Identification and Analysis of Business and Software Services—A Consolidated Approach

Thomas Kohlborn, Axel Korthaus, *Member*, *IEEE*, Taizan Chan, and Michael Rosemann

Abstract—Although the benefits of service orientation are prevalent in literature, a review, analysis, and evaluation of the 30 existing service analysis approaches presented in this paper have shown that a comprehensive approach to the identification and analysis of both business and supporting software services is missing. Based on this evaluation of existing approaches and additional sources, we close this gap by proposing an integrated, consolidated approach to business and software service analysis that combines and extends the strengths of the examined methodologies.

Index Terms—System architectures, integration and modeling, software architectures.

1 Introduction

1.1 Motivation

Due to new communication and delivery channels, global competition, and deregulation of markets, companies from all over the world are able to conduct business with each other. However, these changes in an organization's environment do not only provide chances and possibilities, but also exhibit several threats and risks since organizations are confronted with an increased number of competitors. Additionally, the demand for new customized products has resulted in a shorter time to market, which influences the design and execution of business functions and processes [1].

To deal with changes and to enhance the agility and responsiveness of an organization, recent ideas promote the restructuring of organizations based on core competencies [2], [1]. Since companies focusing on core competencies are highly specialized, they need to collaborate with external partners to deliver a complete product or service to the end user. Thereby, a business network or ecosystem of collaborating companies emerges, wherein each company focuses on its core competency as well as on collaboration to fulfill customer demand. To expose standardized endpoints that can be used to leverage a partner's offered operations within the network, the concept of service orientation can be applied [3].

Generally, a service can be seen as an abstract resource that represents a capability [4] offered by the service provider that performs some kind of action on behalf of a service consumer at some time and place and through some channel [5]. Service orientation on the business level enables organizations to expose and offer operations as business services to business partners in order to facilitate on-demand collaboration opportunities. A *business service* is a specific set of actions that are performed by an organization [6]. Since the operations of an organization can be analyzed on different granularity levels, business services can represent these operations on different levels as well. Thereby, they are aligned with the capabilities of the organization in order to reflect the actual operations of the company [7].

Such an on-demand business needs to leverage its existent technology [1]. To support the agility of organizations in that respect, service orientation on the technical level fosters the utilization of software services and enables a close business and IT alignment that typically has been proven to be beneficial for an organization [8], [9]. Software services expose application functionalities that can be reused and composed based on business needs. A *software service* describes part of an application system which can be consumed separately by several entities. Hence, a software service supports the execution of a business service.

Software services have found their most promising incarnation in the form of Web services as the technological implementation of a service-oriented architecture (SOA) [10]. While the concept of SOA is not new, no common definition has evolved yet [11]. Most publications allocate the SOA concept merely to the technical domain [12], [13], [14]. Some authors, however, also emphasize the potential of applying the concept analogously in the business domain to refine business models [15], [16]. A definition that is applicable on both the business and the technical level and thus supports this holistic SOA view is the one provided by the Organization for the Advancement of Structured Information Standards (OASIS) in their SOA Reference Model [17]:

> *SOA is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains.*

- *pp. T. Kohlborn, A. Korthaus, and M. Rosemann are with the Business Process Management Group, Information Systems Cluster, Faculty of Science and Technology, Queensland University of Technology, 126 Margaret Street, Brisbane QLD 4000 Australia. E-mail: {t.kohlborn, axel.korthaus, m.rosemann}@qut.edu.au.*
- *T. Chan is with the School of Information Technology, Faculty of Science and Technology, Queensland University of Technology, 2 George Street, Brisbane QLD 4000 Australia. E-mail: t.chan@qut.edu.au.*

Manuscript received 15 July 2008; revised 21 Jan. 2009; accepted 17 Feb. 2009; published online 26 Feb. 2009.
For information on obtaining reprints of this article, please send e-mail to: tsc@computer.org, and reference IEEECS Log Number TSC-2008-07-0064.
Digital Object Identifier no. 10.1109/TSC.2009.6.

Thus, the concept of service orientation is applicable on the business level as well as on the IT level and can therefore be seen as a comprehensive approach for the complete enterprise [18], [1].

1.2 Research Questions

In order to achieve the benefits of service orientation, comprehensive and detailed approaches must be in place that can be utilized by organizations to identify and analyze services in early phases of service adoption. Thus, two main research questions arise:

1. What are underlying concepts of relevant, extant methods and approaches for identifying and analyzing services in an organization?
2. If shortcomings exist, how can these different concepts be consolidated, enhanced, and extended to provide an integrated and comprehensive guideline for the seamless analysis and identification of business and software services in an organization?

The first question addresses the need to identify a comprehensive, relevant set of service analysis approaches that needs to be analyzed in order to compare the relative strengths and weaknesses of the individual approaches. Based on the answer to the first question, the second research question focuses on the compatibility of the underlying concepts for service analysis associated with each approach. Should the evaluation reveal any gaps or shortcomings of the analyzed approaches, an additional literature review will have to be conducted to identify suitable concepts to enhance or extend the approaches.

The remainder of this paper is structured as follows: In Section 2, we derive criteria for the comparison and evaluation of existing approaches, justify the selection of the sample, and present the results of the analysis. Subsequently, we derive requirements for a consolidated approach and unify various concepts from the existing approaches. In Section 3, we present our integrated approach to the derivation of both business and software services, which is illustrated by an application example. Section 4 summarizes and concludes the paper.

2 Comparison and Evaluation of Existing Approaches

2.1 Derivation of Evaluation Criteria

In order to be able to assess how far existing approaches meet the general requirements stated in Section 1, we had to derive corresponding evaluation criteria. A review of the related literature published by [13], [19], [20] built the foundation for this task.

SOA concept. To analyze whether both service concepts (business and software services) are supported, we selected the SOA concept as a criterion reflecting whether an approach's primary focus is on the derivation of business services (BS), software services (SS), or both (BS/SS).

Delivery strategy for SOA. To address organization-specific requirements (e.g., the need to leverage existing legacy systems), different delivery strategies for a SOA are conceivable. Hence, we need to analyze whether a particular approach supports the top-down strategy (T), where services are derived based on the analysis of business requirements [21], the bottom-up strategy (B), which focuses on the derivation of services based on an analysis of legacy systems on an as-needed basis [22], or the meet-in-the-middle strategy (M), which combines the other two strategies [23]. The delivery strategy criterion was included to reflect which strategy an approach recommends to implement an SOA. To provide information about the particular starting point and focus of an approach, we included a textual comment in our analysis.

Lifecycle coverage. Approaches also differ with regard to their coverage of the SOA life cycle. While some proposed SOA development approaches aim at offering support for the full SOA life cycle, other approaches are more focused on a subset of the activities to be performed in the life of an SOA. In our analysis, we use a trivalent scale (0, +, ++) with the following semantics: 0 stands for methods that focus on service identification and analysis only, while + represents methods with a service analysis and design focus, and ++ finally marks more comprehensive approaches that include phases like implementation, etc.

Degree of prescription. A service analysis method can be rather prescriptive and define a rigid, heavy-weight process with lots of details, or it can describe a more lightweight, flexible, less structured process that is adaptable and allows for an agile approach. In our analysis, we use a trivalent scale (0, +, ++) with the following semantics: 0 stands for methods that are very lightweight, while + represents methods with a moderate degree of prescription, and ++ marks highly prescriptive approaches.

Accessibility and validity. To be useful, a service analysis approach should not only be well documented, but the documentation must also be accessible and the validity of the approach should be made clear. The documentation should provide many details, examples, ideas, case studies, etc., to provide useful guidance in practice. Sometimes, methodologies proposed by vendors or industry players are proprietary. Typically, detailed knowledge about these approaches cannot be easily accessed, whereas nonproprietary approaches are openly available. Moreover, this criterion captures whether an approach has been validated or illustrated by presenting real case studies, whether it uses fictitious examples only or even remains on a purely theoretical level without any examples at all. In our analysis, we used a three-valued tuple to describe the documentation (textual comment), the availability (trivalent scale with 0 standing for a proprietary, not openly available method, + representing a method that is at least partially documented for public use, e.g., in the form of papers about single activities that constitute the method, and ++ denoting a fully open method), and the validation, e.g., in the form of industry case studies, etc. (textual comment).

2.2 Selection of the Sample

After having defined the criteria for the comparison of extant approaches, we had to identify those prominent service analysis methods that were the most appropriate candidates for our analysis. We did not aim at a representative sample, but rather at a comprehensive mix of predominant approaches reflecting the broad spectrum of different characteristics. To this end, we made sure to not only

include academic work, but also approaches developed by the largest providers of packaged business applications such as IBM, Microsoft, and SAP. We ensured that the sample included both top-down and bottom-up approaches. Another requirement was that each approach to be included explicitly referred to the concept of service orientation. The sources of the methods are manifold, as they range from journals (e.g., [24]), to conferences (e.g., [19]), to books (e.g., [21]), to white papers (e.g., [25]).

The final sample of analyzed extant approaches can be found in the first column of Table 1, which also shows the characteristics of each approach with regard to a subset of the selected criteria. The ordering of the list of approaches in Table 1 allows the visual identification of different classes of approaches according to the underlying SOA concept (BS, BS/SS, and SS), which are separated by thicker lines in the table. For the category of approaches that are only concerned with software services (SS), subclasses can be identified depending on the delivery strategy for SOA (T, T/M, M, M/B, or B) separated by dashed lines.

2.3 Method Comparison and Evaluation

The comparison of the 30 extant service engineering methods as reflected in Table 1 was conducted independently by two coders, whereby the second coder restricted the analysis to a random control sample. The results of the coding process of the control sample were consistent with the results of the original coder. In the following, we describe our observations that resulted from the analysis of the 30 methods for each criterion used in the comparison process.

SOA concept. SOAs containing primarily business services are less prevalent than SOAs for IT infrastructure. Jones [15], OASIS [29], and Sehmi and Schwegler [7] propose approaches that do not directly apply to the concept of a business service. Nonetheless, the underlying concepts can be adopted for the identification of business services. Flaxer and Nigam [27] and IBM [26] explicitly define business services, but a detailed approach for the identification of these services is missing. Kaabi et al. [30] identify business services based on goal-modeling, which can then be supported by software services. All other approaches focus on the derivation of software services, although the term "business service" is used to distinguish between services that encapsulate business logic and services that encapsulate application logic.

Delivery strategy for SOA. The delivery strategy for SOA is partially dependent on the underlying SOA concept. Approaches which address the analysis of business services postulate a top-down strategy for the delivery of services. Regarding the scope and depth of these approaches, none of them provides a detailed description of how to include existing (legacy) systems into the analysis. For example, Sehmi and Schwegler [7] propose a pure top-down approach that describes how a business model can be implemented using software services. As their method has been incorporated in Microsoft's Motion Methodology [50], all details are not openly available. The starting points for the business service analysis vary widely. Jones [15] and OASIS [29] postulate a method that does not necessarily rely on any models or documentations, but on the collaborative analysis of the business of an organization. IBM [26] addresses business services as provisions of business components, whereas Flaxer and Nigam [27] propose analyzing business entities to identify business components and business services subsequently. A bottom-up strategy for business services could not be identified.

TABLE 1
Comparison of Service Analysis Approaches

Approaches	SOA concept	Delivery strategy for SOA	Lifecycle coverage	Degree of prescription	Accessibility and validity
	{BS/SS/ BS+SS}	{T/M/B}, {text}	{0/+/++}	{0/+/++}	{{text}, {0/+/++}, {text}}
[15]	BS	T, domain decomposition	0	+	examples, 0, n.a.
[24]	BS	T, component decomposition	+	+	examples, +, applied in industry
[25]	BS	T, business entities	0	+	examples, +, case studies
[7] [26]	BS+SS	T, capability decomposition	++	+	examples, +, applied in industry
[27]	BS+SS	T, domain decomposition	++	++	examples, 0, n.a.
[28]	BS+SS	M, goal modelling/IT	+	+	example, 0, n.a.
[29]	SS	T, process models	0	++	examples, 0, focus groups
[30]	SS	T, attributes analysis	+	++	examples, 0, n.a.
[31]	SS	T, use cases	++	++	case study, 0, case study
[32]	SS	T, use cases	+	+	example, 0, n.a.
[13]	SS	T/M, process models	+	++	case study, 0, case study
[21]	SS	T/M, process models, entities	++	++	examples, 0, n.a.
[33]	SS	T/M, use cases	++	0	n.a., ++, applied in industry
[34]	SS	T/M, business events/IT	0	++	examples, 0, n.a.
[19]	SS	M, process models/IT	0	++	case study, 0, case study
[35]	SS	M, process models/IT	+	0	case study, 0, case study
[36]	SS	M, processes and solution maps/IT	+	+	examples, ++, applied in industry
[37]	SS	M, processes and application portfolio analysis	0	+	examples, 0, n.a.
[38]	SS	M, business requirements/IT analysis	++	0	n.a., +, n.a.
[39]	SS	M, domains and processes/IT	++	0	example, ++, n.a.
[40]	SS	M, use cases/IT	0	0	n.a., 0, n.a.
[41]	SS	M, multiple starting points	+	+	examples, 0, n.a.
[42]	SS	M, processes/ IT	+	++	case study, 0, case study
[43]	SS	M, domain analysis/IT analysis	++	+	n.a., ++, applied in industry
[44]	SS	M, process models/IT	++	+	examples, 0, n.a.
[45]	SS	M/B, domain analysis/source	+	+	case study, 0, case study
[46]	SS	B, source code	+	+	case study, 0, case study
[47]	SS	B, IT analysis	+	0	few examples, 0, n.a.
[48]	SS	B, class diagrams	0	+	example, 0, n.a.
[22]	SS	B, source code	+	+	case study, 0, case study

Approaches addressing the derivation of software services postulate one of the three described delivery strategies. The top-down approach is supposed to derive a high quality SOA that is built on well-designed services and service compositions. However, depending on the size of the company and on the scope of the SOA initiative, a top-down strategy may consume significant resources, such as money and time, without showing an immediate outcome, since the upfront analysis has to be conducted before actually deriving services ([21], [36]).

Contrarily, a pure bottom-up strategy to deliver software services typically comprises activities that analyze existing legacy systems in order to define fine-grained services that can be linked to business processes and business requirements [22]. Hereby, one can distinguish between two types of bottom-up analyses. Noninvasive legacy approaches encompass methods that do not change the structure of the legacy code ([48], [51]). They propose building wrappers around the functionalities and components of the legacy system, so that they can be used in a service-oriented environment. Invasive legacy approaches aim at self-contained software services that encapsulate the functionalities provided by the legacy systems by restructuring the respective legacy code ([46], [22], [47]).

An interesting point is that most approaches postulate a meet-in-the middle strategy that takes into account business requirements as well as existing legacy systems to combine the advantages of both strategies. Thus, the advantages of a high quality SOA have to be weighed against reality constraints applied by the legacy systems. Arsanjani [44], as well as Zimmermann et al. [40], describe what an overarching approach could look like, but they fail to go into detail, as their approaches are proprietary.

Lifecycle coverage. Regarding the lifecycle coverage, it is obvious that the scopes of the methods vary widely. While certain methods specifically focus on the analysis phase (e.g., [19]), others also address the service design phase (e.g., [13]) or even address the complete lifecycle (e.g., [21], [24]). However, as a unified, standardized life cycle for services or when SOA is not prevalent in literature, authors addressing the life cycle of a service propose such a life cycle in the course of their publication.

Degree of prescription. Some of the approaches do not provide any structured guideline or process to derive services. These approaches give general suggestions about what to do, but do not provide information on how it should be done (e.g., [40]). Thus, they can be used or must be used in a flexible manner, as detailed application steps are missing. Nonetheless, most of the analyzed approaches provide some kind of procedural model to identify business services or software services. The steps or phases for the identification are very approach specific. For example, Rahmani et al. [49] propose a Model-Driven Architecture (MDA) approach that focuses on the derivation of three specific models (three phases) to identify services. Jones [15], on the other hand, proposes four steps within his identification framework that can be used to identify services.

Accessibility and validity. There is a strong correlation between the documentation of the method within the respective publication and the application in practice. Typically, if the method has been applied in practice by conducting a case study, the case study will be described within the paper. If the method has not been applied in practice, typically just basic examples are presented.

2.4 Conclusion from the Evaluation of the Existing Approaches

Although service orientation is affecting different levels within an organization, no unified way of how to identify and analyze services could be identified. Our analysis of existing approaches that have been formulated by academics as well as by industry vendors showed that the scope and depth of these approaches vary widely. For reasons of scope, we cannot provide a detailed analysis and description of the strengths and weaknesses of each approach in this paper, but we refer the reader to [52] for those details.

The most important insight that came out of our evaluation was that none of the examined approaches is comprehensive and integrated enough to cover both main SOA concepts (business and software services) to an adequate extent. Hence, we recognized an urgent need for a consolidated approach that provides guidance on the derivation of both business services and supporting software services to achieve a close business and IT alignment. This conclusion is supported by the literature, where the gap of a unified, consolidated approach has been stated in a number of publications [19], [13]. As a consequence of the proliferation of the service idea on both the business and the software level, there is now a demand for service engineering methodologies that cover both business and software services and provide an integrated, holistic approach to ensure business and IT alignment and agility. Service engineering in general is still regarded as a research challenge in the literature about current SOA research roadmaps (e.g., [53], [54]).

Having addressed the first research question, we had to propose a holistic approach that provides an organization with a methodology to not only understand and document its existing capabilities from a service perspective, but more importantly, to identify potential new services that may be provided on either or both the business and technical levels in order to answer our second research question.

2.5 Requirements for the Consolidated Approach

For the adequate design of the consolidated approach, we derived the following requirements from our analysis of the existing approaches.

For the consolidated approach, we chose the (T) strategy for the derivation of business services because none of the analyzed approaches provides any insight about possible ways to define business services in a bottom-up manner. For the derivation of software services, we propose an (M) strategy since most organizations will have existing application systems in place that cannot be easily replaced and need to be addressed by an SOA initiative [44].

None of the existing approaches directly addresses the identification of business services. Thus, for the design of the consolidated approach, their underlying concepts had to be adapted and enhanced by drawing input from additional related sources. An approach that details how to derive

software services based on business services is missing as well. Consequently, the consolidated approach needs to address this transition, e.g., by proposing process decomposition as an ancillary tool that can provide the additional information required to identify software services.

Regarding the identification process, the consolidated approach is intended to provide a structured guideline for organizations introducing service orientation. Thus, it will be divided into two parts (for the derivation of business and software services, respectively) with different phases and activities. Both parts comprise a preparation phase, an identification phase, and a detailing phase. Additionally, a prioritization phase will be described as a fourth phase during the derivation of business services to provide the link between the two parts. The inputs and outputs for each phase will be described as well to derive a detailed procedural model.

In addition to the criteria shown in the table, we also analyzed which types of services are described by the analyzed approaches and which service design principles they recommend. Since these aspects do not directly influence the structure of the consolidated approach, but are nevertheless important foundations for its development with regard to content, we will address them in the following.

The analyzed approaches describe different *types of services* based on different objectives. As mentioned in Section 1, business services can exist on different granularity levels. However, no classification scheme could be identified from the analyzed approaches. Only the analysis of additional sources, such as Bieberstein et al. [55], who structure business services according to their strategic importance and organizational ownership (e.g., group services, business unit services, team services, etc.), provided adequate details to classify business services.

In the case of software services, two different types of services could be identified. On the one hand, *atomic services* provide certain functionalities themselves and do not depend on other services. *Composite services*, on the other hand, combine atomic services to offer aggregated and/or extended functionalities.

We further classified *atomic services* into three types: utility services, task services, and entity services. *Utility services* are typically business-logic agnostic [25] and serve to provide reusable, cross-cutting functionalities related to processing data within legacy application environments [21], [56], [37], [13]. *Entity services* represent a business-centric service with a service boundary encompassing one or more business related entities. They often create business objects and ensure abidance by business rules and data completeness [13], [12], [25], [56], [19]. A *task service* is directly related to a business task of a process. It is modeled for specific processes to meet immediate requirements of the organization and therefore contains specific business logic. Encapsulated in a task service, one may provide centrally accessible functionality that is used consistently throughout the organization, e.g., for complex calculations that have been encapsulated in libraries and business frameworks traditionally [21], [12], [25].

Composite services can be categorized into two classes: logic-aggregation services and data-aggregation services. A logic-aggregation service or *process service* acts as the parent controller of entity, task, and utility services [19]. Thus, it invokes their operations based on the underlying process, which it represents [25], and controls and maintains the state of the process for its clients. The process logic and related business rules are transferred to the sphere of control of a process service, embedded in a process definition accessible to the process service [21]. *Data aggregation services* compose certain operations of entity services to expose their functionalities as a package, which enhances data consistency [57].

The aspect of *design principles* refers to the way desired characteristics of SOAs, such as loose coupling, are reflected in an approach. Most of the approaches do not elaborate on how desired service characteristics can be achieved. The use of service design principles can guide the service derivation process and ensure that the resulting services will feature the corresponding service characteristics [56]. We consolidated and identified nine design principles, namely, contract orientation, abstraction, autonomy, coupling, statelessness, cohesion, discoverability, reusability, and composability [52]. Although some of the principles focus more on realization issues (e.g., contract design, implementation, etc.), a subset of these principles should be applied during analysis to ensure a sound service design right from the beginning and to avoid misalignments or errors pervading through to the service implementation activities [23].

Service contract design is very important since the agreement on a service contract means the establishment of a dependency relationship between service consumer and provider [58]. A service contract establishes the terms of engagement as well as any semantic information the service owner wishes to make public [56], such as a description of the operations that are exposed to the service environment [59], [12].

Autonomy refers to the level of independence of a service. This means a purely autonomous service has full control over its environment, which results in increased reliability and predictability since external unpredictable influences are minimized [56]. Thus, a service's contract should not overlap with any other contract and the underlying realization or implementation of one service should reside under the control of that service [36]. Service normalization is another aspect that aims at designing the operations in a nonredundant manner [60].

The concept behind the principle of *coupling* is applicable on different elements of a service and an SOA in general and refers to the level of dependency between two or more elements. The observable type of coupling can be identified as the interdependency of multiple services and service compositions. As soon as one service calls an operation provided by another service, the service is dependent on the functionality offered by the other service's operation and the services are coupled. Services that are loosely coupled have a high reusability potential and are easy to maintain. Thus, the coupling between services should be minimized [36], [19], [45], e.g., by optimizing the allocation of operations to services.

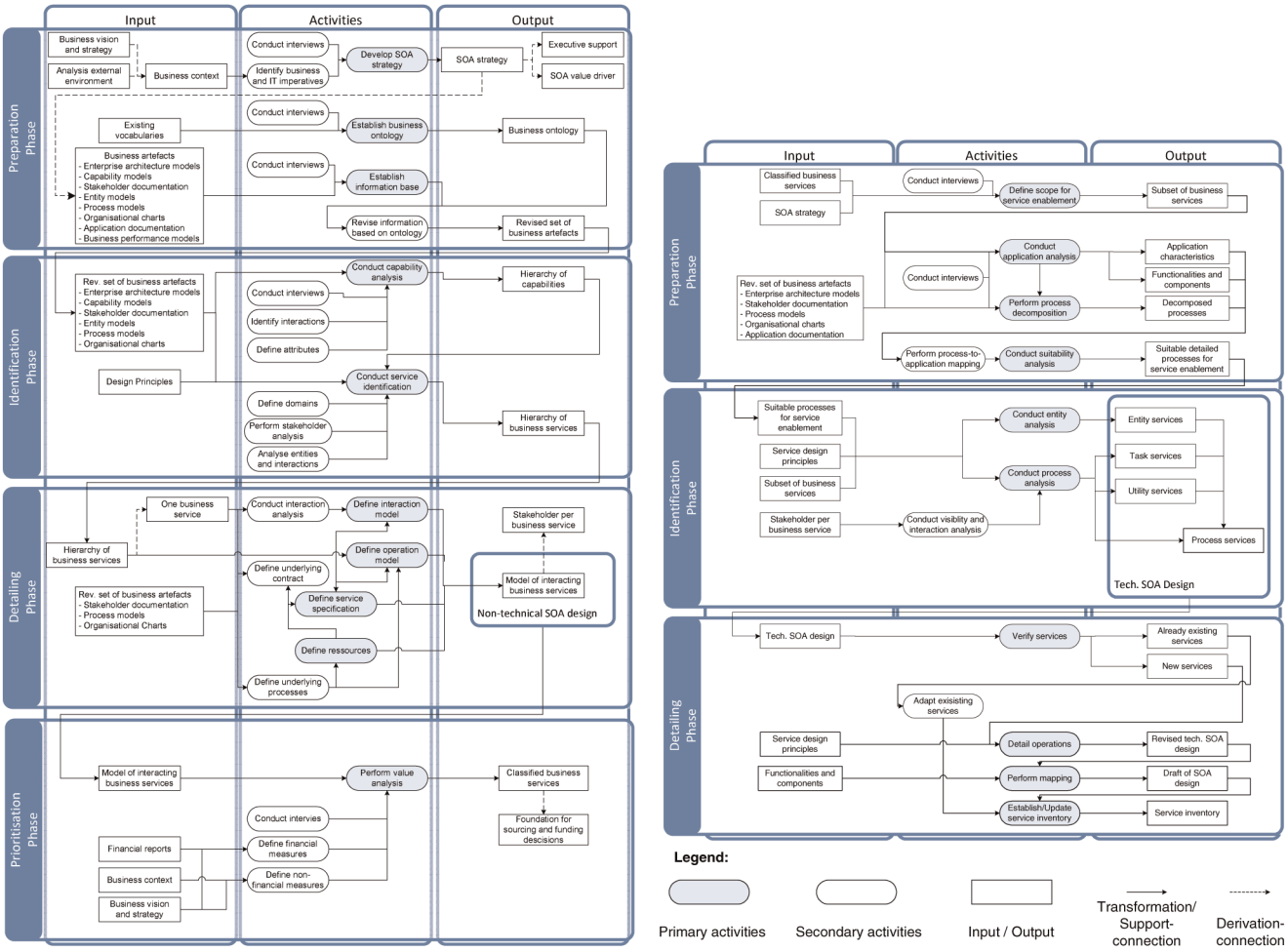

Fig. 1. Phases, activities, and inputs/outputs for the derivation of business and software services.

The idea of *service cohesion* refers to the strength of functional relatedness between operations in a service. Papazoglou and van den Heuvel [45] propose three levels of service cohesion, among which functional cohesion is the most preferable. Other types of cohesion can be analyzed by regarding the input and output parameters of an operation or the sequence of operation invocations. The functional relatedness can be interpreted as the interdependencies between operations or resources [19], [58].

The principle of *reusability* has a straightforward underlying concept: to make the service useful for more than one single purpose. Thus, services should be applicable in different situations and under unforeseen circumstances and be used by different service consumers [23], [19]. This can be achieved by defining an agnostic functional context and defining the encapsulated logic as generically as possible, so that the service is not bound to one single usage scenario. Furthermore, the service contract should encompass generic and extensible operations, so that, for example, they are not restricted to one specific type of input parameter [56].

All those requirements and consolidated principles or foundations were incorporated in the design of the consolidated approach, which will be presented in detail in the remainder of this paper.

3 The Consolidated Approach

3.1 Overview

This section describes the consolidated approach for the identification and analysis of both business and software services as visualized in Fig. 1. It combines the strengths of the analyzed approaches as well as extending and adopting certain facets to provide a comprehensive procedural model for service analysis. Business services will be derived that can (partly) be supported by IT functionalities represented by software services. The approach is subdivided into two main parts, each with different phases.

To illustrate the consolidated approach, we will recurrently revert to a typical example of an organization seeking to understand and improve its service orientation approach. The example refers to Dash Inc., a manufacturing organization specializing in manufacturing standardized car dashboards for major car manufacturers. It has also been delivering customized products to private end-consumers as well as to organizations that modify car interiors. One year ago, the organization decided to engage in an SOA initiative, and a project team was formed with the goal of analyzing how the organization could benefit from service orientation. The example details the different activities of the project team applying the consolidated approach.

3.2 The Derivation of Business Services

The first part covers the identification and analysis of business services by detailing, adapting, and consolidating existing service analysis approaches that focus on the business domain of an organization. This part will be structured into four distinct phases, each comprising a specific set of activities that may use the outputs of previous phases as inputs. Refer to Fig. 1 for an overview of the inputs, activities, and outputs for each phase.

3.2.1 The Preparation Phase

The *preparation phase* serves as the foundation for service identification [19]. First, the SOA strategy will be defined to set the scope and objectives of the following analysis efforts. As a start, one has to conduct a business context analysis [42]. The objective here is to find key areas of the business context that have to be supported by the SOA initiative. This provides the underlying basis for the identification of business and IT imperatives which serve as the motivation or driver for the SOA initiative to be documented in the SOA strategy [25], [42]. The SOA strategy influences the selection of related and adequate business artifacts that serve as the foundation for the service analysis phase. Executive support and sponsorship, which is crucial for the SOA initiative, may also be built upon the business context analysis and the specification of an SOA strategy [42].

Second, a business ontology providing a shared vocabulary among the involved stakeholders should be established. Since an SOA initiative may affect various parts of an organization, a common vocabulary helps to avoid misunderstandings between any involved parties and can be an essential step toward the integration of the business perspective and the technical perspective within an organization [21].

Third, an information base for service analysis needs to be established, i.e., the business documents that serve as the foundation for the identification and analysis of business and software services have to be compiled [21], [19], [13]. The business documents may comprise details about the enterprise architecture, capabilities, organizational structure and responsibilities, the stakeholders, business processes, and entities, and should be enhanced and validated by interviews with senior and top management.

Example (Dash, Inc.). At Dash Inc., the team recognized that their company was the market leader in manufacturing standardized and customized dashboards in the Australian market and the business strategy addressed the need to maintain this position and to enhance customer interaction, especially in the area of customized products. The identification of the business and IT imperatives showed that the integration of customers in the organization's processes should especially be focused on to enhance customer satisfaction. An SOA strategy (with value drivers such as "improve business agility," "improve customer integration," etc.) was formulated based on these imperatives, and executive support for the service analysis effort was established. The team was able to access an existing ontology and decided to extend this ontology throughout the analysis phase. To establish their information base, the Dash Inc. team collected multiple documents including enterprise architecture models, business process models, organizational charts, entity models, and application documentation that described the existing application landscape. Based on interviews, it turned out that no realignment of the documents with the existing ontology was required.

3.2.2 The Identification Phase

Having built the foundation, the *service identification phase* comprises activities that lead to the derivation of a set of business services. As a starting point, business capabilities are identified and analyzed. Moreover, a domain analysis, a stakeholder analysis, and an analysis of the interactions and entities further support the identification of business services.

According to [61], a business capability is a particular ability or capacity that a business may possess or exchange to achieve a specific outcome. However, a capability abstracts from *how* the desired outcome is actually achieved, but rather describes the externally visible behavior and the expected level of performance [7]. Thus, the complete organization can be seen as a federation of capabilities that interact with one another to achieve a valuable outcome for a business network. Capabilities can be used to develop a business model that is particularly stable against changes in the external and internal environment of an organization. The reason for this is that capabilities are not affected by changes in the organizational structure of an organization, or by changes in the flow of the underlying processes as long as the purpose or outcome remains the same [61].

Different structured and unstructured approaches to the identification of capabilities within an organization as well as related approaches based on process classification frameworks can be found in the literature (e.g., [62], [7], [63]). Capabilities can be modeled on different granularity levels to establish a hierarchy of capabilities until fine-grained business capabilities can be identified. This could be done, for example, by interviewing several senior managers of an organization and using APQC's Process Classification Framework [63], or the capability decomposition framework published by [7], as a guideline for capability identification. The required depth of the decomposition depends on the business needs that can be addressed by the SOA strategy. Business capabilities should be described and detailed by certain attributes, such as input/output parameters, owners, entities, etc. [7]. These attributes inform the process of determining appropriate boundaries of business services and provide an overview of how parts of the contract need to be designed related to service performance. A capability model also visualizes the relationships between all the different capabilities of an organization [64].

Having established different layers of business capabilities, one has to identify the boundaries of business services which encapsulate certain capabilities and expose them to the service environment. To avoid the proliferation of business services, often, some capabilities should be grouped together to form a business service that provides cohesive functionalities and is loosely coupled to other

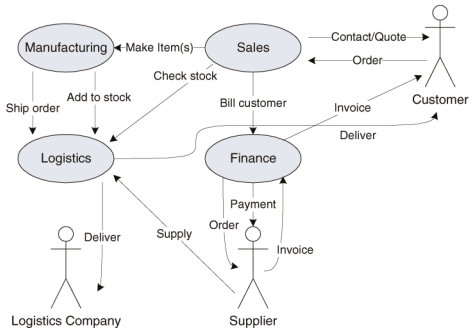

Fig. 2. Capability decomposition [15].

Domains \ Capabilities		Sales management		Manage promotions	
		Determine stock	Manage quotes	Pricing	Advertising
Manufacturing	Testing				
	R&D				
	Production				
Sales	Presales	P		P	P
	Quoting		P		
Logistics	Packaging				
	Shipping				
Finance	Credits				
	Billing				
Customer			C		
Supplier		C			
Logistic Company					

P = Provider; C = Consumer

Fig. 3. Mapping of domains and capabilities.

business services. This can be achieved by defining service domains.

A service domain can be described as a sphere of control that contains a collection of tasks to achieve related goals [65]. The domain decomposition approach proposes to establish a domain model on different levels of granularity that visualizes the interactions among the different domains as well as the interactions external to the domains [15]. Each level is derived by decomposing a service domain on the previous level.

The first step involves the definition of the service domain's scope, asking what the boundary is of a service domain within an organization that comprises certain capabilities. The objective of service domain modeling is to represent what the organization does and to place boundaries between the domains [27], [16], [26], [15]. Thus, services that will be provided by a particular domain are owned and managed by this domain. This step should closely be aligned with any governance activities that address ownership concerns [66].

The next step focuses on the interaction between different service consumers and service domains [15]. The objective is to identify and understand the external stakeholders of the organization [67]. After the identification of service domains and actors, the reasons for the interactions between the domains themselves and between the domains and the actors have to be identified [29], [15].

The fourth step concentrates on the allocation of capabilities to the identified service domains and their interactions. The domain decomposition and the capability decomposition are orthogonal to each other. Based on this first level of the domain model, one may already allocate capabilities to certain domains using certain heuristics and indicators, e.g., design principles such as low coupling and high cohesion [64]. Furthermore, one can use the attributes annotated to the identified capabilities to analyze the belonging to domains. The allocated capabilities must support the identified interaction between the domains.

The last step involves the decision of which domain should be decomposed further. Each step of the approach has to be performed for each specific domain of the developed diagram derived on the previous level [15].

Each service domain can be regarded as a provider of one or more business services. Since each domain on level k is decomposed separately, it is possible that one identifies domains that are shared on level k+1. This leads to the identification of business services that are reused under different circumstances within the organization [15]. The classification of business services can be made according to the strategic relevance of one specific service [55]. By analyzing the different actors and interactions, one can also identify certain virtual services, which are not core to the organization and do not belong to a direct domain [15].

Example Dash, Inc. Since Dash Inc. had not documented any capabilities before, the project team decided to identify the capabilities of the organization by interviewing senior managers and utilizing the process classification framework that had been developed during past BPM projects. Hereby, they identified capabilities based on the main clusters of the process classification framework. The first cluster encompassed all processes that were related to the design and development of new products or services. The second cluster grouped capabilities based on the generation of demand and the third cluster grouped capabilities together that addressed the delivery of products and services. Subsequently, the team annotated details such as interactions between capabilities, attributes of single capabilities, underlying entities, the existing owner and interrelationships.

Four primary domains were identified that represented the business of Dash Inc., namely finance, logistics, manufacturing and sales. Furthermore, three general classes of stakeholders were identified to be relevant for the service analysis phase: customer, supplier and logistic companies. The customers of the organization included end-consumers, car manufacturers and car customizers. The execution of the first three steps described above resulted in the following domain model (see Fig. 2).

The mapping between capabilities and domains was supported by using the template visualized in Fig. 3.

The capabilities were allocated to the respective service domains. Additionally, potential provider-consumer relationships between domains and external stakeholders were indicated.

Finally, the first three steps were applied again to decompose the domains even further until fine-grained service domains were identified. The capabilities that had been allocated to the high-level domains were reallocated to the finer-grained domains. The boundary for each business service was drawn based on entity mapping, ownership considerations and interaction analysis. Subsequently, each service domain was responsible for one or more business

services. The ownership considerations were closely aligned with the governance structure of Dash, Inc.

3.2.3 The Detailing Phase

At this stage, capabilities, domains and business service boundaries have been identified. The *detailing phase* comprises activities concerned with the relationships and interactions between business services. The input for this phase includes the identified business services as well as a set of relevant business artifacts. Each business service has a set of associated resources (e.g., process models, contract, owner, etc.) and exposes one or more operations to its environment that can be utilized according to the business service contract as part of the business service specification. Each operation has two underlying models, namely, the interaction model and the operation model, which describe the behavior of the specific business service operation [16], [6].

Business Service Resources. Each business service has an owner who is responsible for the specific service. The owner of that specific business service is related to the service domains that have been identified in the previous phase. A business service comprises one or more capabilities based on ownership concerns, functions, related interactions and entities. At this stage, the underlying processes of the capabilities have to be identified. The value or output of one capability can be based on one single process as well as an interaction between different processes. Consequently, the design of a business process across a business network or within a single organization (e.g., order-to-cash process) can be modeled as a choreography of interacting business services that may be under different ownership domains.

Operation Model. The operation model of a business service operation details how the specific operation is executed. Hereby, the processes utilized by the business service operation as well as invocations of operations of other business services to achieve the desired goal stated in the business service contract have to be addressed. This model does not need to be published to the service environment since it is not necessary for the service consumer to understand how the specific outcome of a service is realized [6]. The operation model orchestrates the identified processes based on the functionality that should be provided by a business service operation. An orchestration describes the behavior that a service provider performs internally to realize a certain provided service. This may include the cooperation with external business partners and the interaction with other services, although the focus lays on the fulfillment of one service operation [3].

Interaction Model. The interaction model as part of the service contract is used to define the commitments that have to be made by the service consumer and provider [6]. The interaction model can be derived using service blueprints, which are known from the area of service marketing [68]. The so-called "line of interaction," which reveals the division of activities between the organization and its customer, is the main focus of the analysis. Services are always invoked by a service consumer. The participation can range from a one-time invocation of the service to a deep integration of the service consumer into the service delivery process of the service provider [68]. The organization has to analyze what parts of the service can or should be managed by the customer and how the exception handling mechanisms have to be defined. One solution to overcome the risks associated with extending the line of interaction toward the consumer is an extensive standardized description of the service contract [69]. The interaction model is the explicit way a business service defines how the collaboration will be realized [3]. The interaction can be modeled as a process, indicating that the interaction between the service provider and consumer is purely behavioral. The interaction model for a business service operation represents a choreography that can be seen as the external view on a collaboration between a service provider and its consumer to achieve a certain goal through a sequence of commitments [21]. It abstracts from any tasks that the service provider performs internally.

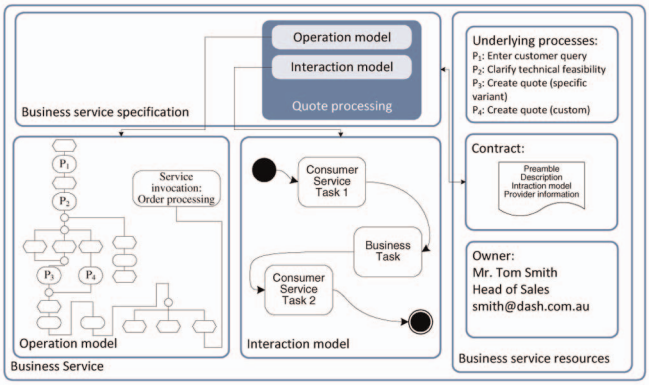

Fig. 4. Detailing the quote service [3].

By providing these models for all identified business services, interaction and dependency relationships between those services can be visualized as an output of this phase.

Example (Dash, Inc.). In the following, the coarse-grained business service "Quote service" will be used to exemplify the activities that needed to be conducted. This service was owned and managed by the service domain "Quoting" within Dash, Inc. and derived from the capability "Manage quotes." The "Quote service" was classified as a business unit service within the group service "Sales." The owner of the sales service was the head of the sales department. The "Quote service" had four underlying processes as shown in Fig. 4. The "Quote service" exposed one operation, namely, "Quote processing."

The *operation model* for the "Quote processing" service operation at Dash, Inc. utilized all four processes and specified which business service had to be subsequently invoked. If a query was received from a customer, it would be entered into the system before the technical feasibility was clarified. A feasible query would be analyzed regarding its relation toward a standardized product, a variant of a product, or a fully customized product. Based on this differentiation, a quote was created and transmitted to the customer. If the customer accepted the quote, the business service "Order processing" would be invoked.

With regard to the *interaction model*, the first task for the service consumer was to send a query with adequate

information to the organization. The query could be sent via e-mail or could be received by phone. Based on the SOA strategy, the project team had already considered offering this service via the Internet to provide another communication channel between the customer and the organization. If a query was received, the organization would analyze the query for its feasibility and notify the customer in case the query was not feasible. If it was feasible, a quote would be created. The second task of the service consumer was to accept or decline the quote. Alternatively, the customer could also propose changes to the quote.

After the elements of the business service had been specified, the project team decided to postpone the detailed *specification* of the service until potential supporting software services were identified. The detailing of the business service was delegated to the top management and service owner since they provided the specific knowledge about the details of the service, e.g., billing, provision, etc.

3.2.4 The Prioritization Phase

When all business services have been detailed and analyzed, the *prioritization phase* proposes activities that enable an organization to structure their services based on specific measures and the value they deliver for the organization. This phase is used as a link to the second part, which is concerned with the derivation of software services.

A value classification enables the organization to gain an understanding of which business services contribute most to the value of an organization and its competitive advantage. This information is fundamental to make sourcing and funding decisions. A value chain or business network analysis may enable the organization to understand its core critical functions as well as supporting functions.

There are different methods available for performing service value analysis. Jones [15], for example, proposes a service classification matrix for ad hoc service classification based on the value a service delivers. A more structured approach is the Analytical Hierarchy Process (AHP), which is based on the principle of decomposing a complex problem into finer-grained problems that can be "solved" more easily and explicitly takes into account both financial and nonfinancial data [70]. Hafeez et al. [62] propose the application of the AHP to determine the key capabilities of an organization. Thus, this approach can easily be adapted to classify business services. Although the AHP uses a structured and formalized approach, one has to keep in mind that it is purely subjective in nature as it relies on judgments made by human decision-makers.

Example (Dash, Inc.). At Dash Inc., the project team decided to conduct a value analysis based on the service domains and their underlying capabilities utilizing AHP. The operating profit, sales growth and return on capital employed were chosen as financial measures. Customer satisfaction, market share, and new product introduction were chosen as nonfinancial measures. Two AHP models were developed to derive the relative importance of the financial and nonfinancial measures of the distinct business capabilities. A pair-wise comparison between the measures was conducted to identify the priorities of the different measures. The outcome of the AHP is visualized in Fig. 5.

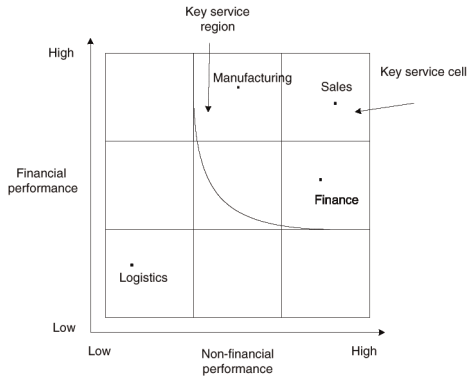

Fig. 5. The outcome of the AHP [62].

Based on the outcome of the AHP, the project team concluded that the service "Logistics" had a low financial performance and a low nonfinancial performance, whereas the sales service had the exact opposite characteristics.

3.3 The Derivation of Software Services

The second part of the consolidated approach describes how software services can be identified and analyzed that support business services in order to achieve close business and IT alignment. Similar to the first part of the consolidated approach, this part will be structured into distinct phases, each comprising specific activities. Refer to Fig. 1 for an overview of the second part of the consolidated approach.

3.3.1 The Preparation Phase

The prioritized business services and the SOA strategy will be used as an input to define the scope for software service enablement and to identify which area within the organization might benefit the most from service enablement [42]. Each of the business services has at least one underlying process that describes the activities that have to be performed in order to deliver the value or outcome of the respective business service. In order to provide a basis for the identification of software services, the respective process has to be decomposed and enriched by the information that has been the outcome of an application analysis [40]. Based on the process decomposition and the application analysis, the process has to be analyzed to rate the potential benefits of service enablement. Manual processes or processes that run stable in isolation might not be the best candidates for service enablement [31]. The outcome of the preparation phase is a set of business processes that is suitable to be service enabled and that is decomposed to the most granular process steps with annotated roles and application systems.

Example (Dash, Inc.). Based on the analysis of the value contribution of the underlying business service capabilities, Dash, Inc.'s project team selected the "Sales" business service and its underlying services to define the *scope* of the software service enablement. The focus

lay on finding software services to support the business services in terms of customer integration and flexibility. Data flow and invocation calls were traced throughout the applications to identify interfaces and interdependencies. Based on the processes of the business service, detailed *process models* were developed. (In the following, the already described "Quote service" will be used as the foundation to derive software services. Specifically, the process "Create quote (variant)" will be used to exemplify the further analysis steps.) After the scope, the applications, and the processes were specified, a *process-to-application mapping* was conducted to analyze the suitability of the process and application for service-enablement. Hereby, the applications that were associated with the "Sales" business service were annotated to the respective process models. Furthermore, the project team identified which process steps were executed manually, semi-automatically, or fully automatically. As the next step, the project team conducted a *suitability analysis* for the application and processes. The application related to the "Quote service" had been developed in-house. Hence, the source code was available, as well as the knowledge to customize the source code if required, so that the project team decided that the application was suitable to be service enabled.

3.3.2 The Identification Phase

The identification phase purely targets the derivation of software services. To this end, two main sources of information are needed. On the one hand, entity models are needed to derive entity services related to the core business objects of each business service. On the other hand, detailed process models should be in place to be analyzed for service enablement. This phase comprises the following steps mainly based on [21], [19]:

1. *Identify corresponding entity.* Taking the output of the previous phase as an input for service identification, one should first identify entity services since they are very generic and reusable in nature. They are not tightly coupled to processes, meaning that the provided interface of that service is not process-specific [21]. To define the boundary of an entity service, one has to analyze the actual context of the service. This can be achieved by examining the selected process models. Processes might be analyzed to define the entities that are processed and the operations that are used for processing the entity. Furthermore, entity models [21], class-diagrams [49], or brainstorming techniques [42], in combination with the analysis of the selected process, can point to preliminary entity services.
2. *Analyze visibility and takeover of process steps.* Having identified the main entity of the process and the corresponding potential entity service, the "line of interaction" and "line of visibility" of the process have to be analyzed, taking into account the interaction model (cf. [19], [68]). The "line of interaction" specifies the parts or functions of the process that may be taken over by the service consumer. Especially with multiple channels facing the consumer, one has to decide what process functions may reside in the sphere of control of the service consumer. The "line of visibility" defines how much of the process should be visible for the stakeholders. The stakeholders may comprise external business partners (e.g., customers, suppliers, etc.) as well as internal partners (e.g., subsidiaries). By analyzing process functions based on their visibility and interaction potential with stakeholders, one can identify potential groupings of functionality that must/should be explicitly exposed to the organization's stakeholders by means of services [19].
3. *Identify potential service operations.* Once the process itself has been decomposed into its most granular process steps, one has to identify potential service operations. Each process step can be regarded as a potential service operation [23]. However, all process steps that represent solely manual tasks or process steps that are executed by a legacy system which cannot be service enabled have to be excluded from the potential logic that can be encompassed by a service.
4. *Extract process logic.* This next step encompasses the extraction of process logic from the process itself. One may extract business rules, conditional logic, exceptional logic, and sequence logic [21]. Some process logic may not actually be visualized by an individual action or process step, depending on the modeling conventions (e.g., modeling language) an organization has used while deriving the process models.
5. *Define logical context(s).* The remaining process steps should be grouped based on their logical context [21]. Thus, the identified context confines the service boundary. Hereby, the principle of service cohesion plays the most important role as operations should be grouped together that are functionally related [45]. Depending on the scope of the service identification, one may address reusability and define service operations that have a high potential to be consumed in different scenarios [56]. However, any added operations should still relate to the logical boundary of the service. The design principle of coupling can be applied to identify sequential dependencies between operations. Sequential operations, which are only dependent in one way, may be combined inside a service. One may also identify process steps that are recurring within that process, which can be grouped together into a single service. New services may be created as well, depending on new logical contexts that may be identified. The visibility and takeover potential annotated to the process steps may provide a guideline for the grouping of process steps and, consequently, for the definition of task and entity services. Marks and Bell [43] propose developing two different services if the service invokers can invoke the associated operations of a service at different times and in different contexts (similar to [19]). Furthermore, one can identify services that are purely technology-related and business logic-agnostic. Thus, these services can be classified as utility services. At this stage, we have identified task, entity, and utility services.

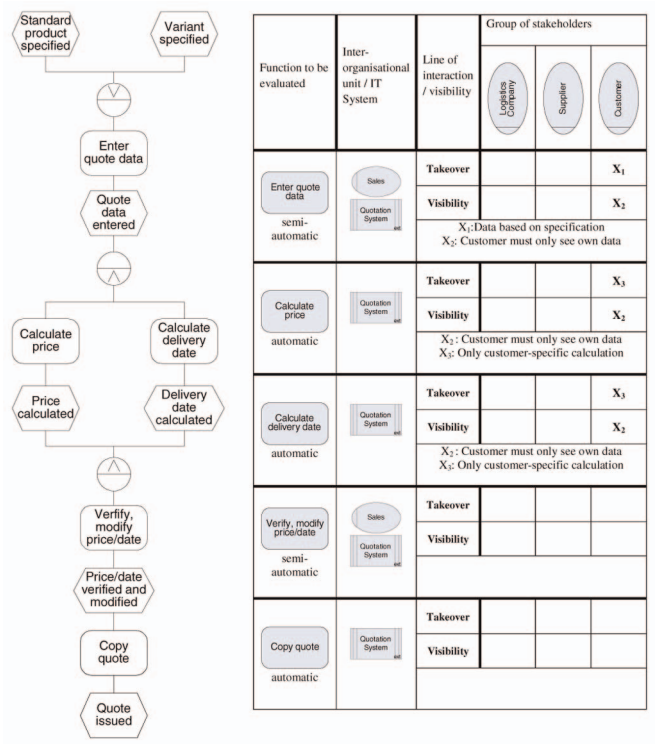

Fig. 6. Analyzing the "Create quote" process [19].

6. *Define compositions.* In order to identify the potential for composition and consolidation of services and to evaluate the appropriateness of service boundaries, usage scenarios for the service have to be developed [21]. Furthermore, one can discover in which situation logic is missing and can shift the related business rules to task services or process services. Consequently, new services may be created. The main objective is to specify process services that compose the task, entity, and utility services related to the underlying process. Based on the visibility and interaction analysis, one may create process services that are exposed to a specific set of stakeholders [19].

Example (Dash, Inc.). Dash, Inc.'s project team identified the "quote" entity to be relevant for the business service by analyzing the process model. Based on the targeted reusability considerations, the "quote" entity service provided the CRUD-operations since these operations were most often utilized by other software services in different scenarios.

The results of the analysis of Dash, Inc.'s process "Create quote" is visualized in Fig. 6.

Since all process steps within the "Create quote" process were supported or executed by IT, all steps were regarded as potential service operations. The project team identified the parallel split after the process step "Enter quote data" and documented this issue for further analysis regarding the composition of services. Since all steps focused on the manipulation of the "quote" entity, the first draft of the logical boundary of the service, based on regarding the service cohesion principle, encompassed all process steps. Based on the principle of reusability, the project team decided to model the operations "Calculate price" and "Calculate delivery date" as two separate services. That way, both services could be utilized independently without invoking the complete entity service. Furthermore, both services were related to different underlying documents. For example, the "Calculate price" service was regarded as a task service that utilized different documents about prices based on the specific customer. Since the functionality of the entity service would not change if the customer changed, the operations "Calculate price" and "Calculate delivery date" were outsourced to two separate task services. The "Calculate price" operation was grouped together with the "Modify price" operation to form the "Price" task service. Similarly, the "Calculate delivery date" operation and the "Modify delivery date" were factored into the "Delivery date" service. The "Copy quote" operation comprised purely business-agnostic logic. Hence, the project team classified this operation as a separate utility service.

The project team focused on a close business and IT alignment. Thus, the process was represented by one process service that composed the entity and utility services as well as the task services. Furthermore, the process service invoked the operations of the composed services based on the process flow. The interaction and takeover analysis of the process steps identified that the operations "Enter quote data," "Calculate price," and "Calculate delivery date" were also executable by the customer. The project team decided to encapsulate these operations in a second process service that could be utilized by customers independently of any sales representatives or account managers. Thus, six services were identified.

3.3.3 The Detailing Phase

After the services have been identified, one has to verify that already existing services do not overlap with the newly discovered ones. If the service or service operation already exists, one has to decide if the existing service can be extended or adapted. Services may already be in place if processes have recurrent process steps. Thus, by analyzing different processes in terms of communalities, one can abstract from certain process steps and group them together to form a service which can be shared by these processes.

New services should be detailed further in order to identify additional overlaps with already existing services and to make the services more reusable and autonomous. Since a service can encompass one or more operations, one can identify the input and output parameters in order to minimize the coupling between different operations and the underlying systems [19], [13]. Feuerlicht [60] proposes an approach for service interface design that builds on the principles of cohesiveness and coupling. This may lead to the identification of operations that overlap within or between services, which leads to the specification of new services or operations. Hence, the newly identified services have to be verified to ensure that these services do not already exist.

The identified operations and services should be mapped to the existing application layer in order to identify missing functionalities or the need for additional services (gap analysis) [23]. For each operation candidate within the

TABLE 2
Detailing of Task, Entity, and Utility Services [19]

Atomic services	Operation	Input Parameter	Output Parameter	Service Consumer
Quote (Entity)	create()	quote data [payment and delivery conditions]	quoteID	CU (customer)
	update()	quote data [payment and delivery conditions (delta)]	notification	
	read()	quoteID	quote data	CU
	delete()	quoteID	notification	
Price (Task)	calculatePrice()	materialID, values	price	CU
	modifyPrice()	quoteID, new Price	notification	
Delivery date (Task)	calculateDeliveryDate()	quoteID, values	delivery date	CU
	modifyDeliveryDate()	quoteID, new delivery date	notification	
Copy (Utility)	copy()	data	notification	

TABLE 3
Detailing of Process Services [19]

Process Service	Service Consumer	Function	Service	Operation
Enter quote		enter quote data	Quote	create()
		calculate price	Price	calculatePrice()
		calculate delivery date	Delivery date	calculateDeliveryDate()
		modify price, delivery date	Price	modifyPrice()
			Delivery date	modifyDeliveryDate()
		copy	Copy	copy()
Calculate quote	CU	calculate price	Price	calculatePrice()
		calculate delivery date	Delivery date	calculateDeliveryDate()
		enter quote data	Quote	create()

identified software service, one has to analyze the underlying processing requirements and the application logic that needs to be executed in order to make decisions about the sourcing of the specific logic. One may also break down the application logic requirements into smaller steps in order to identify new operation candidates within a proposed service [21]. However, it may be possible that all the operation candidates identified in the previous phase are supported by the application portfolio and do not need to be revised. Finally, one needs to analyze the original service compositions and determine whether any changes need to be made concerning the inclusion of new services or operations [21].

The last step of the service analysis phase includes the exposure of the service candidates in a service repository. This will be used during further analysis and identification efforts to prevent a proliferation of services with the same functionalities. For each service that has been identified and verified, service-specific details have to be documented [13]. These details comprise the service identifier (e.g., the name) including a service description, the service type, input and output parameters, and the service consumers [19].

Example (Dash, Inc.). After the services were identified, Dash, Inc.'s project team verified that the services did not already exist. Since the project team split up into different groups to identify services more efficiently, one team had already fed their identified services into the service repository. The entity service had already been specified and could be reused. As the principle of targeted reusability had been taken into account, the needed operations were already defined. The other services were new and had to be detailed further. To identify further reusability potential, the different operations were detailed regarding their input and output parameters. The project team decided that the utility service "Copy quote" should be made more reusable by extending the allowed parameters. Thus, the service should not only copy quotes, but different data types. Table 2 and Table 3 were created.

After the operations were detailed, the application functionalities were mapped on the identified service operations. By utilizing the list of the application's functionalities and components, the project team concluded that no changes to the application functionality had to be made other than providing an interface for the potential services. However, this was not in focus of the project team. Finally, all the newly identified and detailed services were fed into the service repository to provide other teams and projects with the possibility to reuse already identified services. Hence, the first draft of an SOA for Dash, Inc. was established. Internal service domains were identified as well as external stakeholders. These actors utilized the business services that had been identified as well. The business services were owned and managed by their respective service domains. The processes underlying each business service were used to identify process services that provided IT support for certain operations that were exposed by their related business service. The process services in turn composed finer-grained tasks, entity and utility services. Their functionalities were mapped on the IT systems to identify reengineering requirements. Further projects were founded to analyze the first draft of the SOA regarding its benefits for Dash, Inc. in more detail.

4 Conclusions and Future Work

In this paper, the results of a comprehensive analysis of 30 extant service identification and analysis approaches were used to propose a structured, consolidated approach that combines their relative strengths and suggests enhancements and extensions based on the identified shortcomings of the extant approaches. The paper pointed out how software services can be used to support business services to achieve close business and IT alignment. Additionally, the presented approach provides an organization with a methodology to not only understand and document its existing capabilities from a service perspective, but more importantly, to identify potential new services that may be provided.

Although the consolidated approach combines methodologies that have successfully been applied in practice, its applicability in different contexts of private and public sector organizations needs to be validated. The first results of a case study at an Australian Statutory Authority seem promising and are currently leading to refinements regarding the operationalization of the consolidated approach. We have also started another case study at an Australian government agency to gain further valuable insights regarding the validity of the proposed approach, particularly in the business service context. Finally, we are investigating the application of our approach at one of Australia's leaders in banking, insurance, investment, and

superannuation. We work with the insurance division of that company to apply parts of the methodology in the context of the reorganization and service-oriented implementation of a motor claims business process.

In addition to making refinements based on the empirical insights gained, we plan to extend our approach by putting an emphasis on the consideration of important aspects that have not been prominently featured in the current version of our approach. This includes, for example, the aspect of designing services to enable the establishment of business performance models that will measure performance based on the enactment of these designed Web services within the SOA. Increased visibility into business processes and business data has been characterized as an often overlooked benefit of SOA-based approaches [71]. By incorporating adequate activities and guidelines into our methodology to enable organizations to come up with business performance models and plan about things like Business Activity Monitoring and event processing right from the beginning, we aim to support organizations in taking advantage of this opportunity from the outset to gain greater business value from their SOA initiatives.

ACKNOWLEDGMENTS

Parts of this research have been funded by a research project within the Australian Research Council (ARC) Linkage Schema (grant code LP0669244), including financial support from SAP Research and the Queensland Government. The authors would also like to thank Marlon Dumas for being the initiator of this research and providing valuable input.

REFERENCES

[1] L. Cherbakov, G. Galambos, R. Harishankar, S. Kalyana, and G. Rackham, "Impact of Service Orientation at the Business Level," *IBM Systems J.,* vol. 44, no. 4, pp. 653-668, 2005.

[2] J. Hagel III and M. Singer, "Unbundling the Corporation," *Harvard Business Rev.,* vol. 77, no. 2, pp. 133-141, 1999.

[3] J.L. C. Sanz, N. Nayak, and V. Becker, "Business Services as a New Operational Model for Enterprises and Ecosystems," *Proc. Eighth IEEE Int'l Conf. E-Commerce Technology and Third IEEE Int'l Conf. Enterprise Computing, E-Commerce, and E-Services (CEC-EEE '06),* p. 61, 2006.

[4] W3C, "Web Services Glossary," http://www.w3.org/TR/ws-gloss, 2004.

[5] M. Dumas, J. O'Sullivan, M. Heravizadeh, D. Edmond, and A.T. Hofstede, "Towards a Semantic Framework for Service Description," *Proc. IFIP TC2/WG2.6 Ninth Working Conf. Database Semantics: Semantic Issues in E-Commerce Systems,* 2001.

[6] J. Sanz, N. Nayak, and V. Becker, "Business Services as a Modeling Approach for Smart Business Networks," Technical Report RJ10381 (A0606-001), IBM Research Division Almaden Research Center, 2006.

[7] A. Sehmi and B. Schwegler, "Service-Oriented Modeling for Connected Systems—Part 1," *The Architecture J.,* vol. 7, pp. 33-41, 2006.

[8] P.P. Tallon and K.L. Kraemer, "Investigating the Relationship between Strategic Alignment and IT Business Value: The Discovery of a Paradox," *Creating Business Value with Information Technology: Challenges and Solutions,* N. Shin, ed., pp. 1-22, Idea Group Publishing, 2002.

[9] J.C. Henderson and N. Venkatraman, "Strategic Alignment: Leveraging Information Technology for Transforming Organizations," *IBM Systems J.,* vol. 32, no. 1, pp. 4-16, 1993.

[10] G. Alonso, *Web Services: Concepts, Architectures, and Applications.* Springer, 2004.

[11] M.W. A. Steen, P. Strating, M.M. Lankhorst, H.W.L. ter Doest, and M.E. Iacob, "Service-Oriented Enterprise Architecture," *Service-Oriented Software System Engineering: Challenges and Practices,* Z. Stojanovic and A. Dahanayake, eds. pp. 132-154, Idea Group Publishing, 2005.

[12] D. Krafzig, K. Banke, and D. Slama, *Enterprise SOA. Service-Oriented Architecture Best Practices,* fifth ed. Prentice Hall, 2006.

[13] F. Kohlmann and R. Alt, "Business-Driven Service Modelling—A Methodological Approach from the Finance Industry," *Proc. First Int'l Working Conf. Business Process and Services Computing (BPSC '07),* pp. 180-193, 2007.

[14] C. Legner and R. Heutschi, "SOA Adoption in Practice-Findings from Early SOA Implementations," *Proc. 15th European Conf. Information Systems (ECIS '07),* pp. 1643-1654 2007.

[15] S. Jones, *Enterprise SOA Adoption Strategies. Using SOA to Deliver IT to the Business.* C4Media, 2006.

[16] N. Nayak, A. Nigam, J. Sanz, D. Marston, and D. Flaxer, "Concepts for Service-Oriented Business Thinking," *Proc. IEEE Int'l Conf. Services Computing (SCC '06),* pp. 357-364, 2006.

[17] OASIS "Reference Model for Service Oriented Architecture 1.0," http://docs.oasis-open.org/soa-rm/v1.0/soa-rm. pdf, 2006.

[18] C. Schroth, "The Service-Oriented Enterprise," *J. Enterprise Architecture,* vol. 3, no. 4, pp. 73-80, 2007.

[19] K. Klose, R. Knackstedt, and D. Beverungen, "Identification of Services—A Stakeholder-Based Approach to SOA Development and Its Application in the Area of Production Planning," *Proc. 15th European Conf. Information Systems (ECIS '07),* pp. 1802-1814, 2007.

[20] E. Ramollari, D. Dranidis, and A. Simons, "A Survey of Service Oriented Development Methodologies," *Proc. Second European Young Researchers Workshop on Service Oriented Computing,* 2007.

[21] T. Erl, *Service-Oriented Architecture. Concepts, Technology, and Design,* fourth ed. Prentice Hall, 2005.

[22] H.M. Sneed, "Integrating Legacy Software into a Service Oriented Architecture," *Proc. Conf. Software Maintenance and Reengineering (CSMR '06),* pp. 3-14, 2006.

[23] S. Inaganti and G.K. Behara, "Service Identification: BPM and SOA Handshake," *BPTrends,* vol. 3, pp. 1-12, 2007.

[24] M.P. Papazoglou and W.-J. van den Heuvel, "Business Process Development Lifecycle Methodology: Bridging Together the World of Business Processes and Web Services," http://infolab. uvt.nl/pub/papazogloump-2006-89.pdf, 2006.

[25] SAP, "Enterprise Service Design Guide," http://www.sap.com/platform/netweaver/pdf/BWP_ES_Design_Guide. pdf, 2005.

[26] IBM, "Component Business Models,"http://www-935.ibm.com/services/us/imc/pdf/g510-6163-component-business-models. pdf, 2005.

[27] D. Flaxer and A. Nigam, "Realizing Business Components, Business Operations and Business Services," *Proc. IEEE Intl Conf. E-Commerce Technology for Dynamic E-Business (CEC-EAST '04),* pp. 328-332, 2004.

[28] A. Sehmi and B. Schwegler, "Service-Oriented Modeling for Connected Systems—Part 2," *The Architecture J.,* vol. 8, pp. 35-39, 2006.

[29] OASIS, "SOA Blueprints," http://www.oasis-open.org/committees/download.php/15965/05-12-00000.001.doc, 2005.

[30] R.S. Kaabi, C. Souveyet, and C. Rolland, "Eliciting Service Composition in a Goal Driven Manner," *Proc. Second Int'l Conf. Service Oriented Computing (ICSOC '04),* pp. 308-315, 2004.

[31] J.-H. Sewing, M. Rosemann, and M. Dumas, "Process-Oriented Assessment of Web Services," *Intl J. E-Business Research,* vol. 2, no. 1, pp. 19-44, 2006.

[32] M. Bell, *Service-Oriented Modeling. Service Analysis, Design and Architecture.* John Wiley & Sons, 2008.

[33] D. Adamopoulos, G. Pavlou, and C. Papandreou, "Advanced Service Creation Using Distributed Object Technology," *IEEE Comm. Magazine,* vol. 40, no. 3, pp. 146-154, Mar. 2002.

[34] Y. Kim and K.-G. Doh, "The Service Modeling Process Based on Use Case Refactoring," *Proc. 10th Int'l Conf. Business Information Systems (BIS '07),* pp. 108-120, 2007.

[35] Sun, "SOA RQ Methodology. A Pragmatic Approach," http://www.sun.com/products/soa/soa_methodology.pdf, 2006.

[36] B. Gold-Bernstein and W. Ruh, *Enterprise Integration: The Essential Guide to Integration Solutions.* Addison Wesley Longman, 2004.

[37] A. Erradi, N.N. Kulkarni, and P. Maheshwari, "Service Design Process for Reusable Services: Financial Services Case Study," *Proc. Fifth Int'l Conf. Service-Oriented Computing (ICSOC '07),* pp. 606-617, 2007.

[38] D. Quartel, R. Dijkman, and M. van Sinderen, "Methodological Support for Service-Oriented Design with ISDL," *Proc. Second Int'l Conf. Service Oriented Computing (ICSOC '04),* pp. 1-10, 2004.

[39] P. Allen, "The Service Oriented Process," http://www.cbdiforum.com/secure/interact/2007-02/service_oriented_process.php, 2007.

[40] O. Zimmermann, P. Krogdahl, and C. Gee, "Elements of Service-Oriented Analysis and Design. An Interdisciplinary Modeling Approach for SOA Projects," http://www-128.ibm.com/developerworks/webservices/library/ws-soad1, 2004.

[41] Z. Stojanovic, A. Dahanayake, and H. Sol, "Modeling and Design of Service-Oriented Architecture," *Proc. Int'l Conf. Systems, Man, and Cybernetics (SMC '04),* vol. 5, pp. 4147-4152, 2004.

[42] E.A. Marks and M. Bell, *Service-Oriented Architecture. A Planning and Implementation Guide for Business and Technology.* John Wiley & Sons, 2006.

[43] S.H. Chang and S.D. Kim, "A Systematic Approach to Service-Oriented Analysis and Design," *Proc. Eighth Int'l Conf. Product-Focused Software Process Improvement (PROFES '07),* pp. 374-388, 2007.

[44] A. Arsanjani, "Service-Oriented Modeling and Architecture. How to Identify, Specify, and Realize Services for Your SOA," http://www.ibm.com/developerworks/library/ws-soa-design1, 2004.

[45] M.P. Papazoglou and W.-J. van den Heuvel, "Service-Oriented Design and Development Methodology," *Int'l J. Web Eng. and Technology (IJWET),* vol. 2, no. 4, pp. 412-442, 2006.

[46] F. Chen, S. Li, and W.C. -C. Chu, "Feature Analysis for Service-Oriented Reengineering," *Proc. 12th Asia-Pacific Software Eng. Conf. (APSEC '05),* pp. 201-208, 2005.

[47] Z. Zhang, R. Liu, and H. Yang, "Service Identification and Packaging in Service Oriented Reengineering," *Proc. 17th Int'l Conf. Software Eng. and Knowledge Eng. (SEKE '05),* pp. 620-625, 2005.

[48] E. Nadhan, "Seven Steps to a Service-Oriented Evolution," *Business Integration J.,* vol. 1, pp. 41-44, 2004.

[49] A.T. Rahmani, V. Rafe, S. Sedighian, and A. Abbaspour, "An MDA-Based Modeling and Design of Service Oriented Architecture," *Proc. Sixth Int'l Conf. Computational Science (Part 3) (ICCS '06),* pp. 578-585, 2006.

[50] R. Merrifield and J. Tobey, "Motion Lite: A Rapid Application of the Business Architecture Techniques Used by Microsoft Motion," http://msdn2.microsoft.com/en-us/library/bb736727.aspx, 2006.

[51] W. Al-Belushi and Y. Baghdadi, "An Approach to Wrap Legacy Applications into Web Services," *Proc. Int'l Conf. Service Systems and Service Management (ICSSSM '07),* pp. 1-6, June 2007.

[52] T. Kohlborn, "A Consolidated Approach for Service Analyis," master's thesis, WWU Münster, http://eprints.qut.edu.au/archive/00013682, 2008.

[53] M. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, "Service-Oriented Computing: State of the Art and Research Challenges," *Computer,* vol. 40, no. 11, pp. 38-45, 2007.

[54] K. Kontogiannis, G.A. Lewis, D.B. Smith, M. Litoiu, H. Muller, S. Schuster, and E. Stroulia, "The Landscape of Service-Oriented Systems: A Research Perspective," *Proc. Int'l Workshop Systems Development in SOA Environments (SDSOA '07),* p. 1, 2007.

[55] N. Bieberstein, S. Bose, L. Walker, and A. Lynch, "Impact of Service-Oriented Architecture on Enterprise Systems, Organizational Structures, and Individuals," *IBM Systems J.,* vol. 44, no. 4, pp. 691-708, 2005.

[56] T. Erl, *SOA: Principles of Service Design.* Prentice Hall, 2007.

[57] A. Minhas and F. Vogt, "Service Orientation for a Dynamic Enterprise," *Proc. Int'l Conf. Logistics and Supply Chain Management (ILSCM '06),* pp. 1-7, 2006.

[58] C. Legner and T. Vogel, "Design Principles for B2B Services—An Evaluation of Two Alternative Service Designs," *Proc. IEEE Intl Conf. Services Computing (SCC '07),* pp. 372-379, 2007.

[59] J. O'Sullivan, D. Edmond, and A.T. Hofstede, "What's in a Service?" *Distributed Parallel Databases,* vol. 12, nos. 2-3, pp. 117-133, 2002.

[60] G. Feuerlicht, "Design of Service Interfaces for e-Business Applications Using Data Normalization Techniques," *Information Systems and E-Business Management,* vol. 3, no. 4, pp. 363-376, 2005.

[61] U. Homann, "A Business-Oriented Foundation for Service Orientation," http://msdn2.microsoft.com/en-us/library/aa479368.aspx, 2006.

[62] K. Hafeez, Y.B. Zhang, and N. Malak, "Determining Key Capabilities of a Firm Using Analytic Hierarchy Process," *Int'l J. Production Economics,* vol. 76, no. 1, pp. 39-51, 2002.

[63] APQC, "Process Classification Framework v4.0.0," http://www.apqc.org/pcf, 2006.

[64] U. Homann and J. Tobey, "From Capabilities to Services: Moving from a Business Architecture to an IT Implementation," http://msdn2.microsoft.com/en-us/library/aa479075.aspx, 2006.

[65] S. Jones, "Toward an Acceptable Definition of Service," *IEEE Software,* vol. 22, no. 3, pp. 87-93, 2005.

[66] P. Malinverno, "Service-Oriented Architecture Craves Governance," Technical Report G00135396, Gartner Research, 2006.

[67] T. Donaldson and L. Preston, "The Stakeholder Theory of the Corporation: Concepts, Evidence, and Implications," *The Academy of Management Rev.,* vol. 20, no. 1, pp. 65-91, 1995.

[68] C. Lovelock and J. Wirtz, *Service Marketing: People, Technology, Strategy,* fifth ed. Prentice Hall, 2004.

[69] S. Fließ and M. Kleinaltenkamp, "Blueprinting the Service Company. Managing Service Processes Efficiently," *J. Business Research,* vol. 57, no. 4, pp. 392-404, 2004.

[70] T. Saaty, *Decision Making for Leaders. The Analytical Hierarchy Process for Decisions in a Complex World.* RWS Publications, 1990.

[71] M. Pezzini, "Findings: Greater Business Process Insight Is an Unexpected Benefit of SOA," Technical Report G00153427, Gartner Research, 2007.

Thomas Kohlborn received the MS degree in information systems at the University of Münster, Germany, in 2008. The major subjects he studied were information systems, business studies, and controlling. He wrote his Master's thesis at the Queensland University of Technology in cooperation with SAP Research Brisbane and the Queensland Government, Department of Public Works. Currently, he is enrolled at the Queensland University of Technology as a PhD candidate working on service analysis, design, and governance.

Axel Korthaus received the MS degree in information systems from the University of Mannheim, Germany, where he also received the PhD degree in 2001 and worked as a postdoctoral lecturer and researcher until 2008. He currently works as a postdoctoral research fellow at the Queensland University of Technology and manages a project about service ecosystems, which involves SAP Research Brisbane and the Queensland Government, Department of Public Works, as partners. His main areas of interest include service-oriented architectures and software engineering. He is a member of the IEEE and the IEEE Computer Society.

Taizan Chan received the PhD degree from the National University of Singapore in 1998. He is currently a senior lecturer in the Faculty of Information Technology at the Queensland University of Technology. His research interests include economics of systems management, the impact and value of IT, and enterprise systems management. He is the author/coauthor of over 30 refereed publications in international journals. He was a visiting senior fellow in the Wharton School of Business, University of Pennsylvania, in 1995 and 2001, and a visiting scholar in the Haas School of Business, University of California, Berkley, in 2004.

Michael Rosemann received the MBA degree in 1992 and the PhD degree in 1995 from the University of Münster, Germany. He is a professor of information systems and a co-leader of the Business Process Management Group at the Queensland University of Technology, Brisbane, Australia. He is the author/editor of six books, more than 140 refereed papers, and an editorial board member of seven international journals.