The ROI of SOA

Based on Traditional Component Reuse

By Jeffrey Poulin, Ph.D. and Alan Himler, MBA

The ROI of SOA Relative to Traditional Component Reuse
Initial Quantified ROI on SOA Initiatives

The rationale of this white paper is to present the costs and savings of Service Oriented Architectures (SOAs) compared to traditional component-based software development. SOAs are quickly becoming a significant influence in the mainstream of all industries. Although numerous sources expound on the technical advantages of SOAs as well as listing praises for their intuitive and qualitative benefits, until now no one has provided reliable and quantifiable results from SOA implementations currently in production.

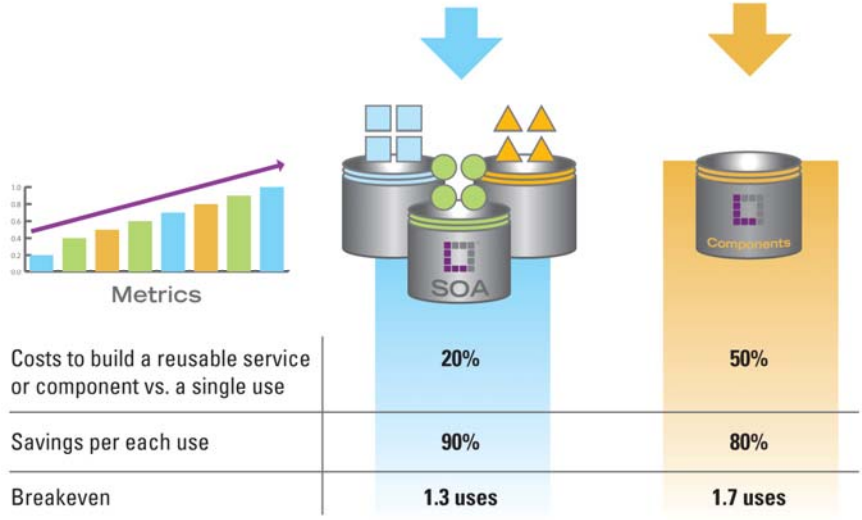
To seek out answers, LogicLibrary surveyed "early-adopter" Logidex™ clients who are implementing large-scale SOAs. Their responses show significant cost savings over the "traditional" non-SOA approach to reuse as follows:

1. Building components for an SOA requires *only* ~20% additional investment over development for one-time use, compared to the ~50% additional investment required to build traditional reusable components.
2. After an initial learning curve, the development costs for reusing an SOA service are about ~½ of the costs required for traditional component-based development and integration, which represents a ~90% total savings over development from scratch.
3. The level of reuse in a typical SOA application increased to 25% from an average of only 10% in a traditional component-based application, which drives significant cost savings on future SOA projects.

This paper answers the questions that executives ask before investing in a new technology:

"How much money will I *really* save?"

"What is the ROI?"



In short, after a small investment in SOA organization, process, and tools, **SOAs cost 20% less to implement and save 50% more with each reuse** than traditional component-based development. In addition to consistently observed (but hard-to-quantify) increases in quality and productivity, **the level of reuse in SOA development averaged 2.5 times more than non-SOA development**, leading to measurable development cost reductions in the surveyed organizations.

The ROI of "Traditional" Reuse

To understand and compare the business advantages of an SOA to traditional software development, we start with a quick review of the reuse investments required and the benefits returned prior to the introduction of SOAs. First, reuse is not "free." Setting up the people, training, processes, tools, and components that fit into that architecture requires an initial investment and commitment from the development organization. We call this investment the "Relative Cost of Writing for Reuse (RCWR)." Based on data collected over the past 15 years, this investment is approximately 1.5 times (meaning 50% more) cost over building software for one-time use.[1]

Second, reusing components has benefits both during development and support ("maintenance"). As with RCWR, data shows that the "Relative Cost of Reuse (RCR)" during development is only 20% of the cost of development without reuse; i.e., traditional reuse provides an 80% development savings.[1] The maintenance savings over the life of the reused product can be estimated in several ways, such

as by calculating the historical cost to fix an expected number of errors in deployed software. The sum of these two benefits is the total cost avoidance of reusing software.

The ROI of Reuse with SOAs – Breaking Down the Survey Results

This survey seeks to quantify how the established "Relative Costs" of reuse change with the adoption of SOAs. To do this, we surveyed LogicLibrary clients who are in various stages of deploying large-scale SOA implementations. Although the number of respondents does not make the results statistically significant, the data begins to form a consistent picture about the costs and benefits currently experienced in industry.

Question 1: *How much effort does it take to build a typical SOA service compared to a functionally-equivalent component for just one-time use?*

Responses ranged from an additional 15% effort up to 2x effort, with a median at 20% additional effort. Although the range is similar to that for traditional component-based development (where RCWR=1.0-2.2), the SOA median is 20% below the additional 50% effort typically experienced prior to SOAs. An RCWR=1.2 for SOA is also consistent with and substantiates our early observations of 20-40% additional effort to build SOA services. [3]

Question 2: *What cost savings (if any) do you experience with using services compared to building and deploying an equivalent component?*

The responses ranged from 20% to 85% savings; e.g., RCR= .15-.8. This is a much wider range than for non-SOA development, where RCR= .03-.4. In part, this is because two of the respondents were early in their respective implementations and indicated that they were not yet over the expected learning curve and therefore were just beginning to experience the benefits of SOAs. In addition, the respondents may have interpreted the term "build" as either "to write the code for" or simply "to compile," as it would be used in an SOA environment. Taking this into account, the realistic range RCR with SOA is .15-.5, which suggests that the cost of reusing SOA services is (conservatively) approximately ½ the cost of reusing components in traditional software development.

Question #3: *How much effort do you save (if any) when integrating SOA services compared to integrating an equivalent component?*

The responses ranged from "additional 10% effort" (this was again attributed to the "learning curve") to savings of between 25% and 80%, with a median of 50% savings compared to component-based development. This result strongly supports the assertion that the ease of integrating services is one of the primary drivers of cost savings with an SOA.[2]

Taken together, the responses to questions #2 and #3 suggest that RCR=.1 with SOAs. Furthermore, this conclusion is consistent with our first quantifiable result with SOAs of RCR=.08.[3] Table 1 summarizes these values, showing the relative costs of reuse in SOAs to the cost of building software (1) without reuse and (2) to the cost of development with traditional component reuse.

Traditional Reuse	Additional investment to build each Component		Savings for each Reuse of the Component	
	RCWR	% cost	RCR	% savings
Without SOA				
Traditional component-based development vs. No Reuse	1.0-2.2; median=1.5	0% to 220%; median=50%	.03-0.4; median=0.2	97%-60%; median=80%
With SOA				
SOAs vs. Traditional component-based reuse			.15-.8; median=.5	95%-20%; median=50%
SOA vs. No Reuse	1.15-2.0; median=1.2	15%-200%; median=20%	.1	90%

Table 1- Costs and benefits with and without SOAs

The values of RCWR=1.2 and RCR=.1 with SOAs quantify the effort required to develop the services to populate an SOA and the savings when subsequently reusing those services. Because we do not yet have enough data on how SOAs affect maintenance costs, that portion of the ROI model remains unchanged, with estimated maintenance savings based on historical data.

The next two questions focused on how SOAs affected the level of reuse practiced by an organization:

Question #4: *What percentage of reuse is in one of your typical legacy applications?*

Responses ranged from 0% to 20%, with an average of 11.25% and median of 10%.

Question #5: *What percentage of reuse is in one of your typical SOA applications?*

Responses ranged from 10% to 60%, with an average of 26.25% and median of 25%.

The responses show a 15% increase in reuse level in a typical SOA application as compared to a non-SOA application, which is an astonishing 2.5x improvement. Assuming a 90% savings for each reused service, this by-product of SOAs would by itself reduce the total development cost of a typical SOA application by 13.5%.

Question #6: *How does the quality of reusable SOA services compare to the rest of the SOA application?*

All responses supported the observation that reusable SOA services are of higher quality than services constructed for one-time use, as measured by one of (a) defects found during development, (b) defects found during production, (c) response time, and (d) availability. Although traditional reusable components typically enjoy 10x higher quality than non-reusable components (as measured by defects per deployed line of code) there was insufficient data to make a similar conclusion about reusable SOA services.

Question #7: *How does SOA affect development productivity compare to pre-SOA development?*

All responses supported higher productivity through the use of reuse of existing services, standards-base integration, and the ability to focus on new capabilities rather than reinventing core services. This productivity increase came, however, after an initial productivity loss required to write and to deploy the reusable core services as well as a short “learning curve” needed to understand how to deploy applications in an SOA. The productivity gains were measured by comparisons to historical product development efforts, but there was insufficient data to quantify the effect of SOAs on productivity.

Question #8: *How does SOA affect time-to-market compared to pre-SOA development?*

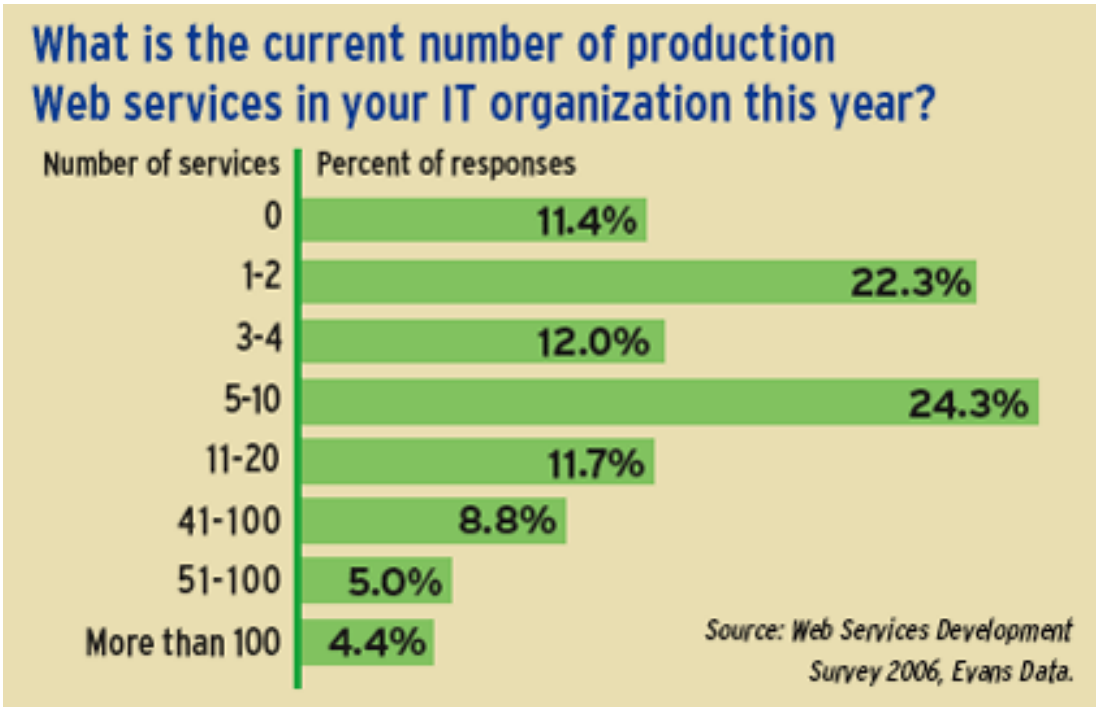
As with questions #5 and #6, responses supported the claim that SOA shortens time-to-market, as measured by historical development time lines and by comparing first-time development effort with subsequent integration efforts. As with the previous questions, there was insufficient data to quantify the effect of SOAs on time-to-market. After allowing time for additional experience with SOAs, we intend to repeat these three questions and report on actual results.

Question #9: *What cost savings does your organization gain (if any) from SOAs compared to pre-SOA development?*

The responses to this question are dependent on the size of each SOA application as well as the number of applications the organization has built since adopting SOAs. Therefore, although answers spanned a wide range ($500k to $1.75m), SOAs are clearly responsible for early and significant returns in organizations that support SOA architecture, processes, and tools.

Question #10: *How many total reusable services do you currently manage in your reuse library?*

Responses ranged from 10-60 services, with an average of 26 services and a median of ~20 services. While this may seem like a small number of SOA services, it is consistent with the "best practices" that we have seen with SOAs; namely, that a small collection of carefully-selected core services is sufficient to support significant cost savings and is preferable to larger, less well-selected collections of services.



Conclusion

The results of this survey not only strengthen the business case for reuse, they validate and quantify the benefits of SOAs over traditional component-based development. These survey results attach hard numbers to the more qualitative of SOA including; reducing integration expense, increasing asset reuse, increasing business agility, and reduction of business risk. We have found from this initial survey that SOAs:

1. Require only 20% additional cost to build core reusable services (e.g., RCWR=1.2)
2. Save 90% of development costs for each reuse of a SOA service (e.g., RCR=.1)
3. Quickly increase levels of reuse by 2.5x on applications.
4. Lead to further, as yet to be quantified savings in quality, productivity, and time-to-market.

Given these results, SOA enjoys a positive Return on Investment (ROI) with only 1.3 uses, compared to an ROI after 1.7 uses of a component in traditional development (assuming a RCR of .2 and a RCWR of 1.5). The resulting ROI model provides powerful motivation to invest in SOAs, especially whenever two future development efforts must develop similar services.

About the Authors

Dr. Jeffrey Poulin, System Architect at Lockheed Martin Systems Integration, Owego, NY.

Dr. Jeffrey Poulin has over 14 years of experience as the technical lead on large systems development and integration projects. He currently serves as Systems Architect at Lockheed Martin Systems Integration. Dr. Poulin's landmark work on the business case for software reuse forms the basis for the reuse measurements and return on investment (ROI) models used by industry and government organizations worldwide. In addition, he has authored over 70 technical publications, including Measuring Software Reuse: Principles, Practices, and Economic Models, a book published by Addison-Wesley.

Alan Himler, Vice President of Product Management and Marketing at LogicLibrary, Pittsburgh, PA

As vice president of product management and marketing at LogicLibrary, Alan is responsible for product strategy, direction and introduction of new products. He works closely with customers' in creating and presenting pro-forma ROI analysis concerning component reuse and SOA projects.

References

[1]Poulin, Jeffrey S., Measuring Software Reuse, Addison-Wesley, Reading, Massachusetts, 1997.

[2]Marks, Eric and Michael Bell, Service-Oriented Architecture (SOA): A Planning and Implementation Guide for Business and Technology, John Wiley and Sons, New Jersey, 2006.

[3]Carlson, Brent and Jeffrey Poulin, "Measuring Reuse ROI," Java Pro Live!, San Diego, CA, 12-14 September 2005.