# Reuse Framework for SOA

**Srikanth Inaganti**

## Abstract

The success of SOA initiatives depends on fostering reuse of services and components deployed. Without reuse, basic business problems such as time to market or reducing Business-IT lag, cannot be achieved. Many organizations that embarked upon the reuse journey under different initiatives or contexts with attractive ROI projections, met with either partial or no success due to political and cultural barriers. Since SOA is all about promoting the design of services that are inherently reusable (which opens up opportunities for leveraging the existing investments in terms of using business logic embedded in applications to build new applications), appropriate IT processes are to be institutionalized as part of SOA governance to extract maximum reuse potential. This article will address some of the hurdles to promoting reuse culture and will discuss solutions. The main focus of this article is to discuss the overall framework for business logic or code reuse in the context of SOA, such as the architectural checkpoints to be brought into existing IT processes, service design considerations for reuse, and the importance of defining acceptable reuse metrics to make enterprise SOA transformation a huge success.

## Context

CXOs are continually being asked to do more with less – i.e., with tight budget allocations and schedules. Coupled with this, the demands for making IT more agile and making it a tool to achieve the maneuverability required in the market space. makes reuse more important than ever. The majority of CXOs are already convinced and are looking at SOA as the solution to operational efficiency and maneuverability. One of the pillars of successful transformation of enterprise to SOA has been reuse.

Reuse is not new to the IT world. IT leaders have been trying to achieve the best possible thinking about reuse so that one day the IT industry can emulate the manufacturing industry in terms of developing a system using readily available parts, analogous to the components, services, etc., that are in the SOA world. The machinery that brings together all parts to make a system is analogous to the service infrastructure components in the SOA world comprising Service Bus, BPEL engine, Service Registry & Repository, and Service Management toolkit, etc. The IT industry is still far away from mass production, but reuse will be one of the first steps towards that.

It should be noted that the old method for reuse is in the use of shared libraries for application development. But that proved to be not scalable due varied application customization requirements and targeted platforms. But in the SOA era, developers and designers are dealing with defining new contracts for the same business logic that resides on the network in order to cater to multiple applications, which is little better than pre-SOA era.

In the SOA era, accessing the business logic residing over the network – a logic developed by somebody else – and new methodologies such as composition, and contract development, might lead to fear of loosing control for both application and business owners unless a comprehensive reuse framework under SOA governance is established and managed to the competitive advantage of the enterprise. Since SOA brings IT and business together in identifying and defining the services as a first step towards reuse, the focus that business team brings in terms of organization business goals would certainly augment reuse. Promoting reuse culture would reduce the lag between business and IT, and without a reuse culture the business team cannot go to market in time: This is a fundamental SOA principle.

## Overall Reuse Framework

The following are the various steps that should be taken in order for reuse to become a culture within the enterprise.

Copyright © 2007 Srikanth Inaganti. All Rights Reserved. 

**Identify reuse opportunities** The first step for any reuse effort is to look at the existing IT landscape and determine a list of business critical applications. Among them, one must analyze the common data entities being shared across applications; common business logic would determine the first level of reuse. For example, search functions -- locating a retail store, locating a dealer, etc. – can be potential candidates for such services.

As a second step, analyzing the business processes would lead to more coarse grained business services that have some lower degree of reusability compared to the ones identified from IT landscape study. However, if business processes are designed and implemented with reuse in mind, IT systems would accommodate or be extendable to the new customers, partners, and suppliers emanating from mergers and acquisitions. For example, a health care system developed with the required flexibility, based on SOA principles, can serve multiple providers across different geographies without major effort.

**IT standards** Defining IT standards and enforcing them across the enterprise would lead to considerable of cost savings in terms of reusing the enterprise level licenses for software.

**Shared Infrastructure** Take a look at how judiciously hardware and software, including network infrastructure components, can be shared across multiple applications or business units to achieve operational efficiency.

**Reuse Asset Repository** Institutionalize the enterprise level reuse asset repository that hosts all the service or component related artifacts for all users within the enterprise, including process templates, service specification, architecture and design documents for each service, frameworks, best practices, and guidelines, etc. Normally, users of this repository will be associated with projects from business and IT domains during the design and development time in various capacities, such as project managers, architects, technical leads, designers, developers, LOB executives, etc. This is a direct access.

The other points of access to asset repository will be through a KM (Knowledge Management) solution and via automated retrieval tools that will help making some informed decisions. It is suggested to make this repository a part of one of the back-end content sources for the enterprise KM (knowledge management) solution. By aggregating and classifying the repository content, and indexing it into the KM related search engine, the repository content becomes available to a variety of users within the enterprise. Implementing effective role based personalization would help targeted content delivery.

Tools based on automatic information retrieval, coupled with pattern matching technologies from Autonomy, Verity etc., would help to deliver the right content at the right time to LOB executives, enterprise architects, portfolio managers, and application owners even before they search and, more importantly, when they **need** of the information. But these tools would also have the side effect of consuming LAN bandwidth and slowing down the client machine on occasion.

**Seek Attention and Communicate Benefits** Advertise reuse initiatives across the enterprise and conduct workshops on reuse. Ensure that notifications are sent out to all stakeholders whenever the repository is updated and new training workshops are announced.

**Organize** Establish EARB, Service Factory Team, SOA Program Office. For effective reuse opportunity identification and promotion, the EARB (Enterprise Architecture Review Board) team is the core team of EA Governance structure to be institutionalized. This team is a group of enterprise architects that have good knowledge of business critical systems within the enterprise, and are familiar with organizational IT policies, IT standards, goals and objectives, etc.

There should be a central services team, managed under a factory model, looking after the collection of the requirements, designing, and construction of the enterprise level services or components. The services team interfaces with the application development team, after service usefulness to the application is confirmed, and throughout the application development cycle, to make sure that functional requirements are met by service/component, for smooth application-service/component integration and also so that non-functional requirements are met. It should be

Copyright © 2007 Srikanth Inaganti. All Rights Reserved. 

noted that another major responsibility of the services team is to make sure SLAs for other applications are not affected due to addition of new service client.

As a best practice, the EARB team should get involved in all the project discussions during the business discovery phase itself. This provides the EARB team with an opportunity to give necessary recommendations on reuse to the business and IT management team funding the project that might lead to potential cost saving opportunities.

IT PMO could find the appropriate business and service owners from reuse asset repository to interact with for further investigations of reuse within the context of applications.

The SOA program office should be set up to oversee enterprise SOA transformation. This office should comprise of enterprise architects, application architects, data architects, operational architects, and business executives. This office coordinates the activities between EARB, Services factory team, and IT PMO.

**Architectural (EARB) Review Check Points:** Refine the existing IT/SDLC processes with additional reuse check points, with one review to propose reuse opportunities, a second review to enforce reuse, if found necessary, and a third review to make sure application and service/component environments are ready for cut-over. All these reviews must have appropriate participation from EARB, SOA Program Office, and service factory.

**Involve Business Team** Thorough involvement of the LOB executives is required not only at the time of identifying the services and components [2], but also in selling the components and services within the enterprise. Business executives should be made part of the service factory and must carry the objectives of promoting the reuse within the enterprise. Their domain knowledge, and their suggestions coupled with their focus on business goals would really help the service factory teams to design the services and components to be more generic and improve their consumability.

**Define, Track, and Report Acceptable Reuse Metrics** Funding has always been a problem to reuse initiatives since executive management wants some kind of proof that investment will generate a profitable return. Formalizing the reuse metrics to be tracked will help get buy-in from the executive management. Some of the reuse efforts either ended up in failure or terminated in the middle due to lack of acceptable and measurable metrics.

**Data Ownership** Since the service or component will have data stored on its side, data ownership will definitely be a concern for all potential service consumers even within the enterprise, especially when it involves business critical data. Hence, it is essential for the service factory team to define adequate security measures while maintaining service consumer friendly data access policies. Otherwise, fear of loosing control over their own data would lead to lot of resistance in accepting the component or service by application and business owners, leading to reuse promotion difficulties.

**Reuse Pain Points** Service identification, and building, deploying and building composite applications, are relatively easy tasks when compared with promoting reuse and showing continuous improvement in terms of ROI from time-to-time. Most of the organizations embarked upon the reuse journey, with attractive ROI projections, either met with partial or no success for one or more of the reasons mentioned below.

1. Lack of expected generalization in service or component design
2. Lack of supporting IT processes
3. Lack of acceptance on reuse metrics
4. Lack of organizational support
5. Lack of Standards, Guidelines, and Best practices
6. Lack of sufficient SOA testing to cover for after-the-fact interoperability issues leading to loss of service or component customers
7. Lack of awareness and necessary advertisement
    a. Service discovery
    b. Poor communication

Copyright © 2007 Srikanth Inaganti. All Rights Reserved. 

## Suggested Solutions

This section primarily covers service design issues involved in promoting reuse, the reuse check points that are necessary within SDLC, and reuse metrics to be tracked over a period of time. This would help in sustaining the executive sponsorship as well as in augmenting the SOA transformation in right direction.

### Service Design for Reuse

The challenges involved with service design from the point of view of reuse are

- How to make the design generic enough so that it works in all the scenarios
- How to design for future unknown or unspecified requirements

It is common in the IT world that services and components evolve out of existing applications that are targeted for one application – after putting in some decent amount of effort to generalize. In order for the generalization efforts to be successful, the architect must understand the sources of information for its lifecycle in terms of how it is created, read, updated, and deleted within the enterprise. In other words, understanding data flow through business processes within the enterprise is required to generalize the service interface contract. Otherwise, the contract definition for a service may not withstand all the scenarios and may trigger many versions. It is recommended to look at agile methodologies or refactoring techniques for service development and evolution. Note that effective collaboration across organizational units is required to gather information on how data is used across different functions.

As a best practice, it is essential to capture the service evolution. The following data has to be captured in a service repository.

- Idea behind the service inception, and its requirements
- Assumptions considered
- Design decisions made and reasons

The historical data would be useful for the architects to understand whenever they encounter situations that force them to consider either replacing the existing service with a new one or making modifications to the existing service.

### Reuse Review Check Points

Effective IT/EA governance recommends at least three EARB (Enterprise Architecture Review Board) architectural review controls for any project or application implementation to enforce IT standards and promote reuse.

The first architectural review needs to be done right at the project inception phase or after the business discovery phase. The intent of this checkpoint is to

- Bring the reuse perspective into the inception phase of the project itself
- Application development teams are aware of services and components that can be used potentially
- Enforce IT standards to make sure that proven tools are used in the project, and assess the project feasibility
- Plan for and refine the estimated budgets after reuse is taken into account

The second review needs to be done after the design with the intent of

- Making sure that proposed reusable components and services, as per previous review feedback, are being considered for implementation
- Conveying the best practices around the usage of services and components
- Ensuring all architectural, design, and deviation approvals are closed

Copyright © 2007 Srikanth Inaganti. All Rights Reserved. 

With the above two architectural reviews, all the reuse related aspects can be identified, communicated, tracked, and closed to the competitive advantage of the enterprise. Note that EARB, in concurrence with IT PMO, can suggest the need for additional reviews before giving a go ahead to the implementation or construction phase if there are pending items to be addressed in the context of reuse.

The final review checkpoint is to make sure that application is transitioned to operations smoothly with all required integration points:

- Ensure both application and service/component environments are ready for cut-over
- Ensure that application and service/component interface points are adequately load and stress tested
- Ensure necessary security measures are taken, and review the operational readiness

## Tracking Reuse Metrics

Reuse metrics form an extraordinarily important role in the plan for they play a key part in gaining and sustaining the commitment of executive management. Defining acceptable reuse metrics will help in the exact measurement of reuse and resultant cost savings. This section provides high-level guidelines to calculate the ROI on reuse, but is not intended to cover a detailed list of line items for ROI calculation.

The following are a few important reuse metrics to be gathered and published to executive management.

- Number of services and components deployed over a period of time
- Number of projects using components and services (progressive numbers on service client would help retaining the executive support)
- Amount of USD savings across various projects/applications (based on the current estimates of application development)
- Metering and Billing:
    1. Defining the charge back models for each service that is acceptable to business and IT divisions. Assign a USD value to each request depending on component or service complexity, service criticality to business, investment made, projected ROI, etc.
    2. Keep measuring the usage (number of service requests) of all the components and services. Direct requests come from all applications directly using the component or service. Indirect requests come from other business services compositions.
    3. Keep comparing the one time development, deployment costs, and ongoing operational, maintenance, and hardware upgrade costs against the value created, as shown below in the conceptual view – to justify continued support to leadership. Metering and billing should also involve charge back model for service infrastructure components (such as common security components, service registry, service repository, service management tool kit, BPEL Engine and service bus, etc.) to recover the one-time expenses incurred for procurement under deployment costs and to measure the value being generated for all applications being serviced.

The diagram below demonstrates a comparison of investment cycles between applications and services. Please note that the S-curve is symmetric for explanatory purposes only. In reality, it will be distorted from the one shown above.

The BLUE colored S-curve depicts the typical investment lifecycle for an application. Service would also follow the same pattern until it is deployed. Once it is operational, due to its reuse by multiple applications and other services via composition, the investment lifecycle of service follows a different path that shows cost savings and chargeback cycles alternatively.

BLUE colored Spikes in the diagram represent the additions to service client portfolio and represent the resultant cost savings to the enterprise. To measure the accumulated cost savings from time-to-time, CFO organization involvement would be useful to get the vendor development

Copyright © 2007 Srikanth Inaganti. All Rights Reserved. 

rates. Following simple math would give cost saving for service usage from composite application development perspective.

**Cost saving = (No of lines of service code/ No of lines of application/service client code) * Dollar value of the application**

The payback period is a function of operations and maintenance costs and charge back values. Assuming the cost savings are constant, for service investment to break even early, it's operational and maintenance costs should be low and charge back value should be high. If the operational and maintenance costs are high and chargeback values are low, then the service payback period would be longer.

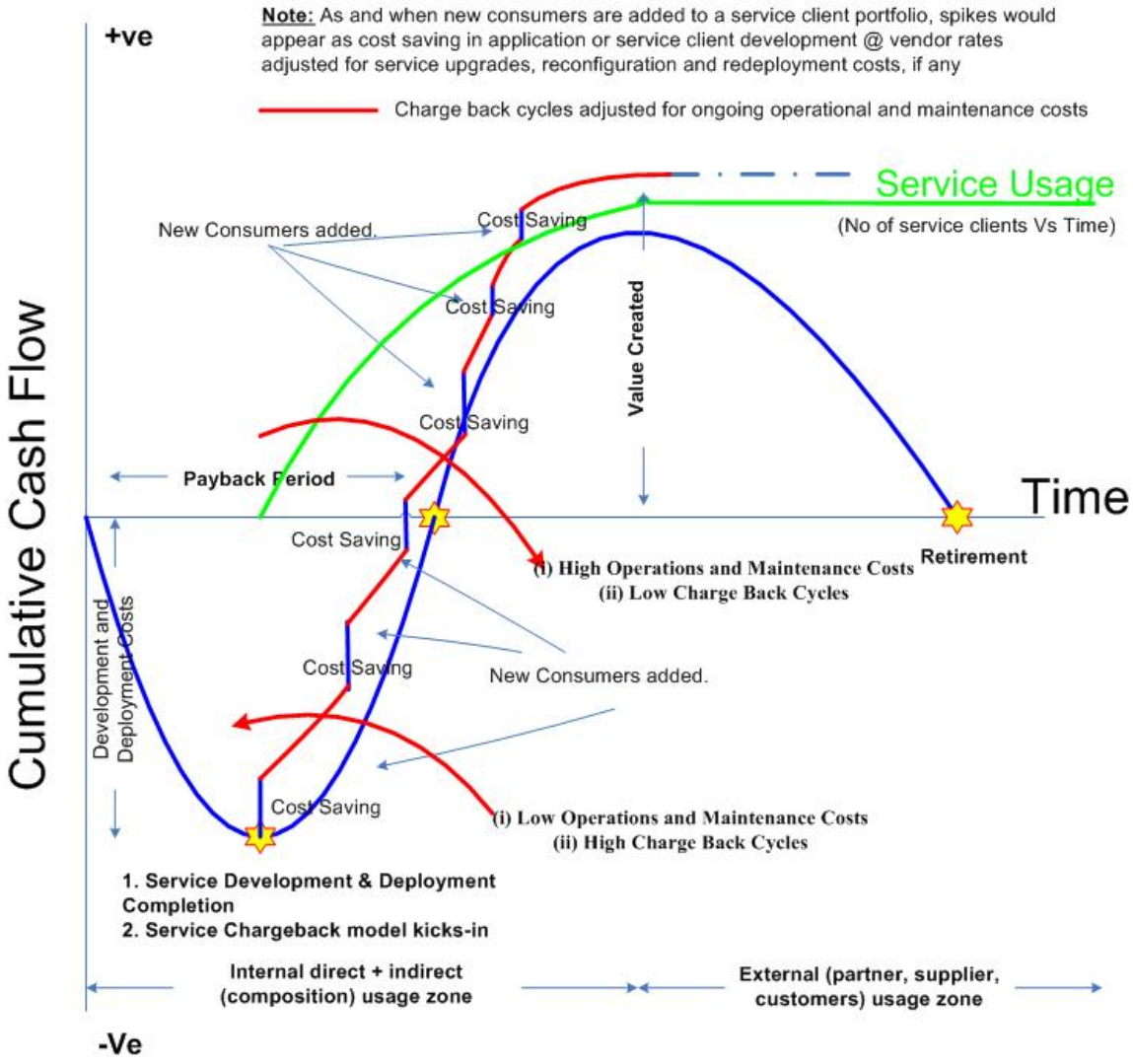


**Figure. Service Investment Lifecycle**

**Value created = (Accrued cost savings from service clients development + cumulative chargeback for service usage – ongoing operational and maintenance costs)**

If reuse culture is promoted within the enterprise, the cumulative value created by all services and components would exceed one time investment costs of service infrastructure (software) components including hardware.

Copyright © 2007 Srikanth Inaganti. All Rights Reserved. 

The application potentially becomes a suitable candidate for retirement when the value created is nullified by the ongoing operational and maintenance costs due to reasons such as high cost of developer resources, proprietary technologies involved etc. But services case is slightly different based on the type of services.

Once business service reaches the maximum direct usage within the enterprise, the value created can be further improved by exposing it to outside of the enterprise to partners, suppliers and companies within your domain. However for composite business services that comes under business processes or specific domains, productizing the services or possibility of exposing externally to create enterprise mashups should be the end goal right from the beginning for better value generation. This is due to the fact that they are coarse grained and may not be reusable across domains within the enterprise.

Value created from framework and infrastructure related services might be linear and may not reach the plateau as fast as business or coarse-grained services since they can be used in almost all the applications.

## Miscellaneous

The following table depicts the probable solutions for other pain areas.

| Pain Area | Suggested Solution |
|---|---|
| Lack of organizational support | ▪ Garner executive support via IT strategy, Funding to IT initiatives<br>▪ Development Managers, Developers support: People who work at grass roots level also be mentored to promote reuse, Should be more committed towards strategy and its goals<br>▪ Evangelization should be core function of architecture, technology, and COE teams with appropriate funding |
| Lack of Standards, Guidelines, and Best practices | ▪ Prevalence and usage of different products or technologies or styles to solve the same problem. For example, using different web service styles like REST, J2EE web services, etc. As a best practice, service provider must specify the invocation example for each type of technology/client/consumer within the enterprise.<br>▪ Sharing best and worst practices across the organization<br>▪ Evangelization of new technologies – through COEs |
| SOA Testing (for future-proof integration) | ▪ Interoperability testing should be mandatory during all testing cycles of service development.<br>▪ While service is developed, all probable message exchange patterns available within the enterprise should be tested. |
| Lack of awareness and necessary advertisement on services and components | ▪ Commission the enterprise level asset repository that holds all service, component related artifacts that should include client list<br>▪ Increase awareness: Enterprise wide email notifications to be sent for all important updates to the repository<br>▪ Advertise the ownership of services and components<br>▪ Institutionalize the service or component team engagement protocols |

Copyright © 2007 Srikanth Inaganti. All Rights Reserved. 

## Conclusion

This article covered the importance of reuse in the context of SOA, some of the bottlenecks in the enterprise and suggested solutions to promote reuse. Enterprise reaps the maximum benefit if realigned towards reuse in terms of fine tuning the SDLC processes with additional reuse check points by the enterprise architecture team as mentioned in this article, apart from business executive and IT management support. This article explained the need to have appropriate metrics and to provide high-level guidelines for ROI calculation from reuse. In this context, it also highlighted the appropriate EA and SOA linkage.

## References

[1] Service Identification: SOA and BPM Handshake, BP Trends Journal March 2007.

[2] TOGAF v8.1: Architecture Board

[3] Code Reuse in Enterprise interview with Charles Stack, CEO, Flashline Inc.

[4] FTP Online: Top 5 Asset Reuse Best Practices For SOAs

[5] SOA + Information Architecture = Code Reuse (finally!) By Jason Bloomberg, Document ID: ZAPFLASH-10222003

[6] The Val IT Framework from IT Governance Institute

## Glossary of Terms

| Acronym/Abbreviation | Definition |
|---|---|
| BPEL | Business Process Execution Language |
| CFO | Chief Financial Officer |
| COE | Center Of Excellence |
| CXO | Chief Executive Officer, Chief Financial Officer, Chief Operating Officer, Chief Information Officer etc |
| EA | Enterprise Architecture |
| EARB | Enterprise Architecture Review Board |
| ESB | Enterprise Service Bus |
| J2EE | Java 2 Enterprise Edition |
| KM | Knowledge Management |
| LOB | Line Of Business |
| IT | Information Technology |
| IT PMO | Information Technology Project Management Office |
| REST | Representational State Transfer |
| ROI | Return On Investment |
| SDLC | Software or System Development Life Cycle |
| SLA | Service Level Agreement |
| SOA | Service Oriented Architecture |
| SOA CC | SOA Competency Center |
| TBD | To-Be-Determined |
| USD | United States Dollar |

Copyright © 2007 Srikanth Inaganti. All Rights Reserved. 

## Acknowledgements

I would like to thank Dr. Udaya Bhaskar Vemulapati, General Manager, Wipro Technologies for giving the required time and support in many ways to help produce this article as part of our SOA Center of Excellence efforts. Thanks to my colleague Srinivasa Rao Chintala for his feedback and review of the article.

## Author

Srikanth Inaganti is a Senior Enterprise Architect in the Enterprise Consulting and Architecture Practice division of Wipro Technologies. Srikanth Inaganti, PMP, E-Architect, ECAP Group, Wipro Technologies srikanth.inaganti@wipro.com, Office # 91 40 3079 5110; Mobile # 91 9849058064

Copyright © 2007 Srikanth Inaganti. All Rights Reserved. 