Meta-Services – Towards Symmetric Service-Oriented Business Ecosystems

Rainer Schmidt[1], Axel Kieninger[2], Robin Fischer[2], Christian Zirpins[2]

[1]HTW-Aalen
Rainer.Schmidt@htw-aalen.de

[2]Karlsruhe Service Research Institute, Universität Karlsruhe (TH)
{Axel.Kieninger|Robin.Fischer|Christian.Zirpins}@ksri.uni-karlsruhe.de

Abstract. Service-Oriented Business Ecosystems (SOBEs) contain numerous procedures and rules that act upon services, but are not considered to be services themselves. Thus an asymmetry is created by dividing SOBEs into first-class entities that are services and others that are not. This separation creates inefficiency and conceals the value contribution of non-first-class entities. Using so-called *meta-services* it is possible to encapsulate procedures and rules homogeneously as services, thus creating a symmetric architecture of the SOBE. This symmetric architecture allows reducing the administrative effort for SOBEs by reusing procedures and using a common administration for services and meta-services. Meta-services are a value factor often underestimated by service designers, but highly appreciated by service users. By defining meta-services, the possible interactions between customer and service provider can be defined clearly, including mutual expectations. E.g. it is possible to define the information due in the case of a complaint. Furthermore, meta-services allow to clarify the potential adaptations to changing requirements to the customer and thus to delineate the spectrum of flexibility offered. The transparency created by meta-services allows defining contracts in a more precise manner and thereby avoiding misunderstandings and deceptions on both sides.

Keywords: Meta-services, Service-Oriented Business Ecosystems, Service Dominant Logic, Service Value Networks

1 Introduction

Service-Oriented Business Ecosystems (SOBEs) [1] contain numerous procedures and rules that act upon services, but are not considered to be services themselves. This creates an asymmetry by dividing SOBEs into services that are first-class entities and equivalent procedures that are not. This separation impedes optimizations of SOBEs, because redundant procedures and structures are created. An example is the so-called IT service management, which is applied in many enterprises to administrate IT-services using approaches such as ITIL [2]. Although IT service management provides numerous procedures to support services – such as “Incident Management” for

handling customer complaints – these procedures themselves are not regarded as services. In consequence, a multitude of auxiliary concepts – such as "policies" in ISO 20000 [3] – is created to capture and administer these procedures. Often, the content of these auxiliary concepts is highly redundant to already existing procedures in the service lifecycle.

Furthermore, the value contribution of these procedures and rules acting upon services is concealed. As they are not regarded as first-class entities, their value-contribution is often not adequately considered. In practice, however, e.g. the possibilities to raise and settle a complaint are highly influential on customers' evaluation of a service. Customers want to know clearly, how they can interact with the service provider to change or improve services.

Therefore this work will introduce the concept of *meta-services* on top of a SD-logic foundation. Meta-Services allow encapsulating procedures and rules acting upon services as services themselves and thus to handle them equally to services. Especially, they clearly define the potential interactions between customer and service provider, e.g. in case of service failure or when a service has to be adapted due to changing customer requirements. Furthermore, considering meta-services creates a more complete view on the value created in a SOBE. Thus optimizations of the SOBE are more effective, because they are based on a more complete and detailed view on the SOBE.

The paper proceeds as follows. Firstly a business-oriented view on SOBEs is introduced in sec. 2 that adopts the SD-logic approach. Based on this view, a framework for meta-services is developed in sec. 3. It introduces a meta-service notion for SD-logic and identifies functional and non-functional service properties as operands for meta-services using aspects. Furthermore, a concept for a meta-service taxonomy is proposed and the role of meta-services as value factor for SOBEs is analyzed. To exemplify meta-services, sec. 4 compares the current definition of the ITIL Incident Management with a meta-service-based approach, using service value nets. Related areas of work are analyzed in sec. 5. Section 6 gives a conclusion and outlook.

2. The SD-logic Approach to SOBEs

The definition of SOBEs builds on a service definition that considers business and software views. Thus, it is necessary to take the business perspective on services into account. While business-related service definitions have a long history[1], recent research converges more and more towards a common paradigm called SD-logic [5].

2.1 SD-logic Foundations / SOBEs based on SD-logic

Following SD-logic, a service is defined as the "application of competences (knowledge, skills and resources) by one entity for the benefit of another entity in a non-coercive (mutually agreed and mutually beneficial) manner" – whereas the value co-

[1] Traditionally, service has been defined by the absence of four properties when compared to material products: intangibility, heterogeneity, inseparability and perishability. However, this definition has been criticized for being negation-based [4].

creating entities (e.g. an IT provider and its customer) are also referred to as "service systems" [6].

According to this definition of service, Maglio et al. [6] define a service system "as an open system capable of improving the state of another system through sharing or applying its resources and capable of improving its own state by acquiring external resources". The resources shared, applied or acquired by service systems may be divided into four basic classes, namely people, organizations, information and technology [7].

The main purpose of service systems is to "design, propose, agree and realize" value propositions with other service systems [7]. Service systems may occur in various forms and granularity, e.g. as businesses, government agencies, people families, community groups, and open source communities [7]. By running common service processes – and thus co-creating value – with other service systems, they integrate and coordinate resources [8]. Thus, there is no service provider producing services in isolation, but service is always the common effort of two or more service systems. Services can be best described as a bi- or multilateral process [9] of two or more service systems co-creating value, as shown in fig. 1. The process nature of a service is also confirmed in [10].

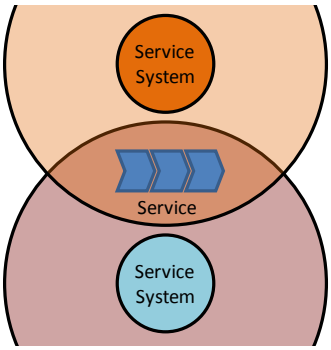

Figure 1: Service Co-Creation

Applying these considerations leads to homogeneous service-oriented architectures [11], where value is co-created within a network of interconnected service systems. We consider such networks of interconnected service systems that co-create value to be service value networks (SVNs). Therein, a SVN may span over multiple organizations. It may also be under the control of an individual enterprise. A SOBE consists of one or more SVNs that co-create value and service systems that offer not yet utilized service-propositions. Figure 2 exemplifies our understanding of services, service propositions, SVNs and SOBEs: Two SVNs are shown. The upper one is an intra-enterprise SVN, containing only service systems within one enterprise. The lower SVN is a cross-enterprise SVN, because two of its service systems are in another enterprise. Furthermore, there are three service systems belonging to a third enterprise. These service systems are not yet participating in value co-creation – they only offer service propositions.

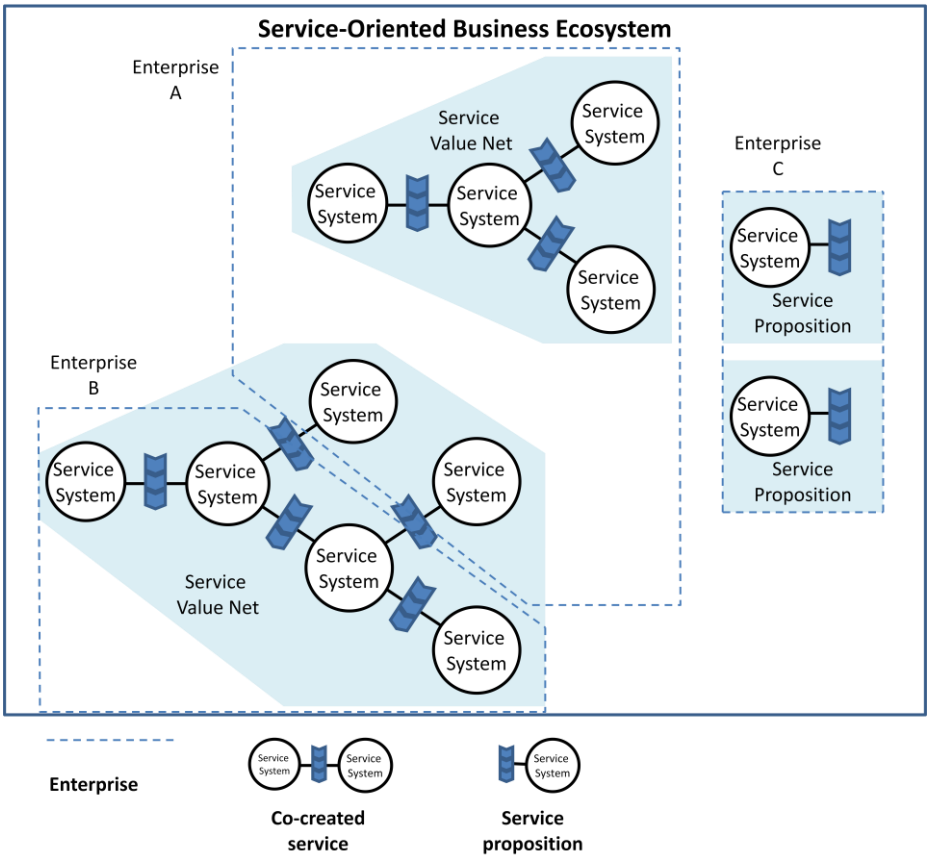

Figure 2 : Service-Oriented Business Ecosystem

2.2 Scenario

The concept of a SOBE based on SD-logic can be illustrated by the gastronomy domain. Basically, the preparation of a meal is a classical service. In this context, a complete SOBE has developed over the time. Starting point is the preparation of meals in different kinds of restaurants. There are gourmet restaurants, where you find very complex internal service value nets and fast food restaurants, which are integrated into huge external service value nets. These value nets also include enterprises, which are not part of gastronomy such as the producers of the boxes, used by the fast food restaurant and the logistics company, delivering the boxes.

The service system concept, as defined in SD-logic [5], can be easily found in a gourmet restaurant. The preparation of a gourmet menu is distributed to several stations, such as for cooking the meat, the sauces, the baking of the dessert etc. Thus a multitude of service processes is coordinated to provide the final service to the customer. In the fast-food restaurant, the use of goods to transport service can be observed when applying SD-logic. Instead of preparing the ingredients of the fast food

themselves, many services are outsourced: bread is bought from a bakery; the frozen hamburger is bought from a meat factory etc. These ingredients "import" the services originally provided in the fast food restaurant from other places. Even more important, they allow concentrating these services and thus achieving economies of scale.

In the gastronomy domain, the non-functional properties of services are of huge importance. E.g., fish is not just ordered for the menu, but it is also expected to be delivered in time, meaning not too early and not too late.

3. A SD-logic based Framework for Meta-Services

When taking a closer view on SOBEs, it becomes obvious, that not only the functional and non-functional properties of services are important, but also procedures, actions etc., which act upon the services rendered. For example, it is necessary to be able to complain about delayed services or to initiate the change of services. Manifestations are complaint and suggestion boxes. Raising a complaint or a suggestion initiates a process, which acts upon the service. This process need not be executed by the service provider himself. Also another service provider may execute it. The complaint is analyzed or the suggestion for a new or changed service is considered. Thus, the management of complaints and suggestions can be regarded as services themselves. They are services that act upon services, thus we term this kind of services *meta-services*.

Subsequently, a framework for meta-services based on SD-logic shall be developed in this section. First a meta-service notion is developed and the characteristics of meta-services are clarified. Then the possible operands of meta-services, the functional and non-functional properties of services are investigated. This investigation forms the basis for a meta-service taxonomy. Finally, the influence of meta-services on the value of services in SOBEs is analyzed and a cube model to represent service value is introduced.

3.1 A Meta-Service Notation for SD-logic

Basically, SD-logic describes service as a common process between two or multiple service systems as discussed above. However, to establish the provisioning of services between two service systems or in a SVN, a lot of "procedures" are necessary to act upon services. Following the life cycle of services, a number of example meta-services can be identified.

First, requirements for a service have to be collected. These requirements include both requirements concerning functional and non-functional properties of the service. Second, the appropriate service has to be designed. Then, based on the design, the service has to be implemented and tested including both functional and non-functional requirements. If the collected requirements imply the change of an existing service, appropriate operations have to be applied. Furthermore, the reuse of several existing services and their orchestration may provide the required service functionality. Next, implementation resources have to be assigned to put the service into operation. To do so, an adequate administration of the resources is an important prerequisite. During operation deviations from the specified service-functionality and service levels (non-

functional properties) have to be detected, recorded and investigated. Finally, services that are no longer required have to be retired and the assigned resources have to be released.

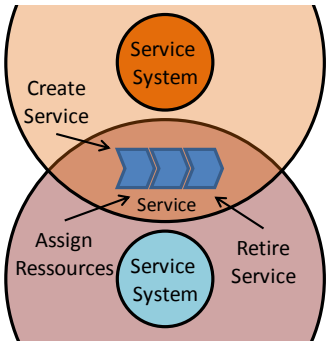

Figure 3: Examples for meta-services

From an abstract point of view, meta-services act upon one or more services and modify their functional or non-functional properties [11] as shown in fig. 3. Often, this kind of alteration implies changes to service systems that co-create a service. Meta-services are co-created by two or more meta-service systems. Meta-service systems differ by the fact, that meta-services and not services are co-created by them.

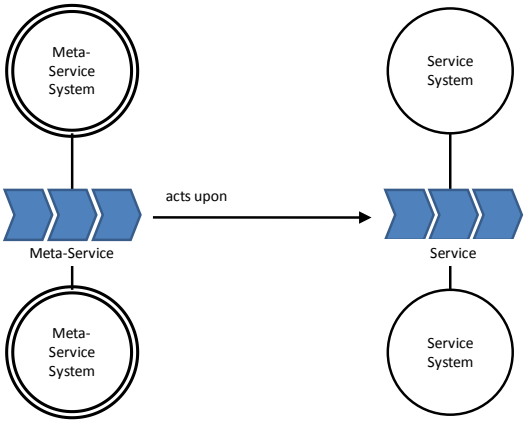

Figure 4: Meta-service acting upon a service

Formally, a meta-service differentiates from a service mainly by its operand type that is a service (see fig. 4). Their functional and non-functional properties have the same structure as those of services. A meta-service is described by a process, just as a service. Furthermore, meta-services and services share non-functional properties such as availability and throughput. E.g. for a complaint meta-service, it may be defined, how long it takes to process a complaint. Of course, meta-services may also be object

of other services. Such *meta-meta-services* are the entry point to a recursive structure of meta-service relationships.

Meta-services are operations on functional and non-functional properties of services. Therefore, both functional and non-functional properties of services shall be described in more detail using an aspect-oriented approach to define meta-services with more precision.

Functional Service Properties as Operands for Meta-Services

Functional properties of a service can be described as a process, which – according to SD-logic – consists of "a series of activities where a number of different types of resources are used" [12]. This service process can be further divided into five so-called aspects[2]. Aspects are disjoint sets of the process description. Using aspects, meta-services can be differentiated with respect to the aspects that their operand belongs to:

The *hierarchical aspect* describes how the service process is composed of sub-processes and activities. E.g. the hierarchical aspect describes which preparation activities are necessary to deliver a menu in a restaurant.

The *behavioral aspect* defines, when and under which preconditions activities are performed, that means the control flow. The control flow consists of sequence and gateway elements. To create a menu, it is very important to obey to the order of preparation steps to avoid, that the meat is well done but the side dishes are overcooked.

Furthermore, there is a flow of information between activities. In the informational aspect the information that is exchanged between activities is defined. In a restaurant e.g., the chef announces the orders and the cooks for meat, fish etc. reply whether they are ready.

The *resource aspect* complements the hierarchical aspect by describing how the activities specified in the behavioral aspect are executed using one or several resources and whether they are operant (active) or operand (passive) resources. Thus it describes the implementation. Resources may be people, organization, information or technology. The resource aspect specifies not only the resource itself, but also the procedure needed to obtain and return the resource, for example from the customer. In a restaurant the resource aspect represents the existence of cooks, ingredients etc..

The *interaction aspect* describes the in- and outgoing messages during the service execution. Due to his inter-organizational nature, this aspect is missing in most intra-organizational business processes. The interaction aspect is required to reflect the fact that there is no hierarchical, but a market-based coordination between organizations. The interaction aspect differs from the information flow between the activities, because it is not orchestrated but part of a choreography, as differentiated in BPMN (see e.g. [16]). Interactions may be a simple one-way communication, bidirectional or follow complex protocols. The difference between the control flow in the behavioral aspect and the interaction aspect becomes obvious, when we compare the strict commands of the chef to the cooks with the orders arriving from the guests in the restau-

[2] Aspects have been introduced in the workflow [13] and software engineering domain [14]. An approach to unify the usage of the term aspect in both domains has been done in [15].

rant. The commands from the chef have to be obeyed, the orders of the guest have to be fulfilled, but with certain degrees of freedom.

As shown by example, meta-services may have operands belonging to one or several of the functional aspects identified. However, aspects for representing non-functional properties of services may as well be operands of meta-services as shown below.

Non-Functional Service Properties as Operands for Meta-Services

In recent years non-functional properties of services have been widely discussed in literature as for example in the field of Software Engineering (see e.g. [17], [18]). In [19] O'Sullivan examines these properties of services in general abstracting from a particular (computer) science perspective. According to his work non-functional properties of services are constraints associated to service functionality and may be divided into nine different aspects. In the following these aspects – namely availability, price, payment, discounts and penalties, obligations, rights, quality, security and trust – are briefly introduced. Contrary to the aspects representing functional properties, the aspects for the representation of non-functional properties cannot be assigned to a single entity in the process graph. Instead they represent cross-cutting issues distributed over the whole process graph.

The *availability* of a service defines the time or location when or where a prosumer is able to accept a provider's proposition to co-create value. Regarding the *price* of a service there are different charging techniques a provider may select to specify the value of his work. The amount of money a prosumer is charged may depend on proposition activities (e.g. enabling service availability) as well as on co-creation activities (e.g. units of measure co-created) of the provider. The corresponding *payment* is agreed on in the beginning of a service relationship. There may be *discounts* that a prosumer receives depending on terms of payment (payment related discounts, i.e. how to pay) or on attributes of the prosumer (payee related discounts as e.g. membership to associations). Within the scope of their cooperation a prosumer and a provider agree to meet certain *obligations* (as for example to provide operand or operant resources). In case of non-compliance with these obligations the respective party will be *penalized*, i.e. has to bear the consequences defined. By now the provider usually owns the intellectual property associated with a service process. A prosumer just has a limited set of *rights* (as for example the right to comprehend, the right to retract, the right of premature termination, the right of suspension and the right of resumption). However, the co-creation of service process specifications will bear an influence on the current legal situation. The *quality* of a service should be assessed from a prosumer's [20] point of view. O'Sullivan recommends measuring perceived service quality along five dimensions (namely reliability, responsiveness, assurance, empathy and tangibles). *Security* aspects are of increasing interest – particularly with respect to IT-enabled services. Managing security means to reduce concerns regarding identity, privacy, alteration etc. Also, mutual *trust* between parties involved in a service relationship is of high importance.

The non-functional properties of a service cannot be changed directly because they represent goals for the process described by the functional properties and they are "cross-cutting" [14]. E.g. availability is not just a property, which can be modified directly. Instead, changing the availability means – possibly – to change the service process and the resources assigned to it.

Thus, meta-services acting upon non-functional properties need the support of meta-services acting upon functional properties to adapt the service process appropriately. This implies that changes of non-functional properties may require a kind of design method, to identify the changes necessary to functional properties caused by the change of non-functional properties.

3.2 Concept for a Meta-Service Taxonomy

There are different approaches to introduce a taxonomy of meta-services. A very straightforward approach is to use the phases of a service lifecycle. However, there is no common understanding about the phases in the service lifecycle. Therefore, other concepts for taxonomies have to be developed.

The first approach to classify meta-services is scope (see tab. 1). The scope of a meta-service may be aspects, services or SVNs. Change Management is – such as a meta-service – scoped on aspects and changing a service to adopt it e.g. to changed customer requirements. Meta-services that scope on aspects may be further differentiated whether they scope on single or multiple aspects and which aspects they act upon. Configuration Management [3] is an example for a meta-service usually acting upon resources, whereas Change Management may act-upon all aspects. Brokering is a meta-service that is scoped on whole services. According to the requirements defined, a service from a set of services is selected. The internal structure of the service, however, is not changed. An example for a meta-service that operates on service value nets is orchestration. An orchestration meta-service coordinates the execution of multiple services and hereby abstracts from the internal structure of the services to be orchestrated.

Scope		Example
Aspect(s)	Single Aspect	Configuration Management
	Multiple Aspects	Change Management
Service		Brokering
Service Value Nets		Orchestration

Table 1: Classification of meta-services according to their scope

3.3 Meta-services as Value Factor for SOBEs

Based on the considerations above, not only functional and non-functional properties contribute value but also meta-services do. Thus, these three dimensions define a service and the volume of the spanned cube is equivalent to the service value created. The value judgment a service system applies within a particular service relationship is

depending on the types of service systems involved [19]. There are different forms of value judgments, as for example monetary value or reputation value [19].

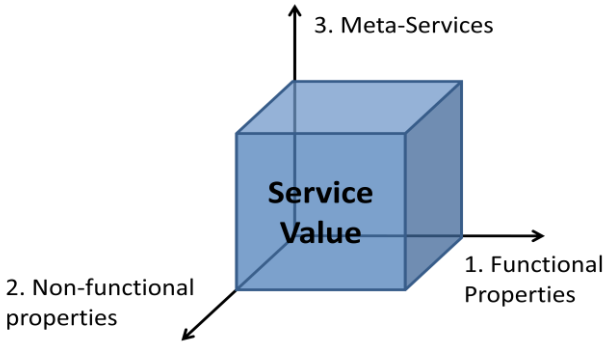

Figure 5: Meta-Service as a value factor for services

Following the cube-model developed above (see fig. 5), the value of a service is determined by its functional and non-functional properties as well as the meta-services available. For example, even if a fast food restaurant is offering its meals for lunch break at a very low price, its value to the customers may be very low, if they have to wait too long. Furthermore, the customers may also choose another restaurant, if they notice that complaints are not handled by the restaurant management.

Meta-services are an often underestimated value factor by service designers, but highly appreciated by service users. By defining meta-services, the possible interactions between customer and service provider can be defined clearly, including the duties of customer and service provider. E.g. it is possible to define the information due in the case of a complaint. Furthermore, meta-services allow to clarify to the customer the potential adaptations to changing requirements and thus to delineate the spectrum of flexibility offered. The transparency created by meta-services allows defining contracts in a more precise manner and thereby avoiding misunderstandings and deceptions on both sides. Without proper meta-services only imprecise and cloudy assumptions can be made. Especially the impossibility to sanction violations of the QoS agreed upon may make the service worthless to the customer. The expenses avoided by the availability of meta-services can be associated with certain kinds of transaction costs [21].

4. Study: Modeling ITIL Incident Management as Meta-Service

To clarify the structures developed, we performed a study on Incident Management [2]. In ITIL [2], *Incident Management* is a very popular process. Its task is to process the interactions with the customer and to maintain the service levels agreed upon. Primary goal of Incident Management is to reestablish the service as quick as possible. This may include the use of so-called work-arounds, which reestablish the service but do not solve the deeper cause of the incident. E.g. the rebooting of a PC enables the user to continue, but does not provide a solution to the underlying problem, a

faulty driver, for example. Incident Management uses other processes such as Problem, Change and Configuration Management [2].

4.1 The current Definition of Incident Management

Incident Management differentiates between standard incidents and so-called major incidents that require measures beyond the standard procedures, e.g. in case of an emergency, as shown in fig. 6. The first step in processing incidents is recording. All incidents are clearly described and associated with unambiguous identifications, such as a ticket number. After recording, incidents are prioritized using procedures for evaluating their business impact. Classification is done to associate the incident with one or multiple possible causes. The solution is developed using information from so-called known errors. These are known malfunctions of hard- or software associated with a work-around or remedy. However, a deeper analysis may be necessary, because no applicable known error was found. In this case, a work-around extends the time for deeper problem analysis by enabling the user to continue his work. This problem analysis is not part of Incident Management, but is part of Problem Management, which is directed at finding the deeper cause of incidents. Solution developed are implemented using Change- and Configuration Management [2] assuring a proper implementation of the solution. If no solution is available, an escalation procedure is initiated. This may be the case, if no appropriate resources are available. If a solution is developed, the customer has to evaluate it. The closure of an incident is only possible if the customer agrees.

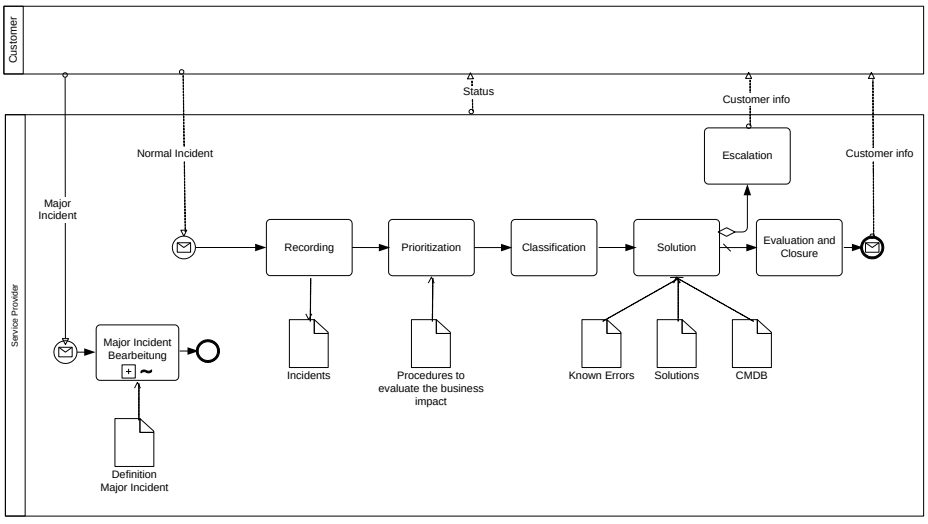

Figure 6: Incident Management

In practice the coordination of Incident Management with Problem Management etc. often creates huge difficulties, because units from different organizations have to be coordinated [22]. The cause for these difficulties is the inappropriate structure of

Incident Management. Although it is a meta-service using a number of supporting services such as Problem Management, Change Management etc. it is not organized in a service-oriented way. Therefore, many proven means for organizing the co-creation of services are not applied. Furthermore, the QoS of Incident Management is a major issue in many IT organizations. Incidents are not processed or only processed slowly. Often, there are huge piles of "old" incidents no one feels responsible for.

4.2 Incident Management as Meta-Service

A meta-service-oriented redefinition of Incident Management regards Incident Management as a service provided by a SVN, as shown in fig. 7. Starting point is the service desk, which co-creates the Incident Management service with the customer. The prioritization of an incident is done in cooperation with Service Level Management. Service Level Management not only provides the service level agreements of the service, but also the actual status of service level fulfillment. Using the latter information, a service that is near to violate the service level agreement is higher prioritized than a service that is not. The classification of the incident is done in cooperation with Problem Management that also provides information about known errors. Either a solution originating from a known error is applied or a new solution is developed together with Problem Management. The solution developed is implemented using the change management service. It is responsible for correctly implementing the change into to the production system. To do so, Change Management checks if the changes requested are in compliance with the corporate rules by cooperating with Configuration Management. If yes, the changes are implemented by change management. Finally the incident is evaluated in cooperation with the customer and – hopefully – closed.

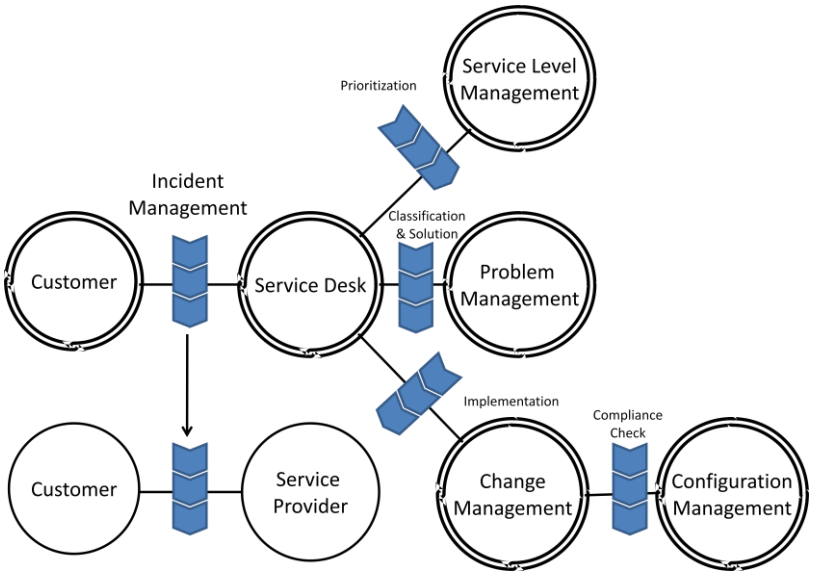

Figure 7: Incident Management service value network

5. Related Work

Although there is work that uses the term meta-service, there is no common understanding of meta-services. The closest usage of the term meta-service can be found in [23]. There, meta-services are services for the monitoring, optimizing and adjusting of resource services in a grid environment. Another definition of meta-service is found in [24]. Here meta-services "map an existing grid workflow to a service by overriding attributes of the grid workflow". Thus, meta-services are a kind of intermediary. In [25] a concept for meta-service discovery is introduced. However, topic is not the discovery of meta-services, but a mechanism for service discovery. The same applies for the meta-service discovery in [26]. Meta-services following the definition in [27] are services processing meta-data, they manage meta-data in [28]. In [29] a meta-service represents "a set of services which have similar function". Thus a meta-service "can be seen as a delegate of some similar services", but not as a service acting upon other services. In [30] a single composition service for e-services is classified as meta-service. However, no general framework for meta-services is presented. The composition of services is termed as meta-service in [31], too. In [32], a multi-agent architecture for the management of business processes is developed. In this context, meta-service are used as synonym for certain non-functional properties of services. In [33] QoS provisioning and maintenance are identified as meta-services.

A variety of meta-services can be found in the field of IT Service Management, which is specified in frameworks like Cobit [34] and ITIL[2]. In section 4, Incident Management as a typical example for a meta-service within ITIL has been introduced (Service Operation). Other meta-services presented in ITIL are for instance Demand Management ensuring resources are allocated appropriately (Service Strategy), Availability Management caring about agreed availability service levels to be met (Service Design), Change Management covering changes to service assets and configuration items (Service Transition) and the "7 Step Improvement Process" implementing corrective actions to enhance a service (Continual Service Improvement).

SOBEs are closely related to homogeneous and service-oriented enterprise architectures [11], co-creating value within a network of interconnected service systems, i.e. SVNs. Recent research on SVNs such as [35], adds revenue flows into the basic SVN considerations that merely focus on the notions of value or service composition. The authors aim at providing means to measure data that is relevant for determining information such as key performance indicators based on service network notation (SNN). However, a concept to integrate services that modify services or entire SVNs (i.e., meta-services) is not included into the research of SNN.

The characteristics of meta-services to change functional and non-functional aspects of other services technically manifests in Aspect Oriented Programming. By weaving in code fragments into service instances, their behavior can be changed at runtime. This approach e.g. is used to change BPEL processes at runtime [36] and therefore may point ways, how to operationalize our concept of meta-services.

6. Conclusions and Outlook

Encapsulating procedures to act upon services as meta-services simplifies the management of services. Since there is no longer the need for separate administration structures that handle services and management tasks, but only one service catalogue that contains both services and their lifecycle procedures. Furthermore, meta-services help to introduce polymorphism into service management and thus reduce redundancies. Procedures hitherto defined for services and their lifecycle procedures separately may now be unified into one. An example is Change Management: changes to services and meta-services should follow the same rules and can be implemented the same way.

The benefits of introducing meta-services to SOBEs are similar to those identified in [37] for SOA. First, there is an increased interoperability of the procedures encapsulated as meta-services, because standardized interfaces are used instead of ad-hoc integration and information is easier to exchange. The use of meta-services also increases federation, because meta-service-systems allow integrating resources, while maintaining autonomy and self-governance of resource administration. Clearly, the definition of meta-services also facilitates outsourcing. Due to their clear description, it is easier to use external service systems, e.g. for Incident Management. The clarified description of meta-services and their capability to improve the interaction between customer and service provider also increase the alignment of customer requirements and services and the organizational capability to react to changed requirements.

But there are also some issues to be solved. The inclusion of meta-services into the consideration of SOBE has been done in a qualitative manner only. Therefore, a quantitative analysis of the value contribution of meta-services is an important topic for future work. Another upcoming task is the refinement of the meta-service taxonomy. In particular, the granularity of meta-services has to be investigated in order to identify if further criteria for meta-service differentiation have to be found.

Furthermore, the recursive nature of meta-services should be investigated in more detail. In practice there are deliberate escalation mechanisms, which allow driving disputes about QoS up to the top management. An interesting question is, whether escalation mechanisms are a further kind of meta-meta-services.

7. References

[1] S. Tai, N. Desai, and P. Mazzoleni, "Service communities: applications and middleware," *Proceedings of the 6th international workshop on Software engineering and middleware*, ACM New York, NY, USA, 2006, pp. 17-22.

[2] S. Office, *The Official Introduction to the ITIL 3 Service Lifecycle*, Norwich: The Stationery Office Ltd., 2007.

[3] J. Bon, A. Jong, and J. Wilkinson, *IT Service Management: An Introduction, Based on ISO 20000 and ITIL V3 (ITSM Library)*, Van Haren Publishing, 2007.

[4] C. Lovelock and E. Gummesson, "Whither Services Marketing?: In Search of a New Paradigm and Fresh Perspectives," *Journal of Service Research*, vol. 7,

2004, pp. 20-41.
[5] S.L. Vargo and R.F. Lusch, “Evolving to a new dominant logic for marketing,” *Journal of Marketing*, vol. 68, 2004, pp. 1-17.
[6] P. Maglio, S. Vargo, N. Caswell, and J. Spohrer, “The service system is the basic abstraction of service science,” *Information Systems and E-Business Management*, vol. 7, 2009, pp. 395-406.
[7] P. Maglio and J. Spohrer, “Fundamentals of service science,” *Journal of the Academy of Marketing Science*, vol. 36, Mar. 2008, pp. 18-20.
[8] J. Spohrer, L.C. Anderson, N.J. Pass, T. Ager, and D. Gruhl, “Service Science,” *Journal of Grid Computing*, vol. 6, 2008, pp. 313-324.
[9] C. Grönroos, *Service Management and Marketing: A Customer Relationship Management Approach*, Wiley, 2000.
[10] J.L.C. Sanz, N. Nayak, and V. Becker, “Business Services as a New Operational Model for Enterprises and Ecosystems,” *Proceedings of the 8th IEEE International Conference on E-Commerce Technology and the 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)*, 2006, pp. 61-65.
[11] R. Schmidt and A. Kieninger, “DYNSEA – a dynamic Service-Oriented Enterprise Architecture based on S-D-Logic,” *First Workshop on Service oriented Enterprise Architecture for Enterprise Engineering*, Auckland: 2009.
[12] J. Spohrer, P.P. Maglio, J. Bailey, and D. Gruhl, “Steps Toward a Science of Service Systems,” *COMPUTER*, 2007, pp. 71-77.
[13] M. Weske, “Workflow management systems: Formal foundation, conceptual design, implementation aspects,” *University of Munster, Germany*, 2000.
[14] C.V. Lopes, G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. Lopes, J. Loingtier, and J. Irwin, “Aspect-oriented programming,” *In Proceedings: European Conference on Object-Oriented Programming*, 1997.
[15] R. Schmidt and U. Assmann, “Extending Aspect-Oriented-Programming in order to flexibly support Workflows,” *Proceedings of the ICSE98 AOP Workshop*, 1998, pp. 41–46.
[16] B. Silver, *BPMN Method and Style: A levels-based methodology for BPM process modeling and improvement using BPMN 2.0*, Cody-Cassidy Press, 2009.
[17] M. Glinz, “On non-functional requirements,” *Proc. RE*, vol. 7, 2007, pp. 21-26.
[18] L. Chung, “Representation and utilization of non-functional requirements for information system design,” *Advanced Information Systems Engineering, Proc., 3rd Int. Conf. CAiSE*, Springer, , pp. 13–15.
[19] J.J. O'Sullivan, “Towards a precise understanding of service properties,” 2006.
[20] D. Tapscott and A.D. Williams, *Wikinomics: How Mass Collaboration Changes Everything*, Portfolio, 2006.
[21] O.E. Williamson, “Transaction-cost economics: the governance of contractual relations,” *The journal of Law and Economics*, vol. 22, 1979, pp. 233-261.
[22] F. Zielke, “Incident Management,” *ITIL V3 umsetzen*, Düsseldorf: Symposion Publ., 2009, pp. 77-122.
[23] Z. Du, F.C. Lau, C. Wang, W. Lam, C. He, X. Wang, Y. Chen, and S. Li, “Design of an OGSA-Based MetaService Architecture,” *Grid and Cooperative*

Computing – GCC 2004, 2004, pp. 167-174.
[24] S. Lee and J. Choi, "Meta Services: Abstract a Workflow in Computational Grid Environments," *Computational Science – ICCS 2005*, 2005, pp. 916-919.
[25] J. Buford, A. Brown, and M. Kolberg, "Meta service discovery," *Pervasive Computing and Communications Workshops*, Pisa: 2006, p. 6.
[26] A. Friday, N. Davies, N. Wallbank, E. Catterall, and S. Pink, "Supporting service discovery, querying and interaction in ubiquitous computing environments," *Wireless Networks*, vol. 10, 2004, pp. 631-641.
[27] J. Hau, W. Lee, and S. Newhouse, "Autonomic service adaptation in ICENI using ontological annotation," *Grid Computing, 2003. Proceedings. Fourth International Workshop on*, 2003, pp. 10-17.
[28] J. Hau, W. Lee, and S. Newhouse, "The ICENI semantic service adaptation framework," *UK e-Science All Hands Meeting*, 2003, pp. 79-86.
[29] S. Qi and W. Hai-yang, "Meta Service in Intelligent Platform of Virtual Travel Agency," *Semantics, Knowledge and Grid, 2005. SKG'05. First International Conference on*, 2005, pp. 116-116.
[30] F. Casati, M. Sayal, and M.C. Shan, "Developing e-services for composing e-services," *Lecture notes in computer science*, 2001, pp. 171-186.
[31] C.H. Crawford, G.P. Bate, L. Cherbakov, K. Holley, and C. Tsocanos, "Toward an on demand service-oriented architecture," *IBM Systems Journal*, vol. 44, 2005, pp. 81-107.
[32] T. Norman, N. Jennings, P. Faratin, and A. Mamdani, "Designing and implementing a multi-agent architecture for business process management," *Lecture Notes in Computer Science*, vol. 1193, 1997, pp. 261-276.
[33] R.F. Rodrigues and L.F.G. Soares, "Inter and intra media-object QoS provisioning in adaptive formatters," *Proceedings of the 2003 ACM symposium on Document engineering*, ACM New York, NY, USA, 2003, pp. 78-87.
[34] V.H. Publishing, *IT Governance based on Cobit 4.1 - A Management Guide*, Van Haren Publishing, 2007.
[35] M. Bitsaki, O. Danylevych, W.J. van den Heuvel, G. Koutras, F. Leymann, M. Mancioppi, C. Nikolaou, and M.P. Papazoglou, "An architecture for managing the lifecycle of business goals for partners in a service network," *Proceedings of the 1st European Conference on Towards a Service-Based Internet*, Springer, 2008, pp. 196-207.
[36] A. Charfi and M. Mezini, "Ao4bpel: An aspect-oriented extension to bpel," *World Wide Web*, vol. 10, 2007, pp. 309-344.
[37] T. Erl, *Service-oriented architecture: concepts, technology, and design*, Prentice Hall PTR Upper Saddle River, NJ, USA, 2005.