Service-Oriented Architectures: Myth or Reality?

Haresh Luthria, Digital Business Strategies

Fethi A. Rabhi, University of South Wales

// A study of SOA benefits reports a mixed reaction from the organizations that have implemented such systems, especially when service-oriented thinking fails at an organizational level, financial responsibility is misallocated, or mature tool chains are lacking. //

SURVEYS OF IT professionals worldwide indicate that knowledge and awareness of service-oriented architecture (SOA) is significant, with a majority of companies actively working on SOA initiatives.[1] There are also indications that the number of SOA projects "continues to grow in spite of tough times, or even because of tough times."[2] These findings mirror the general optimism in trade journals and magazines about SOA and Web services in particular as a popular choice for businesses looking for flexible systems development.

Services represent discrete business functions, and SOA facilitates their implementation as well as the communication between them and their consumers. SOA is often promoted as not only a technical architecture but also a design approach for enterprise environments that offers cross-platform compatibility, agility, and cost-efficiency. With so much hype surrounding SOA, it's not surprising that academics, vendors, and IT professionals interpret the concept differently (see the sidebar, "Service-Oriented Architecture and SOA Benefits").

The arguments favoring SOA are widely publicized, but little empirical evidence substantiates their claims. The relative abundance of analyst and vendor reports extolling the virtues of service-oriented computing stand in stark contrast to the few empirical academic studies of SOA's use in practice, indicating the need for scholarly benchmarks of SOA adoption.

Most academic studies focus on the technological innovations related to SOA. Only a few studies consider its practical adoption in an organizational context. Here, we present results based on a broad study across several organizations to better understand the challenges of realizing the benefits of SOA in practice.

Hit or Myth? We Report, You Decide

In 2008–2009, we conducted a rigorous study of a how a spectrum of companies use SOA in practice (see the sidebar "The SOA Case Study Protocol").[3] We wanted to examine issues that influence SOA adopters—from a business unit's understanding of SOA at the conceptual level to the industry's lack of standards at the implementation level. Here, we extend the study's results by focusing on interviewee perceptions of constraints and issues related to SOA implementation from a practitioner's perspective. We can then map interviewees' perceptions to SOA organizational benefits that appear in the academic literature (see the sidebar, "Service-Oriented Architecture and SOA Benefits").

Visibility into Business Flows

Many of our study's interviewees saw a challenge in clearly articulating what SOA meant, with vendors providing views that differed from senior management, business owners, and technologists.

To senior executives, services were

0740-7459/12/$31.00 © 2012 IEEE

SERVICE-ORIENTED ARCHITECTURE AND SOA BENEFITS

The term "service-oriented" is used in various contexts for different purposes. When it's used together with "architecture," it has a technical connotation of decomposing complex program logic into smaller, autonomous, distinct units.

SOA: A DEFINITION

A *basic service* represents a discrete business function implemented in software and wrapped with a formal, documented interface that doesn't depend on how other services operate and that standards-based communication mechanisms can locate and access.[1] A *composite service* assembles simple or other composite services to implement broader business functions.

The concept of a service-oriented architecture (SOA) includes a set of desirable design characteristics for promoting interoperability, reusability, and organizational agility as well as a service-oriented business process-modeling paradigm.[2] The term *SOA* is now commonly used to designate anything contributing to an enterprise platform based on service-oriented principles.

ORGANIZATIONAL BENEFITS

The academic and technical literature suggests several benefits offered by SOA approaches to system design.

Visibility into Business Flows

Effective SOAs tend to be well-defined, process-centric architectures that support process design and monitoring more from a business perspective than a systems perspective. This makes design easier, along with the automation, monitoring, and modification of business processes.[2] It also helps identify business services that constitute an organization's core competencies. Services that aren't core competencies can then be candidates for services in which vendors offer relevant expertise.[3]

Plug-and-Play Infrastructure

Transforming an enterprise's business processes to services using standards-based communication protocols opens new avenues to strategic partnerships with suppliers, partners, and customers.[3] In turn, a new business model emerges based on rebundling intra- and interenterprise business processes as seamless services.[2]

Leveraging Legacy Systems

A service-based approach to system design allows existing and proven legacy system functions to be encapsulated as services on a new standards-based integration platform.[2] The services can encapsulate single functions or comprise several smaller services that represent legacy functions on diverse hardware and software platforms.[2,4]

Rapid Development, Reduced Costs

Over time, developed services become an organization's core asset—a library of tested, ready-to-use, compatible components.[2] This potentially reduces the time to pull well-tested functionality together to meet new market needs. Potential reductions in development and testing costs can increase service modularity and potential reuse.[4] In addition to offering cost efficiencies, reusing existing components also reduces risk by limiting the introduction of new potential points of failure.[2]

References

1. T. Erl, *Service-Oriented Architecture: Concepts, Technology, and Design*, Prentice Hall-Pearson Education, 2005.
2. K. Channabasavaiah et al., "Migrating to a Service-Oriented Architecture," *IBM Developer Works Tech. Report Series*, 2004; www.ibm.com/developerworks/library/ws-migratesoa.
3. F. Curbera, "The Next Step in Web Services," *Comm. ACM*, vol. 46, no. 10, 2003, pp. 29–34.
4. M.N. Huhns and M.P. Singh, "Service-Oriented Computing: Key Concepts and Principles," *IEEE Internet Computing*, vol. 9, no. 1, 2005, pp. 75–81.

interchangeable with channels or business product offerings—that is, with what companies offer their customers. Technologists, on the other hand, considered Web services to be synonymous with service orientation. Business owners responsible for defining business processes were quick to label any concept that includes the word "architecture" as a purely technical concept. Consequently, no service-oriented thinking appeared at the business unit level. This limits service orientation to the technical domain, where technologists push the SOA case for modeling business processes as Web services.

Most of the firms we studied recognized process modeling as critical to successful SOA implementations, with a few of them adopting business process modeling (BPM) as a strategy. Some firms had purchased BPM tools and tried to implement modeling but couldn't sustain it. In addition to the

THE SOA CASE STUDY PROTOCOL

We used a qualitative case study approach to this research,[1] conducting semistructured interviews with key decision-makers at 15 firms. Table A summarizes the organization profiles and interviewee roles within each firm. The firms were in various stages of implementing SOA, either for themselves or their clients. Most of them already had experience migrating targeted business functions to a service-based deployment. We knew that firms 1, 10, and 11 had unsuccessful SOA implementations. We contacted the remaining firms from their involvement in three industry SOA conferences where representatives had presented aspects of their SOA implementations.

We asked a broad set of questions addressing implementation details, challenges and concerns, benefits realized, and lessons learned to guide each hour-long interviews. Next, we analyzed transcripts of the individual interviews using a methodologically rigorous, two-phase thematic method. The first phase involved multiple passes of the data to identify discrete data segments with relevant semantic meaning and tag them as concepts. In the second phase, we grouped the coded concepts into broader categories that reflected recurring themes in the data. Finally, we reviewed these themes to identify similar patterns across the data from all the firms interviewed.

Case study research has some generally accepted limitations, specifically those relating to rigor and generalization of results. This study attempted to mitigate these limitations by ensuring contextual richness in the collected data and the inclusion of multiple cases, such as firm 10, that didn't consider SOA viable. In addition, we designed and used a structured case study protocol and maintained a chain of evidence, including emails, and a case study data repository. To ensure that the study didn't just investigate what the conceptual research model represented, we extracted all results strictly from the data, providing a high level of confidence that the concepts identified were strongly linked to the data and weren't being fit into a preconceived pattern.

Our approach to identifying firms and interviewees also has a potential limitation because of potential bias. We tried to avoid bias by selecting three firms that we knew had unsuccessful SOA experiences and by selecting firms that presented different perspectives on what they asserted were successful SOA deployments. Selecting firms that presented at three separate conferences also allowed for some randomness in the sample.

Reference

1. H. Luthria, "The Organizational Diffusion of Service-Oriented Computing: Investigating the Realization of Business Value from Service Oriented Architectures," PhD thesis, Australian School of Business, Univ. of New South Wales, Australia, 2009.

TABLE A

A snapshot of the organizations interviewed.

Firm	Industry sector	Interviewee(s)
1	Banking	Head of strategy
2	Banking	Business development executive; technical architect
3	Banking	Business development executive
4	Banking	CIO
5	Banking	Enterprise architect
6	Banking	Enterprise architect
7	Insurance	Two technology managers
8	Insurance	CTO
9	Insurance	CIO
10	Insurance	Enterprise architect
11	Software products and services	CTO; VP of strategic accounts
12	Software products and services	Technical architect
13	Software products and services	Technical architect
14	Software services	Two technical architects; product manager
15	Software services	Principal of practice

sheer complexity of the tools, the firms we studied cited a lack of comprehensive functionality for handling the complex realities of systems development. Available BPM tools and languages support the initial development cycle well, enabling the initial description of business processes as models and the simulation of various business scenarios. However, any changes during the development cycle involved nontrivial changes to the models.

We can broadly characterize the management process for defining business flows used by the firms we interviewed that had purchased BPM tools. First, analysts define the business processes diagrammatically using a variety of tools, usually Visio or Excel spreadsheets, and in some instances, such as business requirements, a word-processing package. Systems analysts then import the diagrammed flows and requirements into the BPM tool as documentation. However, the translation of these models into use cases or technical requirements is performed manually, and mapping upstream business processes to downstream technical services is maintained manually. Consequently, the firms used their BPM tools primarily as static repositories. Can such use of BPM tools really enable better business process monitoring and accelerate process improvement?

Plug-and-Play Infrastructure

At an implementation level, the study interviewees were uniformly skeptical of the promise of plug-and-play, referencing the lack of industry-level SOA standards. The firms treated each integration effort, either across internal business applications or across business partner domains, as a one-off project—even when many of their systems were implemented as Web services.

Two of the firms we interviewed were initiating integration efforts after mergers. They found that even adopting IFX (www.ifxforum.org), an open standard for financial data exchange defined by a forum of business and technology professionals, didn't reduce the challenges of disparate data context or semantic relevance when integrating synergistic business systems.

After acquiring a partner bank, a third firm had to settle for running two separate systems, maintaining two distinct user accounts despite an effort to install a service-oriented connectivity infrastructure. In addition to data format variations among individual firms, the lack of service standards in the vendor space led to tough development decisions. Some firms had to back away from using best-of-breed solutions for certain business problems, relying instead on developing or modifying solutions available for their current vendor platforms.

Firms in the study also questioned the practicality of implementing services that span partner domains. Breaking applications into discrete services could cause ownership issues when they span boundaries—even internal departmental boundaries—or when the applications are merely shared across business domains. If a service is created for a business unit and other units can use it, there is considerable debate over how to maintain the service in the future. Will the business unit that created the need for the service own it, or should the common IT infrastructure team own it? If a specific user's needs change, should the infrastructure team change the common service or customize another incarnation of it?

Identifying business ownership is critical not just for easier maintenance but more significantly for compliance with industry regulations. A loose confederation of services could pose access and security challenges not just across external domains but also within internal domains because of contextually dependent security rules. According to the CIO of one firm in our study, "It's hard enough to identify the business owner of an internal account or application. Imagine doing that for a business service." Although such issues have come up in earlier modular or component-based approaches, services exacerbate them because higher-level composite services can cross business domains. Firms must keep this in mind when considering a model that promises seamless plug-and-play with upstream and downstream business partners.

> How does the migration to a service-oriented environment impact the significant value invested in legacy systems?

SOA might support a plug-and-play infrastructure in theory, but most firms we interviewed were implementing it with home-grown or preferred-vendor standards, which varied even within organizations depending on the business system platform. The glut of standards and the lack of widely accepted ontologies create undue complexity. Consequently, the goal of easy integration, even internally, seems difficult at best and unachievable at worst.

Leveraging Legacy Systems

Beyond the variation in interface protocol and data interchange standards, critical design-related issues and questions also prove challenging to enterprise implementation of SOA—for example, how does the migration to a service-oriented environment impact the significant value invested in legacy systems?

The software service providers we

interviewed advocated exposing key business functions implemented on proven legacy systems as services over time, one by one, thus providing a safer and less expensive migration path to a service-oriented infrastructure. Their approach to migrate their clients consisted of wrapping enterprise application integration (EAI) or related middleware frameworks with service interfaces to effectively create an enterprise service bus (ESB). Across the remainder of the firms we interviewed, however, there appeared to be no clear or generally accepted way to leverage proven legacy systems in a service-oriented environment. Most were still grappling with how to use legacy assets, with many taking the easy way out by rewriting legacy business functions as Web services. One of the software service providers we interviewed cited the example of a client bank that had taken three years to build a new service-based loan system and "scrapped its old loan system… [that] had been in use for 12 years." SOA proponents encourage the use of existing legacy assets, but this appears to be easier said than done.

> In short, it isn't easy to determine how broad or narrow a service's scope should be.

Rapid Development, Reduced Costs

Services aren't typically reused for rapid development or cost reduction. Even firms that appeared to have a significant and relatively successful implementation of service-oriented systems across the enterprise were strongly concerned about the lack of reuse for three reasons.

First, the smaller the service scope, the harder it seems to manage, because applications using the service could be of varying levels of criticality to the firm. The CIO of one firm we studied highlighted an example: a seemingly benign calendaring function created as part of a low-priority meeting scheduler application was introduced into the service catalog. When the service encountered a failure, its owners weren't available to fix it because the owning application wasn't "tier 1," meaning it wasn't high priority. A mission-critical application that used the service had to contend with extended downtime as a consequence. Although this case seems particularly extreme, other firms in our study echoed similar concerns.

Second, delivery pressures reduce developers' risk tolerance because of the lack of visibility into existing services' feature capability and code quality. Reuse of services by different users or by higher-order services could potentially compromise security as well by changing the context in which the service is used.

Third, according to the managers and architects in our study, developers are motivated by the charge they get out of creating something new. Consequently, they tend to avoid reuse, preferring instead to recreate a service rather than use an existing one developed by someone else.

Nevertheless, the technical teams in this study advocated keeping services limited in scope to encourage reuse. The coarser a service is, the smaller the chance to reuse it in different business contexts—conversely, the smaller the services' scope, the harder it is to deal with service quality issues such as performance and transactional integrity, as some firms have experienced. Performance could also be impacted if services are too fine-grained because network connectivity between services could cause unpredictable response times. If the service is too coarse, it could compromise modularity. Similarly, the rollback of database updates across services was a challenge, forcing services to be designed to minimize the loss of transactional integrity. Consequently, there's considerable confusion about optimal service granularity among most of the firms in the study, with developers taking the easy way out and recreating or rewriting service functionality. The resulting glut of redundant services causes versioning and integrity issues.

A service's optimal granularity or scope requires a trade-off between its potential for reuse and better management of quality attributes. In short, it isn't easy to determine how broad or narrow a service's scope should be.

What SOA Practitioners Said They Needed

Our study provided a unique insight not only into SOA's practical challenges but also into what practitioners believe they require to achieve its advertised benefits.

Service Ecosystems

Many of the firms in our study indicated that what could help in using SOA to integrate with business partners and in creating widely accepted standards is the establishment of industry-level "service ecosystems." Such an ecosystem would comprise a group of partner organizations within an industry discipline sharing their services for use across the group. Until such time that such broader cooperative efforts are taken on, the vision of a seamless and flexible bundling of intra- and interenterprise business processes as services will remain elusive.

Simpler Business Process Modeling Tools

Practitioners look to vendors to provide

process-modeling tools with comprehensive but simple functionality. The BPM-purchasing firms in this study found that the sheer complexity of the vendor tools required extensive technical training that daunted even their technology teams, let alone the nontechnical staff. This caused business units to view modeling as a technical activity, pushing the responsibility of modeling business processes as services, which should arguably be borne by business units, to technology teams. The latter, in order to translate business processes to representative services, then had to be trained on the business aspect of services (a nontrivial effort) in addition to taking on the technical training related to using the tools. Bridging this implementation gap has proved to be one of the more significant challenges for the firms in this study.

ABOUT THE AUTHORS

HARESH LUTHRIA is a principal at Digital Business Strategies, where he consults on IT projects—primarily with financial services companies. His research interests focus on integrating organizational management with IT. Luthria has a PhD in information systems from the Australian School of Business at the University of New South Wales. Contact him at h.luthria@digitalstrategies.biz.

FETHI A. RABHI is an associate professor in the School of Computer Science and Engineering at the University of New South Wales. His research interests are in service-oriented software engineering with a focus on business and financial applications. Rabhi has a PhD in computer science from the University of Sheffield. He's a member of the Australian Computer Society and the IEEE Computer Society. Contact him at f.rabhi@unsw.edu.au.

Rigorous Governance

Governance of which services should be created, not just how they're created, is very important. This is because the technical design and implementation of business processes as services hasn't proved that simple. Emerging case studies support this observation.[4] Moving to a SOA shouldn't necessarily require replacing existing legacy systems, but there are no design guidelines to help leverage legacy applications. Until there are successful examples and established design patterns, organizations might be misled into rewriting their core applications as Web services, just as many of the firms in our study did.

Service reusability is touted as a way to reduce development costs and enable rapid product development. But it's important to understand that reusability isn't an attribute specific to services. All the modular approaches to system development that preceded service-orientation enabled reuse. Business pressures, always hard to ignore, are a disincentive to reuse in multiple ways. Reuse in practice isn't purely enabled by technology but is a function of organizational culture and strong governance to encourage, and even enforce, reuse.

SOA Awareness and Training

SOA users require appropriate awareness of what SOA is and what it is not. In our study, we found that business owners tended to simplify the significance of the service concept and treat enterprise offerings as service offerings, delegating the task of BPM to their technical teams. Developers working on SOA initiatives, on the other hand, needed a tempering of expectations in terms of viewing SOA as a leading-edge technology skill. Technically inclined architects and developers tended to jump on the SOA bandwagon to increase their technical skills, but were disappointed when they realized that most of the effort in implementing SOA lies in process engineering and implementing appropriate governance structures. Consequently, technical people weren't too keen on implementing SOA at an enterprise level, limiting their service solutions solely to the technical domain. Business and technical units, it appears, need to change their current mindsets, move their thinking more toward the center, and view service-orientation as a joint business and technology solution.

In general, firms struggle to realize the full potential of large IT investments,[5] and the adoption of SOA as an enterprise strategy is a significant and nontrivial investment as companies upgrade their business and technology infrastructures.[6]

Most of the available literature on SOA is broadly optimistic, leading potential users to mistakenly assume that moving to it is a relatively small step. This general misconception was well captured by the application architect for one firm in our study. Talking about the academic viewpoint and his business team's expectations for SOA, he said, "They think, 'Buy SOA, and you're set. All your problems will be solved.' This is not so."

We aren't suggesting that SOA isn't all that it's made out to be. In fact, the analytical arguments promoting it are quite sound in describing its potential benefits. But there are some constraints and issues associated with its implementation. SOA neither exacerbates nor alleviates the lack of proper systems engineering, project management, and program governance. Potential users shouldn't assume that the burden of such discipline can be eschewed just because the processes have been defined as granular services. SOA is an approach to building business systems that requires significant investment in process management and governance mechanisms for the service life cycle, and it might not necessarily be an instant solution to all an enterprise's business problems.

References

1. J. Ladley, "SOA Survey Results," *Enterprise Data J.*, 2010; www.enterprisedatajournal.com/article/soa-survey-results.html.
2. J. McKendrick, "Forrester SOA Survey: Seven out of Ten Very Happy, Plan Expansions," *ZDNet*, 2010, www.zdnet.com/blog/service-oriented/forrester-soa-survey-seven-out-of-very-happy-plan-expansions/5069.
3. H. Luthria and F.A. Rabhi, "Organizational Constraints to Realizing Business Value from Service Oriented Architectures: An Empirical Study of Financial Service Institutions," *Proc. 6th Int'l Conf. Service Oriented Computing* (ICSOC 08), Springer, 2008, pp. 256–270.
4. D. Krafzig, K. Banke, and D. Slama, *Enterprise SOA: Service-Oriented Architecture Best Practices*, Prentice Hall PTR, 2005.
5. H.C. Lucas, *Information Technology and the Productivity Paradox: Assessing the Value of Investing in It*, Oxford Univ. Press, 1999.
6. N. Bieberstein et al., "Impact of Service-Oriented Architecture on Enterprise Systems, Organizational Structures, and Individuals," *IBM Systems J.*, vol. 44, no. 4, 2005, pp. 691–708.