Managing complex service portfolios: A business case from a full service provider

Stephan Klingner[1], *Martin Böttcher*[1], *Michael Becker*[1], *Arndt Döhler*[2], *Frank Schumacher*[1]

[1]University of Leipzig, [2]Intershop Communications AG

Due to ongoing and far-reaching changes in the service sector, service companies are faced with various challenges. The increasing demand for individualisation as well as a growing competition necessitates flexible, effective as well as efficient methods for developing and managing complex service portfolios. This paper presents findings and possible solutions gained in a business case conducted with a full-service e-commerce provider.

1. Introduction

Similar to the intensified customer-dependent individualisation of offers in the market of industrial engineering in the last centuries, the service sector nowadays faces comparable challenges. Customers demand services highly aligned with their specific needs, which leads to the need for broadly diversified company service portfolios. Therefore, individualisation results in a new quantitative dimension of service offers and a required flexibility regarding their composition in equal measure.

At the same time, a growing competitive environment intensifies the necessity for a higher productivity. To counteract the challenge between a higher individualisation on the one hand and the increasing economical requirements on the other, adequate methods and tools for developing, structuring, and composing services are needed.

The business case presented in this paper illustrates the challenges, approaches, and findings of the full-service e-commerce provider Intershop Communications AG[1] regarding the structural evolution of their service portfolio and its adaptation according to the new demands. The Intershop Communications AG maintains a holistic and comprehensive service portfolio for various kinds of online businesses, comprising services such as software development, hosting, shop management, online marketing and others.

Scientifically this process was supported by the department of Business Information Systems of the faculty of mathematics and computer science of the University of Leipzig. The conducted research work comprised a literature review, the development of the theoretical groundwork, and the engineering of a software prototype. The prototype serves as a supportive tool for structuring services and is used for evaluation purposes of the theoretical findings.

[1] http://www.intershop.com

Subsequent, the specific challenges and problems of the business case are described in further detail. Chapter 2 presents the process of developing an actual solution for Intershop, beginning with the theoretical foundation gathered in a literature review. Moreover, a metamodel as a framework for further implementations is described. Based on that, a software prototype is introduced in chapter 4. Chapter 5 concludes the paper.

2. The business case

The Intershop Communications AG is a medium-sized company providing global e-commerce solutions since the early 1990s. The company's customers profile consists to one-third of customers active in the B2B market; two-thirds operate in the B2C segment. Several key accounts such as Daimler, Telekom, or Hewlett-Packard generate the revenue's majority.

Although in the beginning the selling of e-commerce software was the core business of Intershop and services were only offered as augmentation, in the recent years this focus has been shifted for the benefit of service offerings. Nowadays, the quota of the turnover achieved with selling of standard software lies at approx. 11%, whereas services generate a portion of 89% (Intershop, 2011).

The transformation from a software development company to an all-encompassing e-commerce provider resulted in various new challenges regarding the development and management of the service portfolio (Böttcher, et al., 2011). First improvements were made by the subdivision of the service portfolio into modules, such as Online Marketing, Content Management or Technical Operations. The aim of modularisation was to provide an easier option for customer specific configurations. This improved flexibility allows for an easier adaptation of costumer's needs, which on the one hand extends the offered portfolio and on the other hand supports the application for public tenders. However, for a holistic and effective service management more powerful approaches were required, since so far modularisation was conducted on a coarse-grained level and in a less formal way. The collection of requirements was the first step towards a metamodel for the description of the service portfolio.

One articulated wish was the support for a more precise price calculation and profitability analysis. So far, the profitability of a service module was only measured very coarse-grained and not standardised on company level. Therefore, the offered services are based on a mixed calculation, which can decrease the company's profit as well as competitiveness. Comparably, a detailed productivity analysis of single modules as well as of specific customer configurations was desired. Similarly, as Intershop sometimes also operates e-commerce services itself, a configuration dependent risk evaluation was required, to support price calculation and investment decisions.

Furthermore, the so far applied modularisation lacks the possibility to describe complex logical and temporal dependencies in a formal way, since the portfolio is mainly described using common office software, such as Excel or Word. To provide configurators for more complex services, however this is required, since single modules might require the previous execution of other modules or a combination of modules might be invalid.

Another major challenge was the intra-company transparency regarding the service portfolio. Up to now, a list containing all available services and components is maintained centrally and customer-specific services are assembled individually. New features were introduced company-wide during quarterly conducted sales meetings. At the same time many account managers, especially of key customers, have developed their “own” modules as customer-specific addition or modification of the company’s portfolio. To increase the transparency of the portfolio and also including combinatorial and organisation constraints, an explicit specification of services on a component-level is aspired.

Based on these requirements a solution was developed, as deduced and described in more detail in the following chapter.

3. Solution

The management of complex portfolios is a common task in other domains such as industrial and software engineering (Böttcher & Klingner, 2011). Therefore, a literature review was conducted to identify approaches and concepts already utilised and established in other areas. Similarly, literature focussing on service specific aspects was included as well.

A fundamental requirement for the centralised, unified, and company-wide management and maintenance of all offered services is standardisation. Furthermore, various benefits can be gained through standardisation. Ulrich defines standardisation as “the use of the same component in multiple products” (Ulrich, 1995), which can be adequately applied to services.

A certain amount of repetition provided, the reuse of standardised components can support economies of scale, as employed using the paradigm of mass customisation (Jiao, et al., 2003). Likewise, in-house transparency as well as the consistency of the portfolio offered to different customers is increased by standardisation.

A further distinction is made by Ulrich, who distinguishes between internal and external standardisation (Ulrich, 1995). Whereas internal standardisation describes the uniformity of components within the borders of a single firm and therefore supports aforementioned in-house transparency and flexibility, external standardisation is characterised by uniform standards for components across multiple companies. The latter allows for an easier outsourcing of the manufacturing of products respectively the executing of services. Likewise, external standardisation also increases the comparability between multiple vendors. Thus, the internal and external transparency is enhanced.

Standardisation is equally important for measuring and improving the quality of services (Zeithaml, et al., 1988). This applies in the same measure for internal and external quality management, since quality parameters can be monitored and assessed in-house or by external suppliers. Customers may benefit from this on the one hand by an improved overall quality and on the other hand by an improved transparency regarding the expected service quality. This results in a better alignment of the expected and the delivered service quality.

However, an area of tension exists between the goal of standardisation and the wish for flexibility. Buckley and Ghauri summarise these concerns in the phrase “the cost advantages of standardisation vs. the revenue advantages of adaptation“ (Buckley & Ghauri, 2004). As elaborated above, standardisation abets economies of scale. But by a comprehensive introduction of standardisation the companies suspect a significant loss of flexibility regarding adaption and customisation of offers. Though the positive influences of adaptation and individualisation on demand and pricing are commonly acknowledged (Franke & Piller, 2004; Wind & Rangaswamy, 2001), companies fear major economical disadvantages by the expected loss of flexibility.

To retain flexibility despite extensive standardisation, the paradigm of modularisation is utilised in industrial and software engineering (Bastide, et al., 2006; Huang, 2000; Sametinger, 1997). Increasingly, modularisation is also transferred and applied in the domain of service engineering (Baldwin & Clark, 1997; Böttcher & Klingner, 2011). By the segregation of monolithic offers into semantically cohesive, atomic components overall complexity is made manageable. Furthermore, by the use of a component-based portfolio adaptations or new offers can be created easily by rearranging components (Uzumeri & Sanderson, 1995). In addition, modularisation improves the manageability of complex portfolios, as costly and sophisticated tasks are also limited to self-contained modules. These positive effects cover a variety of frequently performed tasks, such as the further development, update, or replacement of modules.

Since customers of Intershop often demand similar but not identical solutions, the combination of standardisation and modularisation appeared to be a suitable approach. Thus, during workshop the use of both paradigms was determined. Though Intershop had already segregated their portfolio into distinct components, a more fine-grained and detailed modularisation was required, considering also dependencies between components and following a stricter formalisation.

To reduce the efforts for modularisation and standardisation themselves, the implementation of a supportive tool was decided. The tool was planned as prototype to evaluate whether standardisation and modularisation are suitable approaches and how they can be supported software technically. A detailed description can be found in chapter 4.

Prior to the implementation two aspects needed to be defined. To enable a homogenous understanding of concepts and theory, first a consistent terminology is required. Therefore the term “service module” was defined as follows (Böttcher & Klingner, 2011):

> „A service module offers a well-defined functionality via precisely described interfaces. A service module can be used for composition and can, therefore, itself be part of a more coarse-grained service module. The composition allows a customer-specific configuration, as the customer can assemble a service offering from a given set of service modules. “

In the context of our work, the terms “module” and “component” are considered synonymously. Further possible terms are “service bundle” or “service blocks”, but which are not taken into account in this paper.

The second aspect is the requirement of a formal basis for the prototype. This is provided by the development of a metamodel. The metamodel comprises several concepts, which are components, connectors, key performance indicators (KPIs), and the definition of dependencies, both logical and temporal. Consecutively, the main concepts of the metamodel are described in more detail.

As previously defined, a **service component** respectively service module represents an offer of a well-defined functionality and, therefore, defines a distinct part of a service provision process. Service components can consist of an arbitrary number of child components which describe the corresponding service regarding its possible characteristics in more detail (decomposition). Analogously, more extensive services can be created by aggregating several service components in a more coarse-grained component (composition). These hierarchic relations can be depicted in a graph similar to the Gozinto-graph, common in industrial engineering (Vazsonyi, 1954).

To model complex service portfolios and their logical and temporal interdependencies, a more powerful capability of expression is required. Therefore, so called **connectors** were introduced. By the use of connectors the description of more complex hierarchic dependencies between components are enabled. By the definition of cardinalities extensive logical dependencies can be described, such as Component A requires either the Components B and C or the Components C and D (Figure 1). Furthermore, quantitative rules can be defined by using cardinalities in the form of *(min, max)* tuples as well, such as Component A needs at least 4 child components.

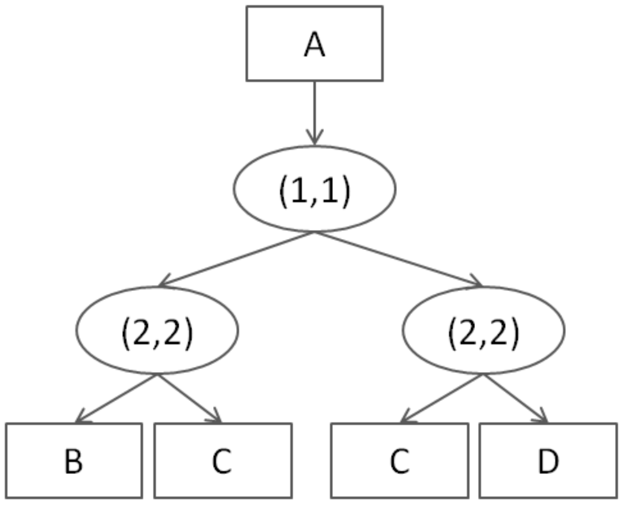

Figure 1: Defining logical, hierarchical dependencies using cardinalities

To create a valid graph, a few constraints need to be considered. First of all, the graph needs to be acyclic, since otherwise the hierarchic paradigm of the graph would be foiled. Secondly, components must not have other components as direct child nodes, but need to be connected by the use of connectors. And lastly, to preserve tree structure of the graph, a node must not have more than one parent node.

Due to the aspired software support for price calculation and productivity analysis of service portfolios respectively configurations, the widespread concept of **key performance indicators** (KPIs) was included in the metamodel as well. Each component can contain an arbitrary number of KPIs for measuring respectively describing its performance. The allocation of KPIs on a component level simplifies the complex task of measuring the productivity of services (Böttcher & Klingner, 2011), since on the one hand a service component represents a delimited problem which is easier to measure. On the other hand, components are more homogenous regarding their characteristics than complex and multi-faceted holistic service portfolios. This makes the process of selecting appropriate KPIs much easier. Supplementary, a workshop with Intershop was conducted, to identify important and at the same time simple to measure KPIs.

Each KPI consists of an identifier and a corresponding value. Values can be a single constant, an arithmetic calculation using constants, an arithmetic calculation using referenced KPIs or an arithmetic calculation using a mixture of constants and referenced KPIs. Values of referenced KPIs can be either obtained from other components within the graph or from external controlling software already in use by the company. An example comprising also the concept of KPIs shows Figure 2.

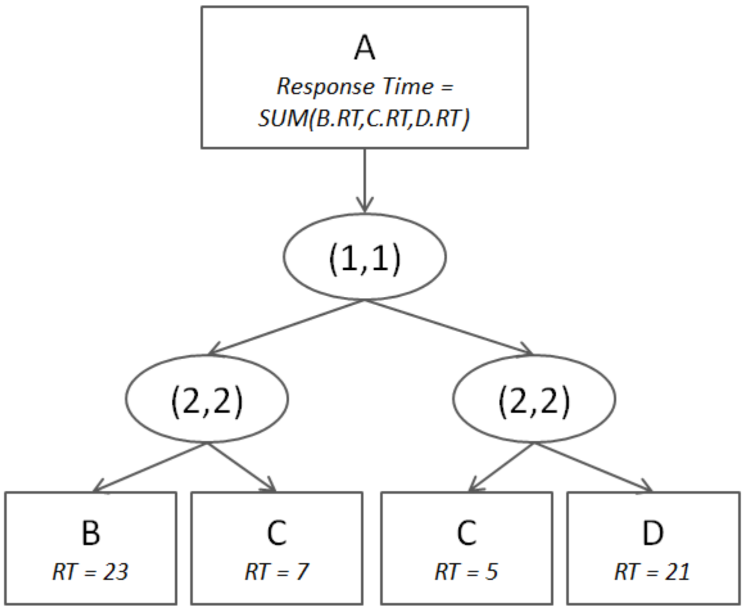

Figure 2: Model of Portfolio including cardinalities and KPIs

The aforementioned concepts constitute the basis for modelling service portfolios. For modelling and particularly configuring complex service portfolios several extensions of the metamodel might be required. Therefore, in a few subsequent workshops two more concepts were introduced. **Variables** represent numerical parameters of the environment where the services shall be provided. **Attributes** represent non-functional properties of the service components such as capacity or supported languages.

4. Prototype

To allow for a better evaluation of theoretical findings, a software prototype called "Service Modeller" was developed in parallel. During several workshops, the metamodel and the prototype were iteratively enhanced. Consecutively, the technical foundations of the tool as well as the realisation of the previously introduced concepts are described.

To provide the highest flexibility and to reduce the entry barriers best possible, the prototype is implemented as an online tool. It is based on the application framework

Microsoft Silverlight. Employing this environment, only a web browser with installed Silverlight plugin is needed to use the prototype.

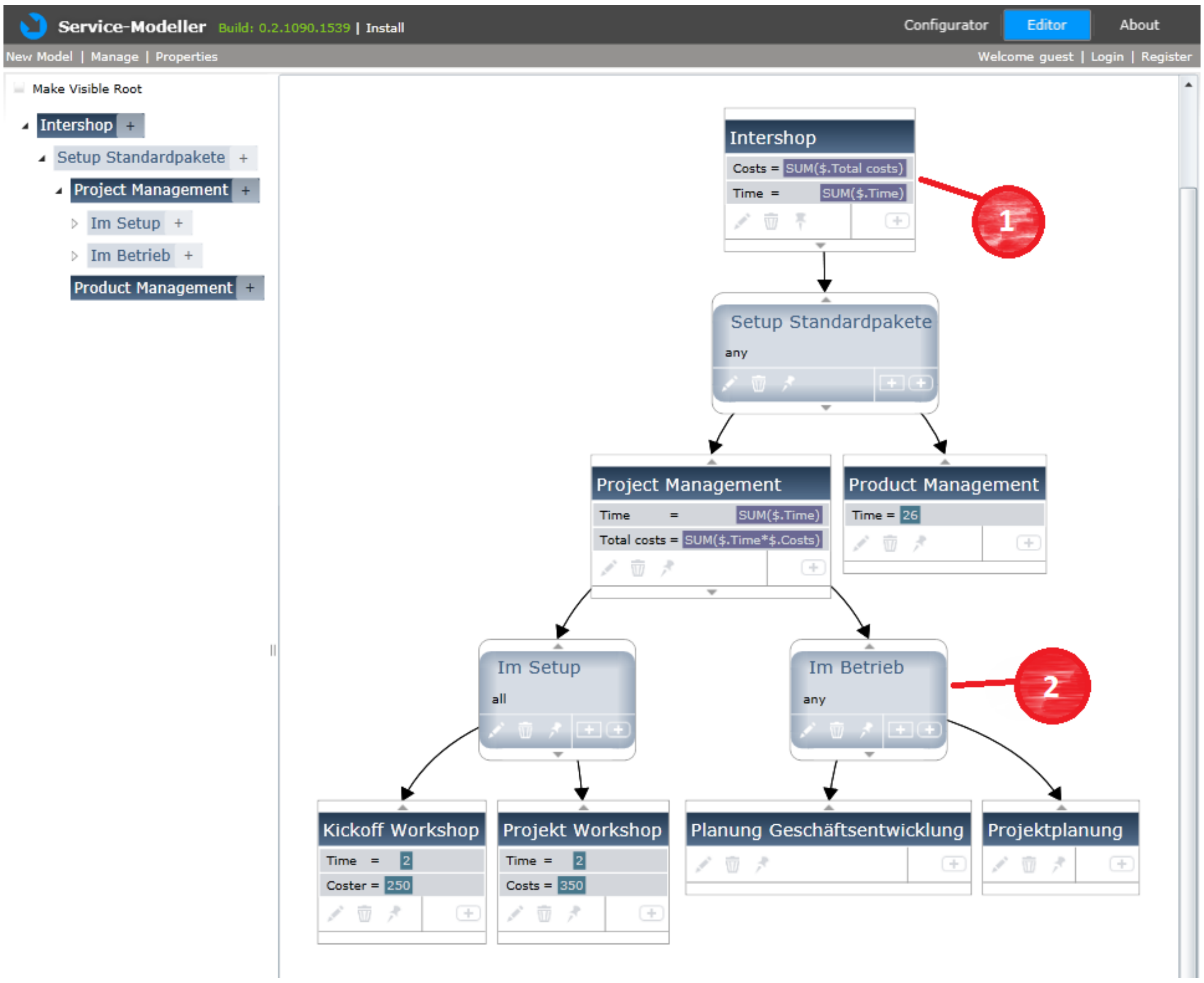

Figure 3: Service modeller prototype – Editor view

The Service Modeller is divided into two major parts. The first view, the editor, encompasses the creation, arrangement and modification of models (Figure 3). Service components (1) can be created and assembled in a tree structure. To describe hierarchical and logical dependencies between components they are linked through connectors (2). These connectors contain rules restricting valid configurations regarding the child nodes of the connector. The rules are

- ANY – choose an arbitrary number of child nodes including none,
- ALL – all child nodes have to be chosen,
- MIN – at least MIN number of child nodes have to be chosen,
- MAX – the maximum number of chosen child nodes is MAX as well as
- RANGE – choose a number of child nodes within the scope of RANGE.

By the use of these rules and the cascading of connectors where applicable, all required complex logical structures could be described.

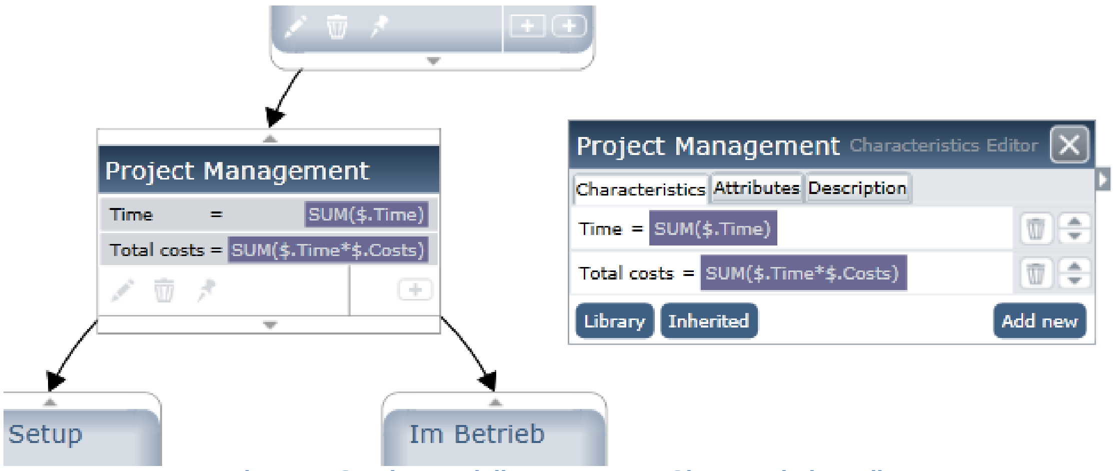

Figure 4: Service modeller prototype - Characteristics editor

Since the evaluation of the productivity of components and particularly of configurations was one requirement of Intershop, components can be augmented with KPIs. As described in the previous chapter, KPI values can be assigned or calculated. Figure 4 illustrates the different options.

1. *Add new:* A new KPI is added to the component. Its value can be either a constant or a calculation, which might also include a manually referenced KPI of a child node.

2. *Inherited:* Shows a list of KPIs of child nodes of the current component capable of being referenced. These KPIs can be sorted either by type or component and can be included in calculations for KPIs of the current component.

3. *Library:* Opens a library containing an extensive collection of common KPIs, including a short description, a source, and a formula which can be imported as initial value for the KPI.

As also be seen in Figure 4, a further tab comprises the definition of attributes for components. These attributes are fixed, not compulsory numeric values describing non-functional properties of the component. During configuration, these attributes can be used for showing only components fulfilling customer's requirements.

The third tab "description" allows the assignment of non-formal text describing functionality and further parameters of the component.

The second major part of the service modeller is the configurator view. Within this view, concrete configurations of previously built service models can be created by selecting required and desired components of the service portfolio. The process of component selection is supported by the previously modelled rules, which can be used to verify the validity of a specific configuration.

To ensure a certain degree of productiveness for configurations, previously described KPIs can be used. Thereby, the calculation of KPIs considers consequentially only KPIs of components selected during configuration.

The definition of environmental parameters and their consideration takes place during the process of configuration. After the definition of all modelled variables the corresponding values are incorporated in the calculation of KPIs.

Besides the aforementioned functionality for modelling and configuring services, the tool implements also the basic concepts for corporate tools like user administration or version management for models.

In future various extensions and enhancements are imaginable. For example, the inclusion of external KPI-values from already existing controlling software is possible. Therefore, adequate interfaces are required. Furthermore, also the planning of necessary resources can be conducted concurrent to and depending on the process of configuration of service offers. Thereby, the availability of resources can influence the pricing, scheduling, or even validity of configurations.

5. Conclusion

The extensive portfolio of services of the partner Intershop Communications AG and their wish for improved approaches and methods to manage their services illustrates the need for new approaches regarding the description and composition of services as well as management of service portfolios very distinctively.

By the development of a metamodel and a tool on that basis, both customers and vendors can benefit of various advantages. Thanks to the modular structure and the configurability, customers can easily create individualised variants of services, perfectly fitting their needs. On the other hand, service providers are put in better positions to benefit from economies of scale.

A supportive tool is a precondition to handle the complex process of creating and managing extensive portfolios. It should cover functionality aspects like the definition of modules, their logical and temporal dependencies as well as the evaluation of these models regarding productivity aspects. Thereby, the prototype described in chapter 4 outlines one possible solution for IT-support of designing, optimising, and configuring service offers.

Further research has to be conducted regarding the aspects of granularity of the described services. Particularly, question like ‘what level of granularity for modularising services is reasonable from an economic viewpoint’ are to be answered. A more fine-grained modelling allows for more configuration variants but also increases the complexity of modelling and even configuration, since an oversupply of options can overcharge potential customers.

Furthermore, the applicability of the solution proposed in this paper in other areas than the e-commerce sector has to be evaluated. The focus on the modelling of mostly service aspects could be a limitation of the developed prototype which needs to be improved or modified in more product-related industries. Likewise, the modification and extension of the underlying model can become necessary.

Literature

Baldwin, C. Y. & Clark, K. B., 1997. Managing in an age of Modularity. *Harvard Business Review*.

Bastide, G., Seriai, A. & Oussalah, M., 2006. Adaptation of Monolithic Software Components by Their Transformation into Composite Configurations Based on Refactoring. In: *Component-Based Software Engineering.* s.l.:Springer Berlin / Heidelberg.

Böttcher, M. & Klingner, S., 2011. Providing a Method for Composing Modular B2B-Services. *Journal of Business and Industrial Marketing*, 26(5), pp. 320-331.

Böttcher, M. & Klingner, S., 2011. *The Basics and Applications of Service Modeling.* [Online].

Böttcher, M., Swialkowski, R. & Fähnrich, K.-P., 2011. Produktivitätsbetrachtung bei der Komponentisierung von Dienstleistungen. In: I. Gatermann & M. Fleck, eds. *Mit Dienstleistungen die Zukunft gestalten – Impulse aus Forschung und Praxis..* s.l.:Campus Verlag, pp. 207-216.

Buckley, P. J. & Ghauri, P. N., 2004. Globalisation, economic geography and the strategy of multinational enterprises. *Journal of International Business Studies*, 35(2), pp. 81-98.

Franke, N. & Piller, F., 2004. Value Creation by Toolkits for User Innovation and Design: The Case of the Watch Market. *Journal of Product Innovation Management*, 21(6), pp. 401-415.

Huang, C.-C., 2000. Overview of modular product development. *Proc. Natl. Sci. Counc. ROC(A)*, 24(3), pp. 149-165.

Intershop, 2011. *http://www.intershop.com.* [Online]
Available at: http://www.intershop.com/investors-financial-reports.html?file=tl_files/media/downloads/en/investors/financial-reports/2010/2010-Annual-Report-ENG.pdf
[Accessed 09 08 2011].

Jiao, J., Ma, Q. & Tseng, M. M., 2003. Towards high value-added products and services: mass customization and beyond. *Technovation*, 23(10), pp. 809-821.

Sametinger, J., 1997. *Software engineering with reusable components.* s.l.:Springer-Verlag New York, Inc..

Ulrich, K., 1995. The role of product architecture in the manufacturing firm. *Research policy*, 24(3), pp. 419-440.

Uzumeri, M. & Sanderson, S., 1995. A framework for model and product family competition. *Research policy*, 24(4), pp. 583-607.

Vazsonyi, A., 1954. The Use of Mathematics in Production and Inventory Control. I. *Management Science*, 1(1), pp. 70-85.

Wind, J. & Rangaswamy, A., 2001. Customerization: The next revolution in mass customization. *Journal of Interactive Marketing*, 15(1), pp. 13-32.

Zeithaml, V. A., Berry, L. L. & Parasuraman, A., 1988. Communication and Control Processes in the Delivery of Service Quality. *The Journal of Marketing*, 52(2), pp. 35-48.

Author(s):

Stephan Klingner
University of Leipzig
Department of computer science
Johannisgasse 26
D-04103 Leipzig
klingner@informatik.uni-leipzig.de

Martin Böttcher, Dr.
University of Leipzig
Department of computer science
Johannisgasse 26
D-04103 Leipzig
boettcher@informatik.uni-leipzig.de

Michael Becker
University of Leipzig
Department of computer science
Johannisgasse 26
D-04103 Leipzig
mbecker@informatik.uni-leipzig.de

Arndt Döhler, Dr.-Ing.
Intershop Communications AG
Intershop Tower
D-07740 Jena
a.doehler@intershop.de

Frank Schumacher
University of Leipzig
Department of computer science
Johannisgasse 26
D-04103 Leipzig
fs@informatik.uni-leipzig.de

Parts of the work were funded by grants of the German Ministry of Education and Research (BMBF) in the context of the joint research project "KoProServ" (01FL09004/01FL09006).