



Topic	Change Governance for the Agile Enterprise – a Service Oriented Architecture (SOA) Perspective
<i>Document</i>	<i>White Paper</i>
Author(s)	Kelly A. Shaw, Ph.D. Research Analyst, Serena Software Brent Carlson, VP of Technology, Logic Library
Date	8/8/2006
Version	1.0
State	Final

Copyright©2006 Serena Software Inc. All rights reserved.



Serena Research Document

Now that the hype has started to die down it is time for organizations to get serious about moving towards SOA-based development. How? By harnessing change while maintaining control over their mission critical applications as they adopt the new architecture – a task easier said than done. These organizations don't just need tools, they need processes, policies and procedures that will help them transition legacy applications and deploy new applications to take advantage of SOA's promised agility and cost savings. SOA isn't magic, and adopting SOA isn't a magic bullet. It requires hard work, planning and an understanding that SOA adoption can be destructive without a strong governance framework.

SOA Promises Agility for the Enterprise

Organizations need agility to maintain strategic advantages in businesses operating on faster and faster time-scales. The difference between gaining and losing market share may very well depend on the ability of organizations to deploy updated or new applications before their competitors. IT is adopting SOA as a means of achieving nimbleness not currently available in more monolithically architected applications. Besides agility, SOA-based application development also holds the promise of reduced application development and maintenance costs through service reuse. So not only can organizations deploy applications more quickly, they can reduce the cost of development and maintenance at the same time.

Agility Requires Strong Governance

Beware: SOA can and does introduce more chaos into the application development process. This is especially true if agile development methods are adopted in conjunction with the new architecture. Applications using SOA are inherently more complex than their counterparts using more traditional development architectures. Not only do applications built using SOA have more moving parts, they may make use of services delivered from anywhere in the world by 3rd party vendors not directly under the control of IT.

Many IT Operations vendors currently touting their SOA Governance solutions concentrate on runtime - that is managing the performance and consumption of services in production. While runtime governance is necessary, it is not sufficient if organizations want to take advantage of SOA's full promise. Governance must start in the design phase, continue through development and carry on through production. IT must recognize that architecture and design-time governance is essential to prevent SOA initiatives from turning into ABOS (A Bunch of Services) – point-to-point services that don't solve business problems and are not reusable.

This can wreak havoc within IT development organizations. First, recent studies show that the cost of building a single application using SOA is often greater than the cost of building the same application without SOA. Only through service reuse do these development costs pay off. Second, because SOA-based applications have more moving parts and may depend on services not under the control of IT, ensuring quality of the running application is extremely difficult. Finally, depending on complex applications where some parts are not under the direct control of IT is risky.

The promise of agility and the cost savings due to reuse will come to nothing without strong governance policies and procedures in place. Note the comments by leading analysts regarding SOA implementation and governance.

Ray Wang, Forrester Research. [11 Entry Points To SOA: How To Prepare For Upcoming ERP Trends In A World Of SOA.](#)

Slow down and make sure strong governance and change management policies and procedures are in place before making the leap to SOA-based development.

Dennis Gaughan, AMR. [A Framework Approach to SOA.](#)



When putting a SOA framework together, Governance is the most important element of the framework. Organizations don't have a viable SOA framework without Governance.

From Gartner:

“Governance is critical to the success of an SOA initiative. In 2006, lack of working governance mechanisms in midsize to large (greater than 50 services) post-pilot SOA projects will be the most common reason for project failure.”

Jason Bloomberg, ZapThink. [SOA Governance and the Butterfly Effect.](#)

Due to the butterfly effect, properly implemented SOA may be just as dangerous as SOA implemented poorly. Put strong governance in place *across the board* (CG, not just SOAG) before attempting to implement SOA else the butterfly will flap its wings and in HQ you have disaster instead of agility.

Serena Software is the leader in Change Governance. We can help you put the policies and procedures in place that will tame the chaos associated with SOA-based development. You can visualize your application to make sure your development staff builds the right application with the right services. There is no room for application rework in an agile enterprise. You can orchestrate your application development process to make sure developers have the tools and services available to do their job quickly and efficiently. Finally, you can enforce the policies and procedures necessary to make sure production changes in an SOA environment are managed and controlled.

How do we make your business more agile?

Visualize the application you want to build and the services you have available

Serena Composer enables business analysts to express application requirements and changes in their own language: visually. Business analysts generally use text documents, whiteboards, notes and screen shots to define how they want applications to behave. With Serena Composer, the business analyst can define the application they really want, and then hand it over to IT without the long and drawn-out process of translating the model in their head into something consumable by IT.

IT can then take the application model and start filling in the technical details with actual calls to existing web services, or with placeholders for services that need to be created. Using Composer's scenario editor, IT can demonstrate the application prototype to business. No wasted effort translating pictures to words to functional specifications to service requirements to API specifications to detailed designs to... Instead, business and IT speak the same language to communicate application requirements.

This won't work for organizations using SOA-based development unless developers and application designers have access to the services; understand where they are, what they do and how they work. IT needs to visualize not only the application, but the services available and necessary to build the application.

Serena's partner, Logic Library, is a leading provider of software and services that help organizations manage their web services. Their flagship product, Logidex, is a design-time repository and registry that helps enterprises manage, reuse and govern web services. With Logidex, IT can browse or search for services and import the service definitions right into the Composer models. Additionally, because Logidex has a tight integration with the most widely used IDEs, IT can do the same imports directly into the source code at development time. This not only promotes reuse of services, but enables IT to develop the right applications more quickly.

Logidex also promotes agility in application design and development by making it easy for

enterprise architects to provide oversight of SOA-based projects without placing undue burdens on either the development staff, or the EA staff. By providing a design-time repository of the applications and services used within an SOA initiative, Logidex enables members of the EA team to provide oversight and design-time governance without requiring time-consuming design documents from the development team. This will help organizations put in effective design-time governance policies and procedures that do not delay or otherwise interfere with tight development schedules.



Orchestrate your development process to be fast, efficient and dependable

Once you have visualized and defined the application you want to build, the next step is to build the application. In an SOA-based development environment you must be able to identify the existing services you want to use, services that must be re-factored to be reusable and services you must build from scratch. Serena Dimensions and LogicLibrary Logidex together help you orchestrate this process to make application development fast and efficient.

Accessing existing services. Logidex has integrations with most of the popular IDEs used in development today. Developers can search for the services they need to use and pull the call directly into the source code whether these services were internally developed, purchased or even reside outside the firewall. This promotes service reuse and pays off one of the benefits of SOA-based development: lower application development costs. Besides lower costs, reusing existing services improves quality since pre-tested application fragments would be used rather than potentially error-prone code written from scratch.

Re-factoring existing services to promote reuse. In many development efforts, especially at the start of an SOA initiative using agile methods, services may be built bottom-up using a project-based mentality. In a perfect world all services would be defined and built with reuse in mind. However, the time constraints associated with agile development methodologies may limit a development team's ability to make service reuse a priority. This may result in a number of services that have the potential to be widely used, but need a re-factoring cycle to bring them into alignment with the requirements of other development projects.

Developing or re-factoring services. Regardless of whether an existing service needs to be re-factored or a new service created, Dimensions, in conjunction with Logidex, can be used to manage the source assets and service metadata for the service. Developers using Dimensions can manage the required changes using change documents that describe the interface and behavior of the service. Using the Dimensions integration with Logidex, the metadata and WSDL for the modified service can be sent to the service metadata repository along with its current development status. The developer implementing the web service can not only use Dimensions to manage the source assets associated with the service, but as the developer actions the change document, the status of the service in Logidex would be updated, keeping the proposed user of the new service informed of the development status. Finally, when the service is ready to be deployed to testing, Dimensions can alert Logidex to register the service with UDDI. At the same time, Logidex can send notifications to interested users that the service is available.

Together, Dimensions and Logidex allows organizations to manage the service development and re-factoring process to make development fast, efficient and reliable.

Enforce your policies and procedures so agility doesn't turn into disaster

Agility is not a license to hack. Agile methods work because everyone knows and understands what needs to be done when and by whom. It also means that strong governance must be in place to stop inadvertent mistakes from causing disasters that could stop innovation and agility in its tracks.

Does it make sense to allow developers to make changes to services in production? Does agility mean you have to give up security, dependability and reliability of your services?

Not at all.

Serena TeamTrack allows organizations to put strong policies and procedures in place to make sure the right people get the right information at the right time to make the right decision without the overhead of manual processes that will slow down your ability to do the right thing fast.

Similarly, should everyone have visibility into the metadata and availability of all services? For example, does it make sense to allow a developer for a new web application to be able to use services that access sensitive internal financial data? Or does this sound like a disaster waiting to happen?



Logidex allows organizations to define strict rules to prevent just such a situation. Project leads can specify which developers can see which services. This protects sensitive services during the design phase. If the developer can't see the service, the developer can't use the service.

The Payoff

Serena, in conjunction with partner LogicLibrary, can help organizations realize the promises of SOA-based development while avoiding the cost, quality and risk problems inherent in moving to SOA both for legacy applications and for new application development. We help you visualize your applications so you can build or buy reusable services. We help you orchestrate your application development to optimize reuse and develop new or re-factored services quickly and efficiently. Finally we enable you to enforce SOA governance policies and procedures that prevent SOA-based development from turning into chaos and disaster.

That's agility.