A reference architecture for feedback-based control of service ecosystems

Robin Fischer, Ulrich Scholten
Karlsruhe Institute of Technology (KIT)
Karlsruhe Service Research Institute (KSRI)
Karlsruhe, Germany
{robin.fischer|ulrich.scholten}@kit.edu

Simone Scholten
SAP Research
Karlsruhe, Germany
simone.scholten@sap.com

Abstract—**With the emergence of digital business ecosystems, new control mechanisms are required to sustainably ensure responsiveness on dynamically evolving consumer demand, as well as goal congruence with the platform providers' strategic goals. Based on system theoretical reflections, we propose a 3-layer reference architecture that collects data on service interactions, aggregates monitoring and feedback information and thus provides data basis for the said control mechanisms.**

Digital Business Ecosystems, Service Ecosystems, Reference Architecture, Feedback Control

I. INTRODUCTION

The IT sector is undergoing a major restructuring process, where an important share of formerly transactional business designs is replaced by Software-as-a-Service (SaaS) solutions. Consumers "demand product and service customization, speed and high levels of quality of service, all in a seamless fashion and preferably from a single provider. In many instances, consumers will only use and continue using products and services, if their value preferences and criteria are met or exceeded by the services provider" [1]. To cope with these requirements without losing the focus on core competencies, companies started delegating the process of value generation into the supply chain [2], limiting themselves to a role of substantiating basic value contribution. This process cannot be reduced to simple outsourcing decisions, but has lead to renewed strategies of innovation and added value creation. Thus, companies are forced to open their business models to benefit from the value creation and the resulting pace of innovation outside the company borders [3,4]. Microsoft for instance interlinks its own services with those of eBay, Equifax, and PayPal to create a dynamic supply chain of interlinked autonomous services [5].

Through these open business models [3] companies aim at integrating own services and third-party offerings into a holistic and synergetic service-portfolio of assured quality and enhanced value proposition. A particular scalable form of open innovation is inherent in Web-based two-sided platform concepts. Examples are the media-covered success of the business application software provider Salesforce.com, Apple's app store or the social utility platform Facebook. These platform providers have to cope with an incorporated flexible network of interconnected autonomous services. The consumer's role is becoming much more central to the supply chain with an increased volatility of preference patterns. SaaS providers require new and innovative mechanisms to control their service ecosystems in response to the enforced dynamism. Software architectures that are designed to cope with such a distributed setting of service ecosystems rarely exist today. However, they are a prerequisite for the monitoring of data and the aggregation of information that is required for the execution of these new control mechanisms.

In this paper, we aim at closing this gap with the design of an architecture for customized feedback to service providers, allowing for accelerated adaption on changing market requirements that are in coherence with the platform provider's strategic goals. We focus on platform-based business ecosystems of two-sided service platforms such as Salesforce, Netsuite or Facebook or more specifically, on ways to optimize their respective service-enabling ecosystem. We begin most fundamentally by introducing our understanding of business and service ecosystems. In section 3, we review system theory to derive a control mechanism in response to the changed requirements on control in dynamic service ecosystems. Subsequently, we propose a reference software architecture that collects data on service interactions, aggregates monitoring and feedback information and thus provides data basis for the feasibility of the control mechanisms, in chapter 4. Finally, we draw conclusions and outline our next steps in implementing feedback-based control.

II. SERVICE ECOSYSTEMS

With Web services technologies and service-oriented architectures (SOA) gaining mainstream acceptance, a new revolution of service-orientation has emerged. These technologies provide the technical foundation for more dynamic and flexible ways of creating customer value in business ecosystems. In this context, digital business ecosystems represent an entirely new class of business design that empowers consumers to create their solutions and service compositions that best suit their needs on-demand [6].

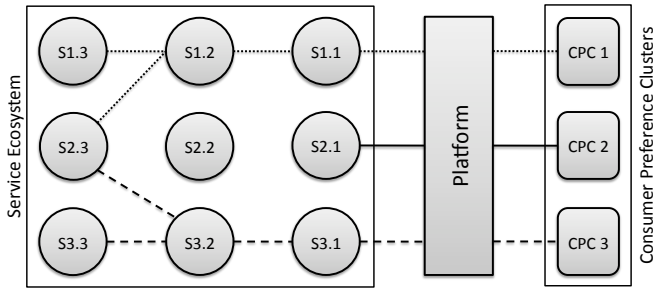

Figure 1. Principal setting of a digital business ecosystem for an SaaS platform

Business ecosystems (see fig. 1) represent an economic community, embracing the platform provider, service providers (being competing service complementors), consumers, and other stakeholders creating customer value in a co-operative manner. According to [6] "businesses should view themselves as a federation of capabilities that collaborate with other enterprises within a business ecosystem." The resulting service value networks (SVNs) can hence be interpreted as an instantiation of a business ecosystem at the time of composite service consumption [7].

Successful platform providers depend on a robust, highly productive business ecosystem of complementary third party companies to co-create the platform's overall value proposition and to support its market adoption [8]. Therein, "the performance of a firm is a function not only of its own capabilities or of its static position with respect to its competitors, customer, partners, and suppliers, but of its dynamic interactions with the ecosystem as a whole" [9]. Business ecosystems are understood as an "economic community supported by a foundation of interacting organizations and individuals [...]. This economic community produces goods and services of value to customers, who are themselves members of the ecosystem" [10]. The participants of the business ecosystem "work co-operatively and competitively to support new products, satisfy customer needs, and eventually incorporate the next round of innovations" [10]. They co-evolve their capabilities and roles over time, and tend to align themselves with the directions set by one or more central companies, the platform provider. Salesforce, for example, offers CRM-as-a-service, while external service providers complement the core service offering through related service like address verification, graphical information systems of financial service applications. These dependencies, however, evoke particular indirect network effects, determining the platform's market success and profitability: The more external companies join the value net, the more valuable the platform becomes. This dynamic causes more users to adopt the platform and more service providers to enter the ecosystem [11]. Thus, service ecosystems develop due to mutual benefits that autonomous partners gain through an interlinking of their respective investments. In doing so, they accept reciprocal dependences, which lead to tremendous implications on economic value creation and capture in distributed ownership [6].

Once having adopted a platform strategy, its long-term success depends on continuous innovation and renewal of the business ecosystem [12], embracing continuous services portfolio optimization and provision of superior customer value. This embraces are two primary risk factors with respect to a robust evolution of such an ecosystem: a) information asymmetry (as service providers lack a comprehensive market view) [7,13,14] and b) goal incongruence with the platform provider's objectives. The risk of missed goal congruence, for example, is currently visible in Apple's app store through a bias towards "games" of low technical quality [15]. The platform provider's challenge with respect to its ecosystem relates to its lack of active control mechanisms as the complementary services are in the service enabler's ownership. Thus, ensuring goal congruence means, ensuring that the goals of the partners within the service ecosystem are consistent with the goals of the platform organization itself [16].

In the following, we will discuss how key market information can be used as feedback to service and platform providers to maintain goal congruency based on concepts of system theory.

III. Introducing Feedback Control to Service Ecosystems

Paying respect to the high dynamics in value nets [17] and to the autonomy of the service providers in the service ecosystem, we make use of system theory in our pursuit of optimizing offerings of SaaS platforms (see figure 1). In the following, we will first provide a general introduction into system theory and thereafter show its application on the previously introduced service ecosystem in the context of SaaS platforms.

System Theory, as pioneered in [18-20] tries to capture complexity, system adaptivity, self-organization, or connectionism in systems. Explicitly, the term system may refer to social, biological, organizational or technical systems. In recent IT-focused research, [21] analyze and describe the interactions of independent players and the resulting macroscopic effects as multi-agent relationship. In this context, they provide profound insight into self-organization in the context of IT-based dynamic systems. According to their findings, self-organization describes "an adaptable behavior that autonomously acquires and maintains an increased order of complexity" [21]. The burden of highly co-operational complexity in value nets can thus be substituted by less intricate, but decentralized auto-adaption processes [22,23]. Complex system theory, hence, provides a starting point to describe the dynamics and service autonomy of value nets, which classical linear process models are unable to capture. In the following, we apply this theory with the intent to create an architecture for information-based optimization of service offerings in SaaS platforms and their service ecosystems.

In particular, we model value nets as controlled systems, a system-theoretical concept to apply feedback in control engineering [24]. Challenging in value nets of autonomous services is their self-organized behavior, which prevents their central or directly control or optimization. In this respect, platform and service providers are driven by different expectations (set values) as regards their performance and the quality of their service offers. Intrinsic motivation, external stakeholders or feedback from the market may influence these

expectations. In spite of this conflict of interests, a macroscopic behavior of the controlled system can reach coherence as long as value is produced to consumers and profit to is generated for platform and service providers, in a self-organized and adaptive way. As soon as the potential for the satisfaction of targets decreases or is not attainable any more, providers leave the ecosystem. Others might join the open ecosystem instead, if they identify enough economic incentives.

We suggest a feedback architecture for a continuous system optimization, addressing the stated problems of information asymmetry and the lack of goal congruence (see section II). These optimizations steps cover a whole range from purely intrinsic to rather regulative control. Whereas the small-cause-large-effect approach is best achieved through targeted feedback information for each service provider, goal congruence can only be enforced through additional restrictive, co-regulative, motivational or sanctional control, being exerted by the platform provider [25]. For that, the latter requires refined static and dynamic view of the business ecosystem.

To do so, we suggest monitoring the entire business ecosystem (see fig. 1), which is turned into the control path of the regulatory process. Monitoring includes consumption and consumer preference data as well as data on the composite service creation, i. e. the service topology. This includes service providers as well as their respective positions and relations within the service ecosystem. Since this kind of information cannot be observed directly in SVNs, we introduce an Extended Analysis (EA) component into the feedback loop. By the use of EA, we are able to record service consumption data in a vector of independent parameters that we call 'actual value' (see fig. 2). A regulator deducts the actual value from the given set value, resulting in a modified control value U (e.g., modification of a service proposition). The modified control value U is defined by the delta of optimization work that is still to be done. Based on the newly adjusted control value, a new actual value is tapped again, which is where the feedback loop closes. Since disturbances may influence the control path by a steady change in the external environment, the actual value may alter. In these cases the feedback, provided to the regulator, allows for rapid re-leveling of actual and set value.

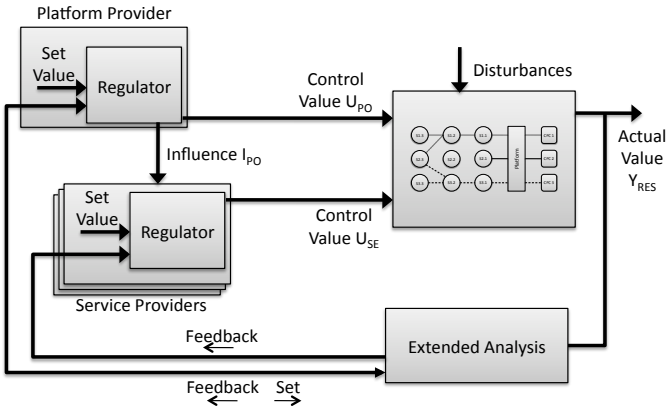

Figure 2. Conceptual Model of Feedback-based Control of Service Ecosystems

Based on received feedback of the actual performance and in alignment with the commercial goals (set value), each service provider may readjust the service proposition. While the service offering is placed within the control path, the service provider takes the role as an outside spectator, influencing the process through the modification of the respective service offering. Hence, the service ecosystem self-organizes. Service providers may modify their value proposition in one or more of the service ecosystem's services (within the control path) by internal "process transformation" [2] or through a selective replacement of their respective supplying services. This causes a "system transformation" [2].

Similarly, the platform provider is part of the loop. Platform providers differ from service providers in possessing a configuration power on the extended analysis component (see fig. 3). In some platform designs, the platform provider may also exert direct or indirect influence mechanisms (IPO) on service providers.

In summary, we identify three fundamentally different types of feedback information that has to be generated by the EA component (see fig. 3): *Preference feedback*, describing the consumers' response to offered composite services; *Structural feedback*, correlating composite services with their supplying value nets, i. e. the service ecosystem's topology; *Interpreted feedback*, giving each service and platform provider the information needed to exert their respective corrective action [7].

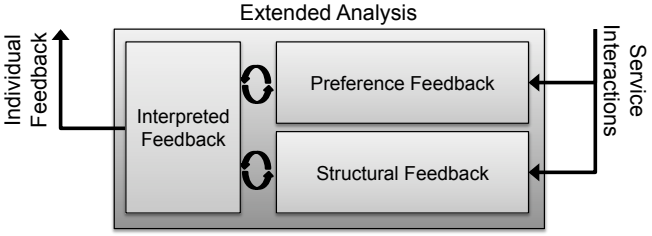

Figure 3. Types of Feedback provided by Extended Analysis Component

IV. A Reference Architecture for Optimization of Service Ecosystems

In the previous sections, concepts of service ecosystems and their respective control have been introduced. We further formulated the overall goal to capitalize on the platform provider's role as keystone in the two-sided market scenario. Through our proposed feedback-based approach, we aim at goal-congruent service evolution of platform and service providers. However, our considerations, so far, have only been of theoretical character.

From a technological point of view, though, there only exists little work that addresses software architectures to optimize service ecosystems. Most closely to our approach, [26] propose a Web Service Manager to monitor and control Web services based on service execution logs and conversation abstractions. However, this approach focuses on the management of Web services from the perspective of a focal corporation. Thus, does not consider the particularities of service ecosystem that arise around platform offerings, i. e.

digital business ecosystems. Still, we may draw technology concepts for the development of our reference architecture.

The reminder of this section, thus, contributes to this void by engineering a reference architecture that provides a framework for monitoring service interactions, analyzing service interaction data, aggregating relevant feedback information, and finally providing customized feedback information to each service provider. The architecture is based on experiments which have been conducted in the frame of the Theseus / Texo project [27].

A. Requirements Enginering

In our effort to develop a reference architecture for SVN optimization, we followed the standard requirements engineering process, i. e. iterated requirements elicitation, analysis, specification and validation [28]. We used longitudinal studies [29] and theory, based on [7,25] such as discussed in section II as major source for requirements elicitation. We will further use experimental prototyping to verify the identified requirements. We used requirements allocation to identify requirements of lower level software components in the proposed reference architecture. In conclusion, we identified the following set of requirements:

1) Service-oriented design (SoD): Service ecosystems are built upon the service-oriented paradigm, correspondingly service-orientation must govern the overall architecture design. In respect of this frameset, the required interoperabilty between platform providers, service providers and feedback provider can be guaranteed. Next, SoD allows for loosely coupling feedback components, which serves the independent development of separate components, i.e. feedback components. Finally, the SoD allows to bind external cloud services (e.g., storage, messaging or database services) to achieve lightweight and scalable implementations.

2) Multi-tenancy: Feedback information is derived from raw data (service interactions) that comprises information of the SVN overall performance. Each service provider, however, must only be granted access to targeted feedback on statistical and quantitative information, relevant for the respective performance. On the other hand, the platform provider requires specific information, enabling the assertion of corrective actions [25]. A multi-tenancy approach pays respect to all involved player, as it is "both robust and secure enough to satisfy tenants […] who are concerned about surrendering control of vital business data to a third party, while also being efficient and cost-effective to administer and maintain" [30].

3) Platform-oriented design (PoD): Like service providers complement a platform provider's service portfolio, SVN optimization tools should allow for customized extensions. Our reference architecture, therefore, is designed to expose base value that can be complemented by higher level feedback providers, i. e. applies SoD not only for external interoperability, but also for future extensibility.

B. Reference Architecture Overview

Our reference architecture for SVN optimization is built upon three logically separated layers of services. Following the above-described PoD requirement, so-called platform core services form the foundation for higher-level service management services and complex feedback services. Each lower-level layer describes its service offering in an interface (see "Service Management Interface" and "Platform Core Service Interface" in fig. 4) that can be accessed from higher-level registered services complementing the platform offering (see "Add-on" in fig. 4) and from within the same layer. Feedback services may access service management and platform core services directly. The so-called "Feedback Interface" is the access point for platform and service providers to either request feedback information, register for feedback, adjust feedback parameters or subscribe for service monitoring. The so-called Client Apps layer (see fig. 4) illustrates our concept of integrating the feedback information from a service or platforms provider's perspective by means of service invocation. We furthermore aim at providing an independent graphical client that provides feedback information in an illustrated way (see GUI client in fig. 4).

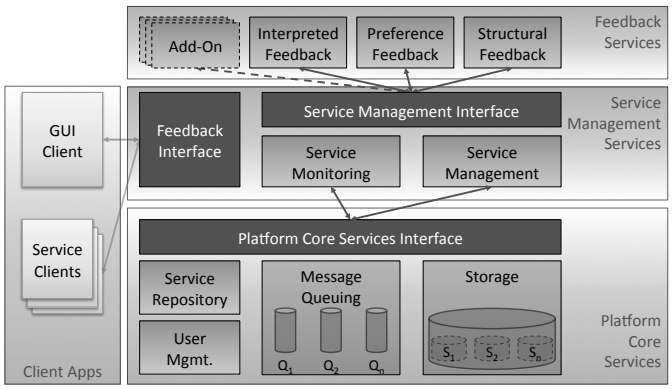

Figure 4. Reference Architecture for Feedback-based Control of Service Ecosystems

Platform core services comprise a storage service, a message queuing service, a service repository, and a user management component. The latter caters for multi-tenancy, while the service repository is required for service providers to register their services for optimization. Only if a service is registered with the SVN optimization tool, service interactions will be monitored and feedback information will be generated. This empowers service providers to independently affirm service monitoring. Next, the message queuing component provides channels for asynchronous communication (e.g., triggering of other services) between service components of all layers (e.g., service registry may trigger service monitoring). Therefore, incoming queues can be dynamically registered for each service (e.g., add-ons may communicate asynchronously with peer services). Most important among the platform core services is the multi-purpose storage service. Depending on a service component's storage requirements the storage service may provide block-storage (e.g., Amazon's S3), relational database functionality or highly write optimized storage (e.g., [31]). In summary, platform core services free higher-level services from taking care of infrastructure and administrative features and in consequence contributes to a design of lightweight service components.

Service management services form the middle layer of the reference architecture. These services are characterized by their aim to provide functionality to manage and monitor service performance at the time of service consumption in service ecosystems. Essential to our concept of feedback-control is the service monitoring component. The dynamics and the distributed setting of service ecosystems, demand ways to track and eventually monitor individual service interactions. We are currently performing experiments on centralized and decentralized approaches of monitoring service interactions based on SOAP proxies and interceptors. The service monitoring component makes extensive use of the storage service as the number of messages to be tracked will be high, and will even rise with the progression of successful service ecosystems. The service management component, on the other hand, is designed to supply service providers with basic service management functionality during service consumption. By the use of this component, service and platform providers may define thresholds for performance indicators (e.g., response time or average service request per day). In doing so, the SVN optimization component may notify service or platform providers, if threshold are exceeded. In a more advanced example, platform operators may be able to define and monitor service level agreements over multiple layers of composite service execution.

Feedback services provide the information that is crucial for our concept of feedback-based control of service ecosystems. According to the previously identified three major types of feedback, i.e. preference, structural and interpreted, we propose one service component for each of the feedback types. Both, the preference and the structural feedback component consume platform core services to retrieve service monitoring data and store aggregated feedback information such as consumer preference clusters and SVN topology information. Interpreted feedback services access this feedback information through the platform core service interface, perform interpretation steps, and store interpreted information. Thus, each feedback service poses individual requirements on the storage service, which again justifies our concept of a multi-purpose storage service. Invocation of feedback services is based on events such as user request, scheduled task, or other platform services. Finally, we anticipate additional feedback services such as third-party performance monitoring or feedback components that can be integrated as add-ons.

V. SUMMARY & FUTURE WORK

In this work, we introduced a reference architecture, which collects data on service interactions, aggregates monitoring data, and provides feedback information in service ecosystems. This information is automatically interpreted and customized for each service provider, and founds the basis for service optimization at the service provider's side. Our reference architecture is further adapted to the platform provider's needs to purposefully exert corrective measures in the pursuit of goal congruence of the ecosystem's evolution. The architectural design is based on a feedback loop and motivated with system theory. It provides a scalable, service-oriented design and is structured in a multi-tenancy approach. By implementing this architecture, we expect to considerably enhance service quality in service ecosystems. Furthermore, the architecture supplies platform providers with means to align the ecosystem to their strategic goals.

In a next step, we aim at optimizing our feedback interpretation. Through experimental data, generated with our architecture from higher quantities of users, we aim at better understanding consumer preference clusters, their evolution over time, and their specific implication per service and platform provider. We will furthermore focus on evaluating algorithms such as proposed in [32] to complete our feedback information by means of value network topologies. Thus, we aim at providing a tool that discovers process and systems transitions in service ecosystems over time. In recent related work, we are furthermore going to examine the role of trust in controlling service ecosystems both from a conceptual and a technological viewpoint.

VI. DISCLAIMER

The project was funded by means of the German Federal Ministry of Economy and Technology under the promotional reference 01MQ07012. The authors take the responsibility for the contents.

REFERENCES

[1] R. C. Basole and W. B. Rouse, "Complexity of Service Value Networks: Conceptualization and Empirical Investigation", IBM System Journal, vol. 47, issue 1, 2008, pp. 53-70.

[2] B. Bernet, "Technologie an der Schwelle des 21. Jh.", In C. Belz, and T. Bieger (Eds.), "Dienstleistungskompetenz und innovative Geschäfts-modelle", Thexis, St. Gallen, 2000, pp. 36-51.

[3] H. W. Chesbrough, "Why companies should have Open Business Models", MIT Sloan Management Review, vol. 48, issue 2, 2007, pp. 22-28.

[4] O. Gassmann, "Opening Up the Innovation Process, Towards an Agenda", R&D Management, vol. 36, issue 3, 2006, pp. 223-228.

[5] Microsoft, "News Press Release: Microsoft Announces Integrated Services with Industry Leaders and New Features for the Next Version of Microsoft Office Small Business Accounting", Redmond, May 2006, http://www.microsoft.com/presspass/press/2006/may06/05-24SBANewFeaturesPR.mspx.

[6] I. Cherbakov, G. Galambos, R. Harishankar, S. Kalyana, G. Rackham, "Impact of Service Orientation at Business Level". IBM Systems Journal, vol. 44, issue 4, 2005, pp. 653-668.

[7] R. Fischer, U. Scholten, S. Scholten, S. Tai, (2009): Information-based Control of Service-enabling Ecosystems. Proceedings of the 2nd Int. Workshop on Enabling Service Business Ecosystems, Athens, 2009, pp. 1-14.

[8] A. Gawer and M. A. Cusumano, "Platform Leadership: How Intel, Microsoft, and Cisco Drive Industry Innovation", Harvard Business School, Boston, 2002.

[9] M. Iansiti and R. Levien, "The New Operational Dynamics of Business Ecosystems: Implications for Policy, Operations and Technology Strategy", Working Paper, Harward Business School, Cambridge, 2002.

[10] J. F. Moore, "Predators and Prey: A New Ecology of Competition", Harvard Business Review, vol. 71, May-June, 1993, pp. 75-86.

[11] M. A. Cusumano, "The Changing Software Business: Moving from Products to Services", Proceedings of the Sixteenth Int. Conf. on Information Systems Development, Springer, New York, 2007, pp. 677-686.

[12] M. H. Meyer and A. P. Lehnerd, "The Power of Product Platforms: Building Value and Cost Leadership", Free Press, New York, 1997.

[13] O. E. Williamson, "The modern corporation". Journal of Economic Literature, vol. 19, issue 4, 1981, pp. 1537-1568.

[14] A. Liening, "Komplexe Systeme zwischen Ordnung und Chaos", LIT, Münster, 1998.

[15] A. Locker, “Der Fluch der Masse – App-Store: Erfolgsmodell mit Tücken”, 2009, http://www.heute.de/ZDFheute/inhalt/4/0,3672, 7927972,00.html.

[16] R Anthony and V Govindarajan, “Management Control Systems”, McGraw Hill, Boston, 2007.

[17] D. Bovet, J. Martha, “Value Nets: Breaking the Supply Chain to Unlock Hidden Profits”, Wiley, New York, 2000.

[18] N. Wiener, “Cybernetics, or Control and Communication in the Animal and the Machine”, MIT Press, Cambridge, 1948.

[19] N. Luhmann, “Systemtheorie, Evolutionstheorie und Kommunikations-theorie”, Soziologische Aufklärung, vol. 2, 1975, pp. 193-203.

[20] I. Prigogine, G. Nicolis, “Self-Organization in Nonequilibrium Systems”, Wiley, New York, 1975.

[21] T. De Wolf, T. Holvoet, “Decentralized Coordination Mechanisms as Design Patterns for Self-Organizing Emergent Applications”. Proceedings of the 4th Int. Workshop on Engineering Self-Organizing Applications, Hakodate, 2006, pp. 40-61.

[22] P. Goos, “Strategisches Innovationsmanagement in fokalen Unternehmensnetzwerken”, Eul Verlag, Lohmar-Köln, 2006.

[23] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, Q. Z. Sheng, “Quality driven Web services Composition”, Proceedings of the 12th Int. Conf. on World Wide Web, New York, 2003, pp. 411-421.

[24] O. Föllinger, “Regelungstechnik – Einführung in die Methoden und ihre Anwendungen”, Hüthig-Verlag, Heidelberg, 1990.

[25] S. Scholten, U. Scholten, R. Fischer, “Composite Solutions for Consumer-Driven Supply Chains: How to Control the Service-enabling Ecosystem?” Proceedings of the 3rd academic symposium on Supply Management, Würzburg, Gabler-Verlag, 2010, in-press.

[26] F. Casati, E. Shan, U. Dayal, M.-C. Shan, “Business-oriented management of Web services”, Communications of the ACM, vol 46, issue, 2003, pp. 55-60.

[27] Theseus Press Office: “The THESEUS-programme“, 2010. http://theseus-programm.de/.

[28] A. Abran, J. W. Moore, P. Bourque, and R. Dupuis, SWEBOK: Guide to the Software Engineering Body of Knowledge. Los Alamitos, IEEE Computer Society, 2004.

[29] U. Scholten, R. Fischer, C. Zirpins, “Perspectives for Web Service Intermediaries: How Influence on Quality Makes the Difference”, Proceedings of the 10th Int. Conf. on Electronic Commerce and Web Technologies, LNCS 5692, Linz, 2009, pp. 145-156.

[30] F. Chong, G. Carraro, R. Wolter: Multi-Tenant Data Architecture, Building Distributed Applications, Microsoft Corporation, Redmond, 2006, http://msdn.microsoft.com/en-us/library/aa479086.aspx.

[31] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, Dynamo: amazon's highly available key-value store. SIGOPS Oper. Syst., vol. 41, issue 6, 2007, pp. 205-220.

[32] S. Basu, F. Casati, F. Daniel, “Toward Web Service Dependency Discovery for SOA Management.” IEEE International Conference on Services Computing, 2008. SCC'08, Honolulu, 2008, pp. 422-429.