Software Engineering Institute

A Research Agenda for Service-Oriented Architecture (SOA): Maintenance and Evolution of Service-Oriented Systems

Grace A. Lewis
Dennis B. Smith
Kostas Kontogiannis

March 2010

TECHNICAL NOTE
CMU/SEI-2010-TN-003

Research, Technology, and System Solutions Program

http://www.sei.cmu.edu

Carnegie Mellon

This report was prepared for the

SEI Administrative Agent
ESC/XPK
5 Eglin Street
Hanscom AFB, MA 01731-2100

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

This work is sponsored by the U.S. Department of Defense. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about SEI publications, please visit the library on the SEI website (http://www.sei.cmu.edu/library).

Table of Contents

Abstract vii

1 Introduction 1

2 A Research Agenda for Service-Oriented Architecture 3
- 2.1 SOA Problem and Solution Space 3
- 2.2 Service-Oriented Systems Development Life Cycle 7
  - 2.2.1 Overview of the Service-Oriented Systems Development Life Cycle 7
  - 2.2.2 Phases 8
  - 2.2.3 Activities 11
  - 2.2.4 Indicators 13
- 2.3 Taxonomy of SOA Research Topics 13
  - 2.3.1 Business 14
  - 2.3.2 Engineering 15
  - 2.3.3 Operations 15
  - 2.3.4 Cross-Cutting 16

3 Research Topics in Maintenance and Evolution of Service-Oriented Systems 17
- 3.1 Tools, Techniques, and Environments to Support Maintenance Activities 17
- 3.2 Multilanguage System Analysis and Maintenance 19
- 3.3 Reengineering Processes for Migration to SOA Environments 20
- 3.4 Transition Patterns for Service-Oriented Systems 23

4 Conclusions and Future Work 25

References 26

List of Figures

Figure 1:	Overview of the SOA Problem Space and Solution Space	4
Figure 2:	Flow Diagram of the Expanded View of the SOA Problem Space and Solution Space	5
Figure 3:	SOA Research Taxonomy	14
Figure 4:	Examples of Business Research Topics	15
Figure 5:	Examples of Engineering Research Topics	15
Figure 6:	Examples of Operations Research Topics	16
Figure 7:	Examples of Cross-Cutting Research Topics	16
Figure 8:	SOA Migration Horseshoe [Winter 2007]	21

List of Tables

Table 1: Mapping Between Phases, Activities, and Indicators 8

Abstract

Despite recent reports that it has failed, the reality is that Service-Oriented Architecture (SOA) remains the best option available for systems integration and leverage of legacy systems. The technologies to implement SOA will certainly evolve to address emerging needs, but its concepts will remain. To address those needs and concerns that SOA is potentially being stretched beyond its limits, a significant and coordinated research program is needed.

The SEI has developed an SOA Research Agenda with participation from a broad cross-section of the research community. The core of the agenda is a taxonomy that classifies topics into the business, engineering, and operations aspects of service-oriented systems, along with a set of cross-cutting aspects. Based on this taxonomy, the agenda outlines research areas, each of which is identified with its rationale, overview of current research, and delineation of research challenges and gaps. This report outlines the SOA Research Agenda. It also provides detail on specific research challenges related to the maintenance and evolution of service-oriented systems. The report concludes with a discussion of next steps in the evolution of the research agenda.

1 Introduction

Service-Oriented Architecture (SOA) is a way of designing, developing, deploying, and managing systems that are characterized by coarse-grained services and service consumers. The services represent reusable business functionality; through standard interfaces, service consumers compose applications or systems using those services.

Despite a highly publicized report that claimed that "SOA is Dead,"[1] the reality is that SOA is currently the best option available for systems integration and leverage of legacy systems. According to a 2007 Gartner Group report, 50% of new mission-critical operational applications and business processes in 2007 were designed around SOA, and that number was projected to be more than 80% by 2010. In May 2009, Forrester Research reported that 75% of IT executives at Global 2000 organizations plan to be using SOA by the end of the year, and that less than 1% report having a negative SOA experience.[2] This suggests that SOA adoption remains a significant goal for IT decision makers. The technologies to implement SOA will most probably change over time, but the concepts will remain.

Although SOA continues to be broadly adopted, it is still difficult to separate fact from fiction about it because of over-heated rhetoric and vendor hype. This makes it difficult to identify the true research challenges that need to be addressed to move the SOA community forward. The SOA research community has gone through a substantial "growth spurt," and faces the need to better channel its research efforts, identify important and promising research challenges, and establish priorities for these challenges.

This need is especially important because of recent concerns that SOA is potentially being stretched beyond its limits [Lewis 2009]. What was initially an approach for asynchronous document-based message exchange that applied primarily to business-oriented applications now has expanded to a larger set of distributed systems with greater expectations for performance, availability, reliability, security and other quality attributes.[3] As a result, the fundamental loosely coupled, stateless, standards-based nature of the relationship between service consumers and service providers in service-oriented systems is changing.

If SOA is to be used in *advanced* ways, significant research needs to be done in areas such as design for context awareness, service usability, federation, automated governance, runtime monitoring and adaptation, dynamic service discovery and composition, event-driven SOA, real-time applications, and multi-organizational implementations.

In order to assure that SOA can meet these expectations, multiple specifications and standards have been proposed and created, and middleware products are becoming more robust. Some of the standards to support Web Services are quite stable such as XML, WSDL, and SOAP. However, standards and technologies that support quality attributes of web services, such as security,

[1] http://apsblog.burtongroup.com/2009/01/soa-is-dead-long-live-services/comments/page/2/

[2] http://blogs.zdnet.com/service-oriented/?p=2053

[3] Quality attributes represent a system's non-functional requirements.

transaction management, and availability remain in a state of flux. To understand and respond to these new expectations and ongoing changes and their implications, a significant and coordinated research program is needed.

The SEI has been developing and evolving an SOA Research Agenda based on a proposed life cycle for service-oriented systems that emphasizes the relationship between business strategy and SOA strategy. The core of the agenda is a taxonomy that is classified into research topics pertaining to the business, engineering, and operations aspects of service-oriented systems, plus a set of cross-cutting aspects. This agenda was developed from the perspective of the research community to synthesize current research efforts and identify research topics that still needed to be addressed. The outline of the research agenda was the focus of several international workshops [Kontogiannis 2007a, 2007b, 2008, Lewis 2008a] that led to the publication of about 40 papers. The ongoing discussions at these workshops, as well as emerging research and new case studies, have helped to evolve and update the research agenda. The goal of the agenda is to provide a set of research topics for academic and industrial researchers that we believe support a strategic approach to SOA adoption and implementation, as well as the trends in usage and implementation of service-oriented systems.

This report presents an overview of the current state of the SOA Research Agenda and focuses specifically on maintenance and evolution, a growing concern as more and more service-oriented systems are deployed. The details of other areas of the research agenda will be the focus of future reports.

Section 2 of this report outlines the basic structure of the SOA Research Agenda. Section 3 identifies a set of research topics for maintenance and evolution of service-oriented systems. Section 4 provides conclusions and outlines next steps in the evolution of the SOA Research Agenda.

2 A Research Agenda for Service-Oriented Architecture

The SEI SOA Research Agenda was initially developed in 2007 on the basis of

- an extensive literature review of topics related to SOA, with the purpose of identifying the state of the practice—including multiple case studies of successful SOA adoption
- interviews with practitioners and researchers to identify both enablers of and barriers to SOA adoption

Case studies, even though mostly vendor-sponsored and product-specific, in particular tend to have a common theme: a strong link between business strategy and SOA strategy. With this in mind, we developed a research agenda that includes these elements:

- an identification of the SOA problem and solution space
- a proposed service-oriented systems development life cycle to support strategic SOA adoption and implementation
- a taxonomy of SOA research areas related to engineering, business, operational, and cross-cutting concerns to address the proposed life cycle

The rest of this section presents the three elements of the research agenda.

2.1 SOA Problem and Solution Space

The term service-orientation implies a distinct set of concerns and activities to different audiences. For example, to software engineers it is all about functional requirements, components, integration techniques, messaging, tools, development environments, and middleware. To business people, it is all about implementing business strategies, enabling leaner IT departments, facilitating agile process models, and driving new service delivery processes. To operational users, it is all about service-level agreements, transparency, flexibility, ubiquitous access to services, and most importantly applications that ease their lives (e-government, e-health, and entertainment).

In a service-orientation adoption setting, an organization should develop a service strategy that takes into account the organization's business drivers, context, and application domain. In order to accomplish its service strategy, the organization has to generate plans to achieve the goals and objectives outlined by the strategy. The execution of these plans requires engineering, business, and operations decisions to be made by the groups identified previously, taking into consideration cross-cutting concerns such as governance, security, risk management, social and legal issues, and training and education. These relationships are shown as problem, planning, and solution spaces in Figure 1.

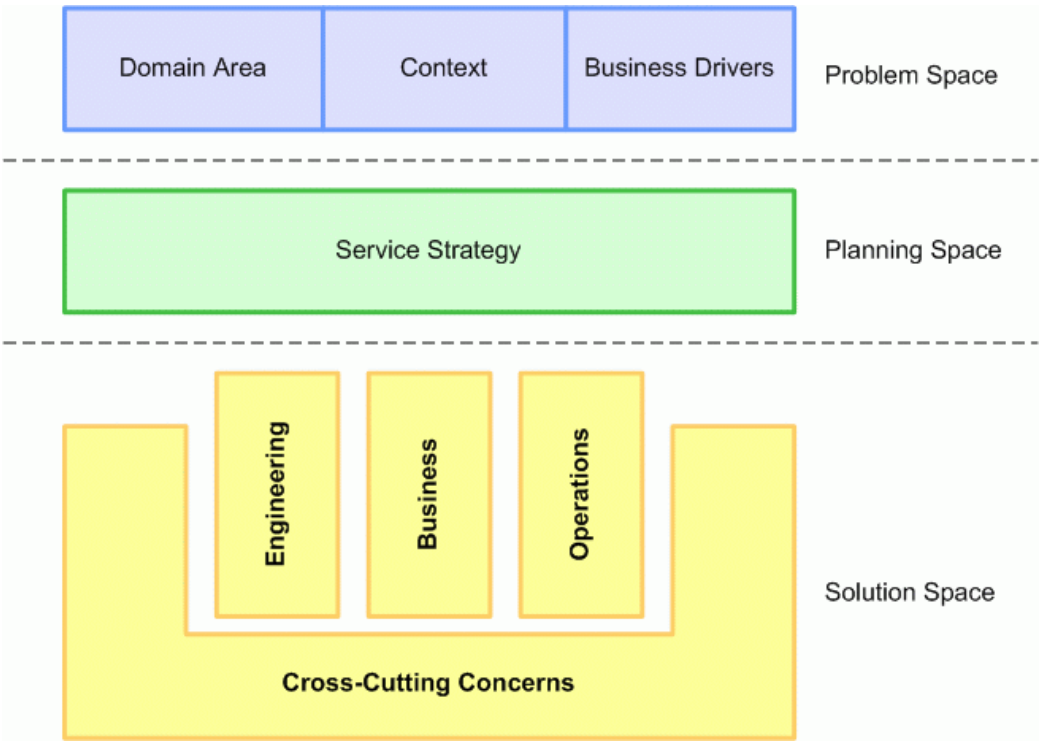

Figure 1: Overview of the SOA Problem Space and Solution Space

Problem Space

The problem space corresponds to the characteristics of the adopting organization as well as the problems that SOA is expected to address. The problem space shapes and places constraints on the strategy but can also enable the execution of the strategy. The elements of the problem space become the drivers for the strategy.

Domain Area: This is the domain in which the organization operates, such as manufacturing, health, government, education, or consulting. It is an important element of the problem space because some domain areas have more rapidly adopted SOA or have created standards or associations to enable SOA adoption.

- Manufacturing organizations that in the past have used Electronic Data Interchange (EDI) for integration with business partners or Enterprise Application Integration (EAI) techniques for systems integration have an easier time adopting SOA because their business processes probably have already been designed around interactions between different systems.
- Health Level 7 (HL7) exists in the health domain to create standards for health information exchange.
- The E-Gov program provides guidance and tools for system integration and SOA to U.S. federal government agencies.

Context: Context is the environment in which the organization operates and in which services will be deployed. It includes organizational issues such as business size, budget, legislation, competitors, market, business processes and technical issues including systems, system users, technology, computing platforms, and available infrastructure.

Business Drivers: The business drivers are the business reasons why the organization is adopting SOA. Different business drivers will lead to different strategies. Examples of business drivers are

- increase information available to customers
- integrate with business partners
- improve business processes
- respond rapidly to business changes
- reduce development costs by increasing reuse

Planning Space

The planning space takes the problem space as input and its output is a set of SOA plans, as shown in Figure 2. In this flow diagram, the rounded squares represent activities and the rectangles represent artifacts.

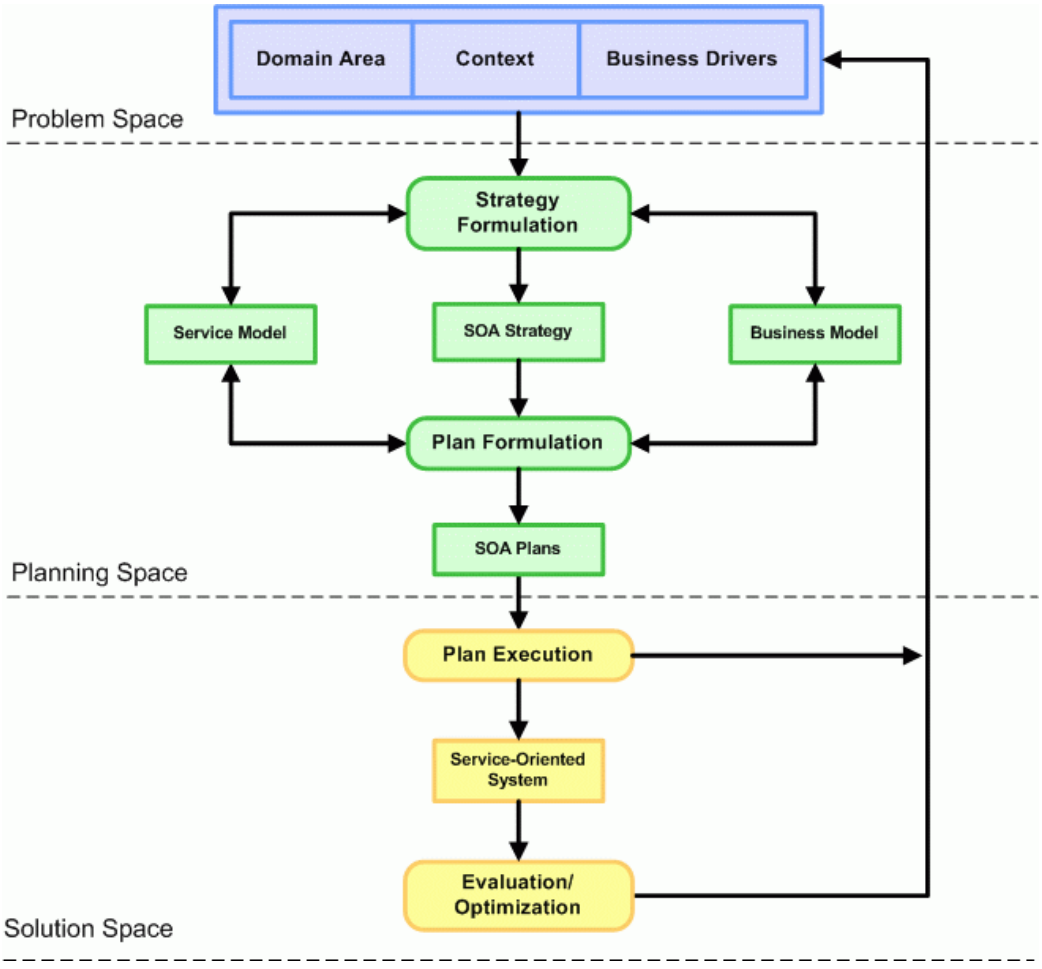

Figure 2: Flow Diagram of the Expanded View of the SOA Problem Space and Solution Space

An SOA strategy should be stated as the way in which SOA is going to address the organization's business drivers for SOA adoption. The feasibility of the strategy, as well as the effort to accomplish it, will be largely determined by the organization's domain area and context, which is why the strategy also needs to take these aspects into consideration.

Developing and deploying service-oriented systems is an iterative process; hence, the term "perpetual beta" is sometimes used to indicate the dynamism of this environment in responding to the need for business agility. The organization's SOA strategy may change over time due to changes in the problem space or to information provided by data collected during Evaluation/Optimization, as shown in Figure 2. Some examples of service strategies are to

- increase information available to business customers through the creation of a customer portal and services related to customer information
- integrate with business partners by focusing on integration of heterogeneous information, back office integration, and identification of business rules
- improve business processes through identification of key processes, elimination of redundancy, consistency between processes, and the creation of services that access legacy systems

The development of the SOA strategy and SOA plans both take into consideration and affect the organization's business model and service model.

- The Business Model is the representation of an organization's business processes.[4] It typically includes the organization's structure, business processes, and information required to support the processes.
- The Service Model is the representation of an organization's existing services and how they relate to the business processes in the Business Model.

The strategy is executed by creating SOA plans with very specific tasks, responsibilities, and time frames. Examples of tasks in an SOA plan are

- business process identification/modeling/management
- service creation/modification/reuse/elimination/consolidation
- technology evaluation/acquisition
- SOA infrastructure setup
- service consumer creation/modification/migration to make use of available services
- data migration/cleansing/consolidation
- legacy system changes to migrate to an SOA environment

Solution Space

In the solution space, the SOA plans are executed to produce a service-oriented system that includes a set of services that provide reusable business functionality, service consumers that use the functionality available from the services, commonly agreed upon service definitions, and an infrastructure that enables discovery, composition and service implementation. During the execution of the plans, changes or wrong assumptions about SOA technology may invalidate the plans and cause the organization to reformulate their SOA strategy, as shown in Figure 2.

Once the service-oriented system is deployed, measurements are gathered to support any metrics designed to test the effectiveness of the SOA strategy and the system itself. This data will help to

[4] Even though business processes are part of an organization's context, these are not always represented in a way that supports service identification decisions. The Business Model is such a representation.

optimize the SOA strategy, if needed, and also help to plan for a next iteration, once again reflecting the dynamic nature of service-oriented environments.

2.2 Service-Oriented Systems Development Life Cycle

A strategic approach to SOA adoption requires an iterative approach to systems development and evolution that reflects the strong link between business strategy and development strategy. SOA adoption is not "all or nothing." As a matter of fact, one of the benefits of SOA adoption is that systems and system components can be deployed gradually. What follows is a proposed development life cycle for service-oriented systems, where each pass through the life cycle corresponds to an iteration in Figure 2. The main differences with other iterative development frameworks, such as the IBM Rational Unified Process (RUP), are an emphasis on activities to establish and analyze the relationship with business goals at the beginning of the cycle, an emphasis on evaluation at the end of the cycle, and the specification/review of business objectives at the end of the cycle so that the requirements for each iteration follow business objectives.

2.2.1 Overview of the Service-Oriented Systems Development Life Cycle

The life cycle consists of

- Five phases (strategic alignment, planning, construction, transition, and production) that include interrelated sets of activities and milestones to support major SOA development and evolution goals
- Nine activities that support the goals of the phases. Because the life cycle is iterative, activities support multiple life-cycle phases.
- Four types of indicators for evaluating the effectiveness of SOA adoption and resulting systems against the goals

Table 1 illustrates the relationship between phases, activities and indicators. The development phases are listed across the top and the activities that are carried out to develop the service-oriented system during these phases are on the left, along with indicators to evaluate the effectiveness of the service-oriented system against SOA adoption goals. These are each discussed in more detail in the following three sections. The number of "plusses" indicates the emphasis placed on the activity or indicator in each phase.

Table 1: Mapping Between Phases, Activities, and Indicators

Phases	P1. Analysis	P2. Planning	P3. Construction	P4. Transition	P5. Production
Activities					
A1: Business Context Understanding	+++				++[a]
A2: Business Objectives Specification	+++	++	+		
A3: Risk Analysis and Initial Requirements Gathering	+++	+++	++	+	
A4: Prototyping and Requirements Tuning		++	+		
A5: Design and Implementation		+	+++[b]	++	
A6: Integration and Testing			++	+++	++
A7: Deployment				+++	++
A8: Maintenance			+	+	+++
A9: Management				++	+++
Indicators					
E1: Financial Indicator Measurements	++	+++			
E2: Technology Indicator Measurements	+	++	++	+	++
E3: User Rating Measurements				++	+++
E4: Compliance Indicators	+	++	+	++	+++

[a] Indicates a state of "perpetual beta"—requirements are not fixed; requirements for each iteration follow business objectives

[b] Implementation, Integration, Deployment, Maintenance, and Management are followed according to the development process used (RUP, Agile, etc.)

2.2.2 Phases

This section focuses on the five phases of the life cycle that are illustrated in Table 1. As stated earlier, each pass through this life cycle corresponds to an iteration of the flow shown in Figure 2.

Triggers for an iteration include

- pilot projects
- new services (or new versions of services)
- new service consumers (or new versions of service consumers)
- service consumer adaptation to use newly deployed services
- business evolution
- business process changes (or a business process reengineering [BPR] effort)
- legacy system changes that affect deployed services
- legacy system adaptation to fit into the SOA environment
- SOA infrastructure changes (or SOA infrastructure initial setup)
- major problem reports

Some of these iterations will be larger in scope than others. Each iteration is composed of the phases that are shown as the column headers in Table 1. These phases are

P1. Analysis

P2. Planning

P3. Construction

P4. Transition

P5. Production

Strategic Analysis Phase (P1)

The strategic analysis phase is composed of three sub-phases: analysis, positioning, and assessment.

Analysis

In early iterations, the main focus of the analysis sub-phase is to understand the business needs for SOA adoption and how SOA fits within the organization. In later iterations, this understanding needs to be reviewed as the business drivers and context change. Other focus areas in this phase are

- business process understanding: This includes the relationship to business needs and the identification of points of automation that would benefit the most from SOA adoption (no need to replicate things that are done well manually).
- initial identification of risks and opportunities
- analysis of the competition in the business processes identified for the iteration
- identification of goals for the iteration

Positioning

The positioning sub-phase is a phase of gap analysis. The focus is to understand the current business and service models and what changes need to be made to satisfy the goals identified. Other focus areas in this phase are

- full understanding of business processes affected
- identification of service-oriented system components that need to be created or will be affected during the iteration
- identification of sources for component implementation (legacy, new, third-party, commercial products)
- identification of additional elements that would need to be in place to accomplish goals: education, culture change activities, building infrastructure, etc.
- understanding of the gap between the as-is and to-be states

Assessment

The assessment sub-phase is a feasibility study. Its focus is on evaluating in more detail the specific tasks, costs, and risks needed to satisfy iteration goals. Other focus areas in this phase are

- risk analysis

- analysis of financial aspects related to funding for the iteration and expected return on investment (ROI)
- definition/update of the organization's SOA strategy
- identification of scope and size of effort for the iteration

Planning Phase (P2)

The focus of the Planning phase is to formulate the plan to reach the goals for the current iteration. Outcomes of this phase are

- concrete plans for the selected strategy
- requirements specifications
- workforce allocation
- technology selection
- infrastructure setup
- tool selection and setup
- definition/refinement of SOA governance elements, including policies, metrics, etc.

Construction Phase (P3)

The focus of the Construction phase is the implementation of the service-oriented system components defined in the plan. The SOA infrastructure is set up to handle new requirements as outlined in the strategy and plans. Provisioning approaches for services could be to build, buy, lease/pay-per-use, reuse, or wrap/adapt legacy systems. If within scope, service consumers such as applications, portals, or internal/external systems would have to be adapted to make use of the infrastructure and available services.

Transition Phase (P4)

The focus of the Transition phase is the integration of service-oriented system components that have been developed into the existing infrastructure. Other focus areas are

- training
- deploying SOA governance elements
- fine-tuning the system management infrastructure

Production Phase (P5)

The focus of the production phase is system operation and management. Other focus areas are

- collection of measurements as defined by the service strategy
- system tuning to meet defined metric goals and service level agreements (if applicable)
- management of problem reports
- kickoff for the next iteration

2.2.3 Activities

During the above phases, activities are carried out to accomplish the goals for the iteration, as shown in Table 1. Many of the activities will be carried out in several phases. In general the higher level activities support the earlier phases of the life cycle.

Business Context Understanding (A1)

The goal of this activity is to understand the context in which the business operates. Internal business processes are documented and analyzed against business goals, the market, and trends. Specific tasks include

- Business Process Management (BPM) activities
- market research
- business intelligence and information gathering
- trend analysis
- identification of key performance indicators (KPIs) for the business domain

Business Objectives Specification (A2)

The goal of this activity is to formalize business objectives and analyze the impact of SOA adoption on the accomplishment of these goals. Specific tasks include

- identification of business objectives
- identification of SOA adoption drivers
- evaluation of SOA adoption goals against business objectives
- selection of appropriate KPIs and evaluation metrics to determine the level of achievement of business objectives

Risk Analysis and Initial Requirements Gathering (A3)

The goal of this activity is to perform risk analysis and to start gathering the requirements for the iteration. Traditional risk analysis techniques apply to this environment: identification of threats, identification of impacts of each threat, assignment of probabilities and associated cost, and cost/benefit analysis. Multiple consultant and vendor firms have written about top risks, reasons for failure, or pitfalls of SOA adoption, based on their experience. While many of the risks apply to systems development projects in general, there are some that are specific to, or aggravated in, SOA environments.

Commonly mentioned business risks are

- performing “big-bang” deployments of service-oriented systems or starting with large projects
- lack of governance
- vendor lock-in
- failure to explain the business value of SOA
- underestimating the severity of organizational culture change

Common technical risks include

- a focus on products rather than architecture
- security
- poor versioning
- poor granularity of services and lack of a system management infrastructure

Requirements gathering activities from any iterative and incremental software development methodology used in the organization should work in this environment as well.

Even though this activity is straightforward, an important aspect to emphasize is that, because these systems are gradually deployed, requirements gathering and risk analysis activities always have to consider the portions of the system that have already been deployed. For example, the requirements for a new service may be redundant or conflicting with a deployed service.

Prototyping and Requirements Tuning (A4)

The goal of this activity is to tune and prioritize requirements based on prototyping results as well as to consolidate multiple stakeholder perspectives. Rapid prototyping for contextual technology evaluation is a key element of service-oriented systems development because of the emerging characteristics of technologies that support SOA implementation. Prototyping results will also serve to validate whether iteration requirements are realistic and whether additional investments in infrastructure are needed.

Even though requirements tuning is straightforward as well, there are aspects related to multiple stakeholders that can make it more complex, such as

- conflicting requirements between multiple business processes that use the same service
- conflicting requirements between direct users of a legacy system and consumers of the services exposed by the legacy system
- conflicting quality of service (QoS) requirements

Design and Implementation (A5)

The goal of this activity is to design and implement the elements of the service-oriented system that are outlined in the iteration plan: services, service consumers, and infrastructure. Techniques from any iterative and incremental software development methodology followed by the organization, such as RUP, XP, or SCRUM[5] should work in this environment.

Design and implementation activities will be highly constrained by existing systems and existing elements of the infrastructure. It is useful to identify and adopt best practices, design patterns, and architectural styles that work well in SOA environments.

Integration and Testing (A6)

The goal of this activity is to integrate all developed components and to test the system end-to-end. A major need for accomplishing this goal is to adapt the existing infrastructure for integrat-

[5] SCRUM is set of guidelines, derived from Agile methods, which aim to correct common failures in the typical development process. For more information, see http://www.codeproject.com/KB/architecture/scrum.aspx#Introduction0.

ing components. This activity is challenging because the many components involved cause versioning conflicts; it can be made easier by the establishment of a system management infrastructure for logging and monitoring.

Deployment (A7)

The goal of this activity is to make the system available for use. During deployment, system parameters need to be fine-tuned to optimize KPIs. Also, the system management infrastructure is set up to collect measurements to support defined metrics, service-level agreement (SLA) parameters, monitoring, logging, and runtime adaptation.

Maintenance (A8)

Maintenance of the service-oriented system and of the SOA-supported business processes make up this activity. In each of these maintenance areas, there will be routine requests such as changes in report formats, increase in performance, or bug fixes. But, there could also be major requests that could trigger another iteration, such as a new service or a major change in a business process.

Management (A9)

The goal of this activity is system management—that is, to make sure that the system operates according to expectations. During this activity, KPIs are monitored and measurements are gathered based on metrics and parameters set up during the Deployment activity. This activity could trigger maintenance requests that result in a new iteration (e.g., system not performing to expectations, security problems, and the like).

2.2.4 Indicators

A strategic approach to service-oriented systems development requires the development of indicators to evaluate the effectiveness of the service-oriented system against the SOA goals. The bottom portion of Table 1 shows a list of potential indicators. These indicators are selected and tuned based on changes in the organization, business goals, and implementation technologies.

- Financial Indicators (I1)—e.g., ROI, business value, maintenance costs, development costs
- Technology Indicators (I2)—e.g., performance, security and auditability, portability, reliability, availability, modifiability and testability, usability, and scalability[6]
- User Rating Indicators (I3)—e.g., user satisfaction and task execution time
- Compliance Indicators (I4)—e.g., SLA violations and audit reports

2.3 Taxonomy of SOA Research Topics

The development of a service-oriented system requires business, engineering, and operations decisions to be made, as well as other cross-cutting decisions. The taxonomy[7] for the agenda, shown in Figure 3, classifies research topics into these decision areas. The research topics correspond to areas where new, different, or additional research is needed to support a strategic approach to ser-

[6] As mentioned in Section 1, these are known as quality attributes.

[7] The original term taxonomy refers to classification of biological organisms. However, the term is now commonly used much more broadly to refer to classifications of themes or topics with an underlying structure. This is how we use the term here: to arrange SOA research topics into categories.

vice-oriented systems development. The identification of topics supports the goal for the SOA Research Agenda of proposing topics for industrial or academic research that can make a difference for strategic SOA adoption.

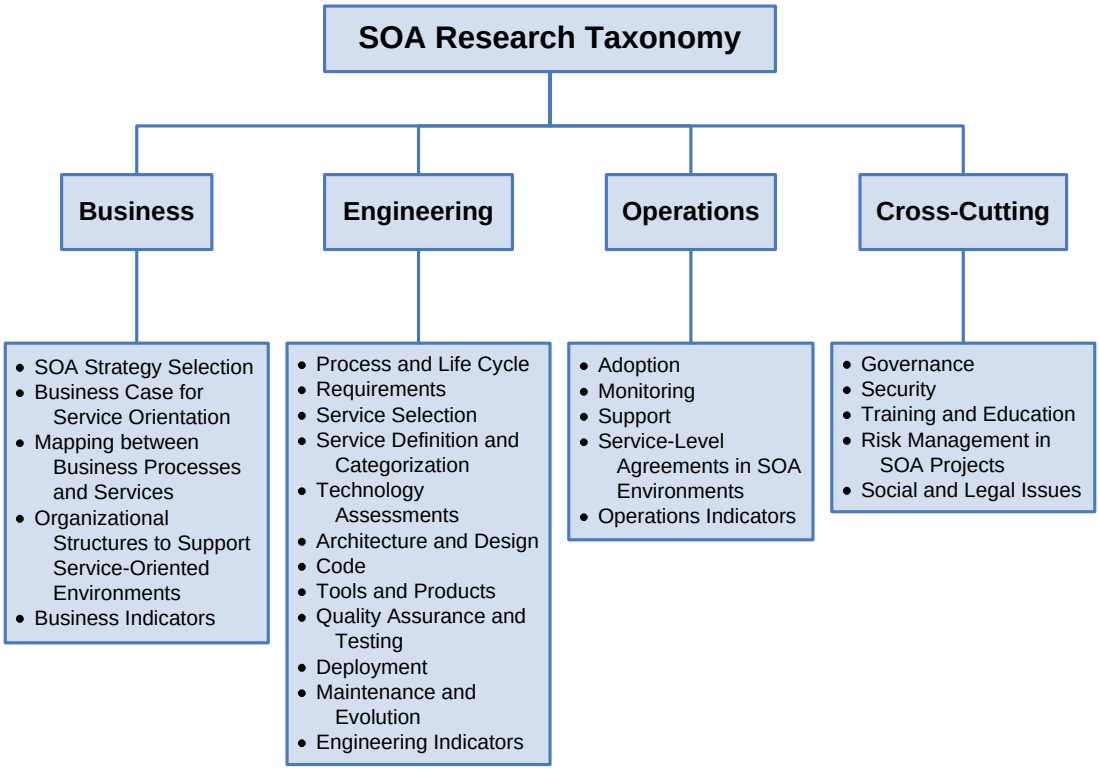

Figure 3: SOA Research Taxonomy

This section identifies the purpose of each decision area of the taxonomy. In the following discussion, we describe the decision area, list the major categories under each one, and show a select number of specific research topics.

2.3.1 Business

Business topics deal with the form and impact of service orientation in an organization. These topics are driven by a desire for on-demand, customizable, trusted, compliant, agile, and measurable systems. Figure 4 shows examples of selected business research topics.

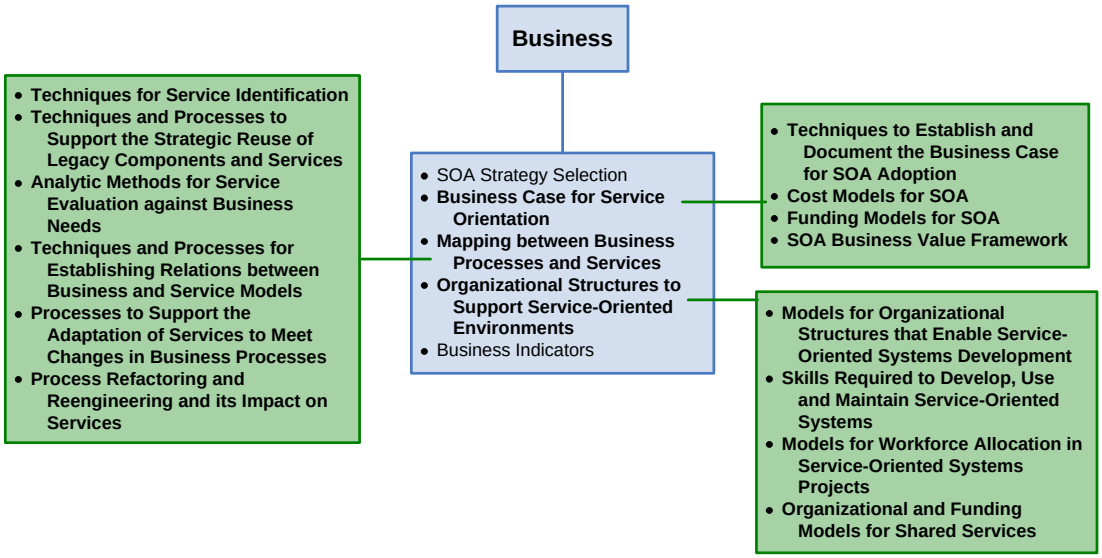

Figure 4: Examples of Business Research Topics

2.3.2 Engineering

Engineering topics deal with the main aspects of the service-oriented system life cycle. These topics are driven by a desire for reliable, secure, open, robust, efficient, and testable systems. Figure 5 has examples of selected engineering research topics.

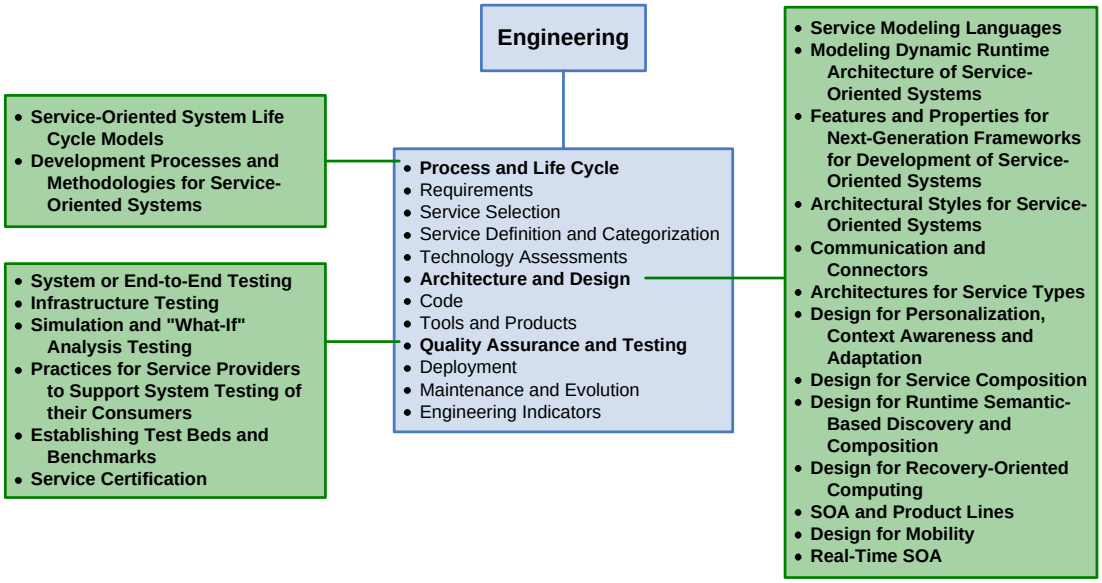

Figure 5: Examples of Engineering Research Topics

2.3.3 Operations

Operations topics deal with the operation, evaluation, and optimization of service-oriented systems. These topics are driven by a desire for ambient, user-friendly, high impact, pervasive, and adoptable systems. Figure 6 has examples of selected operations research topics.

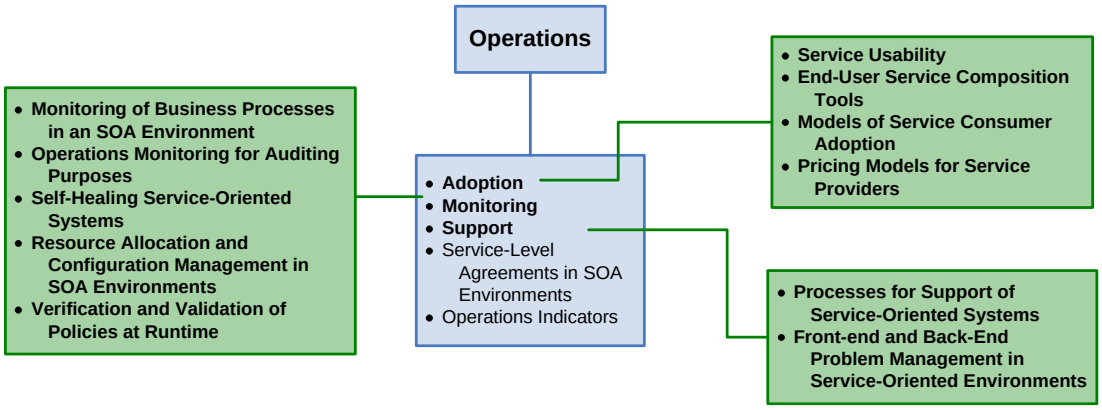

Figure 6: Examples of Operations Research Topics

2.3.4 Cross-Cutting

There are several topics that cannot be classified into any of the previous categories because they have an effect on all of them. Examples of such topics are governance, security, training and education, and social and legal issues. Figure 7 has examples of specific cross-cutting research topics.

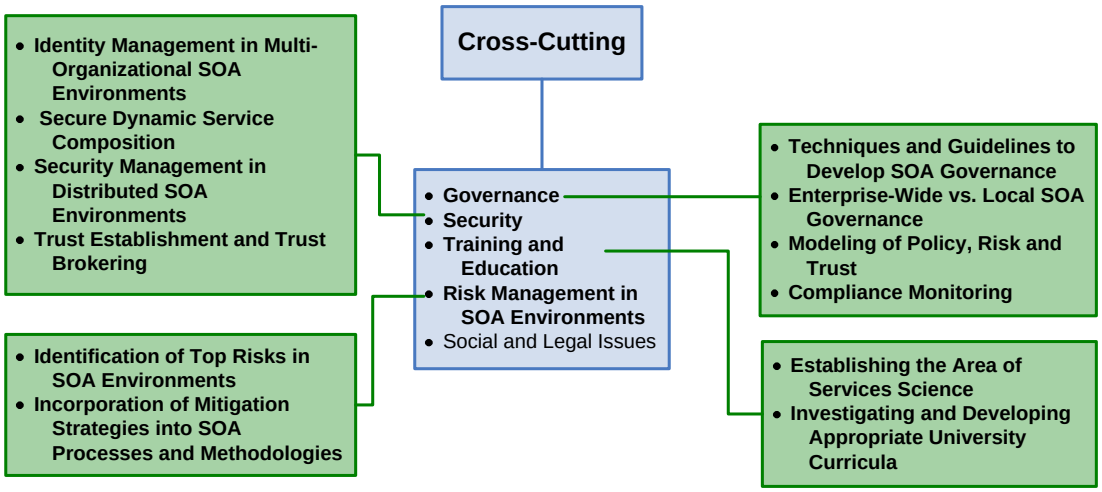

Figure 7: Examples of Cross-Cutting Research Topics

3 Research Topics in Maintenance and Evolution of Service-Oriented Systems

Given that SOA adoption is fairly recent, much of the current research has focused on the earlier stages of the life cycle. However, as service-oriented systems are deployed, major concern shifts to their maintenance and evolution. To show examples of how the SOA Research Agenda is used, this section provides more detail on selected topics in maintenance and evolution.

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 and service providers, (2) shorter release cycles because of the capability of rapidly adapting to changing business needs, and (3) the potential to leverage legacy investments with potentially minimal change to existing systems. An important question is, therefore, what does maintenance and evolution look like in this dynamic, heterogeneous, and potentially distributed development and maintenance environment? We have identified a set of research topics that we believe would help to find answers to this question. The references listed under current efforts are examples of work being done in each area.

In the following subsections, four research areas that are relevant to maintenance and evolution are discussed:

- Tools, Techniques, and Environments to Support Maintenance Activities
- Multilanguage System Analysis and Maintenance
- Reengineering Processes for Migration to SOA Environments
- Transition Patterns for Service-Oriented Systems

Each research area is analyzed in terms of (1) a rationale for why it is an important research topic, (2) current research efforts in the area, and (3) a list of challenges and gaps in the area.

3.1 Tools, Techniques, and Environments to Support Maintenance Activities

Rationale

The complexity of the maintenance process in an SOA environment continues to increase, especially as additional external consumers and providers become involved. As a result

- Impact analysis activities for service providers have to consider a potentially unknown set of users, unless there are mechanisms for tracking service consumers in the SOA infrastructure.
- Impact analysis for service implementation code has to consider direct users of the service implementation code, as well as users of the service interfaces.
- Configuration management becomes more complex, starting from the decision of what to put under configuration management, such as service interfaces, service test instances, configuration files, infrastructure services, and business processes.
- Release cycles between services and consumers, services and infrastructure, and consumers and infrastructure ideally should be coordinated, but may not be when some or all of them are outside the organization.

In addition, the sharing of increasing numbers of services among multiple business processes or consumers challenges maintenance. It is necessary to ask

- Who is responsible for the maintenance of a shared service?
- What happens when multiple business units have different requirements for the same service?
- How is a service evolved in the context of the multiple business processes that use it?

Current Efforts

Even though the maintenance of service-oriented systems is important and growing more difficult, there is not much current research that specifically addresses or provides guidelines for maintenance activities in a service-oriented environment. The work being done falls into the following areas:

- **maintenance processes**—SOA life cycles, such as the iterative one proposed by IBM and others, include maintenance in a post-deployment management phase [High 2005]. Mittal recommends the use of a robust development methodology the first time the service-oriented is rolled out and the use of lighter methodologies to support ongoing maintenance [Mittal 2005]. However, there is no concrete methodology for maintenance of service-oriented systems.
- **change impact analysis**—Work in this area can be seen as taking either a top-down or a bottom-up approach. A top-down approach is to analyze the impact of changes to business processes all the way down to the source code to identify affected system components [den Haan 2009, Ravichandar 2008, Xiao 2007]. A bottom-up approach is to analyze the impact of changes to a service—or its implementation—on the business processes and other consumers of the service [Zhang 2007]. Multiple integrated development environments integrate impact analysis, but the usual assumption is that there is control and full access to all system elements.
- **change management and version control**—This is an area that has received a lot of attention from the research and vendor communities [Brown 2004, Evdemon 2005, Flurry 2008, Laskey 2008, Lhotka 2005, Lublinsky 2007, Peltz 2004, Robinson 2006, Van Leeuwen 2009]. We believe that the reason for this is that the stability of service interfaces is part of the agreement (formal or informal) between service providers and consumers and therefore a very visible aspect of service-oriented development and deployment. The work usually involves the versioning of services—mainly Web Services—and not other components of a service-oriented system. The relationship between change management and SOA is being addressed in governance policies, where it extends beyond physical system components to even organizational components [Berry 2009, Mynampati 2008].
- **organizational structures and roles**—There is some preliminary research into roles and responsibilities for development, maintenance, and evolution of service-oriented systems [Kajko-Mattsson 2007, 2008].

Challenges and Gaps

Over the past 15 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 systems. However, the development of specialized methods and tools to support the maintenance and evolution of large *service-oriented* systems is in the early stages. Current efforts indicate

that maintenance activities for service-oriented systems are seen as being not unlike those for traditional systems. However, most existing service-oriented systems have been deployed for internal integration, where there is still some control over deployed services.

We believe that two factors will force changes to current maintenance practices: the emergence of a market for third-party services and the deployment of more service-oriented systems that cross organizational boundaries. From an engineering perspective, processes to support the incremental evolution of service-oriented systems, configuration management, impact analysis, and versioning in this environment are challenges. From a business perspective, the organizational structures and roles to support maintenance of service-oriented systems as well as models to support the development and maintenance of shared services are areas of much-needed research.

The addition of third-party services that do not require modifications to the service interface will potentially have no impact on service consumers. Changes that do require modifications to the service interface can have a potentially large impact on service consumers. In addition, a change in technology may have a negative effect on provided QoS, even if the interface remains the same. Important research issues are related to

- maintenance of multiple interfaces
- impact analysis techniques for service consumers and providers
- change notification mechanisms for service consumers
- proper use of extensibility mechanisms in messaging technologies (e.g., SOAP[8] extensibility mechanisms)

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 services 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.

3.2 Multilanguage System Analysis and Maintenance

Rationale

One of the benefits associated with SOA, and especially Web Services, is true platform independence. Even though standard interfaces are exposed, the underlying service implementation could be written in almost any language. While this is a huge benefit, it makes looking at the system as a whole difficult.

Current Efforts

The reengineering community has been working on this issue for a number of years to assist in analysis and migration of multi-language systems [Deruelle 2001]. Most work in this area in based on parsing of source code to create common higher level representations that can then be

[8] Simple Object Access Protocol, a standard for web services messages

analyzed using tools. Some of the problems with multi-language analysis are related to the mapping between data types between different languages. In a Web Services environment, this problem might be alleviated because XML Schema data types are used at the service interface level. There is some work in using string analysis to understand Web Services, given that messages are XML-based collections of strings [Martin 2006].

Challenges and Gap

Most research in this area is limited to smaller projects and a few languages, which is a problem for an environment that promotes platform independence. In the case of third-party service providers, access to source code is probably not possible. If that is the case, an important area of research is the identification of the type of information that service providers need to expose to service consumers—in interfaces or service registries/repositories—that wish to do code analysis, as well as tools and techniques to support the process.

3.3 Reengineering Processes for Migration to SOA Environments

Rationale

Because it has characteristics of loose coupling, published interfaces, and a standard communication model, SOA enables existing legacy systems to expose their functionality as services, presumably without making significant changes to the legacy systems. Migration of legacy assets to services has been achieved within a number of domains—including banking, electronic payment, and development tools—showing that the promise is beginning to be fulfilled. While migration can have significant value, any specific migration requires a concrete analysis of the feasibility, risk, and cost involved. The strategic identification and extraction of services from legacy code is crucial as well.

Current Efforts

There are not many reengineering techniques that focus on a "full-circle" model, such as the "SOA-Migration Horseshoe" proposed by Winter and Ziemann and shown in Figure 5 [Winter 2007]. This approach, which derives from a more general model proposed by Carrière [Carrière 1998], integrates software reengineering techniques with business process modeling and recommends applying reverse engineering techniques to extract a Legacy Enterprise Model from the legacy code. Then, the approach applies enterprise modeling techniques to create a Consolidated Enterprise Model from which services are identified using forward engineering techniques. Finally, legacy code is mapped to services via wrapping or transformation.

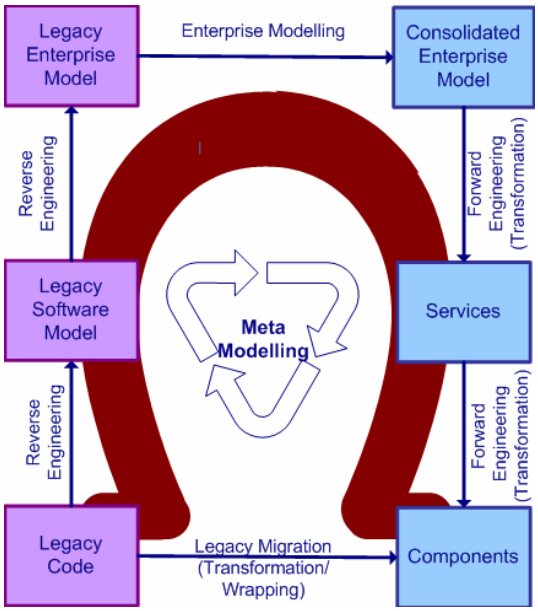

Figure 8: SOA Migration Horseshoe [Winter 2007]

The larger amount of work is on techniques in the bottom portion (open end) of the horseshoe for exposing legacy functionality as services, mainly Web Services [Chawla 2007]. Tools to support this type of migration are

- available as language libraries
- integrated into common Integrated Development Environments (IDEs), such as the Eclipse Web Tool Platform (WTP) and the .NET Windows Communication Foundation (WCF)
- included in infrastructure products such as Apache Axis2 [Eclipse 2009, Microsoft 2009, Apache 2009]

Some work takes into consideration business goals and drivers when making decisions about migrating to an SOA environment, which is consistent with a strategic approach to SOA adoption. The techniques evolved through this work operate on the top portion of the horseshoe.

- The Service Migration and Reuse Technique (SMART) is a method for determining the feasibility of migrating legacy systems to an SOA environment; it accounts for the importance of business drivers as well as characteristics of the legacy system [Lewis 2008b]. The output of this method is the identification of a pilot project and a migration strategy that includes preliminary estimates of cost and risk and a list of migration issues.
- Ziemann et al. agree that "rather than being of a purely technical nature, the challenges in this area are related to business engineering: 'How can a sub-functionality be identified as a potential service, or how can business process models be derived from a legacy system.'" They propose a business-driven, legacy-to-SOA approach based on enterprise modeling that considers both the business and legacy system aspects [Ziemann 2006].
- Not surprisingly, most major vendors have embraced migration of legacy systems to SOA environments and offer services in this area. IBM's Service Oriented Modeling and Analysis (SOMA) focuses on full-system development but has some portions that address legacy reuse: "SOMA facilitates integration with techniques for analyzing legacy applications, cus-

tom and packaged, to identify, specify and realize services for use in a service-oriented architecture. It breaks out the business functions of each existing application, identifying candidate services that can be used to realize business goals under the new architecture. It also identifies potentially problematic areas and highlights areas where new services need to be developed or sourced from an external provider [Arsanjani 2008]." Fuhr et al. have extended SOMA for model-driven migration to SOA environments [Fuhr 2009].

- Cetin et al. propose a mashup-based[9] approach for migration of legacy systems to service-oriented computing environments [Cetin 2007]. The interesting aspect about this work is its inclusion of presentation services, which is not typical. The approach is a combination of top-down, starting from business requirements, and bottom-up, looking at legacy code. Business requirements are mapped to services and integrated through a mashup server, a process which eliminates the need to develop specific applications to access the services.

Finally, there is work related to the identification of services in legacy code, addressing the left portion of the horseshoe.

- In the context of Web Services, Aversano et al. propose an approach that combines information retrieval tracing with structural matching of the target Web Service Definition Language (WSDL) with existing methods [Aversano 2007]. In the approach, first library schema extraction is performed; then it uses feature extraction to build a WSDL document from the legacy code. Finally, it uses structural matching to compare the generated WSDL document with the target WSDL document.
- Also in the context of Web Services, Sneed proposes an approach for salvaging the legacy code, wrapping the salvaged code, and making it available as a web service [Sneed 2006]. In the salvaging step, Sneed proposes a technique for extracting services based on identifying business rules that produce a desired result.
- Canfora et al. propose an approach for exposing interactive functionality of legacy systems as Web Services using black-box reverse engineering techniques on user interfaces [Canfora 2008].

Challenges and Gaps

The ideal reengineering process would be one that implements the full SOA-Migration Horseshoe. The problem, as shown under Current Efforts, is that there are techniques and tools that implement only portions of the horseshoe. Important areas of needed research are the formation of a concrete process to implement the horseshoe and the development of tools (or suites of tools) to support that process. Also, the automation of this process would be worth investigating, though very complex.

As stated by most researchers in this area, the real challenge is mining legacy code for services that have business value. Research topics in this area include

- tools and techniques for analyzing large source code bases to discover code that is of business value
- metrics for "wrapability" and business value to determine reusability [Sneed 2007]

[9] A mashup is a web application that combines data from more than one source into a single integrated tool (http://en.wikipedia.org/wiki/Mashup_%28web_application_hybrid%29).

- application of feature extraction techniques to service identification, given that services usually correspond to features [Sneed 2007]

3.4 Transition Patterns for Service-Oriented Systems

Rationale

One advantage claimed by SOA adoption is that it enables incremental system modernization. For example, legacy system components can be wrapped using Web Services technology initially and replaced with newer components incrementally. As long as the interfaces remain stable, service consumers only have to be modified once to initially access the new services. However, throughout the life cycle of the project, there will be a mix of migrated legacy components, legacy components waiting to be migrated, and legacy components that will not be migrated. Legacy components include application front ends, business logic, data logic, and actual data. A major challenge for incremental migration is therefore the minimization of "throw-away" cost and effort to support intermediate system states.

Current Effort

There is active academic and industrial work related to incremental modernization and enterprise transformation by migrating legacy systems to SOA environments. However, most of this work assumes that the legacy systems will be wrapped and integrated into a service-oriented system, where they will remain relatively stable not eventually replaced.

There are multiple techniques, tools, and consulting services to help organizations migrate legacy systems to SOA environments. Erl, for example, has developed several design patterns that can be used when legacy system components are part of service-oriented systems, such as service façade, service data replication, legacy wrapper, and file gateway [Erl 2009]. Architectural reconstruction and program analysis techniques could be used to isolate "chunks" of code and discover dependencies between components as an effort to study the impact of migrating one set of legacy components vs. another set of legacy components.

Challenges and Gaps

A significant need exists for research that addresses the total cost of SOA adoption—including development, implementation, data migration, and incremental updates. Such a model would also provide one of the inputs for realistic ROI calculations. While existing published studies have not fully addressed calculations of total cost, a study in progress is proposing a Total Modernization Cost (TMC) for replacing a system in a number of steps. In this study, if a new system is built to replace an old system in *n* steps then Total Modernization Cost (TMC) is defined as

$$TMC = \sum_{i=1}^{n} (DC_i + IC_i + DMC_i + TC_i)$$

where

DC = Development Cost
IC = Implementation Cost (Deployment + Infrastructure)

DMC = Data Migration Cost
TC = Transition Cost[10]

If we define Transition Cost as

$$TC = ECC + NTC + TI$$

where

ECC = Existing Code Changes + Maintenance Cost
NTC = New Throwaway Code + Maintenance Cost
TI = Temporary Infrastructure and Operation Cost,

then we would need to identify the process and supporting techniques and tools to determine

- the **right number of increments**
- that would **minimize the throwaway costs** due to
  - temporary infrastructure such as gateways and ETL (Extract, Transform, Load) tools
  - temporary code to deal with mismatches
  - changes to legacy code waiting to be modernized (e.g., adding code to invoke a service, knowing that it will be modernized in a future increment)
- in a **repeatable fashion** such that it could be recalculated when business changes its mind about priorities.

[10] This is work in progress between Parviz Dousti from the Carnegie Mellon University Information Technology Department and Grace Lewis from the Carnegie Mellon University Software Engineering Institute. Results will be published in 2010.

4 Conclusions and Future Work

Many published articles and case studies have focused primarily on SOA implementations within enterprise IT systems, in which applications interact with standard web services in a traditional request-response pattern, predominantly to access data that resides in legacy systems [such as Cetin 2007, Chawla 2007, Kajko-Mattsson 2008, Sneed 2007].

As the use of third-party services becomes the new business model, there needs to be support for SLAs, runtime monitoring, end-to-end testing involving third parties, pricing models for third-party services, and service usability from a design and an adoption perspective. In addition, non-vendor studies and experiments are needed to produce concrete guidance, rather than additional basic research. Some examples of these areas are SOA governance, business case for SOA adoption, ROI for SOA adoption, and development processes and practices for service-oriented systems development.

Also, if SOA is to be used in *advanced* ways, significant research topics need to be addressed in areas as design for context awareness, service usability, federation, automated governance and runtime monitoring and adaptation, dynamic service discovery and composition, real-time applications, and multi-organizational implementations.

In addition, we found several topics, such as use of semantics for service discovery and composition, in which there are significant efforts in the research community, but no support from industry to test ideas. There needs to be more collaborative research between industry and academia to create real practices.

For the maintenance and evolution of service-oriented systems, our research agenda shows that in the short term, maintenance and evolution practices will have to adapt to support this dynamic and changing environment, taking into consideration the emergence of third-party services over which there is less control and visibility. Tools and techniques to support maintenance and evolution activities in these environments, reengineered processes that combine business as well as technical aspects, and capabilities for multi-language analysis are a good starting point.

The research agenda has so far been used primarily to guide the research community on the current status of the state of the art, gaps in existing work, and potential areas of needed research. It also offers a rich potential set of focus for governments, corporations, and standards organizations in understanding the state of the art and setting their research priorities.

The next steps for this project are the publication of the details of other topics in the SOA Research Agenda. In addition, the Research Agenda will continue to evolve to account for new research as well as for emerging challenges as SOA continues to push beyond its original asynchronous document-based message exchanges.

References

[Apache 2009]
The Apache Software Foundation. *Apache Axis 2*. http://ws.apache.org/axis2/ (2009)

[Arsanjani 2008]
Arsanjani, A., Ghosh, S., Allam, A., Abdollah, T., Ganapathy, S., & Holley, K. "SOMA: A Method for Developing Service-Oriented Solutions." *IBM Systems Journal 47*, 3 (2008): 377.

[Aversano 2007]
Aversano, L., Di Penta, M., & Palumbo, C. "Identifying Services from Legacy Code: An Integrated Approach." *Presentation at the Working Session on Maintenance and Evolution of SOA-Based Systems (MESOA 2007). 23rd IEEE International Conference on Software Maintenance (ICSM 2007)*, Paris, France, October 4, 2007. IEEE Computer Society, 2007.

[Bass 2008]
Bass, Len, de Niz, Dionisio, Hansson Jőrgen, Hudak, John J., Feiler, Peter H., Firesmith, Donald, Klein, Mark H., Kontogiannis, Kostas, Lewis, Grace, Litoiu, Marin, Schuster, Stefan, Sha, Lui R., Smith, Dennis B., & Wallnau, Kurt C. *Results of SEI Independent Research and Development Projects FY 2007* (CMU/SEI-2008-TR-017). Software Engineering Institute, Carnegie Mellon University, July 2008. http://www.sei.cmu.edu/library/abstracts/reports/08tr017.cfm.

[Berry 2009]
Berry, Dave. *Avoid Disaster, Embrace People-Processes Like Change Management*. http://blogs.oracle.com/governance/2009/06/avoid_disaster_embrace_peoplep_1.html (June 2009)

[Brown 2004]
Brown, Kyle. & Ellis, Michael. "Best Practices for Web Services Versioning: Keep your Web Services Current with WSDL and UDDI." *IBM developerWorks* (January 2004). http://www-128.ibm.com/developerworks/webservices/library/ws-version/

[Canfora 2008]
Canfora Gerardo, Fasolino Anna Rita, Frattolillo Gianni, & Tramontana, Porfirio. "A Wrapping Approach for Migrating Legacy System Interactive Functionalities to Service Oriented Architectures." *Journal of Systems and Software 81*, 4 (April 2008): 463-480.

[Carrière 1998]
Carrière, Jeromy, Kazman, Rick, & Woods, Steve. "Requirements for Integrating Software Architecture and Reengineering Models: CORUM II," 154-163. *Proceedings of the Working Conference on Reverse Engineering (WCRE'98)*. Honolulu, HI, USA, October 12-14, 1998. IEEE Computer Society, 1998.

[Cetin 2007]
Cetin, S., Altintas, N. I., Oguztuzun, H., Dogru, A.H., Tufekci, O., & Suloglu, S. "Legacy Migration to Service-Oriented Computing with Mashups." *Proceedings of the Second International Conference on Software Engineering Advances (ICSEA 2007)*. Cap Esterel, French Riviera, France August 25-31, 2007. IEEE Computer Society, 2007.

[Chawla 2007]
Chawla, Mohit. & Peddinti, Vijaya. "Exposing SOA Enabled C Apps as Web Services." *SOA World Magazine* (February 2007). http://webservices.sys-con.com/read/314105.htm

[den Haan 2009]
den Haan, J. *A Framework for Model-Driven SOA*. http://www.theenterprisearchitect.eu/archive/2009/06/03/a-framework-for-model-driven-soa (June 2009)

[Deruelle 2001]
Deruelle, L., Melab, N., Boune, M., & Basson, H. "Analysis and Manipulation of Distributed Multi-Language Software Code," 43-54. *Proceedings of the First IEEE International Workshop on Source Code Analysis and Manipulation*. Florence, Italy, 2001, IEEE Computer Society, 2001.

[Eclipse 2009]
The Eclipse Foundation. *Web Tools Platform (WTP) Project*. http://www.eclipse.org/webtools/ (2009).

[Erl 2009]
Erl, Thomas. *SOA Design Patterns*. Prentice Hall. 2009.

[Evdemon 2005]
Evdemon, J. *Principles of Service Design: Service Versioning. Microsoft Corporation*. http://msdn2.microsoft.com/en-us/library/ms954726.aspx (August 2005)

[Flurry 2008]
Flurry, Greg. *Service Versioning in SOA*. http://www.ibm.com/developerworks/websphere/techjournal/0810_col_flurry/0810_col_flurry.html (2008).

[Fuhr 2009]
Fuhr, Andreas, Winter, Andreas, Gimnich, Rainer, & Horn, Tassilo. "Extending SOMA for Model-Driven Software Migration into SOA," *11. Workshop Software-Reengineering 2009 (WSR 2009)*. Bad Honnef, Germany. May 4-6, 2009. http://www.uni-koblenz-landau.de/koblenz/fb4/institute/uebergreifend/sre/conferences/wsr/wsr2009/fuhr.pdf

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

[Kajko-Mattsson 2007]
Kajko-Mattsson, Mira, Lewis, Grace, & Smith, Dennis. "A Framework for Roles for Development, Evolution and Maintenance of SOA-Based Systems," 7. *Proceedings of the International Workshop on Systems Development in SOA Environments (SDSOA 2007). International Conference on Software Engineering (ICSE 2007)*. May 2007. IEEE Computer Society, 2007.

[Kajko-Mattsson 2008]
Kajko-Mattsson, M., Lewis, G., & Smith, D. "Evolution and Maintenance of SOA-Based Systems at SAS." *Proceedings of the 41st Hawaii International Conference on System Sciences (HICSS-41)*. Big Island, HI, USA, January 2008. IEEE Computer Society, 2008.

[Kontogiannis 2007a]
Kontogiannis, Kostas, Lewis, Grace, & Smith, Dennis. "The Landscape of Service-Oriented Systems: A Research Perspective for Maintenance and Reengineering." *Proceedings of the International Workshop on Service-Oriented Architecture Maintenance and Reengineering (SOAM 2007). International Conference on Software Maintenance and Reengineering (CSMR 2007)*. March 2007.

[Kontogiannis 2007b]
Kontogiannis, Kostas, Lewis, Grace, Litoiu, Marin, Müller, Hausi, Schuster, Stefan, Smith, Dennis, & Stroulia, Eleni. "The Landscape of Service-Oriented Systems: A Research Perspective." *Proceedings of the International Workshop on Systems Development in SOA Environments (SDSOA 2007). International Conference on Software Engineering (ICSE 2007)*. May 2007. ACM, 2007.

[Kontogiannis 2008]
Kontogiannis, K., Lewis, G. A., & Smith, D. B. "A Research Agenda for Service-Oriented Architecture," 1-6. *Proceedings of the 2nd international Workshop on Systems Development in SOA Environments* (*SDSOA '08)*. Leipzig, Germany, May 11, 2008. ACM, 2008.

[Laskey 2008]
Laskey, Ken. "Considerations for SOA Versioning," 333-337. *12th International Conference on Enterprise Distributed Object Computing Workshops*. Hong Kong, China, October 16-20, 2008. IEEE Computer Society, 2008.

[Lewis 2008a]
Lewis, Grace & Smith, Dennis. *Proceedings of the International Workshop on the Foundations of Service-Oriented Architecture (FSOA 2007)* (CMU/SEI-2008-SR-011). Software Engineering Institute, Carnegie Mellon University, 2008.
http://www.sei.cmu.edu/library/abstracts/reports/08sr011.cfm

[Lewis 2008b]
Lewis, Grace, Morris, Edwin J., Smith, Dennis B., & Simanta, Soumya. *SMART: Analyzing the Reuse Potential of Legacy Components in a Service-Oriented Architecture Environment* (CMU/SEI-2008-TN-008). Software Engineering Institute, Carnegie Mellon University, 2008.
http://www.sei.cmu.edu/library/abstracts/reports/08tn008.cfm

[Lewis 2009]
Lewis, Grace A. "Is SOA Being Stretched Beyond its Limits?" *Microsoft Architecture Journal* (September 2009).

[Lhotka 2005]
Lhotka, Rockford. *A SOA Versioning Covenant*. SearchWinDevelopment.com. http://searchwindevelopment.techtarget.com/tip/0,289483,sid8_gci1277472,00.html (April 2005).

[Lublinksky 2007]
Lublinsky, Boris. "Versioning in SOA." *The Architect Journal 11* (April 2007). http://msdn2.microsoft.com/en-us/arcjournal/bb491124.aspx

[Martin 2006]
Martin, Evan. & Xie, Tao. "Understanding Software Application Interfaces via String Analysis," 901-904. *Proceedings of the 28th international Conference on Software Engineering (ICSE '06)*. Shanghai, China, May 20-28, 2006. ACM, 2006.

[Microsoft 2009]
Microsoft Corporation. *Windows Communication Foundation*. http://msdn.microsoft.com/en-us/netframework/aa663324.aspx (2009).

[Mittal 2005]
Mittal, Kuna. *Build Your SOA, Part 1: Maturity and Methodology*. http://www-128.ibm.com/developerworks/webservices/library/ws-soa-method1.html (May 2005).

[Mynampati 2008]
Mynampati, Prabhakar. *SOA Governance: Examples of Service Life Cycle Management Processes*. http://www.ibm.com/developerworks/webservices/library/ws-soa-governance/index.html (November 2008).

[Peltz 2004]
Peltz, Chris. & Anagol-Subbarao, Anjali. "Design Strategies for Web Services Versioning: Adapting to the Needs of the Business." *Web Services Journal 4*, 4 (April 2004). http://webservices.sys-con.com/read/44356.htm

[Ravichandar 2008]
Ravichandar, Ramya, Narendra, Nanjangud C., Ponnalagu, Karthikeyan, & Gangopadhyay, Dipayan. "Morpheus: Semantics-based Incremental Change Propagation in SOA-based Solutions," vol. 1, 193-201. *2008 IEEE International Conference on Services Computing*. IEEE Computer Society, 2008.

[Robinson 2006]
Robinson, Ian. *Consumer-Driven Contracts: A Service Evolution Pattern*. MartinFowler.com. http://www.martinfowler.com/articles/consumerDrivenContracts.html (June 2006).

[Sneed 2006]
Sneed, Harry. "Integrating Legacy Software into a Service Oriented Architecture." *Proceedings of the 10th European Conference on Software Maintenance (CSMR 2006)*. Bari, Italy, March 22-24, 2006. IEEE Computer Society, 2005.

[Sneed 2007]
Sneed, Harry. "Migrating to Web Services: A Research Framework." *Proceedings of the International Workshop on SOA Maintenance Evolution (SOAM 2007), 11th European Conference on Software Maintenance and Reengineering (CSMR 2007)*, Amsterdam, the Netherlands, March 20-23, 2007.

[Van Leeuwen 2009]
van Leeuwen, Michaël. *Versioning SOA Services*. http://blog.xebia.com/2009/08/03/versioning-soa-services/ (August 2009)

[Winter 2007]
Winter, A. & Ziemann, J. "Model-Based Migration to Service-Oriented Architectures." *Proceedings of the International Workshop on SOA Maintenance Evolution (SOAM 2007), 11th European Conference on Software Maintenance and Reengineering (CSMR 2007)*, Amsterdam, the Netherlands, March 20-23, 2007. IEEE Computer Society, 2007.

[Xiao 2007]
Xiao, H., Guo, J., & Zou, Y. "Supporting Change Impact Analysis for Service Oriented Business Applications." *Proceedings of the International Workshop on Systems Development in SOA Environments (SDSOA 2007), 29th International Conference on Software Engineering (ICSE 2007)*. Minneapolis, MN, USA, May 20-26, 2007. IEEE Computer Society, 2007.

[Zhang 2007]
Zhang, L., Arsanjani, A., Allam, A., Lu, D., & Chee, Y. "Variation-Oriented Analysis for SOA Solution Design." *Proceedings of the 2007 IEEE International Conference on Services Computing (SCC 2007)*. Salt Lake City, UT, USA, July 9-13, 2007. IEEE Computer Society, 2007.

[Ziemann 2006]
Ziemann, J., Leyking, K., Kahl, T., & Werth, D. "SOA Development Based on Enterprise Models and Existing IT Systems." *Exploiting the Knowledge Economy: Issues, Applications and Case Studies*. Edited by Paul Cunningham. IOS Press, 2006.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	March 2010	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
A Research Agenda for Service-Oriented Architecture (SOA): Maintenance and Evolution of Service-Oriented Systems	FA8721-05-C-0003

6. AUTHOR(S)

Grace A. Lewis, Dennis B. Smith, Kostas Kontogiannis

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2010-TN-003

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116	

11. SUPPLEMENTARY NOTES

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (MAXIMUM 200 WORDS)

Despite recent reports that it has failed, the reality is that Service-Oriented Architecture (SOA) remains the best option available for systems integration and leverage of legacy systems. The technologies to implement SOA will certainly evolve to address emerging needs, but its concepts will remain. To address those needs and concerns that SOA is potentially being stretched beyond its limits, a significant and coordinated research program is needed.

The SEI has developed an SOA Research Agenda with participation from a broad cross-section of the research community. The core of the agenda is a taxonomy that classifies topics into the business, engineering, and operations aspects of service-oriented systems, along with a set of cross-cutting aspects. Based on this taxonomy, the agenda outlines research areas, each of which is identified with its rationale, overview of current research, and delineation of research challenges and gaps. This report outlines the SOA Research Agenda. It also provides detail on specific research challenges related to the maintenance and evolution of service-oriented systems. The report concludes with a discussion of next steps in the evolution of the research agenda.

14. SUBJECT TERMS	15. NUMBER OF PAGES
SOA, service-oriented architecture, services, SOA research agenda, service evolution, service maintenance, service-oriented systems	40

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL