Estimating S O A, As Easy as 1,2,3

Introduction

Service Oriented Architecture (SOA) projects bring a myriad of potential benefits to organizations by creating the promise of business agility. According to Gartner, SOA will be used in more than eighty percent of mission critical operational applications and business processes by the year 2010. The potential benefits of a SOA migration are extensive if the literature is to be believed. From a technological perspective, the flexible nature of the architecture leads to easier integration of new and existing services. The ability to integrate and share data is improved as the standards based integration removes implementation details of the individual systems sharing data. The potential to develop data services enables organizations to remove data duplication and provides access to the right data at the right time. Loose coupling makes it easier and less disruptive to upgrade services as requirements change, leading to potential cost savings throughout the lifecycle.

There are significant advantages of SOA from a business perspective as well. SOA makes it possible for a business to drive Information Technology (IT) requirements instead of letting IT limitations constrain the business. It helps a business remain agile by allowing it to quickly respond to a changing world. A SOA based implementation facilitates ease of communication with business partners and the supply chain. Additionally, the use of a SOA allows a business to eliminate overlapping capabilities and maximize reuse of existing assets.

Analysis of the literature indicates that the SOA vision leads to a belief of implementation efficiencies and cost saving of epic proportions. SOA offers organizations the promise of cost savings, data sharing, interoperability and increasingly agile operations. Cost estimators, analysts and project managers need tools and methods beyond traditional software cost estimation techniques to properly assess the costs of SOA projects. It is true than many aspects of a SOA project are similar enough to other types of software development projects that traditional approaches apply. But there are dimensions of this new paradigm that require shifts in estimating practices. This paper focuses on three such areas: SOA governance, middleware integration and service development.

The proper integration of SOA into an organization necessitates significant changes in governance practices. In order for SOA to be successful, there needs to be governance at an enterprise level to ensure that services are aligned with business processes, that new services meet organizational needs, and that service reuse occurs effectively. New levels of governance and associated change management have cost implications beyond the scope of traditional estimating practices.

The integration of middleware components at the heart of SOA technologies presents scoping challenges outside of those encountered with more traditional implementations, even those with heavy reliance on standard commercial off the shelf (COTS) solutions. For the estimator these challenges require a new paradigm to ascertain the 'size' of the integration task. Service development, in many ways analogous to any new software development task, will experience additional cost concerns associated with the SOA requirement to be highly reusable and potentially useful outside of the scope of the current project.

This paper reports on research focused on providing the estimating community with solutions to the SOA specific issues outlined above. Section 2 provides necessary background information on what SOA is and how it is used. In Section 3, the issues associated with SOA governance are discussed and their cost implications revealed. Section 4 provides methodology for tackling the costs associated with the development of services, while Section 5 presents methods for cost assessment of middleware integration accomplished in the context of a SOA deployment. Section 6 presents conclusions and future work.

What is Service Oriented Architecture?

Service Orientation is not a new concept. We are all providers and consumers for services. If I want to make toast, I plug my toaster into the wall outlet, turn the toaster on and power flows. I require no knowledge of how the power gets from the wall outlet into my toaster, or what substation generates the power. As a service consumer, all I need is the correct interface (my plug) to get access to the electricity and a contract with the service provider, in this case the local power company, which indicates my willingness to pay for the service. Throughout the US, anyone with the same interface and a relevant contract can get access to power in the same way.

In the context of software, SOA is a paradigm that offers software service providers the potential to share their software solutions with consumers using the same basic business model that utilities have used successfully for years. A SOA is an architectural style that allows for distribution of capabilities that need not all be supplied or owned by the same organization or entity with the same notion of transparency that utilities offer electric consumers. Loose coupling of services is the key to successfully deploying service orientated capabilities. There is no need for the service provider to know who the service consumer will eventually be or what specifically they will do with the service.

SOA Building Blocks

The most important aspect of service oriented architecture is that it separates the service's implementation from its interfaces [1]. Through loose coupling and tight standards for interfaces, consumers need only know how to interact with a service, there is no need for them to know or understand what's under the covers. The building blocks of SOA include service consumers, service provider, service registry and the service contract.

> A **service** is a software implemented capability that is well-defined, self contained and does not depend on the context or state of other services. [2]
>
> The **service consumer** is a service, application or other software component that requires a specific service. The consumer locates the service through the service registry, accepts the terms of the contract and initiates the service using the mandated interface.
>
> The **service provider** is the software entity that represents the service being delivered. The service provider makes the service contract available through the service registry and it accepts and executes requests for service that satisfy contractual criteria.
>
> The **service registry** is the network space where service providers publish service contracts and service consumers visit to locate desired services.
>
> The **service contract** is the vehicle through which the service consumer and the service provider seal the deal. If specifies the rules of engagement as far as what the provider will supply, how long the consumer will interface with the service and whether (or how) a particular consumer can be granted access to the service.

SOA Governance

A successful SOA program is likely to require changes in the way that Information Technology (IT) is organized and operates. The extent of change will differ depending on the type of enterprise adopting SOA and the goals that they hope to achieve with SOA.

Successful SOA requires that an organization have good SOA practices. An organization that invests in service oriented technology and then uses that technology to develop 'stovepipes' of SOA is not really 'doing' SOA. Many of the benefits of SOA require that service oriented solutions be approached at an enterprise level. SOA becomes valuable when service design, implementation and usage is governed in such a way that leads to reduced integration expense, increased asset reuse and increased mission thread/business agility.

Successful SOA requires that good SOA governance be in place. IT governance can be defined as "the leadership and organizational structures and processes that ensure that the organization's IT sustains and extends the organization's strategies and objectives" [3]. SOA governance is a subset of IT governance that focuses on exercising control over the adoption and implementation of SOA. This includes developing a strategic adoption approach, and defining SOA standards, policies, contracts and service level agreements. The organizations that are successfully deploying SOA understand what level of governance is appropriate for their organization. These organizations are also successful because they include the need for organizational change and SOA governance into their plans from the very beginning.

SOA governance is a set of activities related to exercising control over services throughout an enterprise. Governance activities can be characterized as either high-level or low-level activities. Low-level governance activities apply to specific services and sub-projects of a more broad SOA initiative. These activities are well handled within existing software estimating contexts and were not the focus of this research. High-level governance activities have enterprise wide application and deal with on-going activities associated with ensuring SOA success at an enterprise level. High level governance activities were the focus of this study.

In large part, SOA is about effectively and efficiently sharing data throughout an enterprise. Because of this, any discussion of SOA governance would not be complete without some coverage of data governance and how it might be handled. Data governance refers to the set of processes an organization puts in place to handle the assessment, monitoring, management, protection and maintenance of their data. Organizations should include data governance in their estimation and planning. The findings of this research in the area of data governance were inconclusive. Most organizations assess data governance as a level of effort (%) of the entire projects effort or they allocate an amount of Full Time Equivalent's to handle governance.

SOA Governance Activities

This research identified four enterprise wide SOA governance activities. These activities are:

1. **SOA Policy and Strategy Development:**

Before any SOA projects are launched there needs to be a vision and plans at the enterprise level in order to establish enterprise goals for SOA and create a roadmap to lead to those goals. Sub-activities include:

- Creation of a detailed vision of the end state
- Creation and enforcement of broad SOA policies
- Creation of strategic SOA adoption plan through analysis of existing IT assets
- Identification of areas of the business where SOA implementations would have the most impact
- Deal with funding issues and incentive programs

2. **SOA Education, Promotion and Marketing:**

A crucial success factor for SOA is buy-in of critical stakeholders and successful education of the enterprise. Sub-activities include:

- Promotion and marketing of enterprise SOA capabilities to stakeholders
- SOA policy education
- Training in enterprise wide SOA procedures
- Enterprise level SOA related communications

3. **Service Provisioning Governance:**

SOA success requires that software projects across the enterprise are aware of what services are available to them from both internal and external sources. Sub-activities include:

- Providing the right services to the right consumers
- Ensuring sharing of both capability and cost responsibility
- Aligning software governance with business governance
- Management of reuse across internal and external domains to achieve maximum agility and economies of scale and scope

4. **Service Performance Monitoring and Optimization:**

As SOA is deployed throughout the enterprise it is important to measure performance against goals and identify areas for improvement and optimization. Sub-activities include:

- Establish and maintain automated service performance monitoring software (generally part of SOA infrastructure technology)
- Utilize performance outputs to facilitate decision making
- Develop and use governance metrics for service specific requirements and enterprise requirements
- Analyze metrics to address failures and identify areas for optimization

SOA Governance Cost Drivers

Traditional software project estimates use software 'size' as the basis of a cost or effort estimate. And while the experts can't seem to agree on the best units for measuring software size, all relate directly or indirectly to the amount of capability being delivered. This is not the case when trying to size the SOA governance activity. Rather the size of the governance activity is a function of the domain of the enterprise and the scope of the current SOA initiative within the enterprise. Clearly there is currently no scale to quantify a domain. In the context of this research, a Domain Size Factor has been established which provides an enterprise with a way of determining domain size relative to well known domains.

Domain Size Factor:

- Measure of the scope of data, processes and relationships common to the domain of a SOA initiative
- Measured through comparison to baseline domains.
- Domain examples include:
  - ➢ Military Strike Community
  - ➢ Law Enforcement Community
  - ➢ Battlefield Situational Awareness (subset of Military Strike Community)
  - ➢ Crime-related Data (subset of Law Enforcement Community
  - ➢ etc…

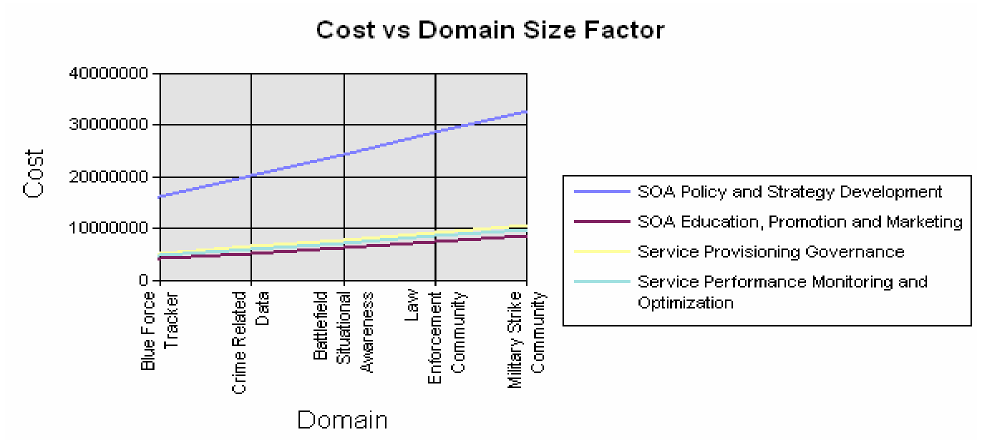

Figure 1 : Impact of Domain Size on Cost by Activity

Independent of the size of the domain, the scope of the SOA initiative currently being estimated is another important factor in sizing the amount of effort and cost that should be planned for the governance activities.

Project Scope Factor:

- Number of entities (organizations, platforms or systems) involved in the initiative
- Extent to which each entity contributes to the information sharing requirements within the domain. Involvement is classified as to what percent of functionality, data or processes of the domain affect the entity as follows:
    - Very Low – less than 10%
    - Low – 10% to 25%
    - Medium – 25% t0 40%
    - High – 40% to 60%
    - Very High – 60% or more

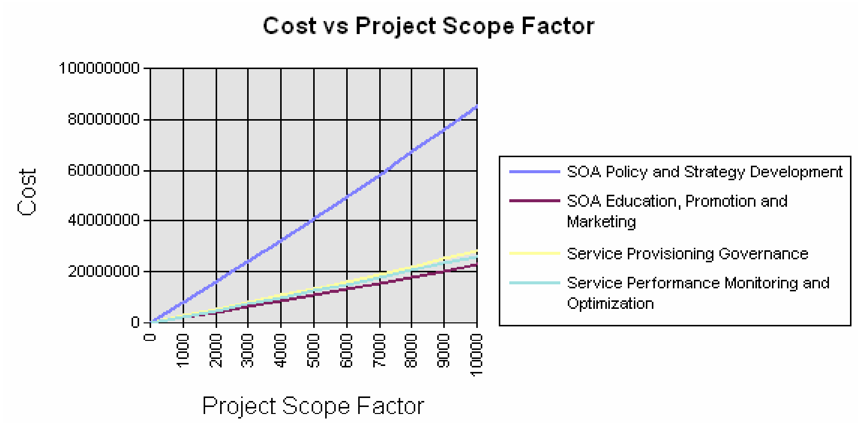

Figure 2: Impact of Project Scope on Cost by Activity

Enterprises do not transition to SOA in one easy pass. SOA evolves within the enterprise, increasing the degree of SOA Maturity through this transition. Immature organizations need to devote more time and energy to activities related to vision, planning and education while more mature organizations need to invest more in provisioning, monitoring and optimization.

SOA Maturity Level

- Emerging SOA Maturity – this level of maturity is characterized by:
  - No formal SOA software development process
  - Little communication between project teams across the enterprise
  - Nonexistent, immature and/or unstable SOA architecture
- Managed SOA Maturity – this level of maturity is characterized by:
  - Enterprise Architecture team is in place with defined reference architecture and development practices that can be applied to SOA initiatives
  - Communication occurs among project teams
  - Policies are in place that provide visibility across the enterprise
- Optimized SOA Maturity
  - A significant and useful set of services is available
  - Governance processes and policies are well established

- ➢ Organization is knowledgeable and has embraced SOA
- ➢ Focus shifts from deploying to optimization

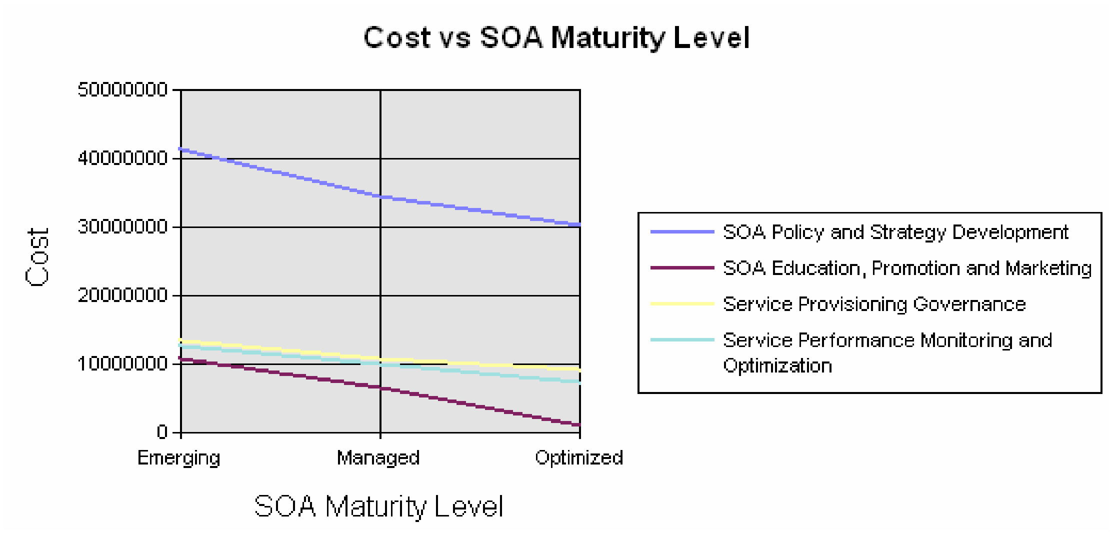

Figure 3: Impact of SOA Maturity Level on Cost by Activity

Two other factors that have been found to influence the amount of enterprise SOA governance required are based on a quantification of security, reliability, documentation, testing and rigor generalized across the enterprise.

- Security Level
  - ➢ Represents the extent to which security is a general requirement for software deployed in this SOA
  - ➢ Levels based on the Evaluation Assurance Levels called for in the Common Criteria for Information Technology [4]

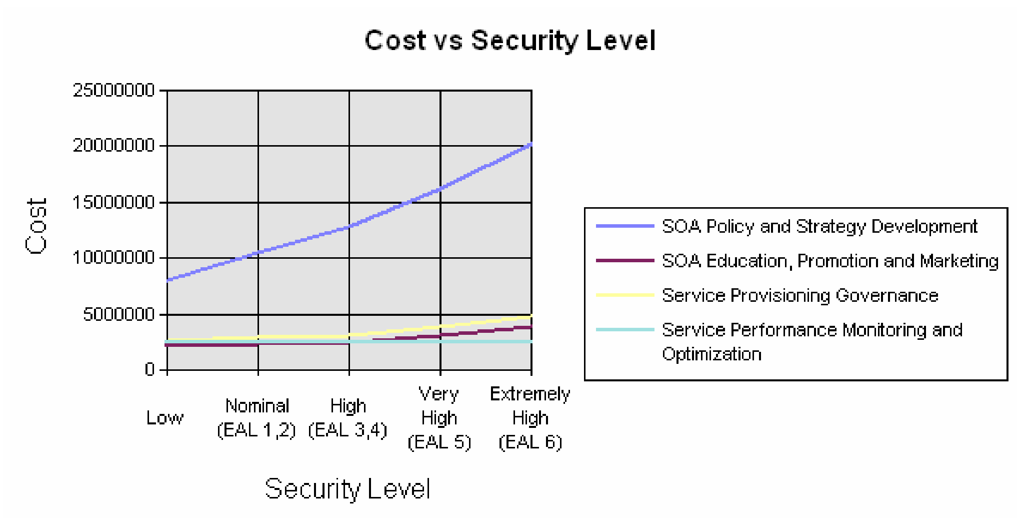

Figure 4: Impact of Security Level on Cost by Activity

- Operating Specification
  - Intended operating environment (commercial, military specification ground, military specification airborne, etc.)
  - Incorporates the degree of reliability, portability, documentation and testing

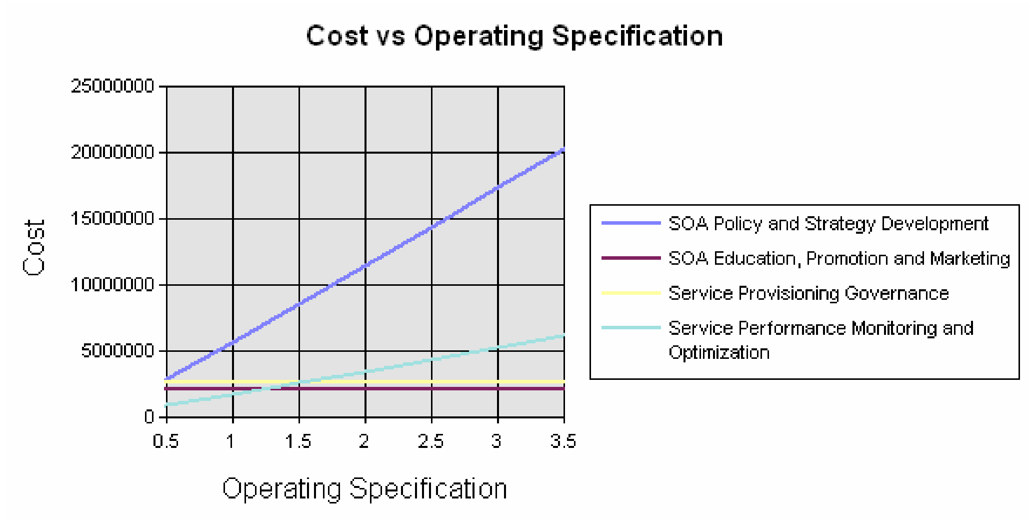

Figure 5: Impact of Operating Specification on Cost by Activity

Service Development

Services need to be developed to implement business rules and mission capabilities. At first glance this sounds like software development that should be estimable using the same methodology that applies to any other types of new software development. Up to a point this is true.

If SOA is being properly governed, services are being developed for multiple stakeholders not just to satisfy the current requirements of a specific project. Each of these stakeholders is likely to have a slightly different view on the best way to deliver a particular service. Negotiating the solution that best meets all stakeholder needs will take additional time and effort. Services need to be developed and tested for maximum reusability. They also need to be designed to not just meet the immediate need for capability or data; they should be designed with some thought toward anticipating non intended uses of the capability.

Relevant cost drivers

It seems that traditional software estimating models can be easily adapted to model service development though there are several considerations that need to be incorporated into such an estimate.

- Operating Specification
    - Intended operating environment (commercial, military specification ground, military specification airborne, etc.). This is a quantification of the amount of rigor required to meet requirements associated with reliability, quality, documentation and test
    - Modeler should indicate that in addition to conditions imposed by the operating environment, the requirement for the creation of reusable capability within that environment should be considered
- Design Reuse
    - Modeler should indicate that significant reuse is expected across multiple platforms
- External Software Integration Complexity
    - If the exercise is to determine cost or effort for the service development, then no external integration is required
    - If the exercise is to determine the cost of a deployment that includes the development of the service and it's integration with middleware or other services to compose an application, standard guidance for software integration should apply
- Productivity and Personnel factors
    - It may not be legitimate to assume that organizationally accepted values for these types of parameters apply to SOA projects in early stages of SOA Maturity
    - Modeler should evaluate how well existing processes and practices can be adapted for SOA before determining the correct value for these types of cost drivers

SOA Middleware Integration

SOA infrastructures contain a set of middleware applications that provide necessary technology components to realize interoperability and promote capability sharing. A wide range of middleware offerings exist and each can be customized for specific organizational or project needs. Common categories of middleware include the following (note that there is often overlap between capabilities provided within each category):

- Enterprise Service Bus – provides a layer of abstraction for messaging between services isolating the coupling between the service and the medium for transporting messages[5]
- Enterprise Service Management – provides management of services via monitoring, metrics collection, alerts on operational status and reporting.
- Registry/Repository – provides capability for discovery and evaluation of available services
- Business Process Management – provides capability to implement and enforce business processes in an integrated holistic approach [6]. BPM provides a platform for modeling business process, reporting and dashboards, content management and tools for communication and collaboration
- Security Middleware – imposes security and identity enforcement

Arguably, the activities associated with configuring and integrating middleware capabilities for Service Oriented Architecture are no different than those associated with any commercial off the shelf (COTS) deployment. In general this is true, although there are certain cost drivers that need to be considered in light of specific characteristics of the SOA initiative. Middleware integration activities can be classified into one of two categories; the initial integration of multiple components to create the environment which enables SOA, and the on-going integration activities that occur as new services are incorporated into that environment. While configuration is an important aspect of each of these types of middleware related activities, the integration component of the former is significantly greater than that for the later.

Accurate assessment of the ‘size’ of middleware functionality being utilized is crucial to successful estimation of the integration effort. In the context of traditional cost and effort estimation, an assessment based on functionality via function points or use cases is the best mechanism for quantifying middleware ‘size’.

Relevant cost drivers

- Functional Size
    - Indicates the amount of functionality that needs to be configured and integrated

  - Measured using function points, use cases, or some other functionality based 'size' metric

- Tailoring Size and Productivity
  - Significant configuration should be anticipated and planned for during initial stages of a SOA initiative
  - It is likely that additional tailoring is required as new services are developed and composed into applications. The extent of tailoring necessary for each initiative needs to be evaluated based on the requirements for that initiative
- Software External Integration Complexity
  - When modeling to determine the effort associated with the deployment of SOA infrastructure, complexity should be assessed based on number of integration points and experience of the integration team
  - When modeling middleware in the context of its requirements for configuration during service deployment, integration complexity should be very low or completely disabled
- Productivity and Personnel Factors
  - It may not be legitimate to assume that organizationally accepted values for these types of parameters apply to SOA projects in early stages of SOA maturity
  - Modeler should evaluate how well existing integration processes and practices can be adapted for SOA before determining the correct value for these types of parameters

Conclusions and Future Work

Research has indicated three areas where traditional software estimating methodologies fall short or require adaptation when applied to SOA initiatives. The three areas identified are SOA governance, service development and middleware configuration and integration. Recommendations for methodology adaptation and new perspective on cost drivers have been presented for service development and middleware integration.

A method for defining domain and project size coupled with cost estimating relationships for SOA governance has been developed and implemented as a prototype. Because the data available for SOA governance costs is very limited, much of this research relied on theory and experiential data. The cost drivers present an excellent framework for future data collection in order to validate and refine the relationships developed through this research. The next step in this research requires identification of emerging and in-process SOA initiatives available for data collection in order to validate these results.

References

[1] McGovern, J, et. al., Java Based Web Applications, Elsevier Science, 2003.
[2] www.service-architecture.com
[3] http://en.wikipedia.org/wiki/Information_technology_governance
[4] www.commoncriteriaportal.org
[5] http://en.wikipedia.org/wiki/Enterprise_service_bus
[6] http://en.wikipedia.org/wiki/Business_process_management#BPM_technology