A Guide to SOA Implementation

Abstract

The information overload on SOA is largely on describing the merits of SOA, principles of SOA and the vast variety of products intended to address SOA needs. There is, however, an acute scarcity of information on SOA implementation to bridge the gap between wanting to get started and actually deploying a game plan where the rubber hits the road. This document is written to identify the factors to be considered, articulate the principles and questions to be asked that will drive the decisions within each enterprise towards creating a road map for implementation.

Separating reality from hype, distilling the essential from the desirable, explicitly addressing the how-to, nuts and bolts of an SOA implementation is the topic addressed in this document. It's about getting going, showing investors the incremental rewards of SOA adoption and continuing on the straight and narrow path that links deployment of technology with business goals.

Whether you are contemplating SOA, are or actually committed and well along the way, a back to basics checklist is a good thing to have. This document includes a review of the following topics that will help to retain focus and direction.

- Your reasons to implement SOA
- First steps and short-term goals
- Long term Objective
- Products and Partnerships
- Return on Investment
- Impact on Business as Usual
- A Stable approach to Change management
- Glossary of SOA Terminology
- Summary and Help Lines

Introduction

The benefits of SOA have been widely described and accepted. The question faced by many is: How to get started? Where to get started? Expectations that may be deemed reasonable and the return on investment(ROI) that can be demonstrated during each phase of investment have to be identified. Further questions abound regarding the choice of products, partners and ramping up of in-house IT capability. Of course, while there is no common one-for-all solution, the implementation of SOA is much like any large-scale migration, with a few notable differences, particularly regarding governance. The purpose of this document is to identify the factors to be considered, articulate the principles and questions to be asked that will drive the decisions within each enterprise towards creating a road map for implementation.

1. Your reasons to implement SOA –Two classes

While there are several frustrating experiences with legacy architecture that serve to propel the drive towards SOA, the drivers usually fall into two classes of compelling reasons, that either singly or together form the impetus to actually get started with SOA.

1) The need to respond quickly to change in business needs
2) The need to reuse technical assets across a larger enterprise

Symptoms include acutely painful situations where “the business” requests rapid and seemingly endless minor changes in business processes that incessantly require significant application level code changes within time lines that seem impossible. On a less painful but equally urgent basis, mergers and acquisitions, integration with 3rd parties/partners etc. extend the borders of e-commerce to include a wider range of activity with suppliers and clients, through public information highways. These requirements, either mandated by management or simply viewed as a step to optimize development and support costs, call for the creation of standardized assets, that once created, can be run anywhere.

2. First steps and Short term Goals

Once the primary driver or business reason that compels a service-oriented approach is articulated and agreed upon by all stakeholders (from the business and technical communities), it becomes relatively easier to establish the milestones that are to be reached. This approach is consistent with one of the commonly observed principles of successful SOA implementations. SOA is almost always best realized through an evolutionary process - an ongoing conversion of services that increasingly perform autonomously and thereby can be reused. This evolutionary approach does not necessarily mean a long drawn out exercise, it simply means that a sequence of steps have to be defined. The steps could become progressively larger and address a wider range of use cases, as each successful installation is completed and seen to work.

If your reasons to SOA happen to fall into either of the two categories described above, then here are some early steps and goals towards the realization of a flexible infrastructure:

1) **"Which business processes are most frequently changed?"** Identification of the application or sets of applications that are impacted by the most frequently required changes requested by the business. This is a first step that gives us an idea of what we will be dealing with, though in small bites. A parallel run, involving both legacy and the different versions, each incorporating progressively more services being exposed in each iteration, will be required during the transition.
2) **Documentation of the Use cases** that will be impacted by changes to these applications; this will include a study of client-based executables or interfaces, invoked by each user community. The purpose of this study is to allow a fuller view of the scope and establish a narrower set of "important victories".
3) **Review of server based application code** with a view to identify and abstract business processes from application functionality. This exercise accomplishes the goal of allowing us to form a reasonable estimate of the work and time lines involved.
4) **Potential latency issues are examined** as a result of re-use and content heavy XML, as well as security issues in opening these assets for access. These considerations will determine the classes and types of products to be used in the enterprise architecture. Once this is done, the first sketch of the target architecture is visible.
5) **Start building and exposing the separate functionality that is identified to be a part of one or more business processes.** At this time, it is important to get started. You need a registry and/or a repository that describes the service, so incoming messages can find them. At first, the SOA could be made available within the firewall, tested, and then subsequently made available through holes in the firewall that are opened specifically with relevant security measures to allow access to nominated users.

These short-term goals represent a set of actions that will lead to a functioning SOA. The SOA can scale up - as we identify further functionality that can be reused, leading to a progressively larger implementation. We call this a Bottom-Up SOA effort.

A Top-Down effort is also required. The purpose of this is to establish some basic ground rules that the entire SOA will live by. This is much like the role-played by a Federal government. In SOA parlance, it is simply called Governance.

3. Long Term Objective

An efficient and agile technical environment is traditionally demonstrated by time and cost measures. Therefore, these measures must govern the path towards the long-term objective and determine the success of its eventual realization.

The success of the target architecture should be measured by:

1) Cost and time to develop and maintain the assets that perform the required business processes and the tangible and intangible returns from executing these business processes.
2) The implementation of automated triggers to fulfill a larger number of routine business activities.

The first criterion is realized through building granular components and making them available for re-use through an expanding repository, progressively made available to a larger community inside and outside the firewall. This community will include third party suppliers and partners and hence the creation and publication of these assets must adopt industry standards.

There are three aspects of the move to SOA that need to be better-understood in-terms of their relevance to each stage of SOA evolution. The purpose of doing so is to retain our long-term objectives within each, but not let them, individually or collectively, paralyze us or prevent us from getting off the ground.

Governance -

The purpose of governance is to avoid lack of conformity between applications. It is difficult to establish all the ground rules when you start. As said before, SOA is an evolving process, in terms of standards, in terms of products and its use within the enterprise. Some fundamental rules common to a federated structure will be defined at the outset and modifications are inevitable.

Service Granularity -

This is one of the most discussed and debated topics today. Since it refers to the scope of functionality that a service exposes, one would need to plan the grain of the service based on the current application infrastructure environment. Waiting until the "right" level of granularity is established, can become a never-ending discussion with no perfect answer; hence, it is important to understand your environment and plan/implement the granularity of your services in a simple and generic way, based on your current and foreseeable needs. SOA is an evolution; coarse grains may have to be decomposed into finer grains at a later time depending on the business process that is required at the time. This fact should not deter us from getting started.

Tools / Products -

BPM, BAM and EDA are some of the terms associated with the realization of SOA. They form part of the second group of long-term objectives and there are a wide range of products available within these categories. It is important to note that BPM and BAM products are not an essential part of an SOA and are in fact not advisable during the early stages. An increasing awareness of their specific benefits could be developed as progress is made and they could be introduced at the right time to enhance considerably the value of an SOA.

4. Products and Partnerships

This section describes the classes of products and partnerships available to better realize the benefits of adopting this methodology.

Many of these products address the same needs, varying in the level to which these needs are to be addressed. The choice of whether to introduce a certain class of product depends on the level of attention required to the functionality that is offered to address that particular need.

For example, some level of authentication and authorization functionality could already be built into your application server. An ESB, introduced at a later stage, for transformations and adaptations is also capable of providing this functionality to a greater depth and range as depicted in the diagram below:

Figure 1: Stage 1

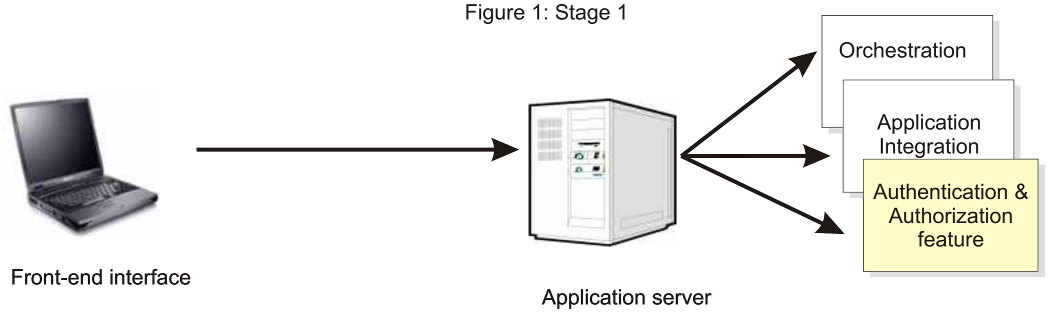


Figure 2: Stage 2

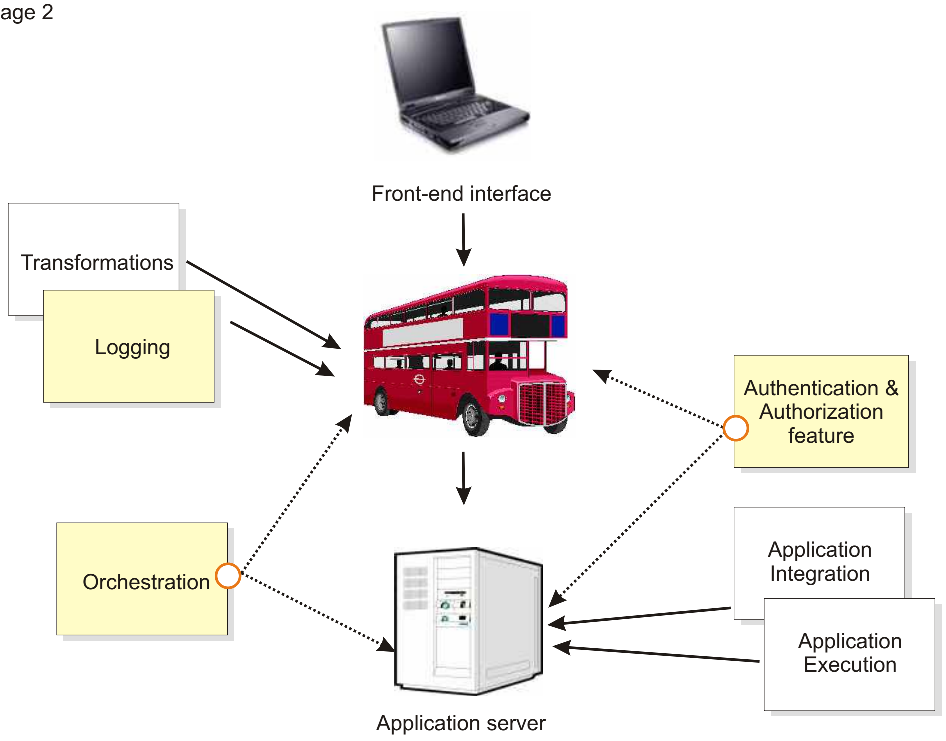
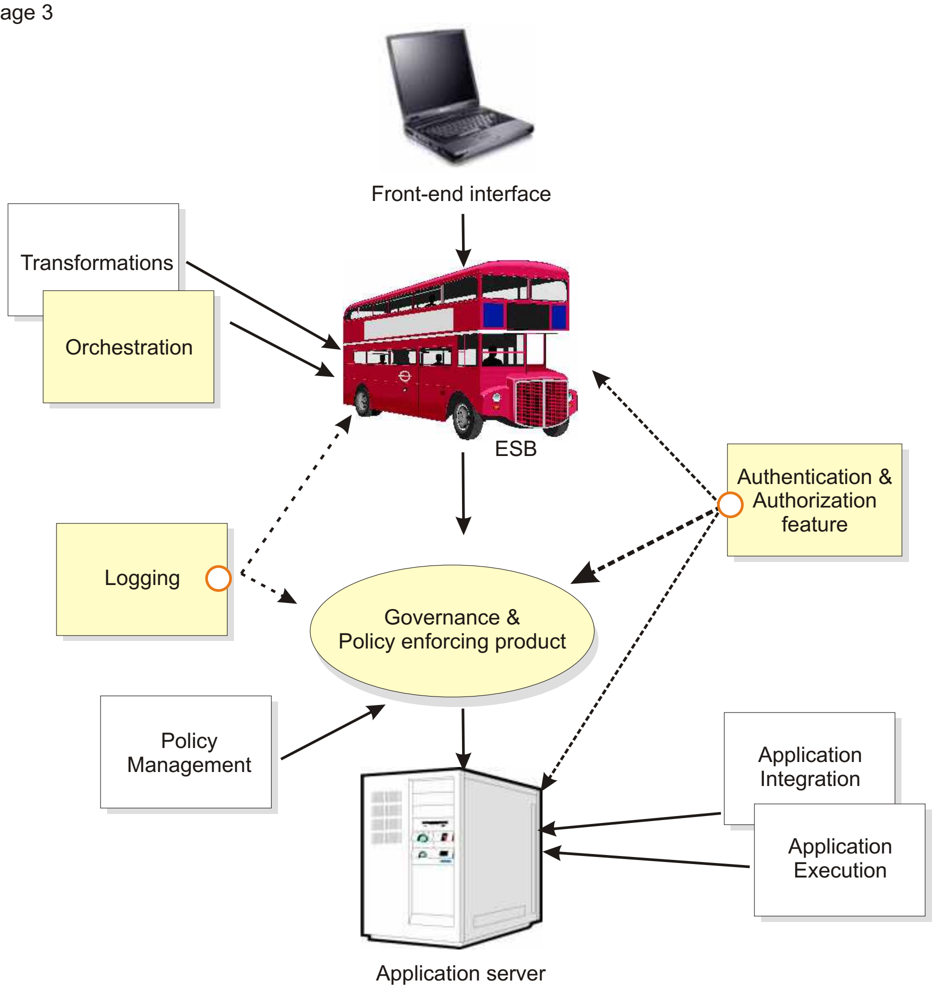


Figure 3: Stage 3



As can be seen, all three products (the application server, the ESB and the governance product) provide the 'authentication and authorization' feature. There is a clear overlap of functionality. Based on where this feature needs to reside and how it should be implemented, a decision would have to be taken on whether to delegate some of the functionality to all products (based on the products' strengths) or to consolidate it into a single, most suitable product, based on the requirement. Similarly, the 'orchestration' feature, which was earlier residing in the application server, is now serviced by the ESB, as the ESB provides more flexibility and a feature-rich orchestration mechanism.

Broadly speaking, products are available to perform one or more of the following functions.

- Governance and Policy enforcing products
- SOA base infrastructure/platform products

A brief description of each product group follows:

Governance and Policy enforcing products

The term “governance” has been emphasized in SOA. The purpose of drawing attention to this aspect of operations is to create a broad set of rules to which the architecture must adhere to, in terms of message format, security, policies to which independent parts of the whole must conform, to avoid mismatches and possible breakdown. Diverse users with varying priorities are now being driven to re-use the same assets; hence it is logical that a federated structure be created. The tools and products available in this category allow the creation of enterprise-defined repositories of the services available, published within acceptable standards of discovery. With the creation of SOA stacks and product suites, governance tools are often clubbed with lower level products that address other SOA needs such as message transformation and brokering associated with integration.

Security issues and trust enablement also play a vital role since application functionality needs to be made increasingly accessible to a wide number of users. Products that facilitate solutions relating to these concerns include an interesting approach to screening, not just the sender of messages, but also content based screening done at the network level, through a new category of network appliances that also serve some translation needs in the message format. Latency and acceleration to deal with the larger number of content heavy messages can also be addressed by this product group. Security is also addressed at lower levels through products such as the ESB.

SOA base infrastructure / platform products

With the increased re-use of assets and application functionality that is the very purpose of an SOA; messages with different and sometimes incompatible formats and protocols tend to flow. There is clearly a need to transform these into a consistent format that will be interpreted in a uniform manner by the applications now being accessed. Hence, a variety of products have been created to accelerate such transformations into the canonical structure required by the enterprise (ESBs, web service hosting platforms, orchestration / BPEL etc.). XML is the standard medium of message routing. XML to XML transformation may also be required to provide the consistency necessary to invoke common assets used by a larger community.

5. Return on Investment

Since SOA usually does not introduce increased functionality, it is often difficult to represent the return on investment to those who fund and approve the initiative. Hence, the return on investment must be demonstrated, using a set of 'what-if' scenarios.

A what-if scenario involves an estimation of the time and effort required to introduce the set of changes that are mandated by a user community through the use of SOA, and through the traditional legacy route. The contrast makes the case. These changes could either be a re-classification in the sequence of an existing business process or the introduction of an additional step in the business process. Either of these changes under the rigid conventional architecture present in most enterprises, presents a significant effort in rewriting the thread that invokes the required functionality. It could also mean a significant effort in modifying the API. SOA now permits re-use of assets, and the effort saved by such reuse is the justification.

Return on investment is also demonstrated by the additional avenues opened, through compliance with the systems used by third parties, who are either clients or providers. Interaction with these valuable partners can now be largely automated and the time, cost, effort and errors associated with human interaction, largely avoided. Increasingly, a hesitation to implement an SOA based communication format, actually erodes an enterprise's ability to enhance its business with more advanced partners, who adopt business processes that require standardized automated communication and triggers. The loss of opportunity is therefore a factor to consider in determining return on investment in moving to SOA.

Last but not the least, in the days of rigorous regulatory requirements, ROI is also demonstrated by the compliance to the statutory and regulatory requirements and savings in penalties that can be avoided.

6. Impact on Business as usual

The impact on business as usual is to be understood and mitigated as is done with a series of releases in any migration. It is in fact made easier by the fact that SOA is neither a new technology, nor a radically different functionality provided to existing users, nor does it necessarily introduce a different user interface. In fact users will often be unaware of the migration except to experience a pleasant reduction in time to implement the changes they seek.

However, behind the scenes, there is a lot of work in terms of the isolation and presentation of technical assets. While certain applications can simply be wrapped in web services and made available through the repository, other applications may have to be broken down and rewritten. This is often necessary where the business process and application functionality are irrevocably tied up through a series of methods, procedures or other code that do not allow use of discrete autonomous parts of the application logic. The concept of loose coupling is the foundation of SOA and hence separation is the first step allowing binding to take place at will.

The two factors to be addressed are the sequence or prioritization, and the impact on current or future users. Hence the documentation of use cases and the documentation of applications including possible elimination of redundant applications are amongst the first steps advocated earlier in the process. If this process is diligently carried out, the impact of separating the logic and the introduction of granular autonomous services will be nullified, provided potential latency issues are addressed. Hence, the evolution becomes a series of releases, each tested in the conventional manner, with calls from within and outside the domain, to verify performance. As in most migrations, two versions of the application continue to exist and remain in use, until one is completely phased out when the full functionality is transferred to the set of independent services that are capable of replacing it through granular deployment.

This process is of course familiar to IT departments who have in the past introduced migrations; and the steps in an SOA migration are quite similar. The one redeeming feature in this migration is that it will hopefully never have to be repeated in its entirety, since SOA is a process to deal with change, without changing the infrastructure. The migration will of course become more sophisticated as it extends, to deal with requests from a broader user base and more layers may be introduced to better address each

7. A Stable approach to Change Management

We are now reaching the long-term objective of an SOA migration. We are creating an infrastructure designed to accept and provide for changes and used by a large disparate set of users irrespective of location, unconnected physically or structurally, except by the Internet.

This is where the introduction and adoption of standards begins to pay off. As developments continue, we will see the introduction of more and more products that serve to do the same thing, except with varying degrees of speed and user options. Almost every feature required in an SOA is now available from one or several of the broad range product vendors, who offer everything from a complete suite to a particular product that could interface with other products in an increasingly standards based paradigm.

The road to SOA is never ending; meaning the adoption of this methodology will continue to be a consideration in the foundation of all development. This does not mean that all assets have to be engineered for reuse, nor does it mean that all application functionality will have to be discovered at the time of use. Traditional hard coding, in the form of fixed methods, procedures or calls will continue to co-exist, and in fact serve better in situations where the performance overhead created through a service oriented approach is not warranted, since the use of the asset is confined to a particular, well contained set of use cases. The concept of SOA is therefore best utilized by applying the principles of SOA towards the construction of assets where their use is known, at least at the time of construction, together with a plan for dealing with use that is not envisaged at the time.

8. Glossary of SOA Terminology

- SOA – Service Oriented Architecture
- BPM - Business Process Modeling and Management
- EDA – Event Driven Architecture
- ESB – Enterprise Service Bus
- BPEL – Business Process Execution Language
- API – Application Programming Interface

9. Summary and Help Lines

Large or small, enterprises have to take the step towards SOA. This document provides a description of the elements to be considered to set in place a road map and take that all important step of getting started, to get going, to begin realizing the benefits that have been described in countless publications on SOA. It hopefully ends the deliberations for a moment and stimulates action.

The question of technical help and the role of service providers are now to be addressed. Product vendors provide a whole spectrum of services in the design and implementation phase. They often have carefully chosen, approved implementation partners who, once having understood the requirements of the enterprise, can greatly simplify the process towards reaching defined goals. It is interesting to note that many of these service partners are certified to work with a number of competing products. This marks a movement to better evaluate and implement a product for a particular enterprise.

SOA is about integration; Integration of services to provide a range of functionality without disruption or destruction of technical assets. Hence, it is not surprising to find the SOA niche is filled with players in the integration space, specialized in Middleware.

Torry Harris Business Solutions Inc, a US based services provider with a large base of technologists located in the UK, India and China has provided cost effective solutions at a design, development and support level to a variety of enterprise clients across the world since 1998. The company specializes in integration, distributed computing, and its focus on SOA is a result of nearly a decade of expertise gathered in the middleware space. The company has partnerships with almost all the leading SOA and integration product vendors. SOA, involving the creation of autonomous parts of a solution, lends itself admirably to the cost effective model of offshore service collaboration. A separate white paper entitled “SOA Implementation with an offshore partner” available for download, explores this model in a more detailed manner. Further information about the company and a variety of white papers on SOA are available at www.thbs.com/soa.

For more information, write to us at soa@thbs.com.