konsensorientierten Informationsmodellierung Table 9 continued Functions Description Search Dynamic value search Allows to search for and identify data objects by means of kn. In F. Ulrich, ed. *Wissenschaftstheorie in O¨ konomie und Wirtschaftsinformatik (The Theory of Science in Economics and IS)*. Wiesbaden: Deutscher Universitats-Verlag, pp. 335–366.

Becker, J. et al., 2002. Refernzmodellierung (Reference Modeling). *Das Wirtschaftsstudium (WISU)*, 30(11), pp.1392–1395.

Behara, G.K. & Inaganti, S., 2007. Approach to Service Management In SOA Space. *Business Process Trends Journal*, 1(February), pp.1–9.

Berbner, R., Grollius, T. & Repp, N., 2005. An approach for the Management of Service-oriented Architecture (SoA) based Application Systems. In *Enterprise Modelling and Information Systems Architectures (EMISA 2005)*.

Brown, W.A., Moore, G. & Tegan, W., 2006. SOA Governance – IBMs Approach.

Bullinger, A.-W. & Scheer, H.-J., 2005. *Service Engineering - Entwicklung und Gestaltung innovativer Dienstleistungen* 2nd ed., Berlin, Heidelberg: Springer.

Bullinger, H.-J., Fähnrich, K.-P. & Meiren, T., 2003. Service engineering—methodical development of new service products. *International Journal of Production Economics*, 85(3), pp.275–287.

Casati, F. et al., 2003. Businessoriented management of web services. *Communication of the ACM*, 46(10), pp.55–60.

Erl, T., 2005. *Service-Oriented Architecture (SOA): Concepts, Technology, and Design*, Prentice Hall.

Erl, T., 2007. *SOA: Principles of Service Design*, Boston: Prentice Hall International.

Fettke, Peter & Loos, Peter, 2007. Perspectives on Reference Modeling. In Peter Fettke & Peter Loos, eds. *Reference Modeling for Business Systems Analysis*. Hershey PA: Idea Group Publishing, pp. 1–20.

Fettke, Peter & Loos, Peter, 2004. Referenzmodellierungsforschung. *Wirtschaftsinformatik*, 46(5), pp.331–340.

Gregor, S., 2006. The nature of theory in information systems. *MIS Quarterly*, 30(3), pp.611–642.

Gummesson, E., 1994. Service Management: An Evaluation and the Future. *International Journal of Service Industry Management*, 5(1), pp.77–96.

Hevner, B.A.R. et al., 2004. Design science in information systems research. *MIS Quarterly*, 28(1), pp.75–105.

Kawaguchi, S. et al., 2004. MUDABlue : An automatic categorization system for open source repositories. In *Proceedings of the 11th Asia-Pacific Software Engineering Conference*. pp. 184–193.

Kohlborn, C.T., Fielt, E. & Korthaus, A., 2009. Towards a Service Portfolio Management Framework. In *Proceedings of the 20th Australasian Conference on Information Systems*.

Kohlmann, F. & Alt, R., 2007. Business-Driven Service Modelling - A Methodological Approach from the Finance Industry, in: Maciaszek, L.A., Abramowicz, W. (Hrsg), Proceedings of the 1st International Working Conference on Business Process and Services Computing (BPSC '07). In *BPSC '07*. Leipzig, pp. 180–193.

Lawler, J. & Howell-Barber, H., 2007. *Service-Orientated Architecture*, Boca Raton: Auerbach.

Lusch, R.F. & Vargo, S.L., 2006. The Service-Dominant Logic of Marketing.

MacKenzie, C.M. et al., 2006. Reference Model for Service Oriented Architecture 1.0. Available at: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm.

Maglio, P.P. & Spohrer, J., 2008. Special Issue: Service Science, Management, and Engineering. *Journal of IBM systems*, 47(1), pp.1–192.

March, S. T. & Smith, G.F., 1995. Design and natural science research on information technology. *Decision Support Systems*, (15), pp.251–266.

OGC, 2007. *The Official Introduction to the ITIL Service Lifecycle*, Office Of Government Commerce.

Österle, H. & Otto, B., 2010. Consortium Research. *Business & Information Systems Engineering*, 2(5), pp.283–293.

Papazoglou, M. & Heuvel, W.-J. v. D., 2006. Service-oriented design and development methodology. *International Journal of Web Engineering and Technology*, 2(4), pp.412–442.

Peffers, K. et al., 2008. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3), pp.45–77.

Puschmann, T. & Alt, Rainer, 2011. Überbetriebliches Service Lifecycle Management. *HMD - Praxis der Wirtschaftsinformatik*, 48(278), pp.58–67.

Raverdi, P.-G.P.-G., 2008. *Service Lifecycle Management*, Available at: http://www.ap-verlag.de/Online-Artikel/20081112/20081112h Siemens Service Lifecycle Management.htm [Accessed March 7, 2012].

Rosemann, M. & Schütte, R., 1999. Multiperspektivische Referenzmodellierung. In Jörg Becker & R. Schütte, eds. *Referenzmodellierung: State-of-the-Art und Entwicklungsperspektiven*. Heidelberg: Physica, pp. 22–44.

Scheer, A.-W., 1992. *Architecture of integrated information systems - foundations of enterprise modelling*, Berlin: Springer.

Scheer, A.-W., Thomas, O. & Adam, O., 2005. Process modeling using event-driven process chains. In M. Dumas, W. van der Aalst, & A. H. M. ten Hofstede, eds. *Process-aware information systems: bridging people and software through process technology*. Hoboken: Wiley, pp. 119–145.

Scheer, A.W., 2001. *ARIS - Modellierungsmethoden, Metamodelle, Anwendungen*, Berling: Springer.
Scheer, August-Wilhelm, 1997. *Wirtschaftsinformatik: Referenzmodelle für industrielle Geschäftsprozesse*, Berlin etc.: Springer.
Simon, H.A., 1998. *The Sciences of the Artificial* 3. ed., Cambridge, MA: The MIT Press.
Treiber, M., Truong, H.-L. & Dustdar, S., 2008. SEMF - Service Evolution Management Framework. *2008 34th Euromicro Conference Software Engineering and Advanced Applications*, (3), pp.329–336.
Weiss Ferreira Chaves, L. et al., 2006. Service lifecycle management infrastructure for smart items. In *Proceedings of the international workshop on Middleware for sensor networks - MidSens '06*. New York, USA: ACM Press, pp. 25–30.
Yelmo, J.C. et al., 2007. User-Driven Service Lifecycle Management – Adopting Internet Paradigms in Telecom Services. In *Proceedings of the International conference on service-oriented computing 2007*. pp. 342–352.