

# Design Principles for B2B Services – An Evaluation of Two Alternative Service Designs

Christine Legner, Tobias Vogel  
Institute of Information Management, University of St. Gallen  
CH-9000 St. Gallen  
{christine.legner, tobias.vogel}@unisg.ch

## Abstract

*A service-oriented architecture (SOA) promises a more flexible intra- and interorganizational integration of heterogeneous application systems. The central design element of a SOA consists of services which encapsulate existing application logic, and can be dynamically combined to form business processes. This paper aims at contributing to the emerging discipline of “service engineering” by investigating the design principles for B2B services. The authors draw on an analysis of the literature to derive the main design principles for services and evaluate two alternative design proposals for a concrete application scenario in the automotive industry. The results comprise on the one hand statements on the quality of the service design in the concrete case and on the other the outlook for future topics of research in “service engineering”.*

## 1. Introduction

A service-oriented architecture (SOA) provides application logic as autonomous, platform-independent services which, according to the vision, can be dynamically combined to form business processes [1]. Thus, a SOA promises a more flexible intra- and interorganizational integration of heterogeneous application systems. Thanks to SOAs, analysts expect to see a 20% reduction in application development and integration costs [2] and predict that by 2009 over 80% of newly developed applications will be based on services [3]. Significant benefits are to be expected if enterprises could leverage SOA to easily encapsulate internal application functionality as a service and expose it to their customers and suppliers [4, 5].

However, many of the benefits of an SOA can only be reaped with the careful and systematic design of services. As a consequence, the discussion of the design principles on which the services should be based, the granularity they should ideally possess and how they are to be derived from the business requirements takes on particular importance. The agreement and general acceptance of common technical and semantic standards and design principles become more important with the number of autonomous partners involved, but – in view of the

absence of any authority to issue directives among equal partners – also grows in complexity [6].

This paper aims at contributing to the emerging discipline of “service engineering” [7] by investigating design principles for B2B services. To this purpose, the authors analyze service design principles formulated in the existing literature on SOA and apply them to a concrete B2B scenario. Two questions are of particular interest in this context: (1) Is it possible to assess the quality of different B2B service designs based on the suggested design principles? (2) Do the suggested design principles satisfy the practical requirements related to B2B service design?

The remainder of the paper is structured as follows: Section 2 reviews the existing SOA literature and deduces a set of service design principles. These design principles are applied to a concrete application scenario in Section 3 in order to evaluate two alternative service design proposals. Engineering change management in the automotive industry was chosen as an example, with two implementation alternatives based on web services available in the form of the OMG PLM Service and the ECR Business Service. Section 4 derives statements on the quality of the service design in the concrete case on the one hand and on the other provides an outlook for future topics of research in “service engineering”.

## 2. Service design

### 2.1. Service typology

The understanding of the SOA concept in the context of this paper is based on the definitions of [8-11]: An SOA is a multiple-layer, distributed information system (IS) architecture which encapsulates parts of the application logic as services. It can be considered an architectural style [12] which relies on services as key design element. Services represent abstract software elements and/or interfaces which provide other applications with stable, reusable software functionality at a business-related level of granularity using widely applied standards.

Different approaches to a more detailed breakdown of the service concept in the form of classifications can be found in the literature (c.f. Table 1). A distinction is frequently drawn according to the type of service functiona-

lity between business services (e.g. checks on credit rating) and technical services (integration and infrastructure services, e.g. user authentication). In addition, services can be split according to their business granularity into “utility services” (support generic functions, e.g. authentication), “entity services” (offer CRUD operations for an entity, e.g. customer), “process services” (offer a process-specific service, e.g. availability check, or go as far as a more comprehensive business process, e.g. order receipt and validation). A further distinction is made e.g. by [13] between atomic (“application”) services and composition services (“enterprise” or “business services”). The former involve services which encapsulate existing application functionality as adapters and make them available at the service level. The design of application services tends to be motivated from the bottom up from the existing application interfaces. In contrast, composition services are identified from the requirements of the business process and can combine business-oriented and technical services (such as a service for requesting information with an access control service). Services are also differentiated according to the style of interaction used and the stateless / stateful property. Finally, aspects of accessibility and standardization reach [14] frequently play a role in classifications. While some services are restricted to internal use only, other services may be exposed to business partners encapsulating business functionality for inter-organizational coordination.

**Table 1. Service typology**

Service Type	Business		Technical / Infrastructural	
Granularity	Business process	Process activity / Task	Entity / business object	Utility
Composition	Composite service (enterprise serv.)		Atomic service (application serv.)	
Interaction Style	Synchronous		Asynchronous	
Exchange Patterns	Request / Response	Notification	...	
State	Stateless		Stateful	
Accessibility	Within the organization (internal)		Outside the organization (external)	

## 2.2. Service design principles

While there is broad agreement on the architectural components and layers of an SOA in the literature, opinions differ more widely when it comes to the service

design principles that should be applied. Diverging opinions exist, for example, on whether the term SOA is tied to the use of web service standards (cf. [15] and [16]), or whether services only support a synchronous “request / reply” interaction style or asynchronous communication as well (cf. e.g. [17]). Many authors see “loose coupling” as a core feature of SOAs. However, while [18] understands this to mean the use of specific communication mechanisms based on standards, for [19] the term refers to the logical independence of architectural components which allows them to evolve separately from one another. [20] associates this with the hiding of internal structures, which other authors tend to classify under the characteristic of abstraction [10, 19].

Analysis of the design criteria for services proposed in the literature highlights the challenge posed by the fact that

- different authors emphasize different design principles,
- they define the design principles with differing degrees of detail,
- there is an overlap in the content of differently designated principles
- or that principles with the same content are given different names.

For this reason, the design principles most frequently cited by science, standardization bodies and consultants are systematized in the next section and the respective concepts clarified (c.f. Table 2). Scientific and practice-related publications provide the basis for the analysis.

**2.3.1. Interface orientation.** Most sources postulate that services have to abstract from implementation details [10] and provide well-defined interfaces described in an implementation-independent manner. Service consumers should not require any information above and beyond the service specification in order to be able to invoke them [21]. Thus, a comprehensive service specification not only contains a technical interface description but also describes semantic and dynamic attributes and quality characteristics of a service [22]. Service interfaces in a SOA represent stable, binding contracts between service providers and users. They are managed in a central repository and are only adapted in clearly defined modification cycles [23].

**2.3.2. Interoperability.** In order to guarantee seamless integration of applications in a heterogeneous environment, a SOA relies on interoperable, standard-based interfaces. Services possess uniform interface descriptions and communicate by means of uniform protocols and data formats [20]. Although a SOA is not tied to a specific technology, web services constitute a more and more applied, ambitious approach to platform- and vendor-independent standards on the transport and

communication layer. Some authors postulate that technical standardization has to be complemented by common semantics for business tasks and data [24]. For this technical and business standardization, SOAs should, if possible, use open and widely applied industry standards [19].

**Table 2: Service design principles (c.f. [10, 24-26])**

Category	Design Principles
<b>Interface Orientation</b>	Abstraction from service implementation
	Comprehensive, uniform service specification
	Stable, managed service contracts
<b>Inter-operability</b>	Technical standardization
	Business standardization
	Use of open, widely applied industry standards
<b>Autonomy and Modularity</b>	High service cohesion and weak logical coupling
	Loosely coupled communication
<b>Business Suitability</b>	Service granularity oriented toward business concepts
	Generalization of services

**2.3.3. Autonomy and modularity.** A SOA decomposes the existing application architecture and structures it into a manageable number of partially autonomous subsystems, i.e. domains and services. In accordance with well-known principles of software component design, functions or resources with high interdependency (cohesion) are grouped together in such a way that their logical, design-time dependency on other subsystems (loose coupling) is as low as possible [22, 27]. Besides the logical de-coupling, loosely coupled communication reduces runtime dependencies. It can be achieved by means of dynamic service addressing via a logical name (e.g. a uniform resource identifier, URI), asynchronous, message-based communication between service users and providers and stateless service interaction [18, 26, 28].

**2.3.4. Business suitability.** Although service granularity, i.e. the scope of functionality a service exposes, is considered a key design decision within a SOA, there is an ongoing debate on the extent to which services should reflect business concepts. Fine-grained services address small units of functionality or exchange small amounts of data. In order to realize complex business scenarios in a

distributed environment, coarse-grained services which exchange a larger quantity of data in one operation and support largely complete process activities are said to be more appropriate [25]. In order to achieve reusability services should also be sufficiently generic to allow their reuse in several processes and / or by several users [24].

### 3. Application in the Automotive Industry

#### 3.1. Background

The obvious potentials of SOA for inter- and intraorganizational integration have led to numerous ideas on how industry standards might be combined with service-oriented concepts. However, there is a broad spectrum of alternative service designs which have been suggested so far. This has also been the case with engineering change management (ECM) in the automotive industry. For the purposes of implementing an emerging industry standard suggested by the Association of German Automobile Manufacturers (VDA), two different design approaches for B2B services are available:

- The ProSTEP iViP Association recommends the PLM Services standardized by the OMG (Object Management Group) for implementation.
- At the same time, a group of automotive OEMs and suppliers developed the “ECR Business Service” which translates the VDA recommendation directly into web services.

If services are used – as in the present case – not only for intra-organizational but also for inter-organizational integration, this gives rise to a number of specific boundary conditions [29]. On the technical layer that includes the use of the internet as physical communication infrastructure and on the organizational layer the higher autonomy of the parties involved. Both approaches leverage web service technology by defining platform-independent, document-oriented web services. There are nonetheless significant differences between the two service design alternatives which are reflected by the operations signatures as exposed in the service interface. Sections 3.3.-3.4. describe the information and function model of PLM Services and ECR Business Service in more detail. The information model refers to the messages which the service expects as input or output parameter, and the fundamental data objects they are composed of. The function model, on the other hand, describes the service operations. In order to evaluate the quality of the different service designs, we will build on the design principles from literature and apply them to the two alternative service design proposals.

### 3.2. Engineering Change Management

The purpose of engineering change management is the real-time propagation of engineering change requests in development, planning and manufacturing processes between supplier and manufacturer. Possible triggers for changes include amongst others modification in product design, quality or safety problems. Once a need for change has been detected and possible solution alternatives have been described, they must then be analyzed for effectiveness and implementability. This is followed by a comprehensive economic and technical evaluation which provides the basis for a decision on the change request and its rollout to production. As a result of the fact that automobile manufacturers are constantly reducing their levels of in-house production, the number of suppliers and engineering service providers, which are involved in a simultaneous product development, increases. This forces automotive OEMs and suppliers to realize collaborative engineering change management processes.

Developed in a joint effort by suppliers, manufactures and software vendors the VDA Recommendation 4965 creates a common understanding of engineering change processes, in particular Engineering Change Requests (ECR) [30]. It defines a role model and a reference process including milestones, phases and synchronization points. Process descriptions include UML diagrams. In addition, it contains a data dictionary and an information model (noted in Express-G) on that base messages are defined. As a pure business standard, the VDA Recommendation initially concentrated on process and data models. In the light of the experience gained from the first pilots, which range from EDI-based implementation to rich client applications accessing multiple PLM systems, the VDA would now like to work toward stronger standardization in the area of IT implementation as well, with the ultimate aim of ensuring the interoperability of approaches and solutions. In this respect a service-oriented approach could offer significant improvements in the design of expandable and scalable architecture by making services available via interactive portals on the one hand and on the other via standardized interfaces for automated processing [4].

### 3.3. OMG PLM Services 2.0

PLM Services 2.0 are a further development of PLM Services 1.0, which the OMG had already standardized in 2005 [31]. The PLM Services covers various applications for product data exchange in simultaneous / collaborative engineering. In Version 1.0, the focus was on exchanging product structure data by coupling rich clients at the supplier's end with the automotive manufacturers' backend systems. In Version 2.0 the PLM Services are being

expanded so that messages from engineering change management can also be exchanged.

**3.3.1. The information model.** In focusing on product data exchange, PLM Services build on the AP 214 STEP data model. STEP AP 214 as ISO 10303-214 describes a core data model for automotive mechanical design processes [32]. The mapping of the ECR information model onto AP 214 data elements is a mandatory prerequisite for the use of the PLM Services. The information model of the PLM Services (Figure 1) contains a generic container referred to as the "PLM\_container", and a context element, the "PLM\_context", which are both encompassed by the so called "PLM\_message". It is the PLM\_message that is exchanged between the PLM backend systems. The superclass PLM\_objects represents the AP 214 data building blocks such as activity, item, date etc. As a result of the complexity of the AP214 data model, hierarchical and logical associations between PLM\_objects are expressed by means of reference relationships, thus resulting in a flat, sequential XML structure. Eventually, the PLM\_context incorporates message type, sender identification along with other control information.

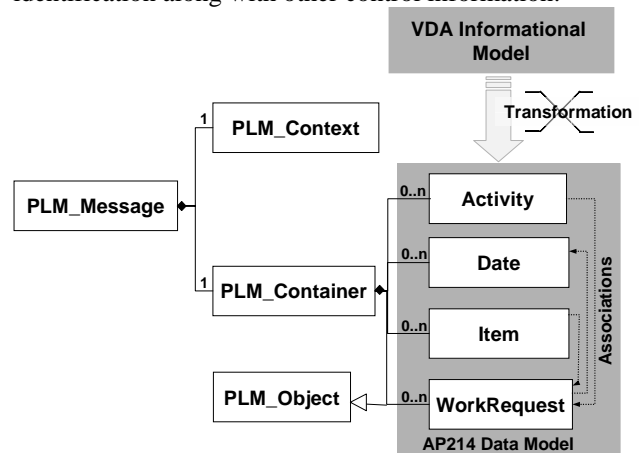


Figure 1. Information model of PLM Services 2.0

**3.2.3. The function model.** The original focus of PLM Services 1.0 was on client-server solutions and designed as a connection-oriented approach. Following successful authentication, the client is given a session ID which is to be retained in subsequent service operations. The operations of PLM Services 1.0 enable querying, up-/down-loading and manipulating (creating, deleting, updating) product data. For the asynchronous transfer of ECR messages, the operation write\_messages was added in version 2.0, with which any ECR messages can be transferred. The write\_message operation expects a PLM\_message which contains the ECR message elements mapped onto AP 214 elements within a PLM\_container. As a consequence of the security mechanisms of PLM Services 1.0, an authentication must be performed before

actual message transfer, whose protocol does not comply with any official WS-Standard. That means that backend coupling on this basis increases the number of interactions between process participants due to connection establishment on functional rather on technical layer.

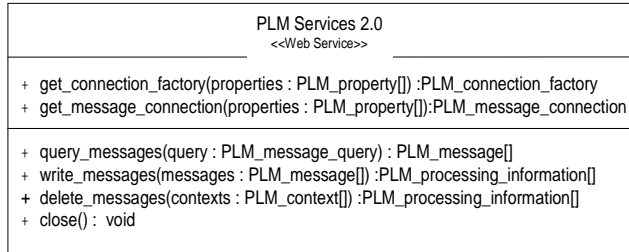


Figure 2. Function model of PLM Services 2.0

### 3.4. ECR Business Service

Development of the ECR Business Service was initiated by a group of automotive companies in order to explore the potentials of service-orientation for B2B integration. In order to translate the VDA recommendation into web services, it builds on successful e-business concepts, such as the process-based XML standards for B2B and A2A integration developed by the OAGi (Open Applications Group, Inc.), a member of both WS-I and OASIS.

**3.2.5. Information model.** The ECR Business Service maps the data objects and messages of the VDA recommendation directly into an XML schema definition. To this purpose, it relies strongly on the information model of the business standard which is formally noted in Express-G and semantically specified by a data dictionary. Each message of the VDA recommendation is based on a combination of different data components which are defined in the data dictionary of the VDA recommendation. They are declared as XML complex types in a separate XML schema file. The development of the message structure follows technology-neutral framework of OAGIS for designing Business Object Documents (BODs). In accordance with the Naming and Design Rules of the OAGi [33], the ECR messages are split into ApplicationArea and DataArea. The ApplicationArea provides the meta information describing the message. Thus far, container elements are available for sender-specific information and for error handling. The actual payload of the message is defined in the DataArea which contains four components for an engineering change message. The MessageRequestContext holds business control information, e.g. DUNS number, which is stipulated as message context by the VDA Recommendation. Whereas the MessageSpecificPart contains the specific data components for the appropriate message, the MessageBasicPart encapsulates the data components

which are common to all messages. The element MessageUserDefinedPart is another container element which is used to extend messages in accordance with bilateral agreements and at the same time to ensure separation from the VDA-specific parts of the message.

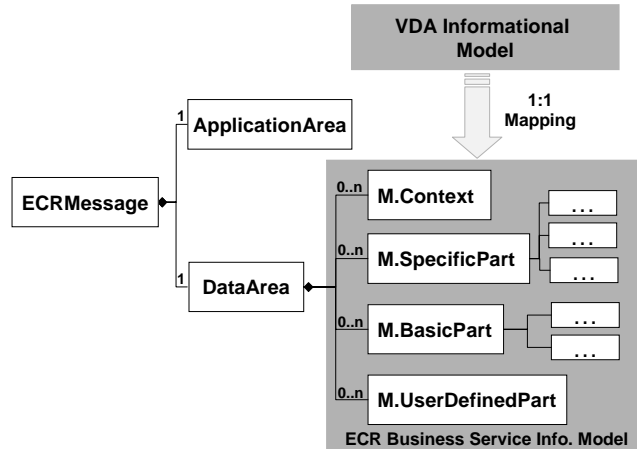


Figure 3. Information model of ECR Business Service (in compliance with OAGIS Naming and Design Rules)

**3.2.6 Function model.** The function model of the ECR Business Service reflects the 11 messages which are defined in the VDA Recommendation and characterize the different interactions. For each of these messages a service operation is provided which expects the XML representation of the message as input parameter. The ECR Business Service merely returns a synchronous acknowledge message which signalsizes correct receipt at the partner's end. In contrast, the business reply to the message is sent asynchronously as ECR processing constitutes a long running transaction (up to several weeks).

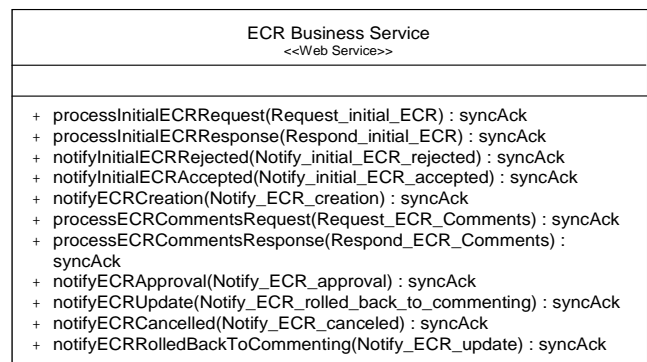


Figure 4. Function model of the ECR Business Service

## 4. Evaluation

### 4.1. Conformity to service design principles

In order to evaluate the two suggested service designs, we refer to the service design principles outlined in Section 2.

Both ECR Business Services and PLM Services 2.0 take into account the aspect of **interface orientation** by providing a WSDL document as comprehensive, uniform service specification. The WSDL document represents a stable interface in the sense of a contract and abstracts from any implementation-specific details. Given the very generic service operation `write_messages` and the generic structure of the `PLM_container`, stability of the PLM Services interface is very high. The ECR Business Service interface is more likely to be affected by changes in the industry standard.

Both service design approaches, ensure **interoperability** by relying on existing technical and business stan-

dards. The ECR Business Service and the PLM Services use the WS-Standards in accordance with the WS-I Basic Profile. In addition, the use of WS-Security and further WS-I Profiles is being prepared. From the point of view of business standardization, both approaches are based on the VDA Recommendation on Engineering Change Management. However, the differences are obvious: PLM Services 2.0 uses the STEP AP214 data model for implementing the standard in order to ensure interoperability in the entire Product Lifecycle Management (PLM) domain. The ECR Business Service directly translates the VDA Recommendation into service design and applies further e-business standards, such as the OAGIS Naming and Design Rules which closely follow the UN/CEFACT standard and thus ensure interoperability across domains.

**Autonomy and modularity** are design principles which structure a SOA in domains and services along the lines of logical dependencies. The ECR Business Services encapsulate the functionality required for processing an

**Table 3: Conformity of alternative service design proposals with SOA design criteria**

	PLM Services 2.0	ECR Business Service
<b>Interface Orientation</b>		
Abstraction from service implementation	Yes	Yes
Comprehensive, uniform service specification	WSDL	WSDL (including semantic annotation)
Stable, managed service contracts	Very stable, through genericity of function and information model	Depending on changes in the industry standard (VDA Recommendation)
<b>Interoperability</b>		
Technical standardization	WS Core standards, WS-I Basic Profile, WS Security (planned)	WS Core standards, WS-I Basic Profile, WS-I Basic Security Profile, WS-I Reliable Secure Profile
Business standardization	Based on industry standards, for the PLM domain	Based on industry standards, restricted to Engineering Change Management
Use of open, widely applied industry standards	VDA Recommendation 4965 (ECR) OMG PLM Services 1.0 / 2.0	VDA Recommendation 4965 (ECR) OAGIS 9.0 Naming and Design Rules
<b>Autonomy and Modularity</b>		
High service cohesion and weak logical coupling	Medium cohesion (encapsulation of functions and data for the ECR process and for product data exchange)	High cohesion (encapsulation of functions and data for the ECR process)
Loosely coupled communication	Stateless, but with correlation ID	Stateless, but with correlation ID
<b>Business Suitability</b>		
Service granularity oriented toward business concepts	No, generic service operation <code>write_messages</code> and generic data container	Yes, business document approach
Generalization of services	Yes, very generic	Medium, possibility of message extensions as stipulated by OAGIS

engineering change request strictly in accordance with the VDA recommendation. The exchange of product data is considered to be outside of this domain, i.e. product data are referenced or exchanged if necessary by calling a distinct service for product data exchange. In contrast, PLM Services 2.0 encapsulate functions and data from the domains of product exchange as well as engineering change management in order to ensure the stability of the interface. This implies logical coupling of these two domains. In order to meet the requirement for loosely coupled communication between the domains, communication is performed stateless in both cases. The sequence of messages on the business layer must then be secured with the aid of correlation identifiers which exist in both approaches.

**Business suitability** postulates that services should possess a coarse service granularity oriented toward business concepts. As already mentioned in the relevant sections, the ECR Business Service directly meets this criterion through direct derivation from the business specification. The design of service operations as well as input parameters caters for this aspect. The PLM Services, on the other hand, are characterized by genericity as only one operation (`write_messages`) is necessary to cover all applications in ECR and even in PLM. This means that the PLM Services are to a large degree generic and can react flexibly to future changes, provided that these can be mapped to the data format STEP AP 214. The ECR Business Service itself, on the other hand, is more specific due to the consistent derivation from the business standard. An extension to include other applications beyond the ECR process is in principle not possible and does not seem opportune in view of the aspect of autonomy and modularity already discussed. Business Suitability also implies that services can be easily orchestrated in internal workflows, e.g. by means of a BPM / workflow tool [16]. A document-oriented approach, as used by the ECR Business Services, facilitates integration of the service into workflows and graphical user interfaces, e.g. in a portal frontend. The PLM Services can theoretically be orchestrated as well. However, as a result of the use of the AP 214 data model, this results in additional transformations between ECR and STEP AP 214 data models.

## 4.2. Discussion

On the technical layer, B2B services build on web service standards, in particular through conformity with the WS-I profile. Nonetheless, the two service designs show considerable differences in the mapping of a business standard onto web service standards. This is evident in the information and function model of the services. Each of the two service concepts emphasizes different design principles, but none of them can be considered a “dominant design” given the service design principles from SOA literature. While PLM Services lend greater weight

to the stability of the interfaces and otherwise give preference to a strongly data-oriented exchange, the ECR Business Service is more oriented toward process-oriented message exchange. This is due to the fact that the ECR Business Service was designed as an inter-organizational business service with a coarse granularity by mapping the interactions outlined in the reference process 1:1 on service operations. It consequently provides a larger number of service operations, whereas the PLM Services were originally intended as an application service and as such were to provide a uniform interface to the different PLM systems. Even if there has been a shift in focus for the use of Version 2.0 of the PLM Services, the connection-oriented, generic interface design hampers its use for inter-organizational application integration.

In view of the two significantly different service concepts, our analysis demonstrates that design principles from SOA literature do not satisfy the practical requirements. On the one hand, the existing design principles are too generic and need to be operationalized, e.g. by introducing quality criteria and by applying engineering principles for service design. On the other hand, service design principles have to be related to the specific type of service under consideration. Whereas genericity and reuse may be key aspects in the design of software components, their relevance for building coarse grained B2B services might be a topic for further discussions. As another example, the generic data container approach which has been realized by PLM Services does not permit schema validation on the message layer and the checking of referential consistency. This negatively impacts maintainability and robustness, which are important aspects given previous experiences from B2B integration. With regard to the specific area of B2B integration, we would encourage the discipline of service engineering to come up with guidelines for designing web services based on industry-wide standard business processes and vocabularies as already postulated by [5]. This involves definition of web services operations and their input and output messages using WSDL. Recent activities by OAGi and UN/CEFACT seem to take on this path.

## 5. Summary and Outlook

The main focus of this paper was the identification of a set of design principles which permit qualitative statements for the evaluation of service design alternatives within the framework of an SOA. From the literature review, the suggested design principles have been classified into four categories: Interface orientation, interoperability, autonomy and modularity and business suitability. The application of the design principles from the literature to the concrete scenario of engineering change management leads to the following findings:

- The design principles are to be weighted differently according to the intended use of the service. In a concrete case, the business derivation and the high cohesion of the service operations and messages of the ECR Services are to be weighed up against the genericity and stability of the interface of the OMG PLM Services.
- The SOA design principles identified in the literature are very generic. In the context of a concrete architecture evaluation they are to be supplemented by general architectural quality features for the SOA area of application. In the case cited in this paper, for example, general recommendations for the design of B2B services should be given greater consideration.

It can therefore be concluded that there is a future need for research service engineering. This will describe categories of services for which common design goals and principles as well as other common features apply (e.g. same development methodologies). A typology of this kind can serve as a design and evaluation pattern for similar and/or recurring design problems.

## 6. References

- [1] F. Curbera, R. Khalaf, N. Mukhi, S. Tai, and S. Weerawarana, "The next step in Web Services", *Communications of the ACM*, 46 (10), pp. 29-34, 2003.
- [2] J. Feiman and M. Hotle, "SODA Reduces Development Efforts", Gartner inc., Stamford 2004.
- [3] S. Hayward, "Positions 2005: Service-oriented Architecture Adds Flexibility to Business Processes", Gartner inc., Stamford 2005.
- [4] O. Zimmermann, V. Doubrovski, J. Grundler, and K. Hogg, "Service-Oriented Architecture and Business Process Choreography in an Order Management Scenario: Rationale, Concepts, Lessons Learned", presented at Oopsla'05, San Diego, California, 2005.
- [5] G. Feuerlicht, "Design of service interfaces for e-business applications using data normalization techniques", *Information Systems and E-Business Management*, 3 (4), pp. 363-376, 2005.
- [6] P. Voigtmann and T. Zeller, "Beiträge zur Integrationsproblematik im Kontext von Electronic Business und Elektronischen Marktplätzen", in *Wirtschaftsinformatik 2003/Band I*, W. Uhr, W. Esswein, and E. Schoop, Eds., 1 ed. Physica-Verlag, Heidelberg, 2003, pp. 215-237.
- [7] M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, "Service-Oriented Computing Research Roadmap", European Union Information Society Technologies (IST), Directorate D - Software Technologies (ST) 2006.
- [8] N. Gioldasis, N. Moumoutzis, F. Kazasis, N. Pappas, and S. Christodoulakis, "A Service Oriented Architecture for Managing Operational Strategies", presented at ICWS-Europe 2003, LNCS 2853, 2003.
- [9] Oasis, "Reference Model for Service Oriented Architectures, Working Draft 09", OASIS Open 2005.
- [10] W3C, "Web Services Architecture, W3C Working Group Note 11 February 2004", last accessed 26.06.2004, <http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/>, 2004.
- [11] J. van Zyl, "A perspective on service based architecture", presented at SAICSIT, 2002.
- [12] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures". University of California, Irvine: 2000.
- [13] D. Krafzig, K. Banke, and D. Slama, *Enterprise SOA - Service-Oriented Architecture - Best Practices*, 1 ed. Prentice Hall PTR, Upper Saddle River NJ, 2004.
- [14] B. Lublinsky, "SOA Design: Meeting in the Middle", *Java Pro* (August), pp. 26-31, 2004.
- [15] O. Zimmermann, M. Tomlinson, and S. Peuser, *Perspectives on Web Services*. Springer, Berlin, 2003.
- [16] W. Dostal, M. Jeckle, and I. Melzer, *Service-orientierte Architekturen mit Web Services*, 1 ed. Spektrum Akademischer Verlag, München, 2005.
- [17] Y. Natis and R. Schulte, "Introduction to Service-Oriented Architecture", Gartner Group 2003.
- [18] A. Brown, S. Johnston, and K. Kelly, "Using service-oriented architecture and component-based development to build web service applications", Rational Software Corporation 2002.
- [19] F.-J. Fritz, "An Introduction to the Principles of Enterprise Services Architecture (ESA)", *SAPinsider*, April-June, 2004.
- [20] M. P. Papazoglou, "Service-Oriented Computing: Concepts, Characteristics and Directions", 2003.
- [21] R. Baskerville, M. Cavallari, K. Hjort-Madsen, J. Pries-Heje, M. Sorrentino, and F. Virili, "Extensible architectures: The strategic value of service-oriented architecture in banking", presented at European Conference on Information Systems (ECIS), 2005.
- [22] M. P. Papazoglou and J. Yang, "Design Methodology for Web Services and Business Processes", in *Technologies for E-Services : Third International Workshop, TES 2002*, A. C. Buchmann, F.; Fiege, L.; Hsu, M.-C.; Shan, M.-C., Ed. Springer, Berlin, 2002, pp. 54-64.
- [23] M. Klesse, F. Wortmann, and J. Schelp, "Erfolgsfaktoren der Applikationsintegration", *Wirtschaftsinformatik*, 47 (4), pp. 259-267, 2005.
- [24] E. Newcomer and G. Lomow, *Understanding SOA with Web Services*. Addison-Wesley, Maryland, 2004.
- [25] J. McGovern, S. Tyagi, M. Stevens, and S. Mathew, "Service-Oriented Architecture", in *Java Web Services Architecture*, J. McGovern, S. Tyagi, M. Stevens, and S. Mathew, Eds. Morgan Kaufmann, San Francisco, 2003, pp. 35-63.
- [26] T. Erl, *Service-Oriented Architecture*. Prentice Hall, New York, 2005.
- [27] S. Vinoski, "Old Measures for New Services", *IEEE Internet Computing*, 9 (6), pp. 72-74, 2005.
- [28] D. Kossmann and F. Leymann, "Web Services", *Informatik-Spektrum*, 27 (2), pp. 117-128, 2004.

- [29] G. Samtani, *B2B Integration - A Practical Guide to Collaborative E-Commerce*. Imperial College Press, London, 2002.
- [30] Association of German Automobile Manufacturers (VDA), "VDA Recommendation 4965 - Engineering Change Management (ECM), Version 1.1", last accessed 25.02.2007, [http://www.vda.de/cgi-bin/paperorder.cgi?search\\_ressort=7&all\\_ranges=1&search\\_now=1](http://www.vda.de/cgi-bin/paperorder.cgi?search_ressort=7&all_ranges=1&search_now=1), 2005.
- [31] PROSTEP iViP Association / Object Management Group, "Product Lifecycle Management Services - OMG Draft Adopted Specification", last accessed 25.02.2007, <http://www.omg.org/docs/dtc/04-05-05.pdf>, 2005.
- [32] R. Anderl and D. Trippener, *STEP, Standard for the Exchange of Product Model Data*. B. G. Teubner, Stuttgart, 2000.
- [33] OAGi, "OAGIS 9 Naming and Design Rules", last accessed 19.6.2006, <http://www.openapplications.org/downloads/oagis/loading90NDR.htm>, 2006.