Towards the Future Internet
G. Tselentis et al. (Eds.)
IOS Press, 2009

doi:10.3233/978-1-60750-007-0-263

S-Cube: Addressing Multidisciplinary Research Challenges for the Internet of Services*

Elisabetta Di Nitto[1], Dimka Karastoyanova[2], Andreas Metzger[3], Michael Parkin[4], Marco Pistore[5], Klaus Pohl[6], Fabrizio Silvestri[7], Willem-Jan Van den Heuvel[8]

[1] Politecnico di Milano, 20133 Milano, Italy, dinitto@elet.polimi.it
[2] University of Stuttgart, 70569 Stuttgart, Germany, karastoyanova@iaas.uni-stuttgart.de
[3] University of Duisburg-Essen, 45117 Essen, Germany, andreas.metzger@sse.uni-due.de
[4] Universiteit van Tilburg, 5037 AB Tilburg, Netherlands, m.s.parkin@uvt.nl
[5] Fondazione Bruno Kessler, 38100 Trento, Italy, pistore@fbk.eu
[6] University of Duisburg-Essen, 45117 Essen, Germany, klaus.pohl@sse.uni-due.de
[7] Consiglio Nazionale delle Ricerche, 00185 Roma, Italy, fabrizio.silvestri@isti.cnr.it
[8] Universiteit van Tilburg, 5037 AB Tilburg, Netherlands, W.J.A.M.vdnHeuvel@uvt.nl

Abstract. The Service Oriented Architecture (SOA) is increasingly adopted by industry as a paradigm for building distributed software applications. Yet, the SOA has currently several serious limitations and many crucial service issues are not addressed, including, for example, how to establish, monitor and enforce quality in an end-to-end fashion, as well as how to build service-based applications that proactively adapt to dynamically changing requirements and context conditions. This paper provides an overview of the service research challenges identified in S-Cube, the European Network of Excellence on Software Services and Systems. S-Cube strives to address those challenges by bringing together researchers from leading research institutions across diverse disciplines. The S-Cube researchers are joining their competences to develop foundations and theories, as well as novel mechanisms, techniques and methods for service-based applications, thereby enabling the future Internet of Services.

Keywords: Service-based Applications, Service Oriented Architecture, Engineering, Design, Adaptation, Monitoring, Quality

1 Motivation

Software services are self-contained, platform-agnostic computational elements, which can be flexibly and dynamically composed to create complex service-based applications. The functionality provided by a service ranges from answering simple requests to executing sophisticated processes requiring peer-to-peer relationships between multiple service consumers and providers. For the service consumer, a software service represents functionality that can be invoked through the service interface. The actual software that implements this functionality is executed, maintained

* The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement 215483 (S-Cube).

and owned by the service provider. Thus, software services take the concept of ownership to the extreme: Not only the development, quality assurance, and maintenance of the software is under the control of third parties, but the software can even be executed and managed by third parties [2].

The Service Oriented Architecture (SOA) is increasingly adopted by industry as a paradigm for building distributed service-based applications [3][6][7]. According to IT analyst Forrester Research, 67% of the largest enterprises were using SOA-based implementations by the end of 2006 and nearly 70% of those indicated that they intended to increase their use of it [1]. These facts make services technology of paramount importance to the European software and telecommunications industry.

Currently, the common practice for developing service-based applications (SBAs) following the SOA paradigm distinguishes between three functional layers [2]:

- ***Service infrastructure:*** This layer supports describing, publishing and discovering services and provides the run-time environment for the execution of SBAs. It provides core functionalities for service communication (e.g., SOAP), service description (e.g., WSDL), as well as capabilities for service discovery (e.g., UDDI).
- ***Service composition and coordination:*** This layer supports the aggregation of multiple (individual) services into service compositions (e.g., using BPEL). Service compositions can in turn be offered to service clients, used in further service compositions and eventually be composed to service-based applications.
- ***Business process management (BPM):*** This layer provides end-to-end visibility and control over all parts of a long-lived, multi-step business process that spans multiple organizations and can involve human actors. BPM provides mechanisms for expressing, understanding, representing and managing an organization in terms of a collection of business processes realized in a service-oriented fashion.

When setting out to build innovative software services and service-based applications of the future, relying on the current layers of the SOA will not suffice. In this paper we elaborate on the issues that are still unsolved and outline the importance of interdisciplinary research to address them. Consequently, this paper provides an overview of the key challenges in Section 2. Then, Section 3 motivates the need for interdisciplinary research, and how S-Cube – the European Network of Excellence on Software, Services and Systems – addresses this need. Section 4 introduces and illustrates the S-Cube research framework. Section 5 concludes the paper.

2 Research Challenges for the Internet of Services

As has been observed in [11][10], many important challenges for building future service-based applications are still to be resolved. For the key areas shown in Figure 1, those challenges are summarized below.

2.1 Engineering and Design

Designing service-based applications shows some peculiarities. Such applications are built by composing services which may be already built and running when the application is deployed. This enables a bottom-up development approach based on the idea of searching for services and building around those that are identified as suitable.

Research is thus required in high-level declarative language concepts for the specification of services that allow lay and experienced users and other stakeholders to express their views and requests in terms of "what is needed" rather than "how it is needed". One direction which could be followed is expressing the requests of the stakeholders at the intentional level, i.e., as high-level business requirements or goals.

Services that compose a service-based application may not be under the control of the organization that is operating the application. This results in the need for defining proper service contracts (such as SLAs) and quality assurance techniques (cf. Section 2.3). Additionally, it must be planned for runtime self-adaptation of the application (cf. Section 2.2) in the case component services become unresponsive or show behaviours that are not acceptable for the application. In general, we say that service-based applications need to be designed to be ready for adaptation. In fact, adaptation cannot be completely specified in advance due to the incomplete knowledge about the interacting parties as well as the application context. In the literature, there are some approaches focusing on design for adaptation. However, a coherent approach that integrates the initiatives of various areas (requirements, design, testing, deployment and operation) to create a proper life-cycle is still to come.

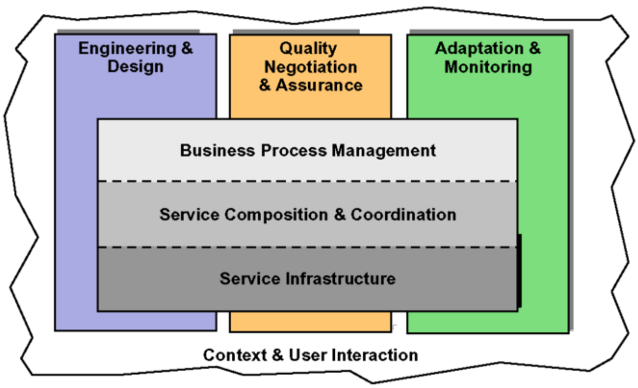

Fig. 1. Areas relevant for service-based applications

2.2 Adaptation and Monitoring

SBAs run in dynamic business environments and address constantly evolving requirements. These applications should hence be able to adequately identify, and react to various changes in the business requirements and application context. These challenges make monitoring and adaptation key elements of modern SBA functionality.

With respect to adaptation and monitoring, the state-of-the-art approaches (e.g., see [8]) need to be extended in several directions. Firstly, a broader perspective is needed on *what*, *how*, and *when* we may monitor and adapt to accommodate to changes and deviations. For instance, a deviation detected through run-time monitoring of a single specific execution of a SBA may trigger an adaptation that can be achieved through the (automated or semi-automated) evolution of the whole application. Furthermore, adaptation decisions, applied for instance by the user or maintainer of the SBA, can be learnt and transformed into adaptation patterns which can then be exploited to simplify and drive these decisions in the future.

Secondly, the definition of monitoring and adaptation itself must be extended: To this end, *monitoring* should subsume all the techniques and tools that allow for identifying, detecting, and even predicting critical events and situations (cf. Section 2.3). In this way, for instance, online testing techniques can be exploited as monitoring tools if they are used for predicting possible execution problems [5]. The same holds for *adaptation*: all the facilities for modifying the application regardless the timing and the effect count for the adaptation problem.

Thirdly, various research disciplines, different application domains (ranging from B2B to user-centric systems), as well as different functional SBA layers need to be considered during adaptation and monitoring. On the one hand, this allows reusing ideas and approaches from existing fields; e.g., using data and process mining techniques for post-mortem business process monitoring in order to gather information about SBA evolution relevant for the adaptation of the latter. On the other hand, only such an integration makes the cross-layer adaptation and monitoring possible in the first place, providing ways to reveal and accommodate to the changes in those elements of the SBA architecture that have an impact on the other layers. Indeed, this is not possible in the current SOA approaches, where the monitoring and adaptation facilities at different layers are considered in isolation.

2.3 Quality Negotiation and Assurance

To provide the desired end-to-end quality of globally distributed service-based applications, the dynamic agreement and assurance of quality becomes a key issue. This requires that not only quality aspects are negotiated and agreed, but also that those are checked during run-time. In a service-based application, different kinds of quality attributes are important [12]: Quality of Service (QoS; e.g., performance, availability), Quality of Experience (QoE; e.g, usability and trust), Quality of Business (QoBiz; e.g., revenue, profit), and Quality of Information (QoI; e.g., accuracy, completeness, relevancy). There is thus a strong need for methods that address quality attributes in a comprehensive and cross-cutting fashion across all layers of a service-based application. Specifically, end-to-end quality provision implies that the dependency between different kinds of quality attributes must be understood. For instance, the interrelation between the fulfilment of different QoI attributes on the infrastructure layer, the satisfaction of QoE on the service composition layer and the achievement of business value (QoBiz) at the BPM layer (cf. Section 2.4) is an open issue.

Further, to address dynamic adaptations of service-based applications, a growing need for automating the negotiation of quality attributes (e.g., stipulated by SLAs) can be observed. However, this issue requires considering user interaction and experience issues that may impact on the negotiation itself. This aspect calls for a multidisciplinary effort in which technology researchers will interact with researchers addressing user interaction issues.

Given the change of the life-cycle for service-based applications (cf. Section 2.1), quality assurance techniques that can be applied at run-time become essential. Therefore, standard and consolidated "off-line" software quality assurance techniques (like testing and analysis) need to be extended to be applicable while the application operates ("online techniques").

Finally, to support the vision of pro-active adaptation (cf. Section 2.2), novel quality prediction techniques need to be devised. Depending on the kind of quality attribute to be predicted, these can range from ones that built on traditional techniques to ones that exploit modern technologies of the Future Internet. As an example for the first case, "correctness" or "performance" (QoS) could be predicted by building on techniques similar to online testing [5] or run-time model analysis [4]. As an example for the latter case, "usability" of services (QoE) could be predicted by extending existing principles of reputation systems.

2.4 Business Process Management (BPM)

Business Process Management (BPM) is the activity associated with modelling, designing, deploying, monitoring and managing information technology aligned to meet the goals of an organisation and its customers [9]. BPM provides entire life-cycle management for multiple business processes that together contribute to the success of a business. Thus, from the BPM perspective of the service network described above, there is a need to define the activities that achieve business goals like lowering costs whilst increasing market share, profits and customer satisfaction.

Currently, there is a gap between business strategy, BPM and business models and their implementation in SBAs. Therefore, the objective of the BPM research area in S-Cube will be to develop fundamental new concepts in service engineering that can drive service implementation, configuration, operation and management from business models and their goals. This requires investigation into, for example, new process languages to enable the reuse of existing service compositions, choreographies, communication and service interaction patterns, the mechanisms of business transactions, collaboration and decision-making within service networks and the verification and demonstration of the compatibility of business process orchestration with respect to compliance with regulation.

As shown in Figure 1, BPM sits above the service composition and co-ordination layer (cf. Section 2.5) that provides functions exposed as services for use in business processes. Thus, integral to this research will be the investigation of how unanticipated changes in the service composition and co-ordination will be dealt in an agile, automated and transparent manner with 'new generation' BPM that provides business activity monitoring (BAM) through the measurement of KPIs and business critical events (cf. Section 2.2).

In summary, and to paraphrase [12], BPM is a natural complement to the techniques of service composition and co-ordination and a mechanism through which an organisation or business can apply and utilize service networks to achieve business goals. S-Cube plans to bring the often-fragmented research of these two areas together through the investigation of mechanisms and models that correlate KPIs with SLAs and business processes (cf. Section 2.3).

2.5 Service Composition and Coordination

Current research results in the field of service composition are designed in isolation from the BPM layer and the service infrastructure layer. While such an approach reduces complexity by abstracting away details from these layers, it does not sufficiently tackle all problems that need to be addressed in a concrete application domain. Therefore, we observe a gap between the requirements of the BPM layer (cf. Section 2.4) and the service compositions that implement them, in particular with respect to the key performance indicators specified on the BPM layer; i.e., the service compositions are not designed such that they can guarantee the desired KPI values.

Additionally, the KPIs on the BPM layer may evolve over time, which needs to be propagated to the service composition layer. Due to the separation of research, this adaptation on the BPM layer currently cannot be propagated to the service compositions, and moreover, the service compositions cannot adapt themselves to meet the modified requirements from the BPM layer. The service compositions require additional support from the service infrastructure, in particular in terms of discovery and selection of services complying to the overall quality attributes of the service composition and not only with the quality requirements of individual tasks. Therefore, we identify the need for the creation of service and service composition models involving quality characteristics and behavioral features. These models will reflect the inherent relationship among the BPM layer and the service compositions.

Based on the models and languages for service compositions, mechanism for service composition adaptation are needed, which are driven by quality attributes and by the requirements of the BPM layer and which are influenced by the service infrastructure. Such mechanisms will inevitably influence the service composition models, i.e., the mechanisms will be supported by the models for service compositions and enabled by corresponding language elements. The mechanism will enable adaptation mechanisms (cf. Section 2.2) which will be identified as necessary for SBAs and which will depend on the technology used to implement the service composition models. For example, for process-based compositions, such adaptations may be realized in terms of control flow changes (i.e., deletion of tasks, inclusion of tasks, etc.).

To enable the monitoring of service compositions (cf. Section 2.2), an event model for event notifications is expected to provide information related to the execution status of individual tasks and about the quality attributes.

2.6 Service Infrastructure

Service infrastructures will need to be scalable and of high performance in order to support for the execution of future service-based applications. Traditional infrastructures have been thought, mainly, to support enterprise applications. This idea has to be extended in order to support the execution of large-scale multi-enterprise service-based applications, which form complex service networks (cf. Section 2.4). This will require the effort of diverse communities like: high performance computing, grid computing, service oriented computing, cloud computing, etc.

In particular from grid computing, the main contributions are expected in the area of self-* infrastructures. Self-* includes self-healing, self-optimizing, and self-protecting [8]. Those self-* properties, in fact, have to be enforced both at a local and a global level. At a local level, self-* capabilities allow services to react to sudden changes in the infrastructure state. At a global level, self-* mechanisms trigger changes that will be propagated to the application.

Future infrastructures have to support effective and efficient service discovery through service registries, which could exploit novel mechanisms of peer to peer architectures. Those have shown – in other contexts (e.g., file sharing systems) – to be a good choice in case of highly dynamic environments. Also, historical information about how services have performed (cf. Section 2.3) could be used to improve the effectiveness of service registries. In response to a query for a service, QoE factors can be taken into account to select the (set of) best service(s) to propose for being included in the application.

Furthermore, novel SOA infrastructures should be designed to include services that are offered through the Internet via Web 2.0.

3 Multi-disciplinary Research in S-Cube

Section 2 has highlighted that many service research activities are fragmented and, as a result, each research community concentrates mostly on its own specific research techniques, mechanisms and methodologies. Thus, the proposed solutions are not aligned with or influenced by activities in related research fields.

In order to address the challenges introduced above, a holistic view and approach to services research is thus required. To this end, S-Cube, the European Network of Excellence on Software and Services (www.s-cube-network.eu), aims to establish a unified, multidisciplinary, vibrant research community. S-Cube is funded for a period of four years by the European Community's 7th Framework Programme. In S-Cube, over 70 researchers and over 50 Ph.D. students from 16 institutions, pursue the following objectives:

- Defining a broader research vision and perspective to shape the software-service based Internet of the future.
- Re-aligning, re-shaping and integrating research agendas of key European players from diverse research communities to achieve a long-lasting foundation for steering research and for achieving innovation at the highest level.
- Inaugurating a Europe-wide program of education and training for researchers and industry to create a common culture and impact on the future of the field.
- Establishing a proactive mobility plan to enable cross-fertilisation between research communities.
- Establishing trust relationships with industry (e.g., via NESSI) to strengthen Europe's industrial competitiveness.

To reach the above objectives, S-Cube members jointly carry out the following activities:

- ***Integration Activities:*** Integration activities tackle fragmentation and isolation of research by different means: (1) The *S-Cube Knowledge Model* will capture terminology and competences of S-Cube members and their research, thereby enabling understanding and eliminating the duplication of research efforts. (2) The *Distributed Service Laboratory* will be established as a research infrastructure to provide access to state-of-the-art collaboration facilities. (3) S-Cube's *program of* *education and training*, together with the *mobility programme*, will lead to cross-fertilisation of knowledge and durable research integration.

- ***Joint Research Activities:*** Work in S-Cube will be guided by the S-Cube research framework, which will be introduced in Section 4.
- ***Spreading of Excellence Activity*:** This activity will ensure a broad dissemination of research results and includes the organisation of international conferences, specialised workshops and summer schools, as well as a European Ph.D. programme.

4 The S-Cube Research Framework

The S-Cube research framework (see Figure 2) guides the joint research activities of S-Cube. In general, the framework distinguishes between principles and methods for engineering service-based applications and the technologies (or mechanisms) which are used to realize those applications. Principles and methods address cross-cutting issues like adaptation and monitoring, as well as quality definition, negotiation, and assurance. Technologies support specific requirements of the individual layers and provide capabilities to the cross-cutting principles and methods.

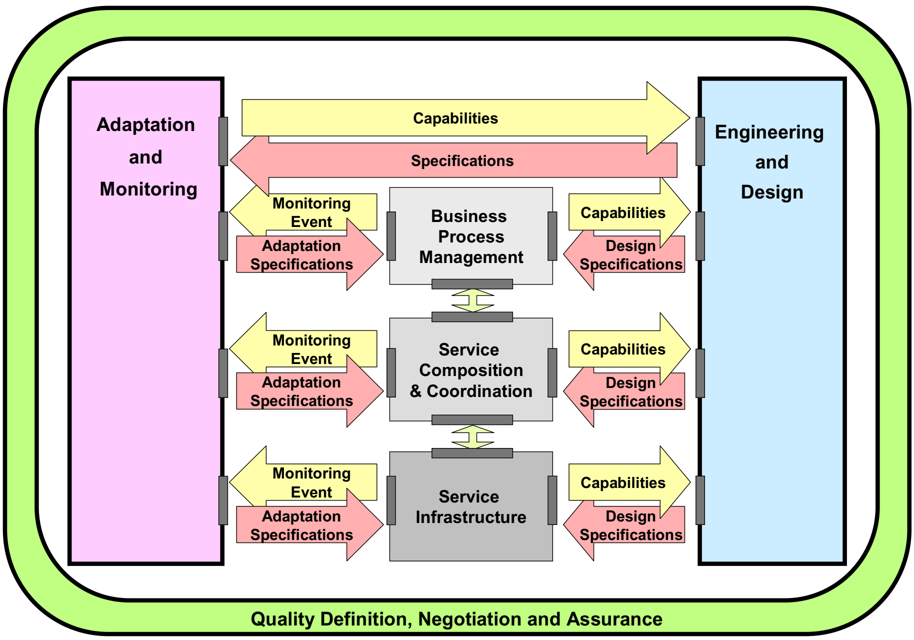

Fig. 2. The S-Cube research framework

What makes the S-Cube research framework unique when compared to the traditional "layered" way of developing service-based applications (see Section 1) is that the framework systematically addresses cross-cutting service issues. Further, the framework sets out to make the knowledge of the functional layers (which currently is mostly hidden in languages, standards, etc.) explicit in order to avoid overlaps and to identify gaps in research. Finally, the framework is designed to handle the complexity of developing and adapting service-based applications.

To this end, the elements of the S-Cube research framework are defined by following a clear separation of two concerns:

Concern 1: Technologies and local principles & methods: The three horizontal layers of the framework are, similar to the traditional SOA layers, responsible for techniques and methods which are applicable locally within the layers. Also, concrete service technologies fall under the responsibility of the layers.

- The ***service infrastructure*** layer provides a high-performance execution platform supporting adaptation and monitoring of SBAs. The platform provides a set of core services, like search engines and virtualisation services to the other layers.
- The ***service composition and coordination*** layer focuses on novel service composition languages and techniques. Especially it provides mechanisms to adapt and monitor service compositions.
- The ***BPM*** layer addresses modelling, designing, deploying, monitoring and managing service networks to meet the goals of an organisation and its customers through the correlation and analysis of KPIs from the service composition and coordination layer with business processes.

Concern 2: Overarching / cross-cutting principles, techniques and methods: In addition to the local principles and methods, principles and methods falling into the following key cross-cutting aspects are addressed:

- The ***engineering and design*** aspect includes all issues related to the life-cycle of services and SBAs. This includes principles and methods for identifying, designing, developing, deploying, finding, applying, provisioning, evolving, and maintaining services, while exploiting novel technologies from the functional layers. An example is exploiting future service search engines for bottom-up SBA design.
- The ***adaptation and monitoring*** aspect includes all concerns with respect to the self-adaptation behaviour of a SBA. Specifically, this comprises languages and methods for defining adaptation goals and different adaptation strategies, which are triggered by monitoring events. An example for an adaptation technique that falls into the responsibility of this aspect is a strategy that correlates the monitoring events across the functional layers, thereby avoiding conflicting adaptations.
- The ***quality definition, negotiation and assurance*** aspect involves principles and methods for defining, negotiating and ensuring quality attributes and SLAs. Negotiating quality attributes requires understanding and aggregating quality attributes across the functional layers as well as agreeing on provided levels of quality. To ensure agreed quality attributes, techniques which are based on monitoring, testing or static analysis (e.g., model checking) are employed and extended by novel techniques exploiting future technologies (like the Web 2.0). Additionally, techniques for ensuring the quality of the actual adaptations are relevant here.

For each element of the framework, interfaces are defined that describe the capabilities that are provided by one element of the framework to another element, resp., the capabilities required by one element from another. As an example, one interface of the service composition and coordination layer defines which kinds of monitoring events (cf. Figure 2) are provided for the adaptation strategies defined in the adaptation and monitoring aspect.

5 Conclusions

The innovation required for devising theories, mechanisms and methods for making the next generation of services and service-based applications become reality, cannot be delivered by any research group in isolation. It requires the synergy and integration of a variety of research communities including but not limited to Grid Computing, Service Oriented Computing, Software Engineering, Business Process Management, and Human Computer Interaction. To this end, S-Cube, the European Network of Excellence on Software Services and Systems, brings together major European research institutions to jointly devise the scientific foundations for future service technologies and methods. The results of S-Cube will thus equip the organizations of the future with the capabilities to develop and evolve innovative software services and service-based applications.

Acknowledgements: We cordially thank all S-Cube members. Their contributions to the state of the art deliverables have been an excellent input for this paper.

References

[1] –. Survey Data Says: The Time For SOA Is Now. Forrester Research, Inc. (2006)

[2] Di Nitto, E.; Ghezzi, C.; Metzger, A.; Papazoglou, M.; Pohl, K.: A Journey to Highly Dynamic, Self-adaptive Service-based Applications. Automated Software Engineering. 15 (2008) 313–341

[3] Erl, T.: Service-oriented Architecture. Prentice Hall (2004)

[4] Gallotti, S.; Ghezzi, C.; Mirandola, R.; Tamburrelli, G.: Quality prediction of service compositions through probabilistic model checking. In Proceedings 4th International Conference on the Quality of Software-Architectures, QoSA 2008, Karlsruhe, October 14-17, 2008. LNCS 5281, Springer (2008) 119–134

[5] Hielscher, J.; Kazhamiakin, R; Metzger A.; Pistore, M.: A Framework for Proactive Self-Adaptation of Service-based Applications Based on Online Testing. In Proceedings ServiceWave 2008 conference, Madrid, December 10-13, 2008, LNCS 5377, Springer (2008) 122–133

[6] Josuttis, N.: SOA in Practice: The Art of Distributed System Design. O'Reilly (2007)

[7] Kaye, D.: Loosely Coupled: The Missing Pieces of Web Services. RDS Press (2003)

[8] Kephart, J.; Chess, D.: The vision of autonomic computing. IEEE Computer. 36(1), (2003) 41–50

[9] Papazoglou, M. and Leymann, F.: Business Process Management. IEEE Encyclopedia of Software Engineering, John Wiley & Sons (2008)

[10] Papazoglou, M.; Pohl, K.: Report on Longer Term Research Challenges in Software and Services. Results from two workshops held at the European Commission, with contributions from Boniface, M.; Ceri, S.; Hermenegildo, M.; Inverardi, P.; Leymann, F.; Maiden, N.; Metzger, A.; Priol, T. European Commission, http://www.cordis.lu (2008)

[11] Papazoglou, M.; Pohl, K.: S-Cube: The Network of Excellence on Software Services and Systems. In At Your Service: An Overview of Results of Projects in the Field of Service Engineering of the IST Programme. MIT Press - Information Systems (2008)

[12] van Moorsel, A.: Metrics for the Internet Age: Quality of Experience and Quality of Business. In: Proceedings 5th Int'l Workshop on Performability Modeling of Computer and Communication Systems, Erlangen, September 15-16, 2001, Arbeitsberichte des Instituts für Informatik, Universität Erlangen-Nürnberg, (2001)