The Landscape of Service-Oriented Systems: A Research Perspective for Maintenance and Reengineering

Kostas Kontogiannis
National Technical University of Athens
Athens, Greece
kkontog@softlab.ntua.gr

Grace A. Lewis
Software Engineering Institute
Pittsburgh, PA USA
glewis@sei.cmu.edu

Dennis B. Smith
Software Engineering Institute
Pittsburgh, PA USA
dbs@sei.cmu.edu

Abstract

Service orientation has been touted as one the important technologies for designing, implementing and deploying large scale service provision software systems. In this position paper we attempt to investigate an initial classification of challenge areas related to service orientation. More specifically, we partition the problems and related issues of service orientation in three domains—Business, Engineering, and Operations domain—along with the notion of Service Strategy as a binding model for these domains. A list of potential challenges for the maintenance and reengineering of service-oriented systems is presented for discussion.

1. Introduction

Over the past decade we have witnessed a significant growth of software functionality that is packaged in the form of services and is accessible through a networked infrastructure. These services are delivered utilizing either open or proprietary network protocols and, are available either on closed corporate Intranets or on open protocols using the Internet. This approach to systems development is commonly referred to as service-oriented architecture (SOA), SOA-based systems, or service-oriented systems.

We have witnessed a gradual evolution from the first generation of service-oriented systems which were based on monolithic components that could be reconfigured at compile time, to the second generation of service-oriented systems which are based on vertically-integrated components that can be adapted and reconfigured at installation, and to some extent at runtime, and towards the third generation of service-oriented systems that will be cross-vertically integrated, context-sensitive, and reconfigurable in an autonomic, ad-hoc manner [Fitz06]

The initially slow but gradually increasing adoption of service-oriented systems is supported by both the technical and the business community. From a technical perspective, service-oriented systems are an approach to software development where services provide reusable functionality with well-defined interfaces; a service infrastructure enables discovery, composition and invocation of services; and applications are built using functionality from available services. From a business perspective, service-oriented systems are a way of exposing legacy functionality to remote clients, implementing new business process models by utilizing existing or third-party software assets, and reducing the overall IT expenditures. From either perspective, and despite their initially slow adoption and the abundance of conflicting standards that support them, service-oriented systems are becoming the solution to bridge the gap between business models and the technical solution to support and adapt changing business needs.

It is clear that the SOA paradigm is having a substantial impact on the way software systems are developed. However, although significant progress is being made in several fronts, current efforts seem to evolve in many directions, without a central compass. There seem to be no clear, commonly agreed upon, overarching themes to focus research activity. As a result, there is a danger that important research needs will be overlooked, while other efforts will focus on issues of peripheral long-term significance in practice.

In this position paper we attempt to provide an initial classification of research issues pertaining to the business, engineering and operation aspects of service-oriented systems, with a focus on some open research

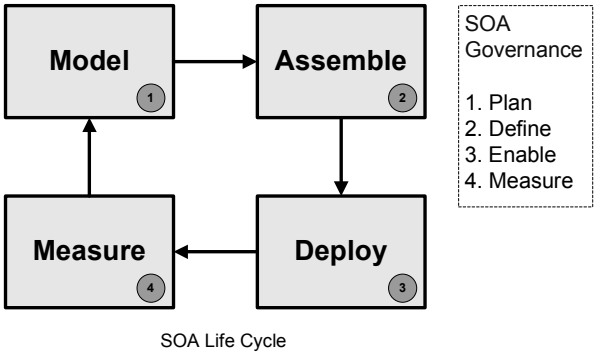

Figure 1. SOA Life Cycle and Governance

challenges related to the maintenance and reengineering of Service Oriented systems.

This position paper is organized as follows. In the second section we present current SOA practice. In the third section we present the main theme areas for issues related to service orientation. In the fourth section we outline some research directions in each of these areas. In the fifth section we focus on the maintenance and reengineering aspects of service-oriented systems. The sixth section presents some emerging opportunities for research in service orientation. The final section provides a summary and outlines next steps.

2. Current SOA Practice

IBM, based on best practices, has defined a life cycle for SOA-based systems that is consistent with other work on SOA life cycle [Borck06, High05, Rodriguez05, Veryard04]. This SOA life cycle, presented in Figure 1, consists of the following phases: modeling, assembly, deployment and management.
Modeling is the process of capturing business requirements, business goals and objectives, and transforming them into business process specifications—the *business model.* The modeling phase also includes the analysis of the model—"what if scenarios" applied to the business processes. The *Assembly* phase deals with the implementation issues: the business models are implementing by either reusing existing services or by creating new services. Functional testing is part of this phase. The *Deployment* phase includes resolving service dependencies, capacity planning, defining the hosting infrastructure, as well as system testing. The *Management* phase refers to the operational activities that keep the applications running as well as the measurement of IT and business performance indicators, logs and traces for auditing, and feedback for other phases of the SOA life cycle.

To ensure successful deployment of the SOA life cycle, it should be done in the context of what is known as SOA governance, as also shown in Figure 1 [Brown06, Windley06]. SOA governance is the process of establishing the chain of responsibilities and communications, policies, measurements, and control mechanisms that allow people to carry out their responsibilities. SOA governance itself has a set of phases: plan, define, enable, and measure. The *Plan* phase documents the existing IT capabilities and defines a governance plan. The *Define* phase defines or modifies the governance processes and the governance infrastructure. The *Enable* phase deploys the governance mechanisms and infrastructure, as well as the policies. The *Measure* phase monitors the compliance with policies and the effectiveness of the governance. SOA governance typically includes policies and procedures, roles and responsibilities, design-time governance and runtime governance [Gold-Bernstein06].

3. SOA Domains

Service orientation means different things to different people. For example, to software engineers it is all about functional requirements, components, integration techniques, messaging, tools, development environments, and middleware. To the business people it is all about implementing business strategies, enabling leaner IT departments, facilitating agile process models, and driving new products. Finally, to the operational users it is all about transparency, flexibility, ubiquitous access to services, and most importantly applications that ease their lives (e-government, e-health, entertainment). In fact, the service-orientation domain can be so diverse and multidisciplinary that the term Services Science has already been adopted by many organizations and academics to denote the variety of issues that cover service orientation [Chesbrough06, Horn05, IBM06].

In this respect, we could consider that service-oriented systems can be viewed from three domains or points of view, as presented in Figure 2. The first is the *Business* domain, the second is the *Engineering* domain, and the third is the *Operations* domain. The following sections discuss these domains in more detail.

3.1. Business Domain

The Business domain deals with the form and the impact that service orientation may take in a given

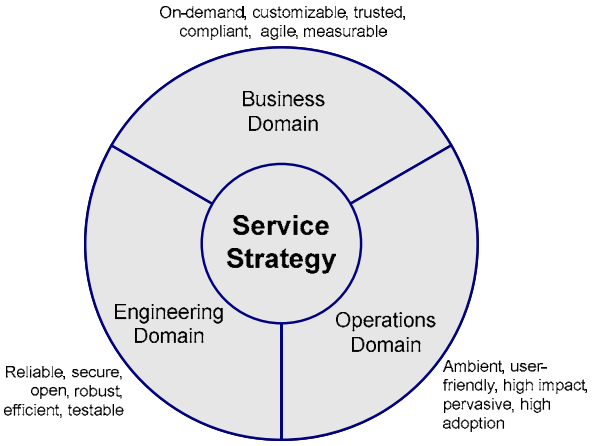

Figure. 2 SOA Domains

corporation, application domain, or context. Some of the drivers for service-oriented systems in this domain are to be on-demand, customizable, trusted, compliant, agile, and measurable. This classification of domains builds upon the notion of a conceptual model of business as considered in [IBM04].

Aspects of this domain therefore include business processes vis-à-vis service orchestration and choreography, conformance and auditing, organizational and financial aspects, evaluation of service-orientation-related IT decisions, new business models, widespread information dissemination, business strategies, workforce management methods, and service performance management.

3.2. Engineering Domain

The Engineering domain deals with the main aspects of the service-oriented system life cycle. Some of the drivers in this domain for service-oriented systems are to be reliable, secure, open, robust, efficient, and testable. Aspects of this domain therefore include process models that can be used to build service-oriented systems, requirement models for denoting functional and non-functional aspects, platform- and computational-independent architectural abstractions, design patterns, logging, model-driven code generation, verification, testing, and maintenance.

3.3. Operations Domain

The Operations domain deals with the deployment, diagnostics, support, adoption, usability, social impact, and pervasiveness of service-oriented systems. Some of the drivers in this domain are for service-oriented systems to be ambient, user-friendly, high impact, pervasive, and adoptable. This domain therefore includes any aspects related to people managing and acting upon services and service- oriented systems as well as, patterns, trends, relationships/differences across languages, cultures, monitoring, and diagnostic frameworks.

3.4. Service Strategy

At the core of an SOA environment is a Service Strategy. We see Service Strategy as the element that ties the Business, Engineering, and Operations domains together, reflecting the influence they have on each other.

Even though it may be difficult to define in precise terms what a Service Strategy is, some of its properties and characteristics can be identified. First, it provides the cause-effect and impact links behind the decisions taken at the Business level, Engineering level and, Operations level in an organization. For example, the Service Strategy will provide the rationale for selecting a technical solution given a business decision or an operational requirement. Second, it provides the top-level flow of activities for the life cycle of a service-oriented system. Third, it provides the common ground for the analysis of a service-oriented system that takes into account different perspectives and points of view. For example, a Service Strategy will provide the rationale for evaluating a system from a technical perspective given the specific constraints of its business and operational context.

4. Research Challenges for Service-Oriented Systems

The following is an attempt to classify research issues in the previously identified domains. The challenges listed under each category are still at a very high level. They are based on a preliminary literature search, expert opinions from academia and industry, as well as the author's experience. This list is by no means complete and it is the intent of the authors to gather feedback from a wide community through exposure of the proposed classification and challenges.

4.1. Business Domain

The research pointers in the business domain focus on activities pertaining to the overall business process as well as on compliance, trust, and analytics.

Management
- Enterprise planning
- Revenue forecasting
- Contract pricing
- Workforce allocation
- Roles and responsibilities
- Service-orientation as a business optimization facilitator
- Coping with the ever-shrinking cycle from innovation to product deployment
- Techniques for risk assessment and mediation
- Support for collaborative work

Evaluation
- Return on investment for service-oriented systems
- Relation of IT metrics with business metrics
- Collection and analysis of data for workforce management and productivity optimization
- Visualization methods and models for assessing the effectiveness of services

Business Processes to Services
- Methods and models for the selection of appropriate services for a given domain, requirements, and objectives.
- Mapping of business processes to services
- Strategic reuse of legacy components and services
- Industry and domain-specific standards for business and service modeling
- Alignment of people across organizations to implement agile business processes

Compliance
- Policy, risk, trust, ensuring a service acts on requests that comply with claims required by policies
- Best practices for compliance monitoring

4.2. Operations Domain

The research pointers in the operations domain focus on activities pertaining to specific application domains, as well as monitoring, support, adoption, and usability.

Monitoring
- Self-healing systems
- Instrumentation
- Diagnostics
- Logging
- Dynamic configurations of large scale systems

Support
- Operations support
- Problem management

Application Domains
- Investigation of application domains and potential impact (healthcare, government, finance/banking, electronics, telecommunications, entertainment electronics, automotive)

Adoption
- Customer adoption processes
- Service usability
- Matching needed services with user skills and abilities
- Portals and collaboration environments
- Social implications

Facilitating technologies
- Service appliances

4.3. Engineering Domain

The challenges in the engineering domain focus on activities that relate to the life-cycle of the system from its requirements specification to its maintenance and evolution.

Requirements
- Models and management of conflicting non-functional requirements
- Requirements elicitation and analysis for service providers
- Information integration vs. service integration

Architecture
- Service-oriented frameworks
- Platform-independent architectural styles
- Data mediation and messaging
- Modeling and use of data and metadata
- Non-functional-attribute-driven design

Design
- Design patterns
- Platform-specific models
- Messaging, invocation, diagnostics and, logging protocols
- Role of component-based software engineering
- Personalization and adaptation
- Semantic description and localization of services
- Service choreography and orchestration

Implementation

- Model-driven approaches
- Template-based code generation
- Language extensions to support service-oriented development
- Transformation frameworks

Testing

- Multi-level testing
  - Technology-Level: messages, description languages, repositories
  - Service-Level: services as components, functional testing
  - Architecture-level: Proof-of-concept, transaction management, quality of service, load/stress testing
  - Global-level Dynamic: Composition, orchestration, versioning, monitoring, and regression testing
  - Acceptance testing
- Test beds
- Benchmarking

Deployment

- Tools to support customization
- Context and location awareness
- Mediation of protocols and messaging
- Infrastructure best practices

Maintenance and Reengineering

- Evolution patterns
- Dependency and impact analysis
- Infrastructures for change control and management
- Tools, techniques and environments to support maintenance activities
- Multilanguage system analysis and maintenance
- Reengineering processes
- Tools for the verification and validation of compliance with constraints
- Round-trip engineering

5. Research Challenges for Maintenance and Reengineering of Service-Oriented Systems

One significant area of research interest in service-orientation is that of maintenance and reengineering. Service-oriented systems are significantly different from traditional systems, resulting in new research issues that need to be addressed. These differences include: 1) the diversity of service consumers, 2) shorter release cycles because of the capability of rapidly of adapting to changing business needs, and 3) the potential to leverage legacy investments with potentially minimal change to existing systems.

The following sub-sections outline challenges and research issues related to the maintenance and reengineering of SOA-based systems.

Evolution Patterns

Multi-tier service-oriented systems have been in operation for more than a decade. The on-going changes in the infrastructure technology (software, hardware, networking) have recently underscored the need to understand how such systems evolve, what forces and requirements drive evolutionary change, and whether patterns and trends can be derived. It is also important to understand the models that can represent evolution patterns and the metrics that can interpret them. We also need to better recognize the relationship that requirements for evolution have on actual activities and overall system quality. Current work on the evolution of legacy applications can serve as a starting point for understanding the evolution patterns of the emerging "new" legacy namely the Service Oriented systems.

Dependency and Impact Analysis

From a service provider perspective, changes in technologies, requirements, or business trends may lead to changes in the service. Changes that do not require modifications to the service interface will potentially have no impact on service consumers. However, a change in technology may have a negative effect on provided QoS even if the interface remains the same. Changes that do require modifications to the service interface can have a potentially large impact on service consumers. Interesting research issues are related to maintenance of multiple interfaces, impact analysis techniques for service consumers and providers in this environment, change notification mechanisms for service consumers, proper use of extensibility mechanisms in messaging technologies (e.g. SOAP extensibility mechanisms).

An emerging trend in the deployment of service-oriented systems involves the use of a set of models that need to co-exist and be kept synchronized. These models can relate to service and server deployment

description, access policy, proxies, invocation stubs, service locator classes and interfaces. In such a case the models need to be kept consistent. The research challenge is to identify dependencies between such models. This will facilitate impact analysis and to help understand how the effect of a change can be better understood, measured and assessed.

Infrastructures for Change Control and Management

Service-oriented systems can be deployed over a wide geographic area and on a set of different server computers. Owners of the service-oriented system may not have control over some of the services used. Despite the fact that robust techniques for configuration management in centralized systems are available, there are open issues with respect to managing change in distributed code bases and code repositories, especially when third-party service are involved. Furthermore, there may be additional requirements for the configuration management of large service-oriented systems. As a result, an open research issue is the development of a unified model for managing and controlling change in such systems.

Tools, Techniques, and Environments to Support Maintenance Activities

Over the past fifteen years, the software reengineering community has investigated and developed a wide range of methods and tools to support the analysis, comprehension and maintenance of legacy applications. However, the development of specialized methods and tools to support the maintenance and evolution of large service-oriented systems is in the early stages. Research needs include processes to support the incremental evolution of service-oriented systems, metrics to assess the impact of a change, specialized monitors to log the important quality indicators of a system, extraction tools, and meta-data and schemas for modeling dynamic and static information extracted from service-oriented systems.

Multilanguage System Analysis and Maintenance

A benefit of service-oriented systems is that they can be designed and implemented as multi-language, multi-paradigm systems. The analysis and maintenance of such systems poses unique challenges including 1) modeling, extraction and representation of information through static and dynamic analysis tools, 2) analysis, exploration and visualization of the extracted information, and 3) tool support and documented processes for the analysis and evolution of multi-language distributed systems.

Reengineering Processes

The reengineering and migration of legacy systems and components into network-centric and service-oriented environments represents an emerging area of research. Open research issues include the definition of migration processes for SOA-based systems as well as frameworks for modeling, denoting, and enacting these migration processes. Such processes may include those related to the migration of legacy components to SOA environments, as well as the migration of existing SOA-based systems to newer SOA environments.

Tools for the Verification and Validation of Compliance with Constraints

Service-oriented systems are implemented as a collection of collaborating services and systems. This collaboration takes the form of workflows that enact specific business processes. A research challenge is to investigate and develop techniques and tools that can be used to verify whether a specific service-oriented system or a specific composition pattern of services implements a given business policy or process, and whether it violates constraints. The compilation of guidelines for best practices for compliance monitoring could be a valuable resource in this area.

Round-trip Engineering for Service-Oriented Systems

An trend in modern software engineering practice is the use of models for representing software artifacts at different levels of abstraction and system life-cycle phases. Such models (requirements models, design models, source code models, test models) should all co-exist and evolve in a synchronized fashion. Model-driven engineering facilitates this co-evolution in a systematic way. An important research need is the investigation and extension of model synchronization techniques to support service-oriented systems. Round trip engineering in this environment involves the transformation of requirement models to code and code back to requirements models, considering that in some cases there will be no code other than a service interface (in the case of third-party systems).

6. Outlook – Emerging Opportunities

There are conflicting views with respect to the usefulness of service-orientation and its related implementations (e.g. Web Services). In one extreme, service-orientation proponents advocate it is the solution to addressing the needs of the next generation software applications. On the other extreme, service-orientation non-believers advocate that it is just a hype that has not delivered on its promises and has been plagued by limited adoption by the industry. We believe that the truth may be somewhere in the middle. On one hand, service orientation is a very promising paradigm for large systems. On the other hand, it is useful and should be considered if and only if there is a proven need, business case, and demand for it. We advocate that service orientation is and should be business-driven and demand-driven. And finally, service-orientation is not the solution to all problems. There are certain types of problems, such as interoperability, integration, choreography, context awareness, design of ultra large scale distributed service-based applications, where service-orientation can be of benefit. There are other types of problems and contexts where the technologies that support service orientation are not enough, or where the investment is too large to justify its implementation. We are also experiencing a healthy growth of technologies that are taking us gradually to the third generation of service-oriented systems, where context-sensitivity, adaptivity, and autonomicity become important requirements.

The outlook for service orientation therefore looks very positive and with it come the challenges to support this paradigm, as well as the opportunities for new research. Maintenance and reengineering practices for service-oriented systems will have to evolve and adapt to this dynamic and changing environment. The quest for third-generation service-oriented systems will also provide a series of issues for maintenance and reengineering that are worth exploring.

5. Conclusions

In this position paper we provide an initial classification of research issues in three domains—Business, Engineering, and Operations. We then focus on the maintenance and reengineering aspects and provide rationale for potential research areas.

Despite the slower than desired adoption of service-oriented systems in industry, we believe that the outlook is very positive, provided that it is always considered and applied within specific contexts, and in order to solve problems that are suited for the technologies supporting service orientation. Furthermore, we argue that for service orientation to succeed there must be a demand and a strong business case for it. Initial research is showing a large amount of research in the Engineering and Operational domain and less in the Business domain. This in itself is a large potential for research if we believe that service-orientation adoption should be business- and demand-driven—hence the need for a Service Strategy.

The next steps for this work are to obtain feedback from research and practitioners on the classification of research issues and their validity. Then, as opposed to classifying the issues into short- and long-term, we will attempt to classify them as important for the success of first-, second-, and third-generation service-oriented systems, with the goal of producing a research agenda in the area of service orientation.

6. References

[Borck06] Borck, J. *Planning an SOA: Gathering Around the Drawing Board*. Infoworld. May 2006. http://www.infoworld.com/article/06/05/08/77665_19FEsoalife2_1.html?s=feature

[Brown06] Brown W. and Cantor, M. *SOA Governance: How to Oversee Successful Implementation through Proven Best Practices and Methods.* IBM White Paper. ftp://ftp.software.ibm.com/software/rational/web/whitepapers/10706900_SOA_gov_model_app_v1f.pdf

[Chesbrough06] Chesbrough, H., Downes, L., Glushko, R., Righter, R. and Saxenian, A. *Designing a "Services Science, Management and Engineering" Discipline and Curriculum*. Position paper for Position paper for Workshop: "Education for Service Innovation". April 2006. http://ssme.berkeley.edu/papers/SSMECurriculum.pdf

[Fitz06] Fitzerald, B. and Olsson C. M. (eds), *The Software and Services Challenge*. Contribution to the preparation of the Technology Pillar on "Software, Grids, Security and Dependability, EY 7th Framework Programme.

[Gold-Bernstein05] Gold-Bernstein, B. and So, G. *Integration and SOA: Concepts, Technologies and Best Practices.*

[High05] High, R., Kinder, S., and Graham, S. *IBM's SOA Foundation: An Architectural Introduction and Overview.* November 2005. http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ws-soa-whitepaper.pdf

[Horn05] Horn, P. *The New Discipline of Services Science.* Business Week. January 2005. http://www.businessweek.com/technology/content/jan2005/tc20050121_8020.htm

[IBM04] IBM Research. *Service Science: A New Academic Discipline? http://www.google.ca/search?hl=en&q=IBM+Research+Service+Science+a+new+academic&btnG=Search&meta=*

[IBM06] IBM Corporation. *Services Sciences, Management and Engineering* (2006). http://www.research.ibm.com/ssme/index.shtml

[Rodriguez05] Rodriguez, J. *New Rules Govern SOA Lifecycle*. July 2005. http://www.looselycoupled.com/opinion/2005/rodri-rules-gov0701.html.

[Veryard04] Veryard, R. *The SOA LifeCycle*. CBDI. August 2004.

[Windley06] Windley, P. *SOA Governance: Rules of the Game.* InfoWorld. January 2006. http://www.infoworld.com/pdf/special_report/2006/04SRsoagov.pdf