Dominik Birkmeier, Sebastian Klöckner, and Sven Overhage

A Survey of Service Identification Approaches

Classification Framework, State of the Art, and Comparison

Due to their modular nature, the adoption of Service Oriented Architectures (SOA) for business applications promises many advantages. The successful introduction of SOA depends on an efficient methodical support of the underlying new development paradigm, though. As the amount of current literature illustrates, especially the development of systematic methods for the identification of suitable services, which can serve as building blocks of business applications, is presently a focal point of interest. The different approaches presented in literature, however, significantly diverge with regard to their concepts and procedures. In this paper, we therefore analyse the current state of the art in service identification and highlight differences between the presented approaches. To evaluate proposed service identification approaches, we introduce a classification scheme with distinguishing factors. We use this scheme to compare and analyse the various approaches. Based on this comparison, we elaborate on individual strengths and weaknesses of approaches from which implications for practice are deduced. Finally, we identify areas of future research that remain to be addressed in order to further advance the state of the art in service identification.

1 Motivation

Today, business application development is facing a whole set of demanding challenges. Among them, managing the complexity of applications, flexibly adapting applications to changes in the business environment, and extending existing applications to quickly implement new functionality are key challenges, which have to be solved by adequate development techniques (Brown, 2000; Cheesman and Daniels, 2001; Papazoglou et al., 2006). With their modular development paradigm, Service Oriented Architectures (SOA) offer a promising contribution to better meet all of these challenges (Kossmann and Leymann, 2004; Papazoglou et al., 2006).

A prerequisite for the success of the new, service-oriented development paradigm in practice is however a sufficient support with adequate methods and tools (Papazoglou et al., 2006). Besides the questions of how services can be described, found in catalogues, and composed (according to business requirements), especially the development of methods and practices for a systematic identification of services is in the focus of scientific as well as practical interest (Arsanjani, 2004; Erradi et al., 2006).

The identification of suitable services, which has to be accomplished at the beginning of the development process, provides the basis for the next design steps as well as for the service composition and usage later on (Erl, 2005). For this reason, it is of central importance for the service-oriented development process as a whole and has accordingly been addressed by a variety of approaches which have been published in literature. These approaches, however, show a significant heterogeneity. They range from ad-hoc findings (which have been gathered by creative thinking or charting an initial project) and general recommendations (which should be considered during the identification of services) to structured methods and algorithmic procedures. Moreover, the underlying service definitions as well as their respective strategy to identify services vary significantly. Due to the short time since a systematic identification of services is in the focus of research,

none of the approaches was so far able to become broadly accepted and dominate the others. Comparative examinations, which assess the evolving approaches and help structuring the area of research, are missing as well. For academia, such an assessment helps identifying complementary approaches, which could be combined to obtain improved results, as well as uncovering unresolved issues for further research. For practice, a detailed comparison reveals consequences for the applicability of the proposed approaches in different development scenarios and contexts.

In this paper, we present a survey of service identification approaches, which we classify according to a detailed scheme with distinguishing factors. To determine relevant factors as well as the eligible service identification approaches themselves, we conducted an exhaustive literature study. Starting from a compilation of service identification approaches already known by the authors, we systematically analysed relevant conferences and journals (both of the information systems and software engineering disciplines) to collect a first set of approaches. We then traced the citations to search backwards and used Google Scholar as well as the Web of Science to look for upcoming approaches. Our research approach is thereby based on the methodology for literature analyses as described by Webster and Watson (2002).

Our survey is structured as follows: Section 2 gives an overview of existing related work to further motivate the research gap. In section 3, we determine and discuss characteristic criteria of service identification approaches. We then refine these criteria into a set of distinguishing factors to evaluate and classify service identification approaches. The distinguishing factors are therefore aggregated into a detailed classification scheme. In section 4, we present the service identification approaches identified during the literature survey and analyse them according to the classification scheme. Commonalities and differences between the presented approaches are highlighted during a comparative discussion based on an argumentative-deductive approach. After describing the current state of the art in service identification and uncovering areas requiring further research, we conclude by summarizing key findings and outlining future directions to further improve existing service identification approaches.

2 Related Work

The development of systematic service identification approaches is, especially due to the short time since this research area is under investigation, still in progress. Nonetheless, there are numerous publications which propose relevant approaches. But the quality of these contributions and therefore their suitability for the desired *systematic* identification of services in the sense of a methodical procedure is varying significantly. To keep track of the ongoing development, classifying existing approaches and providing an overview of the state of the art is advisable in parallel with the creation of new approaches for service identification.

However, scientific literature offers only a few comparisons of service identification approaches, which present a systematic classification. Even though Beverungen et al. (2008) also evaluate different approaches for service identification as part of their contribution on the conception of a SOA, they mainly focus on the comparison of integrated approaches, which cover the whole development process of a SOA. The presented approaches are therefore only partly focused on the identification of services or the identification is only mentioned aside. Specialised approaches for service identification were mostly left unconsidered, as they do not have an integrated procedure. In addition, the comparison criteria stated by Beverungen et al. seemed to be arbitrarily selected and not systematically deduced from literature.

A comparison of approaches for the identification of software components, which could provide important insights for the design of corresponding approaches in the area of service orientation, has been provided by Wang et al. (2005). This contribution only offers criteria which give insights into

the used procedure, but does not mention other aspects. It primarily distinguishes approaches which use a so called domain engineering strategy. Their comparison, however, can hardly be mapped to the approaches presented in this paper, as most of them do not contain a comparable procedure. This might be a first indicator that many of the existing service identification approaches are still in a premature state or at least diverge from established approaches of the component-based software engineering discipline.

3 Classification Scheme for Service Identification Approaches

A thorough analysis of service identification approaches published in literature reveals different layouts, e.g., with respect to their conceptual design and the identification strategy. In order to compare and classify the different approaches, we firstly introduce a set of criteria which characterizes service identification methods and provide insights into the features of such approaches. For the deduction of characteristic criteria, we refer to research focusing on the conception of systematic design methods in the software engineering (Sommerville, 2006) as well as in other engineering disciplines (Pahl et al., 2007). From there, we take the following criteria as being characteristic for systematic methods in general:

- the conceptual foundations, on which the approach is built;
- the procedure that is applied by the approach;
- the underlying model used by the approach;
- the supporting measures, which improve its application in practice.

Examining this rather compact set of general criteria allows a better understanding of whether an approach is able to contribute to the aspired systematic identification of services and where deficiencies exist. While these abstract criteria might not necessarily be complete, they have been proven to adequately describe systematic development approaches in theory (Sommerville, 2006; Pahl et al., 2007). For this reason, we used them as a starting point for building our classification scheme and refined them as documented below to describe service identification approaches in particular.

3.1 Foundations

The conceptual design of service identification approaches is manifested in its foundations. They describe the understanding of central concepts, in particular the underlying service definition of the respective approaches. Furthermore, the different approaches have to be distinguished with respect to their degree of formalisation and their integration into a comprehensive development process model. While the degree of formalisation provides information about how exactly the service identification strategy is described, the latter indicates whether the approach has been designed to work with data created during earlier development activities and provides specific results for later design steps.

3.1.1 Service definition

In accordance with the fact that a standard service terminology has not been established yet, published identification approaches make use of different service definitions which are being discussed in literature. While services can be broadly defined as ‘acts of performance offered by one party to another’ (Lovelock et al., 1999), more specific service definitions focus on a variety of additional aspects (Alter, 2008). Such specific service definitions generally either built upon a more technically oriented viewpoint in the sense of (Web) services as being software components (Arsanjani, 2004; Mcgovern et al., 2006; Natis, 2003; Szyperski et al., 2002) or take a domain-oriented perspective (Alter, 2008; Barros and Dumas, 2006) to concentrate on conceptual aspects such as the actual

business function performed by a service.

Technically oriented service definitions often focus on how to specify and implement services as software artefacts that provide a distinctive functionality. Properties like a loose coupling, reusability, platform independence, or well defined service interfaces are at the core of such definitions (Szyperski et al., 2002). By contrast, domain-oriented service definitions emphasize that services should provide self-contained sets of functionality which are meaningful from a business perspective. In such definitions, a service is typically understood as an activity of a business application system, which supports the accomplishment of a certain set of business tasks (Alter, 2008). Technical aspects are, accordingly, of secondary concern.

Since technically and domain-oriented service definitions diverge in central aspects, they are likely to promote different results when being taken as the basis to identify suitable services. To reflect these general differences in the following, we distinguish between approaches with a domain-oriented focus from those with a more technically oriented service understanding. While a more detailed analysis of the underlying service definitions would also be a desirable research goal, we can only differentiate between the two mentioned archetypes of service definitions in this paper. Gathering and discussing the various service definitions had to be left as a direction of future research. We also did not examine the service definitions underlying the new service science discipline (Chesbrough and Spohrer, 2006), which focuses on the engineering of services in general. Here, we aim at comparing service identification approaches that promote the development of a SOA for a business application system. Service definitions and approaches belonging to the service science discipline are therefore out of the scope of this particular survey.

3.1.2 Degree of formalisation

The degree of formalisation ranges from a presentation of so called ad-hoc findings and general recommendations to structured methods and algorithmic procedures. Ad-hoc findings are based on creative thinking or experiences gathered by charting an initial project. Typically, they offer only a fuzzy strategy to identify services. General recommendations are proven practices that have been repeatedly applied to identify services with certain desirable characteristics. While they are based on more thoroughly researched findings, they usually concentrate on specific aspects or best practices that should be taken into consideration, but do not combine these into a systematic procedure. Structured methods, in contrast, provide the designer with detailed work steps and arrange them into a service identification process. They also provide clearly specified identification criteria. Algorithmic procedures finally comprise a formal plan that combines individual work steps into a comprehensive workflow.

3.1.3 Overall development process model

The identification of services is usually part of a software development project which is guided by a development process model. This process model defines the in- and output of major development phases (such as design, implementation etc.) and coordinates the usage of achieved results in subsequent phases (Cheesman and Daniels, 2001; D'Souza and Wills, 1999).

Service identification approaches should ideally be integrated into an overall process model. Such an integration predefines which development phase has to deliver the information taken as input and how identified services have to be described to be useful as input for subsequent phases of the development process. Renouncing an integration into an overall process model carries the danger that results of earlier phases have only limited value for subsequent phases.

3.2 Procedure

The procedure describes what kind of technique is applied by an approach to identify services. The following aspects are taken into account:

3.2.1 Direction

The analysis to identify services can generally be carried out in two directions. Top-down approaches (Mills, 1971) use domain-specific conceptual models (like business concept and process models) to identify services, which are then specified and mapped onto a software landscape. In contrast, bottom-up approaches start by analysing the existing software landscape and modularising it. Identified modules of this landscape will then be equipped with meaningful domain-specific semantics after the identification.

Since unidirectional top-down as well as bottom-up approaches carry the risk of leading to undesirable results, e.g., by identifying services that might not be suitable from the opposite technical or business-oriented viewpoint, some approaches try to combine both directions and strive for a compromise between a domain and a technology centric view. These will be distinguished as meet-in-the-middle approaches in the following.

3.2.2 Optimisation approach

An important characteristic of service identification approaches is whether they try to find an optimized solution, i.e., services with presumably optimal properties in terms of the designer's preferences. Optimizing approaches, e.g., often try to identify a set of services with maximal cohesion and minimal dependencies, following a principle already stated by Parnas (1972). Such preferences can be translated into mathematical optimisation problems and then be approached with appropriate techniques (e.g., clustering methods). Thereby, one has to distinguish between approaches that use exact or heuristic methods. Exact methods find the overall best solution (a global optimum), while heuristics come up with the best solution that can be found with reasonable effort (a local optimum).

3.3 Model

Generally, the identification of services is based on conceptual models which reflect reality. From a theoretical perspective, a systematic approach for the identification of services should, at least partly, use the model views introduced by general systems theory (Bertalanffy, 1976), which can also be applied to information systems (Yourdon and Constantine, 1979). With respect to the content and complexity of the utilized models, the examined approaches diverge significantly. Differences become apparent regarding the analysed model views, the consideration of legacy structures and system dependencies as well as a differentiation of service hierarchies and predefined service types.

3.3.1 Model views

Independent of the question whether a domain-oriented or technical perspective is being used, socio-technical systems in general and software systems in particular can always be described from three model views, which stem from general systems theory and are widely used in software design methods such as Syntropy, Catalysis, or ARIS (Cook and Daniels, 1994; D'Souza and Wills, 1999; Scheer, 2000): the data view describes processed information objects as well as their respective structure as system attributes. The functions view documents the system behaviour and combines system attributes as inputs and outputs. In addition, a functional decomposition describes the relationship between complex functions and their sub-functions. The process view finally describes the temporal relationships between functions and combines them to workflows.

Basically, the identification of services can take all three model views into account, since only their synopsis provides a comprehensive view. Many approaches, however, only use a subset of these

model views, which leads to specific advantages and drawbacks.

3.3.2 Consideration of existing structures

The identification of services is often performed in an existent software environment with legacy systems or services in place. Therefore it has to be evaluated if the respective approaches consider existing structures appropriately and weave them into their procedure.

3.3.3 Consideration of system dependencies

Services normally provide their functionality only in cooperation with other services. They are consequently developed to be interconnected with other services (Szyperski et al., 2002). Service identification approaches might therefore aim at finding sets of collaborating services with thoroughly analysed interdependencies and explicitly specify the remaining interdependencies with peripheral systems. Others instead concentrate on identifying single services and disregard potential dependencies with the environment.

3.3.4 Differentiation of service hierarchies

During the identification, one can generally distinguish between complex services, which themselves are composed of services, and elementary services, which are not to be further divided into smaller services. Identification approaches which explicitly support such a distinction follow the hierarchical systems concept of general systems theory (Bertalanffy, 1976) and implement a stepwise decomposition until no complex services are identified any more (Atkinson et al., 2002). Others do not explicitly support a stepwise decomposition and leave the structuring of a composition into a hierarchy to the designer.

3.3.5 Differentiation of predefined service types

Some of the service identification approaches distinguish services of predefined types (Beverungen et al., 2008). Often, these approaches differentiate between services whose primary purpose is the management of data (Entity Services) and those which coordinate and execute application-specific tasks (Task Services). Such an identification procedure inherently leads to a separation of data and task-specific services.

It is debatable, however, if such procedures deliver an optimal result, especially since many authors argue for a grouping of data and related tasks into a single part (Parnas, 1972). Other approaches, therefore, do not build upon a distinction of predefined service types.

3.4 Supporting measures

Supporting measures enhance the applicability of service identification approaches in practice. They can be classified into tool support, quality assertions, and evaluation.

3.4.1 Tool support

The practical applicability of service identification approaches can be enhanced by providing software tools which guide the designer through the identification process and help to manage complexity. The absence of such supporting tools hampers especially a possible optimisation of service identification results.

3.4.2 Quality assertions

The quality of an identification result produced by a certain approach has a significant impact, since the implementation and roll-out of new software artefacts is always associated with considerable strategic and financial risks.

Therefore, a service identification approach is ideally able to guarantee the correctness of its result.

Especially for an algorithmic procedure it is important to avoid local optima. If a formal guarantee is not feasible, e.g., due to method-inherent restrictions as in the case of heuristic procedures, approaches should at least support other kinds of quality assertions. They could for example state the maximum deviation from an optimal solution through an upper and lower bound approximation (Jungnickel, 2008) or allow a sensitivity analysis.

3.4.3 Evaluation

An evaluation of service identification approaches ensures their correctness and proofs their applicability in practice. While a plausibility check demonstrates the principal correctness, only comprehensive use cases and best practices reveal terms of use as well as possible areas of application and limitations of a certain approach. Ideally, an identification approach is evaluated by multiple applications in practice and complemented with 'Best Practices', which help to ease its application.

3.5 Classification Scheme

The previously mentioned distinguishing factors are the basis to form a classification scheme as depicted in Table 1. Thereby, values of the identified distinguishing factors have been summarized as a morphological box and will be used to classify individual identification approaches later on.

When looking at the classification scheme as a whole, one might suspect that the depicted distinguishing factors are not independent from each other. A bottom-up approach to identify services might, e.g., probably use a technical service definition. Similarly, an approach that uses matrices to analyse relationships between design elements as part of its identification strategy might probably do this in a formalized (algorithmic) procedure. When analysing the distinguishing factors closely, it becomes obvious that they are orthogonal to each other, however. Accordingly, the apparent coincidences described above can easily be proven to be wrong: first of all, it is quite conceivable that a bottom-up approach might also use a domain-oriented service definition. Such an approach will start to identify services by analysing conceptual models of an existing software landscape. Services will accordingly be identified from existing systems by analysing them from a domain-oriented perspective and mapping results back onto the existing software landscape. In the same manner, identification approaches might as well use matrices to analyse relationships between design elements, but not conduct the analysis in an algorithmic procedure.

4 Classification of Service Identification Approaches

In this section, we provide an overview of the state of the art in service identification, and elaborate on various approaches published in literature. After introducing specialised service identification approaches in section 4.1, we extend the compilation in section 4.2 with general modularisation approaches. The examined approaches will then be compared and classified in section 4.3 according to the previously defined criteria. Section 4.4 concludes with important implications that can be drawn for practice and academia.

4.1 Service identification approaches

4.1.1 Service-Oriented Analysis and Design (Zimmermann et al. and Arsanjani)

Zimmermann et al. examine the applicability and transferability of established software engineering methods to the introduction of SOAs (Zimmermann et al., 2004). Elements of Object-Oriented Analysis and Design (OOAD), Enterprise Architecture (EA) Frameworks and Business Process Modelling (BPM) are combined and expanded to form a Service-Oriented Analysis and Design (SOAD) approach. Although they define quality factors for SOAD and give general recommendations for all phases of the adoption process, they do not

Classification Scheme for Service Identification Approaches				
Service definition	None	Technical	Domain-oriented	
Degree of formalization	Ad-Hoc	General Recommendations	Structured	Algorithm
Development process model	No	Yes		
Direction	Top Down	Bottom Up	Meet In The Middle	
Optimizing approach	None	Heuristic	Exact	
Model views	Data	Functions	Processes	
Consideration of existing structures	No	Yes		
Consideration of system dependencies	No	Yes		
Differentiation of service hierarchies	No	Yes		
Differentiation of predefined service types	No	Yes		
Tool support	No	Yes		
Quality assertions	None	Sensitivity analysis	Quality guarantee	
Evaluation	None	Plausibility check	Use Case	Best Practices

Table 1: Classification scheme for service identification approaches

present an overall process model. While most of the mentioned model criteria of section 3.3 are addressed, both a service definition and concrete recommendations are missing.

In regard to service identification, it is pointed out that a SOA is usually not introduced in a greenfield approach. Therefore, a pure top-down approach would not be sufficient as existing structures have to be taken into account. The poor applicability of classical development methods to identify services is commented by the authors with 'there is room for additional creative thinking.' Zimmermann et al. (2004), but they do not reveal any alternatives. The presented theoretical example, which is used for demonstration purposes, underlines the recommendatory character of their contribution, which can be used as starting point for further research.

Based on the SOAD approach of Zimmermann et al., Arsanjani (2004) articulates concrete recommendations for the identification, specification and realisation of services. Following a mainly technical perspective, especially the identification phase is concretised. The execution of top-down and bottom-up approaches is extended with a goal service modelling in order to find yet unidentified, but needed services. This mainly theoretical approach without any reference examples again does not exceed the state of a loose collection of general advices.

4.1.2 Service Oriented Analysis (Erl)

In his books about the conception and design of service-oriented architectures, Erl describes an approach for the identification of services in the context of an overall development model (Erl, 2005, 2007). Based on an analysis of business processes and existing system structures, service candidates

	Zimmermann et al. [ZiKr+04]	Arsanjani [Arsa04]	Erl ([Erl05], [Erl07])	SAP [SAP05]	Aier [Aier06]	Erradi et al. [ErAn+06]	Inaganti, Behara [InBe07]	Winkler [Wink07]	Beverungen et al. [BeKn+08]	IBM [IBM84]	Szyperski et al. [SzGr+02]	Jain et al. [JaCh+01]	Albani et al. ([AlDi+05], [AlDi06], [AlOv+08])
Year of Publication	2004	2004	2005, 2007	2005	2006	2006	2007	2007	2008	1984	1998, 2002	2001	2005, 2006, 2008
Service definition	None	Technical	Technical	Technical	None	Domain-oriented	None	Domain-oriented	None	n/a	n/a	n/a	n/a
Degree of formalization	General Recommendations	General Recommendations	General Recommendations	General Recommendations	Algorithm	Structured	General Recommendations	Structured	Structured	Structured	General Recommendations	Algorithm	Algorithm
Development process model	✓	✓	✓	✓	✓	✓	✗	✓	✓	✗	✗	✗	✓
Direction	Meet In The Middle	Meet In The Middle	Meet In The Middle	Meet In The Middle	Meet In The Middle	Meet In The Middle	Meet In The Middle	Top Down	Top Down	Top Down	n/a	Top Down	Top Down
Optimizing approach	✗	✗	✗	✗	✓ (heuristic)	✗	✗	✗	✗	✓ (heuristic)	✗	✓ (heuristic)	✓ (heuristic)
Model views	Processes and data	Processes, data and functions	Processes, data and functions	Processes, data and functions	Processes and data	Processes and data	Processes, data and functions	Processes and functions	Processes	Processes and data	n/a	Processes and functions	Processes and data
Consideration of existing structures	✓	✓	✓	✓	✗	✓	✓	✗	✓	✗	✓	✗	✓
Consideration of system dependencies	✗	✓	✗	✓	✓	✓	✗	✗	✓	✓	✓	✓	✓
Differentiation of service hierarchies	✗	✓ (2)	✓ *	✓ (2)	✗	✓ (2)	✓ (2)	✗	✓ (2)	n/a	n/a	n/a	n/a
Differentiation of predefined service types	✗	✗	✓ *	✓	✗	✗	✗	✗	✓ (2)	n/a	n/a	n/a	n/a
Tool support	✗	✗	✗	✗	✓	✓ **	✗	✗	✗	✗	✗	✓	✓
Quality assertions	None	None	None	None	None	None	None	None	None	None	None	None	None
Evaluation	None	None	Plausibility check	None	Use Case	Plausibility check	None	Use Case	Use Case	Use Case	None	Use Case	Use Case

* 11 Service types, ** only sub-steps

Table 2: Classification of service identification approaches

are identified, which can then be refined into services. In a meet-in-the-middle approach, Erl differentiates eleven service hierarchies and classes, which sometimes depart from his general, rather technical definition of services. Dependencies between services are only mentioned aside. Tool support as well as quality assertions are not discussed. The approach is part of an overall development process model, but it only offers recommendations and general instructions instead of explicit procedures. The application of the overall approach is however illustrated in case studies.

4.1.3 Enterprise Service Design Guide (SAP)

In 2005, the SAP AG published an Enterprise Service Design Guide in the context of its SAP Developer Network (SAP, 2005). Besides basics of enterprise services, this guide also comprises the discovery and design phases. Based on a rather technical definition of services, it describes how services can be identified on two hierarchical levels following a meet-in-the-middle approach. The approach offers 16 different indicators in order to help designers of Enterprise Services identifying potential services based on business processes and associated scenarios. The subsequent division into simple and composite services is supported by 10 guidelines. With this document, the authors offer a manual for the identification of services. The application of the approach is left to the designer, though.

4.1.4 EA Builder (Aier)

Aier presents an approach for the identification of an enterprise architecture and its related services based on business processes and IT systems (Aier, 2006). Different architectural views are mapped onto graphs and then partitioned based on a clustering algorithm, which was initially developed for the identification of communities within social networks. The underlying service definition and many other details of the meet-in-the-middle approach remain unclear, however. While a differentiation of service hierarchies and types is mentioned, its realisation in the supporting tool, the EA Builder, is not further addressed. A quality assertion of the results is missing, but the approach has been tested on the basis of a use case.

4.1.5 SOA Framework (Erradi et al.)

The identification of services, as part of an architectural SOA Framework (SOAF), which covers the specification and realisation phases, is presented by Erradi et al. (2006). Based on an analysis of business processes, needed services are identified in a top-down approach. Existing services are extracted from the code base and the related data structures. By comparing needed and existing services, additionally required services are identified. A so-called tool based mining supports the bottom-up analysis of code and data fragments. The top-down analysis of the business processes is realised by a combination of interviews and tools. The approach does not present a concrete definition of services and, besides notes about possible tool support, no tools are mentioned nor does it offer a procedure for matching needed with existing services. While the authors present a case study, their explanations only cover central results and do not document the practical application of the approach at all.

4.1.6 BPM and SOA Handshake (Inaganti and Behara)

A structured approach for the identification of services is offered by Inaganti and Behara (2007). Potential services are identified and opposed to each other in four steps using a top-down as well as a bottom-up approach. Additionally needed services are added in an undefined way. While this contribution describes the identification in a more structured and detailed manner than Zimmermann et al. (2004) and Arsanjani (2004), it rarely exceeds the level of a listing of possibilities and recommendations. Optimisation methods as

well as tool support are not considered. A reference example is not given either.

4.1.7 Identification and design of services (Winkler)

Winkler (2007) presents an approach which covers service identification as well as the design and realisation of services. During the identification phase services are defined based on UML activity diagrams. These services have to comply with three previously defined criteria, namely reusability of services, avoidance of redundant implementation of different services as well as loose coupling of services based on well-defined and simple interfaces. The service identification itself has four subsequent steps: creation of activity diagrams, rework of activity diagrams, identification of services, and analysis of usage frequency. On the basis of an implicit business-oriented service definition, the service identification follows a semi-structured top-down approach. An optimisation of determined services is not part of the identification phase and the compliance of the identified services with the previously defined criteria is not validated. While service hierarchies, service dependencies and structures are mentioned, it stays unclear how these aspects affect the service identification. The whole process is described on the basis of an example from the financial service sector. A supporting tool is not mentioned.

4.1.8 Method for the conception of SOA (Beverungen et al.)

Beverungen et al. (2008) compare different approaches for the development of SOAs. As a result, they offer an own approach, which covers the phases of service identification and specification and is integrated into an overall development process model. Services are identified through a top-down decomposition of business processes. Special attention is placed on an analysis of so-called transfer and visibility potentials of single process steps for business partners. While existing structures and services are taken into account during the identification phase, the dependencies between services are only considered in the specification phase. Service hierarchies are divided into two types, Process and Basic. A differentiation of service types is mentioned, but not integral part of the approach. Although a structured identification process is propagated, further details about such a process are missing. Moreover, neither a possible optimisation nor a supporting tool is mentioned. A use case demonstrates the practicability of the approach, but does not describe any details of the sub-steps.

4.2 General modularisation approaches

4.2.1 Business systems planning (IBM)

IBM describes an approach to systematically decompose business information systems on the basis of business process and data models (IBM, 1984). The approach offers detailed steps and procedures to identify system modules by examining the relations between process activities and data objects in a matrix analysis. Based on a heuristic optimisation procedure, the grouping of process activities is rearranged to minimize the number of shared data objects between the groups of activities. A quality assertion for the results of this optimisation is not given and existing system structures cannot be incorporated into the presented approach. Furthermore, it remains unclear how the rearrangement of process activities should be conducted.

4.2.2 Modularity criteria (Szyperski et al.)

In his book about component-based software engineering, Szyperski introduces 15 modularity criteria which should ideally be satisfied by identified components and services (Szyperski et al., 2002). According to these criteria, services should not only be self-contained with respect to their functionality, but also be independently implementable, installable, and maintainable. In ad-

dition, they should be independent with respect to billing and handling of liability issues. While the criteria are formulated in detail, they do not exceed the level of general recommendations. A structured procedure with concrete work steps to guide the designer is missing completely. The approach can therefore only be characterized as a conceptual framework which might be used to validate identified services.

4.2.3 CompMaker (Jain et al.)

Jain et al. (2001) present an approach which originates from the domain of component-based application development. Based on the Analysis Level Object-Model, a business domain model in UML notation, the approach identifies reusable components following a top-down approach. The domain model contains at least object-oriented class diagrams, use cases and sequence or interaction diagrams. Structural and dynamic relationships between the different objects in the domain model are used to compute the Class Relationship Strength.

In a first step these relationships are used to identify components through a grouping of classes by applying a hierarchical agglomerative clustering algorithm. This initial solution is then improved through automated add, move or exchange heuristics, as well as manual interventions. While classes, as basic building blocks of components, play an important role for the approach, the final results are strongly influenced by the designer's preferences as different measures for the best possible solution can be applied. The identification process is supported by the CompMaker tool and illustrated on the basis of a case study from the automotive insurance sector. An evaluation of identification results is not mentioned.

4.2.4 BCI and BCI-3D (Albani et al.)

In (Albani and Dietz, 2006; Albani et al., 2005, 2008), Albani et al. describe the Business Component Identification (BCI) method, as well as its enhancement to BCI-3D. Both versions identify components by using algorithms that work on process and data structures of a domain model. The component identification follows an algorithmic top-down approach, which is supported by specialised tools and part of the Business Component Modelling Process (BCMP). BCI in its original version (Albani et al., 2005) is adapted from IBM (1984) and considers only the dependencies between single functions and data objects. By contrast, BCI-3D (Albani and Dietz, 2006; Albani et al., 2008) uses graph-based clustering methods, which identify components by combining an opening and an improving heuristic from graph theory. Information about data objects, process steps and actors, plus their relationships is mapped onto vertices and edges of a graph. Weights are then assigned to the edges depending on the relation type and the designer's preferences. A quality assertion for the resulting solution is not given and legacy structures as well as existing dependencies can only partly be included by integrating them into the domain model. Case studies have been conducted for both methods (Albani and Dietz, 2006; Selk et al., 2005). The BCI-3D tool supports the designer during the identification process, but missing advices for the assignment of weights hamper the application of the proposed method.

4.3 Classification and comparative discussion

The identification of services and modules has been an area of continuous research. Whereas all of the specialised service identification approaches were published between 2004 and 2008, the more general modularisation approaches are mainly older and dated before 2004. An overview of the classification results of all approaches is summarized in Table 2. It shows that none of the 9 specialised and the 4 general approaches cover all aspects sufficiently, but rather all of them have their individual strengths and weaknesses. Below follows a differentiated examination and comparison of the approaches.

4.3.1 Foundations

The underlying *service definitions* vary heavily between the approaches. More technical definitions are used in three cases (Arsanjani, 2004; Erl, 2005, 2007; SAP, 2005) and two approaches are built upon domain-oriented definitions (Erradi et al., 2006; Winkler, 2007). However, an interesting observation is that the authors of eight approaches identify services without any definition of what they try to find. This is understandable for the four general approaches not aiming specifically at service identification, but even four of the explicit service identification approaches do not provide any definition (Aier, 2006; Beverungen et al., 2008; Inaganti and Behara, 2007; Zimmermann et al., 2004). Furthermore, even if a definition is given, it is often times imprecise and therefore handicaps a comparison of the approaches.

The *degree of formalisation* is in four cases structured (Beverungen et al., 2008; Erradi et al., 2006; IBM, 1984; Winkler, 2007) and an algorithm is provided for three approaches (Aier, 2006; Albani et al., 2008; Jain et al., 2001). In the remaining six cases general recommendations are given. It is noticeable that the older, general modularisation approaches are overall more formalized than those originating from the newer SOA discipline. On the other hand, most of the newer approaches are embedded in a development process model and thus better support an integrated design than the older ones.

Generally, the amount of information and substeps explained varies noticeably in the evaluated literature, ranging from detailed step-by-step instructions to rather coarse-grained explanations. A low level of detail especially hampers the applicability of an approach. Interestingly, the level of detail is not correlated with the degree of formalisation, which one might have expected.

4.3.2 Procedures

The *direction* of the analysis to identify services is meet-in-the-middle for most (seven) cases, whereas only two specialised approaches (Beverungen et al., 2008; Winkler, 2007) and three general approaches (Albani et al., 2008; IBM, 1984; Jain et al., 2001) follow a top-down course. Also, not a single one of the approaches uses a bottom-up strategy. Generally, it seems like the specific service identification approaches tend to consider technical information more often in combination with business domain information than the more general modularisation techniques which solely rely on the business domain.

One of the explicit service identification methods uses a heuristic to *optimize* the results (Aier, 2006), whereas the other eight do not implement any optimisation. The general approaches on module identification are more advanced in this category, as three of the four methods try to optimize the identified service structure using a heuristic (Albani et al., 2008; IBM, 1984; Jain et al., 2001).

4.3.3 Model

All of the strategies are based on a process *view* of the model. Oftentimes this is completed by additionally considering information on functions and/or data. *Existing structures* and *system dependencies* are each taken into account by nine of the approaches. This is usually granted for the meet-in-the-middle approaches, but for the top-down approaches it can only be achieved through an integration of the information into the domain model. Two third of the proposed techniques for service identification care for *service hierarchies* and include corresponding arrangements in their strategy. The differentiation of predefined *service types* is covered in one third of the papers. Service hierarchies and service types are not applicable for the non service-specific approaches.

4.3.4 Supporting measures

A support of the proposed service identification approaches through corresponding tools is only mentioned in two of the service-specific (Aier, 2006; Erradi et al., 2006) and two of the general

modularisation approaches (Albani et al., 2008; Jain et al., 2001). Quality assertions are so far completely missing for all of the approaches. For the evaluation of the approaches, use cases and plausibility checks are provided in eight cases, but none is evaluated through best practices. Overall, this does not go far enough to ensure a high quality solution. However, this would be crucial for the further development of the procedures.

4.4 Implications

Several implications can be derived from the identified state of the art for researchers in the field of service identification methods, as well as for software engineers of service-oriented software systems. The former ones can use the results from Table 2 to identify areas requiring further research, improve their own approaches and fill out the blanks. To improve the usability of several methods, the analysis of the supporting measures shows that software tools are needed, especially in the case of optimizing approaches. Furthermore, the given quality assertions and evaluations are mostly quite rudimental. Additionally, researchers might be able to use combinations of existing service-specific and general approaches for further improvements of the state of the art in service identification. For example, we see a good chance that the BCI approach from Albani et al. (2008) might be successfully combined with the technique from Aier (2006), since the former one identifies components from a domain model through graph-based clustering methods and the latter one elaborates on algorithms derived from social networks.

A software engineer can use the provided comparison of approaches to select one that is most appropriate for his/her particular development scenario. Above all, the utilized service definition, the direction of the approach as well as the required model views provide insights whether an approach is suited to support a particular development scenario or not. In a greenfield software development project, where services can be identified during the early design phases and without taking existing software systems into account, top-down approaches which use a domain-oriented service definition should be chosen. But in general the decision for a specific approach should be depending upon whether the overall goal during the identification is to identify reusable services or to primarily create a modular system design.

In a scenario where existing software systems have to be modularized or at least to be integrated into the identification of services, meet-in-the-middle approaches should be preferred. These approaches either start from existing software systems and aggregate implementation classes to form services or at least take existing software structures into account during the identification of services. As the classification shows, an integrated service identification approach, which combines the strengths of the mentioned approaches and is able to cover all depicted scenarios, is not available so far. Therefore, it currently depends on the knowledge of the designer, if a suitable approach is chosen and useful results can be achieved. Our paper thus provides useful insights by identifying and detailing on the state of the art.

By analysing the classified approaches from a chronological perspective, it becomes furthermore obvious that the approaches related to the older discipline of component-based application development usually possess more structured, algorithm-based and better evaluated procedures compared to the approaches that specifically support the identification of services. It also has to be highlighted that most of the service-specific identification approaches come without an explicit service definition. For the designer, it therefore often remains unclear which criteria of services are assumed and guaranteed by the approaches during the identification process. In conclusion, service-specific approaches hence appear to be in a comparatively premature state. Additional research effort is therefore required to further advance the state of the art and support an engineering approach to identify services.

5 Conclusions and future directions

In this paper, we compared several approaches for the identification of services, which have been published in literature, and discussed their individual strengths and weaknesses. The discussion was based on a classification scheme that contains various characteristics of service identification approaches as distinguishing factors and has been specifically developed to compare existent as well as future developments. The assembled characteristics were initially based on results from research focusing on the conception of systematic design methods in general and then refined to characterise service identification approaches. We used the resulting classification scheme to compare various service identification approaches and to reveal differences in their conceptual design as discussed in section 4.3.

It has to be mentioned that the approaches vary in respect to their consideration of dependencies between services as well as different service hierarchies and types. A stepwise refinement, which might become necessary during the service identification process, is therefore not supported by all approaches. Approaches distinguishing so-called Entity (Data) and Task (Process) Services are likely to promote different solutions than those which do not build upon such a predefined distinction of service types. This distinction has to be taken into account when a certain approach is selected.

Significant further research is necessary to answer the question if the existing approaches for service identification can be further developed into mature methods with more formalized and detailed procedures. To realize the aspired systematic service identification as part of an engineering process, existing approaches will eventually have to be enhanced in various aspects. In this context, especially the usage of optimisation methods and procedures, which allow at least an estimation of the solution quality, has to be considered.

It appears to be characteristic that the identification of services does not build upon existing, more mature approaches from the closely related (and older) component-based software engineering discipline. Instead, it seems as if research has started anew with the introduction of SOAs. In fact, this course of action can be observed in many areas of the SOA discipline and is therefore rightfully criticized in literature (Szyperski et al., 2002). A more detailed examination reveals that many modularisation approaches, which were developed to identify business components, as defined by Cheesman and Daniels (2001), could well be used for the design of service-oriented architectures. A synergetic examination of these two disciplines could hence significantly accelerate the development of mature methods for service identification.

References

Aier S (2006) How Clustering Enterprise Architectures helps to Design Service Oriented Architectures. In: IEEE International Conference on Services Computing (SCC 2006), IEEE Computer Society, Chicago, Illinois, USA, pp 269âĂŞ–272

Albani A, Dietz JLG (2006) The Benefit of Enterprise Ontology in Identifying Business Components. In: IFIP World Computing Conference, Santiago de Chile, Chile, pp 243–254

Albani A, Dietz JLG, Zaha JM (2005) Identifying Business Components on the basis of an Enterprise Ontology. In: Konstantas D, Bourrieres JP, Leonard M, Boudjlida N (eds) Interoperability of Enterprise Software and Applications, Springer Verlag, Geneva, Switzerland, pp 335–âĂŞ347

Albani A, Overhage S, Birkmeier D (2008) Towards a Systematic Method for Identifying Business Components. In: Chaudron MR, Szyperski CA, Reussner R (eds) Component-Based Software Engineering, 11th International Symposium, CBSE 2008, Springer, Karlsruhe, Germany, Lecture Notes in Computer Science, vol 5282, pp 262–277

Alter S (2008) Seeking Synergies Between Four

Views of Service in the IS Field. In: 14th Americas Conference on Information Systems (AMCIS 2008), Toronto, ON, Canada

Arsanjani A (2004) Service-oriented modeling and architecture — How to identify, specify, and realize services for your SOA

Atkinson C, Bayer J, Bunse C, Kamsties E, Laitenberger O, Lagua R, Muthig D, Paech B, Wust J, Zettel J (2002) Component-Based Product Line Engineering with UML. The Component Software Series, Addison-Wesley, Boston

Barros AP, Dumas M (2006) The Rise of Web Service Ecosystems. IT Professional 8(5):31–37

Bertalanffy LV (1976) General System Theory: Foundations, Development, Applications. George Braziller, New York

Beverungen D, Knackstedt R, Müller O (2008) Entwicklung Serviceorientierter Architekturen zur Integration von Produktion und Dienstleistung: Eine Konzeptionsmethode und ihre Anwendung am Beispiel des Recyclings elektronischer Geräte. WIRTSCHAFTSINFORMATIK 50(3):220–234

Brown AW (2000) Large-Scale, Component-Based Development. Prentice Hall, Upper Saddle River, NJ

Cheesman J, Daniels J (2001) UML Components. A Simple Process for Specifying Component-Based Software. Addison-Wesley, Upper Saddle River, NJ

Chesbrough H, Spohrer J (2006) A research manifesto for services science. Communications of the ACM 49(7):35–40

Cook S, Daniels J (1994) Designing Object Systems. Object-Oriented Modelling with Syntropy. Prentice Hall, Englewood Cliffs, NJ

D'Souza DF, Wills AC (1999) Objects, Components, and Frameworks with UML. The Catalysis Approach. Addison-Wesley, Upper Saddle River, NJ

Erl T (2005) Service-Oriented Architecture: Concepts, Technology, and Design. Prentice Hall PTR, Upper Saddle River, NJ, USA

Erl T (2007) SOA Principles of Service Design (The Prentice Hall Service-Oriented Computing Series from Thomas Erl). Prentice Hall PTR, Upper Saddle River, NJ, USA

Erradi A, Anand S, Kulkarni N (2006) SOAF: An Architectural Framework for Service Definition and Realization. In: IEEE International Conference on Services Computing 2006 (SCC '06), IEEE Computer Society, pp 151–158

IBM C (1984) Business Systems Planning: Information Systems Planning Guide. Technical Report GE20-0527-4, International Business Machines Corporation

Inaganti S, Behara GK (2007) Service Identification: BPM and SOA Handshake. Tech. rep., BPTrends

Jain H, Chalimeda N, Ivaturi N, Reddy B (2001) Business Component Identification — A Formal Approach. In: 5th IEEE International Conference on Enterprise Distributed Object Computing (EDOC '01), IEEE Computer Society, Washington, DC, USA, pp 183–187

Jungnickel D (2008) Graphs, networks, and algorithms, 3rd edn. Springer, Berlin

Kossmann D, Leymann F (2004) Web Services. Informatik Spektrum 26(2):117–128

Lovelock CH, Lewis B, Vandermerwe S (1999) Services Marketing: European Perspectives. Prentice Hall, London

Mcgovern J, Sims O, Jain A, Little M (2006) Enterprise Service-Oriented Architectures: Concepts, Challenges, Recommendations. Springer

Mills HD (1971) Top-down programming in large systems. In: Ruskin R (ed) Debugging Techniques in Large Systems, Prentice Hall, pp 41–55

Natis YV (2003) Service-Oriented Architecture Scenario. Research Report AV-19-6751, Gartner Research

Pahl G, Beitz W, Feldhusen J, Grote KH (2007) Engineering Design: A Systematic Approach. Springer, Berlin, Heidelberg

Papazoglou M, Traverso P, Dustdar S, Leymann F, Krämer B (2006) Service-Oriented Computing: A Research Roadmap. In: Dagstuhl Seminar, Schloss Dagstuhl, Internationales Begegnungs- und Forschungszentrum fÃijr Informatik

Parnas DL (1972) On the Criteria to be Used in Decomposing Systems into Modules. Communications of the ACM 15(12):1053–1058

SAP (2005) Enterprise Services Architecture: Enterprise Services Design Guide. Tech. rep., SAP AG

Scheer AW (2000) ARIS — Business Process Modeling, 3rd edn. Springer, Berlin

Selk B, Kloeckner S, Bazijanec B, Albani A (2005) Experience Report: Appropriateness of the BCI-Method for Identifying Business Components in large-scale Information Systems. In: Turowski K, Zaha JM (eds) Component-Oriented Enterprise Applications (COEA 2005), Bonn, Lecture Notes in Informatics, vol 70, pp 87–92

Sommerville I (2006) Software Engineering, 8th edn. Addison-Wesley

Szyperski C, Gruntz D, Murer S (2002) Component Software. Beyond Object-Oriented Programming, 2nd edn. Addison-Wesley, Harlow

Wang Z, Xu X, Zhan D (2005) A Survey of Business Component Identification Methods and Related Techniques. International Journal of Information Technology 2(4):229–238

Webster J, Watson RT (2002) Analyzing the past to prepare for the future: Writing a literature review. MIS Quarterly 26(2):xiii–xxiii

Winkler V (2007) Identifikation und Gestaltung von Services — Vorgehen und beispielhafte Anwendung im Finanzdienstleistungsbereich. Wirtschaftsinformatik 49(4):257–266

Yourdon E, Constantine LL (1979) Structured Design: Fundamentals of a Discipline of Computer Program and System Design. Prentice-Hall, Upper Saddle River, NJ, USA

Zimmermann O, Krogdahl P, Gee C (2004) Elements of Service-Oriented Analysis and Design — An interdisciplinary modeling approach for SOA projects

Dominik Birkmeier, Sebastian Klöckner, and Sven Overhage

Component & Service Engineering
Research Group
University of Augsburg
Universitätsstrasse 16
86159 Augsburg
Germany
{dominik.birkmeier | sebastian.kloeckner | sven.overhage}@wiwi.uni-augsburg.de