Adoption of Service Oriented Architecture for Enterprise Systems in Education: Recommended Practices

Version 1.0 White Paper

September 2009

Copyright © 2009 by IMS Global Learning Consortium, Inc.

All Rights Reserved.

The IMS Logo is a registered trademark of IMS GLC.

Document Name: Adoption of Service Oriented Architecture for Enterprise Systems in Education: Recommended Practices

Date: 14 September 2009

Executive Summary

Education has many unique challenges associated with integrating business and academic processes and technologies. This paper, a product of the IMS Global Learning Consortium’s Service Oriented Architecture (SOA) project group, filters the information on the current state of SOA concepts, tools and practices and provides guidance on when adoption of SOA is appropriate in Education to overcome some of its core challenges. Based on both theoretical practice and real world expertise, the contributors to this paper have worked to provide real world use cases where SOA adoption has proven to be beneficial.

This document approaches SOA in the same way that many practitioners recommend adopting SOA: incrementally or phased. The framework of an open SOA maturity model is used to give the reader benchmarks for their own institutional adoption of SOA and phased goals for where they want to take their institution in an SOA world. The tools of SOA are evaluated with an incremental approach in mind to inform the reader of when to potentially consider certain common tools used in a SOA. Further, SOA governance is discussed as a concept that is incrementally adopted as an institution moves through the levels of SOA maturity. The intention of this incremental approach to SOA concepts, tools and practices is to give an institution a means to actively begin SOA adoption without a full replacement of the institutional IT infrastructure. Also, the incremental approach to SOA allows an institution to evaluate business processes on an as needed basis to begin the transformation of those processes, if needed, without completely changing the way business is done at an institution.

The intended outcome of SOA adoption as recommended in this paper is to improve interoperability both internal and external to an institution, to realize cost savings over time by adopting reusable and open standard IT services, and to align IT services ever more closely with the services that the institution provides to its ecosystem. These outcomes can be achieved today in education through a phased service oriented approach.

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Scope & Context

This document seeks to present the elements and good practices of a Service Oriented Architecture (SOA) in the context of the needs of Education. It is not the aim of this document to detail every aspect of SOA as there are several existing public resources that provide that information. One such popular, vendor-neutral resource is http://www.soaprinciples.com/, which introduces service orientation in a broad, non-industry specific fashion.

Other industry bodies have also created SOA models and definitions – one example is The Open Group that has a SOA workgroup, linked here http://www.opengroup.org/projects/soa/. Another example is the Organization for the Advancement of Structured Information Standards (OASIS) SOA Reference model work http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm.

Furthermore, there are open methodologies that focus on Enterprise Architecture, such as The Open Group Architecture Framework (TOGAF) http://www.opengroup.org/architecture/togaf8-doc/arch/, which sometimes prove to be excellent starting places for the creation of a Service Oriented Architecture and are thus useful to understand.

The focus of this document is to explain how SOA is applicable to the challenges that Educational institutions typically face including:

- How to best leverage existing technology investments and align them with changing business requirements;
- How SOA can help with seamless third party provisioning of valuable services;
- How to assess which tools and products I need to get started with SOA;
- How to improve access to learning for students that span multiple institutions and organizations;
- Why SOA governance is needed and how to get started with it.

The document begins to address some of these concerns and highlights established patterns in seeking solutions to these problems using SOA.

One goal for this document is to act as a reference and guide in helping Education organizations tackle the types of issues highlighted above using a services based approach.

1.2 Structure of this Document

The document takes the reader through an introduction to the elements of a service-oriented architecture, including the available tooling support to accelerate the creation of an SOA. The latter sections of the document then illustrate the services approach with specific examples and system patterns for creating solutions to the defined problems. The common thread throughout the document is a focus on illustrating how to start with a small SOA project, get a positive Return on Investment (ROI), and then expand to cover larger problems. The detailed structure of this document is:

2. Service Oriented Architecture for Education	Setting the context of SOA for Education;
3. Business Processes and Services in Service Oriented Architecture	Describes the characteristics of a service and of service oriented architecture, and addresses the relation of SOA to the underlying business processes used in education;
4. Relevance of SOA Governance	Describes the importance of SOA Governance and its relationship to Corporate and IT Governance;
5. SOA Tools: Maintaining SOA Principles in the Face of Complexity	Discusses the nature of the tools that are required to support SOA based systems particularly as the maturity of the systems increases;
6. Solving Integration Challenges using SOA	Introduces the concept of the SOA maturity model and looks at the process by which the Open Group's Service Integration Maturity Model can be used to improve the SOA maturity of a system;

7. Integrating Enterprise Applications in an SOA context in Education – An Integration Scenario	Considers the scenario of making it easier to integrate several enterprise applications using the IMS GLC's Learning Information Service specification;
8. Enabling Shared Services & Cloud with SOA – A Virtual Desktop Shared Service Scenario	A review of how the SOA maturity can be improved for a virtual desktop shared service scenario for collaboration between multiple organizations and/or multiple departments;
9. Bringing on New Systems with a Services Approach – Financial Aid Scenario	Considers the scenario where new systems to support student financial aid are introduced using a services approach;
10. Cross Institution Records – A Learner-centered ePortfolio Scenario	Considers the scenario for the creation of cross-institutional ePortfolios and the use of SOA to create trusted systems;
11. Accelerators to Adopting a Service Oriented Approach – Getting a Quicker Return on Investment	Identifies the ways in which the adoption of SOA can be accelerated in terms of organization level, organizational application integration, service design and implementation, and project roll-out;
Appendix A – Abbreviations	The list of the abbreviations used throughout the document;
Appendix B – Glossary of Terms	A glossary of the key terms used throughout the document;
Appendix C – Overview of Relevant Standards & Specifications	A short review of the specifications and standards that are referenced within this document;
Appendix D – The Original Scenarios	A summary of the set of scenarios for the adoption of SOA that are addressed within this document.

2. Service Oriented Architecture for Education

2.1 What is SOA?

Service oriented architecture is an architectural principle that positions IT services as the primary means through which business services are offered by the organization to its ecosystem. Therefore, SOA offers the prospect of better alignment of Academic and Administrative goals and Information Technology (IT) solutions in Education Organizations.

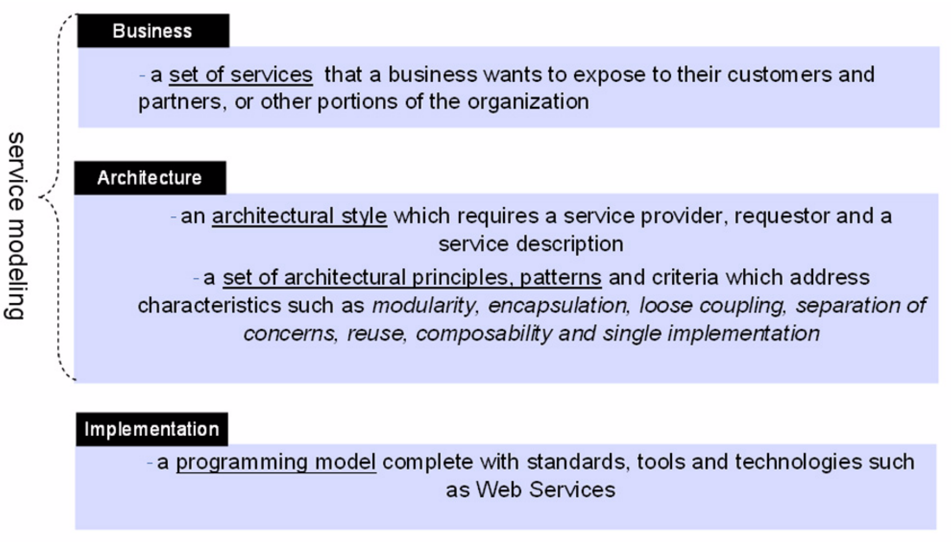
SOA is an approach to IT solutions that is driven by the Organization’s needs. IT solutions are delivered using small modules called “services” that support the explicitly described needs of the Institution. These IT services are designed for use by more than one IT solution (for example many Education systems require information about “students” and IT services about students can be shared). Over time new IT solutions can be built from already existing services. Existing applications can expose themselves as a set of IT services, whilst new IT systems may be implemented in smaller modules with IT service interfaces included in the original designs.

Change is a constant in any environment, and being able to respond to change without investing considerable time and effort in modifying IT systems is the primary benefit of SOA. Furthermore, SOA enables the Education Institution to not only transform internal systems to be more service oriented, but also permits collaboration amongst Educational Institutions, other partners as well as third-party provisioning of useful business functions in a seamless, non-disruptive fashion.

This general view of SOA will be modified by the perceptions of the stakeholders. Within most organizations it will be useful to consider at least three distinct and useful perspectives into “What is SOA?”.

2.1.1 Different Perspectives into SOA

SOA is viewed differently based on perspective of a business owner, an architect or an implementer:



- a set of services that a business wants to expose to their customers and partners, or other portions of the organization

- an architectural style which requires a service provider, requestor and a service description

- a set of architectural principles, patterns and criteria which address characteristics such as *modularity, encapsulation, loose coupling, separation of concerns, reuse, composability and single implementation*

- a programming model complete with standards, tools and technologies such as Web Services

Figure 2.1 Perspectives into SOA.

As highlighted in the Figure 2.1, the enterprise function and architectural view into SOA is primarily about services (both academic and administrative) that the Education Institution offers, and thus to re-align IT systems such that they provision the requisite academic and administrative services.

This alignment is achieved by using tools that have been developed for the business world such as business process analysis. The tools are used to analyze and document academic as well as administrative processes and the lessons learned are fed back into these tools and for the creation of appropriate open standards.

For example, if an Education Institution offers a service to students whereby they can enrol in coursework across multiple Education Institutions, it is important that the IT systems that enable this Enterprise Service are designed as a set of IT Services to meet the requirements of this offering. If IT systems do not align with interoperability standards, such as the IMS GLC Learning Information Services specification [LIS, 09], it becomes increasingly expensive to respond to changes in organizational needs and priorities. This steady rise in cost of change implies that fewer and fewer projects can be undertaken cost effectively, which in turn will exert a steady friction on the competition for resources of the organization and its ability to offer new Enterprise Services as the IT systems will not be able to be developed to support them.

2.2 When Should SOA be Used?

Service oriented architecture is not the right answer for all technology problems faced by Education Institutions; however it does offer value in a variety of situations. Some of the more common situations where SOA lends value are:

- New IT Solutions requiring integration of Enterprise capability from disparate IT systems and programming models
  [*See*, "Integrating Enterprise Applications in an SOA Context in Education – An Integration Scenario" , Section 7];
- Increased efficiency in working with partners for driving revenue, saving costs and for off-loading non-core functions
  [*See*, "Enabling Shared Services & Cloud with Service Oriented Architecture – A Virtual Desktop Shared Service Scenario" , Section 8];
- Leveraging existing enterprise assets by making them accessible for reuse outside their original purpose;
- Permitting organizations to change quickly in response to changes in the marketplace or competitive threats
  ["Bringing on New Systems with a Services Approach – A Financial Aid Scenario" , Section 9];
- Maintaining cross-institution academic records
  [*See*, "Cross Institution Records – A Learner-centered ePortfolio Scenario" , Section 10].

To contrast, some of the situations where SOA is typically not the best answer are:

- Only a small percentage of your organization's IT budget is spent on integration activities;
- A majority of your organization's processes are manual, with little opportunity for automation;
- The operation of your organization is managed by one or two Enterprise Resource Planning (ERP) applications with little or no integration requirements.

Organizations that undertake SOA projects typically start with a small project, restricted to one or two departments within the organization, and only upon achieving success and establishing a positive ROI, expand to the rest of the organization.

As some of the examples above show, succeeding with SOA is not always about transforming the entire organization. Frequently, focusing on a specific problem area or objective is an excellent starting point for increasing the service orientation of an organization.

Using the aid of scenarios that lend themselves to a service oriented approach, this document describes solutions, good practices and use of appropriate standards to accomplish the goals of the organization while preparing it to be responsive to future change.

2.3 Why is SOA Important for Education?

Educational Organizations are interested in services oriented architecture because it promises to reduce the complexity of integrating systems while at the same time aligning the enterprise business processes with the underlying technology used to deliver those processes. However, the lack of a common blueprint for SOA is preventing many Education Institutions from adopting the practice.

It is a matter of fact that the typical Education Institutions must provide enterprise services to an ecosystem of many different people: administrators, librarians, professors, assistants, students, local and global communities, researchers, security and maintenance personnel. Because of the varying needs of this ecosystem, most institutions find themselves forced to manage a complex heterogeneous environment comprised of new and legacy IT systems. Each IT system may require a different server, operating system, messaging platform, database and application. Data must be shared among these systems to support the enterprise processes, so systems integration is a necessity.

However, systems integration is difficult to achieve and manage. As the number of enterprise services (and supporting IT systems) increase, the number of integrations required increase at an even faster rate. At first, traditional point-to-point integration might seem easy to manage, but over time, this approach becomes extremely complex, making it difficult for an institution to analyze the impact of replacing an enterprise service, upgrading the IT system that supports it, or opening the service up to new and different users. Institutions are often hesitant to undergo any transformation because of this underlying complexity, so they become less responsive and less able to keep up with the demands of their user ecosystem. Furthermore, the traditional style of integration was usually undertaken in an inconsistent manner, through ‘nightly batch file updates’, remote-procedure calls, database updates, and the like. As a result, any data that is modified is rarely up-to-date in all systems, leading to errors in fees, course enrollments, addresses, and payments.

In a services-oriented approach, there is a continuum of best practices that can be used to describe activities related to application integration, development and deployment that yields the benefits of a flexible, loosely coupled architecture. Some of the core best practices have been codified as a set of principles. There have been various definitions of these principles including the eight common principles articulated by Thomas Erl [Erl, 07] on his http://www.soaprinciples.com/. Foremost amongst these principles is that of loose coupling[1]. In other words, it does not matter how the services communicated or what the communications looked like, it only mattered that the communication was documented and well understood. For example, a website that was receiving and processing a request was considered SOA. It didn't matter if the request was in an http envelope with a message that could only be parsed in a proprietary way. Many of the early implementations of service-oriented approaches in Education have been at this end of the spectrum.

Service oriented architecture requires more than a services approach. In addition, important fundamental aspects of SOA were missing in those attempts, such as:

- Alignment of Business Services with Application Services;
- Use of standardized mechanisms for discovering & communicating between services;
- The importance of information consolidation as an integral part of a service oriented solution;
- Exposing business capabilities as services, permitting the creation of new business processes.

SOA brings a strong focus on service to IT systems at the enterprise, but it does not do away with the need for enterprise architecture. In other words, it is still important to understand application capabilities, data capabilities and integration requirements between the various IT systems at the enterprise. SOA helps build useful services on top of these underlying IT systems, providing new ways for integrating the various underlying systems in support of a services paradigm. Presence of enterprise architecture greatly simplifies the creation of a service-oriented architecture with its resultant benefits (for more details, see “Solving Integration Challenges using SOA” , Section 6).

This document serves as a foundation to help clarify the elements of SOA as applicable to Education and how SOA can help organizations leverage their investments in current IT systems to address new and increasingly complex requirements from the user ecosystem.

2.4 Does SOA Need Re-architecting of Applications in Education?

Service Oriented Architecture, as pointed out in earlier sections, seeks to make Services as the cornerstone of IT architecture, with service choreography in support of business processes as the primary method of integration. By applying open standard methods of communication (such as Web Services) and open standard data formats (such as from IMS GLC, PESC and others), the goal is to build an architecture that is open, business aligned and flexible.

[1] Classic software engineering best practice recommends low coupling and high cohesion when designing and implementing functionality.

An ideal SOA will thus consist of only services, ready to be combined in new and different ways in response to change in business. In this world view, applications as we know them would be replaced by an inventory of services, many of which would be reusable and ready to integrate.

In most practical situations in Education, service oriented architecture would have to be built upon existing investments in applications that deliver significant current value. Re-architecting or "ripping and replacing" these applications is cost-prohibitive and usually not worth the effort.

Fortunately, SOA delivers value in these situations even as:

- Applications are exposed as Services aligned with the business processes they serve, and thus expand their use beyond their original intent even as additional processes can now leverage the services rendered by these applications;
- Integration between applications is driven by choreography of the services they connect, in support of business processes;
- Service connectivity is open and standardized, permitting easy, flexible integration;
- Infrastructure tasks such as routing requests, mediating protocols and adapting message formats are handled through an Enterprise Service Bus, thus centralizing the burden of mediation and routing.

"Figure 3.1 Application of SoA to education." presents a good reference picture of what SOA would look like typically in an Education institution.

Exposing applications as a set of services is typically a transitional roadmap. "Figure 6.1 Integration capabilities with SOA in education." lays out the typical path of maturation even as an application that has not been re-architected as a collection of Services seeks to participate in a SOA world – starting from file-based exchange through Information as a Service and maturing into Business Capabilities as a Service.

Once an Application has been correctly exposed into a set of business services, these services are readily available for use by other services, service clients or external partners. In effect, they are no different from services that have been built from ground up as they provide the same characteristics of a service as identified in "Business Processes and Services in Service Oriented Architecture" , Section 3.

Therefore, SOA delivers significant value to its adopters even as it encourages application providers to increasingly expose their applications as a set of services/service interfaces for easy use and reuse by the Education community.

3. Business Processes and Services in Service Oriented Architecture

3.1 What is a Service?

IT services are distinguished from general business services i.e. the general operational capabilities provided by a business to its customers. In IT systems, a service is a discrete element of functionality, used to support a business process or function. Typically a service is part of an overall business process of an enterprise (the workflow or set of operations). In a service-oriented computing environment, services become the building blocks used in constructing an overall computing environment or IT system to solve the overarching business problem. To be useful, a service must provide a defined capability to do something. The service must be implemented in software, and must be deployed and managed in some way. Services may be used in combination to support the needs of the business, each service providing one part of the overall operation. While there is no formal agreement on all aspects used to characterize a service, key characteristics include [Erl, 07]:

- *Defined* – services are defined in terms of what they do e.g. the process(es) they perform, the interface used to communicate with the services e.g. how to invoke the service to perform the process, the data that is passed to, or returned from, the service, and how the service is managed;
- *Implemented* – the service has to be implemented in software that: (1) responds to requests to perform its function, (2) performs the requested process, and (3) may return or disseminate results;
- *Deployed* – to be used, the service must be made available for use by others. The definition of the service may separate how it works in the abstract form with specific “bindings” used to convert an instance of the implementation to a functional element that can be accessed in a computing environment;
- *Managed* – the deployed service implementation is under the control of some management authority that insures, to some prescribed level, that the service is available and operates as defined, and provides the necessary underlying IT infrastructure to manage the operation of the service;
- *Reusable* – by providing defined, discrete functionality, independent of how it is used, a service can be used (or reused) at any point in the overall business process where the functionality is needed;
- *Communicating* – a service is accessed by sending it a request i.e. communicating with the service from the IT system or from another service. Results are communicated to those that use the service. The service is part of a consumer/producer environment, with data exchanged between these entities;
- *Abstracting* – services (through their operations) define only what process the service provides and how to communicate with it. How the service is implemented to provide the defined functionality is not defined. The service provides an abstraction boundary, hiding implementation details, or hiding the structure of the resource that the service manages;
- *Composable* – a service may be combined with other services to implement the overall business process e.g. what is behind the abstraction boundary of a single service may be a composition of other services.

Services also have additional implementation- and operation- specific characteristics as opposed to the inherent characteristics above. These new characteristics are

- *Granularity* – the complexity of the business process provided by the service may range in scale. The service may define a small process (or many small processes) where each request performs a simple operation and many requests are needed to complete a business process i.e. *fine grained* or the service may provide a substantial process that maps directly to a business-level step i.e. *coarse grained*;
- *Coupling* – while services need to interact to solve business problems, a service may be defined to be independent of any other service and can function without the knowledge of how other services work, or it may have explicit knowledge about not only what other services are available, but how they perform their operations. Services that are not dependent on other services are *loosely coupled*, while those that require other services are (more) *tightly coupled*[2];
- *Autonomy* – what a service requires to perform its operations is local knowledge only to the service. Autonomy is closely related to both coupling and abstraction;

[2] This does not imply that services should not be composed with other services to achieve higher order business process functionality. Instead it stresses the need for functionally cohesive common services that can be re-used to underpin other application-specific services.

- *State* – to fulfil a request, the operational service may need to know about historic use and invocation of the service i.e. to access data about the “state” of the business operation maintained by the service itself. The service may require no information about state i.e. *stateless*, or it may require knowledge of prior operations, i.e., *stateful*;
- *Discoverability* – service definitions may be made available so that existing services can be found, enabling reuse and composition. Discoverability is independent of the service itself, but part of the overall environment in which the service is defined and managed.

3.2 Business Processes in a Service Oriented Architecture

A Service is a closely related set of operations that are encapsulated together. Generally the service is independent of other services i.e. the service does something useful and discrete in terms of the business process. To model and perform the overall business process, a set of services must be assembled into an overall service environment. Service granularity is defined by the functional context of the whole service. Services that are intended to cover a larger functional context are considered to have a coarse granularity. On the contrary, fine-grained services expose a narrow specialized functionality. Looking from the perspective of the type of service under consideration, you could have:

- Utility services (Infrastructure services) – usually defined by grouping together infrastructural capabilities with a common purpose, for example a Logging Service, and thus granularity is directly defined by the utility functions supported;
- Entity services – these have their functional context scoped by the entities that they manage. For example, granularity of a Student Management Service is defined by the Student entity. In this case Student Management Service would include all the capabilities that are necessary to maintain the Student entity;
- Task services – these typically contain a group of capabilities related to the same business task, for example Tax Calculation Service. Task services tend to be fairly granular;
- Process services – these usually deal with a larger functional context defined by the encapsulated business process. For example, Financial Aid Management Service would include all the capabilities necessary to manage financial aid.

SOA is one of the architectural frameworks (a collection of guiding design principles, a vocabulary of types of components and assembly rules) used to define and build IT systems. As the name implies, SOA is based on building systems that are composed or built from services. Like other architectural frameworks, SOA is defined in terms of a set of specific characteristics that identify how the services are defined (the vocabulary of types) and how they interact (the assembly rules). The key characteristics of SOA follow those outlined previously i.e. defined, implemented, etc. Where there is a range of possible values for a characteristic e.g. state, coupling, SOA defines specific measures of the characteristics (design principles). While these are the preferred values for these characteristics, there are no hard and fast rules to say that an approach that violates one of the characteristics or principles does not correspond to SOA.

As an architectural framework, SOA does not define the technologies used to build an operational system based on SOA. Web Services, while the most common set of technologies used to implement SOA, is not the only one possible. Equating SOA and Web Services is incorrect; SOA is the model, Web Services is a realization of the SOA model used to build operational IT systems using a particular technology platform. The definition of SOA is based upon a set of service characteristics and design principles applied to services, namely: *defined*, *communicating*, *reusable*, *abstracting*, *composable, coarse grained, document oriented*, *loosely coupled, autonomous*, *stateless, discoverable, managed* and *providing QoS*.

While these are the principles that define SOA, using SOA encourages as side effects: reuse; interoperability; layering of abstractions; business modelling; and component technology/vendor independence. It does not guarantee these characteristics of a solution, and as noted, some of the SOA principles may be violated when real IT systems are built and deployed.

The relationship between a business process and its realization using SOA reflects the tension typically encountered when using technology to alter and/or improve a business process. A new technical solution can be used to mimic the original business process (including those activities that were manually undertaken) or the business process itself can be changed to achieve a particular improvement e.g. to reduce duplicated activity, etc. The extent to which a business process should be changed is more of an issue for change management but it is important that the use of the new technology is not perceived as the main or only driver.

3.3 Service Oriented Architecture for the Education

It is not the intention of this White Paper to document an exemplar pattern for the application of SOA to Education as a whole or even for a sector such as K-12/Schools, Higher Education, etc. Instead, we aim to show how the concepts of SOA can be applied to the needs of Education. Any organization providing Education has to address much more than just the actual delivery of learning; from a systems' perspective, while the delivery of learning is the key activity it constitutes only a small amount of the required functionality.

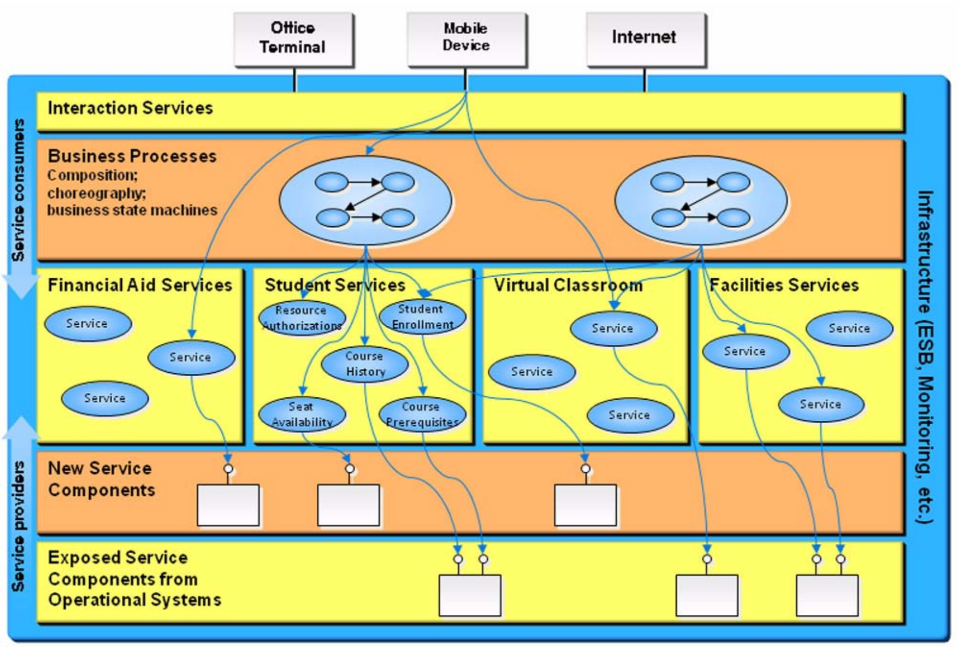


Figure 3.1 Application of SoA to education.

Figure 3.1 is a schematic representation of the use of SOA for a generalized educational provision. The five layers, from the top downwards, are used to provide:

- Interaction services – the delivery of the services to the various user communities. This delivery should be transparent of how the services are supported, must address accessibility issues and should make use of a wide range of delivery devices e.g. mobile, smart-cards, PCs/laptops, etc. Both synchronous and asynchronous interactions must be supported;
- Business processes – the combination and co-ordination of the business services to achieve the specific business processes. For education this would include course registration, course administration, student smart card services, etc.
- Business services – this is the set of services required by the organization. These services must reflect the business processes and so for education include, for example, virtual classroom, financial aid, facilities (library, catering, accommodation, etc.) and student specific services;
- New service components and generic exposed services – each of the IT services has to be supplied. This provision could be via the creation of new IT services or the adoption of generic IT services more broadly available. For example, access to off-campus libraries could be through the appropriate local public library portals whereas course registration would require use of the specialized on-campus student information service.

Underlying this set of services is the communications infrastructure and the broader management services.

While the details of the set of business processes and services required will differ for K-12/Schools, HE, Community College, etc. the underlying approach is identical. Indeed, these different sectors will make use of common generic services e.g. access to the local public library portals. This service approach means that a service provider can make changes in their provision without changing the way in which a user makes use of the service. This reduces the cost of maintenance and support, permits the service to be improved without requiring changes to the services that make use of it and improves the user experience by providing a consistent experience.

One important example of the use of SOA in education is the work by IMS GLC on supporting interoperability between student administration and learning systems. The IMS GLC Leaning Information Services (LIS) specification [LIS, 09] has been created to address some of the interoperability issues for education (in particular information exchange between student information and learning management systems). The LIS specification is based upon the IMS GLC Abstract Framework (IAF) [IAF, 03] and the IMS GLC General Web Services (GWS) Basic Profile v1.0 [GWS, 05]. The IAF defines the IMS GLC approach to creating SOA-based interoperability specifications that are realized by Web Services in the form of GWS. This work has taken over two years of development by many of the student information and learning management systems vendors. Vendors that adopt this specification considerably simplify the work they are required to implement to integrate with other student information and learning management systems products. End-users benefit from having systems that can be easily integrated with their other student information and learning management systems.

4. Relevance of SOA Governance

No one would expect an automobile to function over a long period unless three key things were true:

a) It was well designed;

b) Its constituent components, fittings, connecting valves and hoses all work efficiently with minimal friction;

c) It is well maintained with regular oil changes, tune-ups and the like.

IT systems and architectures are no different and are dependent on initial design, integration strategies between systems and ongoing maintenance – all of which are ultimately governed by the set of policies that the IT department adopts.

This section discusses how these practices, called SOA Governance, apply to SOA, why they are important and how to best apply them in a typical Education Institution.

4.1 What is SOA Governance

In a nutshell, “governance” means the set of practices, policies, and processes that are put in place to increase the likelihood that an entity functions as intended. Corporate governance is the set of practices and policies that dictate the way a corporation is controlled and directed, while IT Governance is the set of practices and policies that ensure that IT investments are generating the intended value for the Organization.

SOA Governance is the set of practices and policies that ensure service oriented architecture and its component services represent and conform to the objectives and requirements of the institution. It involves not only IT support processes to ensure regulatory compliance and quality of service, but also organizational processes to ensure that stakeholders are receiving value by adopting SOA.

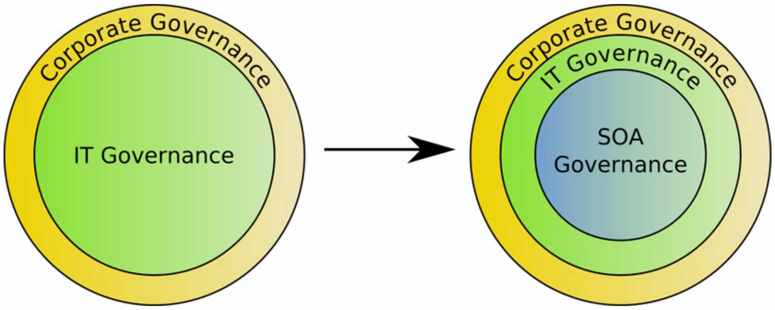


Figure 4.1 Relationship between SOA Governance and IT governance.

4.2 Practising SOA without Governance

Often, new service-oriented projects are undertaken without explicit SOA governance in place. This works if the scope of the project is small, but, as the architecture grows to include new service providers and consumers, the overall reliability and predictability of the architecture begins to decline; this is typically the result of a lack of governance. For example, a lack of attention to IT service boundaries may have resulted in duplicating some old IT service functionality in a new IT service. Or perhaps the demands placed on a certain IT service may have increased dramatically because of a new IT service consumer, and this may have been unaccounted for – resulting in the inability of the IT service to meet its Service Level Agreement (SLA). Furthermore, as enterprise services evolve and new IT service versions are deployed, careful consideration must be taken each time to assess impacts of the changes to other IT services and provide backward-compatibility.

In order to prevent many of these potential problems from occurring, some amount of formal governance is critical even in a new service-oriented project to ensure delivery of consistent and reliable IT services. Fortunately, one does not have to adopt all elements of a full-fledged SOA governance solution to gain many of its benefits, and instead can concentrate on using appropriate parts of governance to gain maximum benefit depending on scope and nature of the individual project at hand.

4.3 Incremental SOA Governance

It is instructive to note that, in most cases, there exists some form of governance over the architecture, although it is often informal. Informal governance is often inconsistently applied, usually proving to be insufficient as the scope of the architecture expands. In order to ensure that the goals of the SOA are understood and achieved, some amount of formal governance should be practiced.

But what amount of governance is the right amount? Some approaches dictate heavy governance from the outset, before there is much chance of most of it doing any good. Many simple situations may require only an assigned person to check periodically for common issues but as the architecture matures, more heavy-weight governance is usually needed. Ideally, an organization only practices as much SOA governance as is necessary to handle the architecture in its current state and allow for flexible growth and smooth operation into the future. Therefore the goal is to roll out SOA governance practices to keep up with the state of the architecture, which is a moving target.

This section walks through how governance would grow along with a typical SOA maturity progression and then discusses a general framework for incrementally adding governance practices as they become pertinent

4.3.1 An Exemplary Governance Rollout

Figure 4.2 demonstrates a SOA maturity path for a fictitious institution. The steps on the left side indicate the ways the institution’s services and architecture change, and the blocks on the right are the corresponding governance practices added at each step.

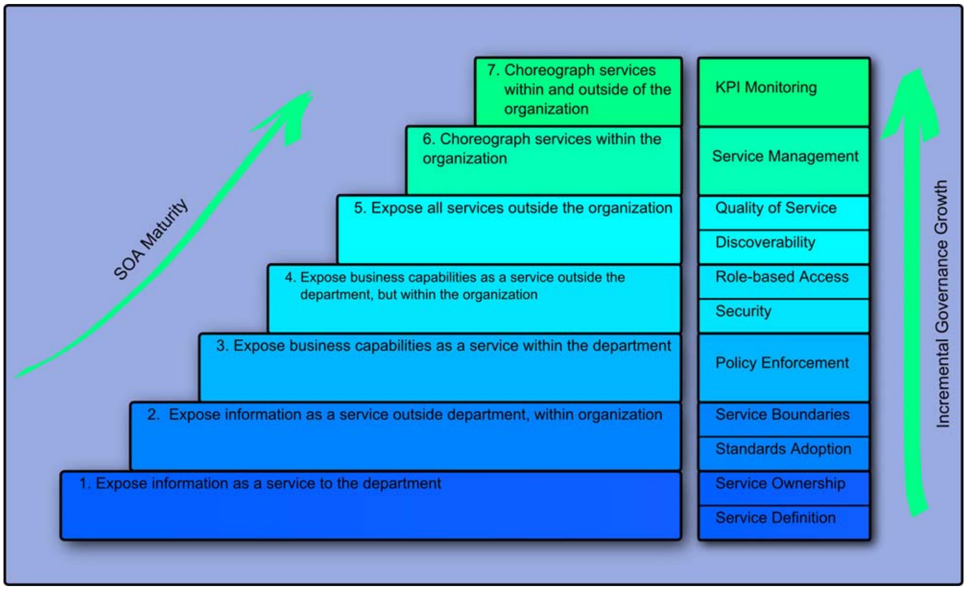

7. Choreograph services within and outside of the organization
6. Choreograph services within the organization
5. Expose all services outside the organization
4. Expose business capabilities as a service outside the department, but within the organization
3. Expose business capabilities as a service within the department
2. Expose information as a service outside department, within organization
1. Expose information as a service to the department
SOA Maturity
KPI Monitoring
Service Management
Quality of Service
Discoverability
Role-based Access
Security
Policy Enforcement
Service Boundaries
Standards Adoption
Service Ownership
Service Definition
Incremental Governance Growth

Figure 4.2 An example of developing SOA maturity.

The engineering department begins by exposing long-running data analysis status and results as a service for graduate students. The first considerations for the department are around designing and modeling the service to function as intended, listed as service design, as well as the need to determine who is responsible for maintenance and who is paying for it, here called service ownership.

In the second step, the department expands its user base to allow other departments to use the service; in order to ensure that the service plays well with the services of other departments, the engineering department starts considering service boundaries to ensure that its services do not overlap in function. Also, in order to maximize interoperability and ease of adoption, the institution employs standard data formats for its services.

The engineering department decides it wants to expose more sophisticated capabilities; the users now want business capabilities from the services, such as re-running previous analyses with different data. In order to support this new level of service requirements, the department starts to implement formal policy conformance auditing for the services. The department intends to automate this as the number of services grow, but decide that it is sufficient to perform the checks manually at this time.

At the next stage, the department exposes these new business capabilities outside the department. Several interested people from each department form a governance board to determine the kinds of security requirements that should be in place for these services with wider usage. Also, because lab results from a particular student are not pertinent to every department, the security board decides to limit record access to the appropriate departments.

The services are becoming fairly robust now, and the institution as a whole is getting a lot of value from the architecture, so the institution decides to expose the services to other institutions. With this new, broader user base, they decide to place overall quality of service requirements on services, including establishing downtime limits during new version deployments. They also recognize that these service endpoints may change over time, so the institution creates a service registry and imposes discoverability requirements on all services to ensure that users can find the services into the future.

With quality of service rules in place, the architecture has reached a point now where the services may be choreographed to execute business processes. This step creates a need for additional control over the lifecycle and state of services, so the institution adds management capabilities to the services and places management requirements on future services.

After seeing success with choreography within the institution, the institution decides to combine its services with other institutions. In order to help assess business value to itself and other institutions, Key Performance Indicator (KPI) monitoring capabilities are added to the architecture, along with monitoring requirements for future services. The data gathered from the monitoring will assist in the decision-making in the future about which services to optimize for increased throughput, augment with new features, or sunset due to lack of usage.

4.3.2 General Practices for Incremental Governance Growth

The above path is only an example; many possible permutations exist of both SOA maturity progressions, and governance practices that are determined to be useful at the next level of maturity. Figure 4.3 depicts the path toward maturity as the number of services, the number of consumers, and the number of services versions grows, as well as the governance practices that are often employed at each level.

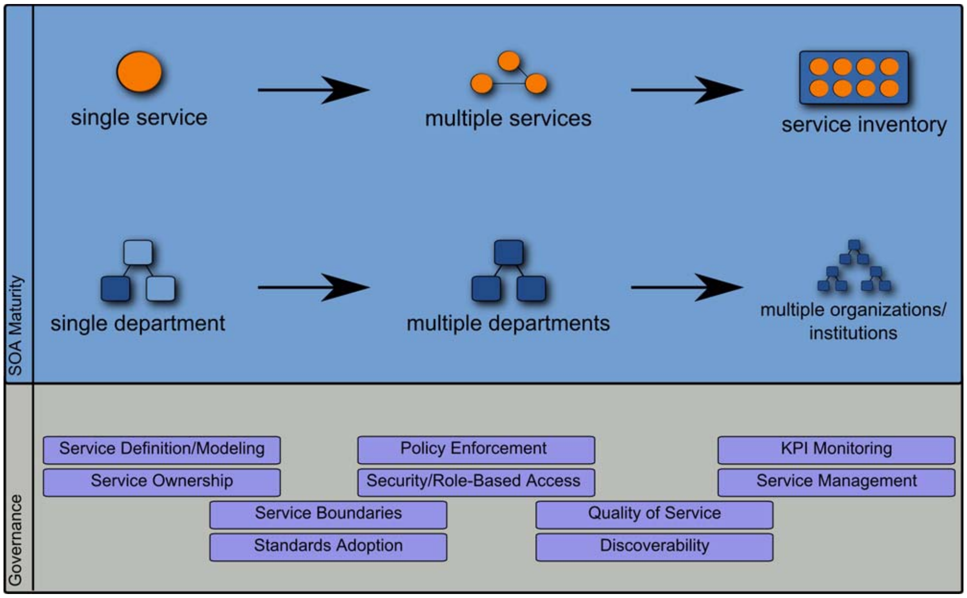


Figure 4.3 Increasing complexity of an SOA rollout.

5. SOA Tools: Maintaining SOA Principles in the Face of Complexity

The service oriented approach is a set of principles that, when applied, create an IT systems environment conducive to accomplishing organizational goals. As the IT systems environment grows, new problems manifest themselves, and attempts to adhere to SOA principles require additional help in the form of development and infrastructure tools, collectively called SOA tools. SOA tools are components that, when applied, enable an architecture to maintain service-oriented characteristics and enterprise alignment in the face of growing complexity. The exact tool types and standards specified here may not be employed directly; often in-house tools are developed to enable SOA functions that have similar characteristics to those mentioned here.

5.1 Incorporating Tools Incrementally

SOA tools are closely tied to maintaining SOA principles. As the architecture moves through the maturity continuum, it becomes worthwhile to utilize new SOA tools to maintain SOA principles effectively. This section enumerates the order that these tools are typically brought into use in the growth of a SOA; this order is depicted in Figure 5.1.

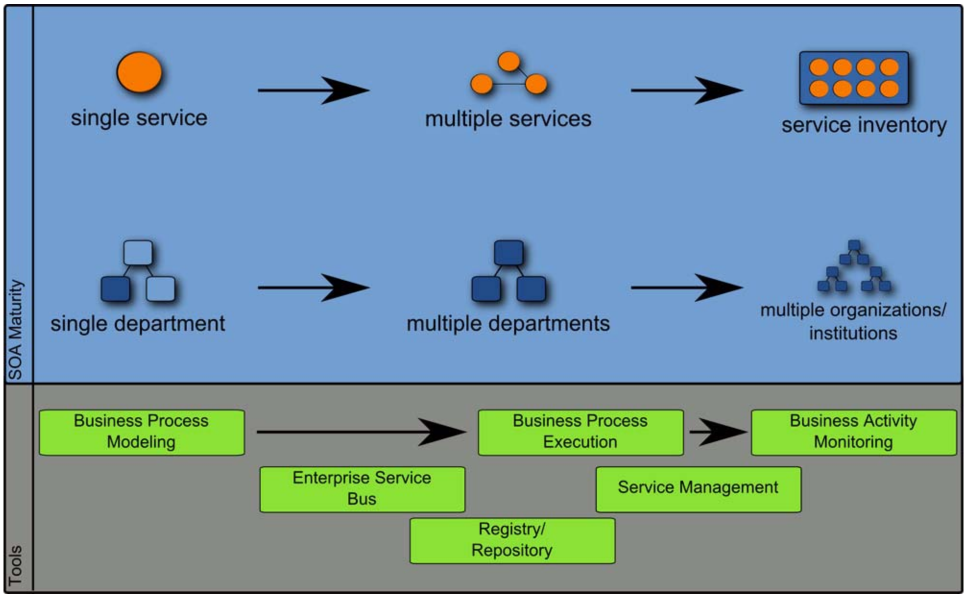


Figure 5.1 Schematic representation of the point of adoption of different tools.

5.1.1 A Blank Slate

An IT systems environment that exposes information as a service to a small group of people is the typical starting point for a service oriented architecture. This architecture addresses few SOA principles, so it stands to reason that few, if any, SOA tools are needed at this stage; this is exactly the case. At this time, the cost associated with employing SOA tools does not typically justify the benefits the tools confer.

5.1.2 Multiple Interacting Services: Business Process Analysis and Modelling

As more services are exposed, and there is a greater degree of interaction between services, at some point questions arise such as which unimplemented functions and services should take priority. Formal business process analysis will help answer these questions. Business Process Analysis involves documenting a business's existing processes, breaking them down into appropriate steps, determining flaws, bottlenecks and superfluous parts, and reworking the processes to improve efficiency or effectiveness; often, an organization learns much from the act of business process analysis including determinations that a given process is not in alignment with the business's goals.

The Unified Modelling Language (UML) and Business Process Modelling Notation (BPMN) are common languages for expressing these business processes. Business Process Modelling Tools exist for modelling and designing business processes using UML and BPMN; they assist with process analysis and rework by providing diagramming capabilities and process simulations. Many plug into existing Integrated Development Environments (IDEs). As opposed to those tools mentioned later, business process analysis and modelling tools are client-side tools that do not impact the operation of the SOA; they typically appear early in the life of the SOA as they can usually provide the artefacts needed to run very powerful tools in the future.

A service-oriented environment is not necessary to derive value from business process analysis – it can be of tremendous benefit in any situation where there are complex processes involved in order to accomplish a common task for the business.

5.1.3 Inviting the Larger Organization: Registries, Repositories and Enterprise Service Buses

With IT service access confined to a small group of users, it is not usually necessary to provide anything other than a web page of URLs and documentation for services, with updates to URLs or service contracts coming through email. If IT service usage is opened to a larger organization, however, it quickly becomes impossible to notify all interested parties about service endpoint address changes and so forth. It becomes necessary to provide a centralized facility for registry and discovery of services, as well as a repository for managing assets related to them. Later, as IT service inventories grow from tens to hundreds of services, it becomes very difficult to manage the environment without a registry and repository.

The repository stores service assets such as WSDLs, XML schema, BPEL documents, service policies and other service metadata, while the registry stores references to the IT services themselves. The registry and repository are often present as a single functional unit of software; for example, software implementing the Universal Description, Discovery, and Integration standard (UDDI) fills the roles of both registry and repository.

Also, as a larger organization is allowed access to IT services, the network in use becomes larger and therefore the routing logic to invoke IT services becomes more complex. Point-to-point integrations start to become an issue, as some services may use different protocols or data formats. The increased number of messages being sent to and from these busier services may require additional decoupling measures such as message processing and queuing capabilities. An Enterprise Service Bus (ESB) addresses these new requirements.

The ESB is complementary to the registry and repository components; while the registry and repository allow the discovery of, and binding, to an IT service, the ESB is a middleware component that facilitates the IT service invocation over a network. It can handle complex routing logic, including content-based routing, protocol mediation, data transformation, and asynchronous queuing. From an IT service consumer's point of view, the IT service is simply invoked over the ESB and all the details are abstracted away, which greatly decouples the two.

There are numerous commercially available ESBs including some free open source implementations. An ESB implementation contains the middleware necessary to provide the function as well as tools to monitor the ESB and simplify its configuration. There are no existing standards regarding ESBs.

Importantly, ESBs permit transformation of data formats from one to the other and as such remove the burden of matching message formats between two applications from the applications and move it to the ESB. There are two distinct approaches to message transformation:

a) Converting from proprietary format of application ‘A’ to proprietary format of application ‘B’. This mechanism effectively creates an implicit point-to-point transformation and thus is not very powerful, as a change in either application implies a change in the transformation logic would be required;

b) Converting from proprietary format of application ‘A’ to an open standard, and then from the open standard to application ‘B’ proprietary format. Even though this approach requires two transformations, changes in either application format are isolated from the other owing to the open standard being used inside the bus. This approach ensures movement of open standards based message formats inside the enterprise service bus, and each connecting application simply needs to convert from the standard to the application specific format. This ensures that the maximum flexibility is delivered for integration purposes.

Outside of providing capabilities for message transformation, protocol mediation and message routing, ESBs are typically designed to handle large volumes of data and can begin going beyond traditional text based messages (XML, etc.) into media rich messages as needed for certain scenarios that deal with rich media as well. Thus the ESB has slowly evolved into a very useful tool to realize flexible, agile service oriented architecture.

5.1.4 Simplifying Service Management

While the architecture has a confined user base and a fairly predictable demand range, much of the administration, configuration and provisioning of services can happen manually. However, as services are exposed to a larger audience such as other institutions, the need arises for more sophisticated service management.

The SOA principle of service management is a blanket term for the monitoring, control, configuration, and provisioning of IT services including security, service level agreements and policy auditing and enforcement. As the number of IT services increases, or more users and services become dependent on an IT service, it becomes increasingly unwieldy to handle these management responsibilities manually.

In order to automate service management, IT services must be instrumented to allow for management, including exposing controls and performance counters. The OASIS Web Services Distributed Management (WS-DM) is a standard that addresses these and other typical management concerns in one specification. Once services are instrumented, various monitoring and management tools exist to automate many pieces of service management while simplifying the rest. This runtime performance and auditing data is usually stored in the service metadata repository.

5.1.5 Making Services Work Together: Business Process Execution and Business Activity Monitoring

In environments with few services and users, it may suffice to allow users to manually string together services to receive needed function, or “hard code” a business process into a script, with changes to the script being made manually as the underlying business process changes. The ability to perform this choreography of composable services is a primary benefit of SOA.

As a service inventory grows and the numbers of represented business processes increase, changes in business processes will create maintenance problems for these scripts and undermine the effectiveness of the architecture in enabling the organization’s goals   It is desirable for an organization to be confident that a given script accurately reflects its underlying business process at all times. Business process execution tools, in concert with business process modelling tools allow such confidence.

5.1.5.1 Business Process Execution

Business Process Execution (BPE) tools use documents or scripts to execute a modelled business process by orchestrating available services, including human sub-processes, parallel sub-processes and fault tolerance. BPM tools can often generate BPE documents, so updates to a business process can quickly be pushed out to the repository in an executable form.

There are several standards related to BPE including Business Process Execution Language for Web Services (WS-BPEL) [OASIS, 07]; notably, the mapping between BPMN and WS-BPEL, its executable form is not complete, so often BPMN use is limited to a subset that maps correctly to BPEL. Web Services Choreography Description Language (WS-CDL) is another XML-based standard that can be used for expressing business process executions [W3C, 06].

5.1.5.2 Business Activity Monitoring

With business process execution in place, the organization can turn its attention to ensuring that the service-oriented computing environment is actually helping to accomplish the goals of the enterprise. Business Activity Monitoring (BAM) tools are, collectively, the tools capable of monitoring not the individual IT assets or services, but the performance and status of the enterprise itself. BAM tools usually provide dashboards for monitoring KPIs.

6. Solving Integration Challenges using SOA

Simplification of integration challenges is a key benefit of a service-oriented approach, allowing information and enterprise capabilities to be re-used and re-composed in new ways in response to changing requirements. Getting the full benefit of a service oriented approach is however a step-wise progression that encompasses improvements along multiple dimensions, operationalized through specific projects.

6.1 Increasing Maturity of Integration with SOA

Most Education institutions typically have multiple applications installed to provide the many different enterprise services expected from the organization. Increasingly, these applications need to integrate their capabilities with each other in order to provide a seamless and consistent experience to students, administrators and staff.

Service oriented architecture provides an enterprise-centric systematic methodology to leverage the data and capabilities carried within these applications, exposed as a set of services and to have these services interact with each other in a flexible and responsive fashion. Having said that, most references on SOA are quick to focus on: Business Services, Choreography of Business Services in support of Business Processes, Composite Services, etc.

The starting point of most integration however needs to begin by taking file-based integration or database sharing towards a more explicit provisioning of ‘Information as a Service’. To move towards a service-oriented approach, the best first step is thus to expose ‘Information as a Service’ from providing applications and prepare the receiving applications as consumers of the service. This has the immediate benefit of:

- Making information widely available within the organization to any system that wishes to access it;
- Standardizing the Information formats thus doing away with point to point transformation requirements;
- Tools such as an ESB can help make this information available widely, while taking care of message routing and transformation.

The IMS GLC LIS specification [LIS, 09] begins to make some of the information in learning systems available through a standardized message based interface. If the standard is implemented as a push-model the corresponding system is classified as at Level 1 of the stair-step towards greater service maturity.

Figure 6.1 shows the rising scope of integration in SOA. Exposing business capabilities such as an IT Service is the next step in the progression towards increasing integration maturity in a service-oriented world. For new services, this is reasonably straightforward, however getting legacy applications to expose their business capabilities may be more complex, depending on the architecture of the underlying application.

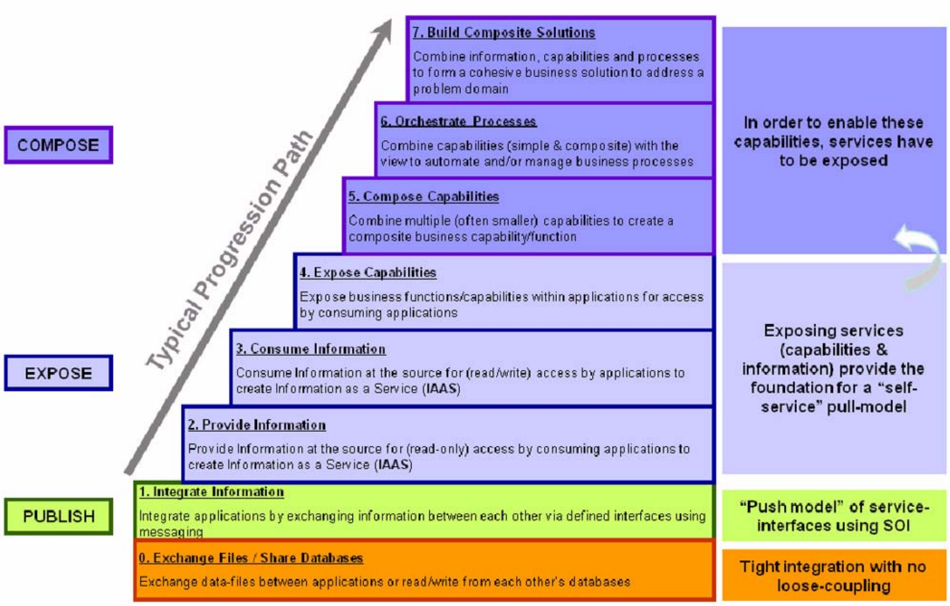
Once an application exposes its capabilities as a set of IT Services, its benefits are manifold as now these applications can more fully participate in any new or modified business processes, be part of composite services and, eventually, play an essential role in the creation of new solutions to meet changing demands.

Each of the steps in the progressive maturity of integration should be undertaken in the context of specific projects with return on investment as the guiding principle.

6.2 Service Integration Maturity Model

As indicated in the previous section, integration maturity is a progressive realization in a service-oriented world, going from tight-integration and tight-coupling to loose-coupling and availability of Information and Business Capabilities as an IT Service. There are multiple dimensions along which service integration maturity can be measured or captured, and it also provides a roadmap for where future efforts should be focused on in order to fully realize the power and benefits of a SOA.

The Open Group has published a very useful draft of a SOA Maturity Model, reproduced here for reference in Figure 6.2 [OpenGroup, 06].



7. Build Composite Solutions
Combine information, capabilities and processes to form a cohesive business solution to address a problem domain

6. Orchestrate Processes
Combine capabilities (simple & composite) with the view to automate and/or manage business processes

5. Compose Capabilities
Combine multiple (often smaller) capabilities to create a composite business capability/function

4. Expose Capabilities
Expose business functions/capabilities within applications for access by consuming applications

3. Consume Information
Consume Information at the source for (read/write) access by applications to create Information as a Service (IAAS)

2. Provide Information
Provide Information at the source for (read-only) access by consuming applications to create Information as a Service (IAAS)

1. Integrate Information
Integrate applications by exchanging information between each other via defined interfaces using messaging

0. Exchange Files / Share Databases
Exchange data-files between applications or read/write from each other's databases

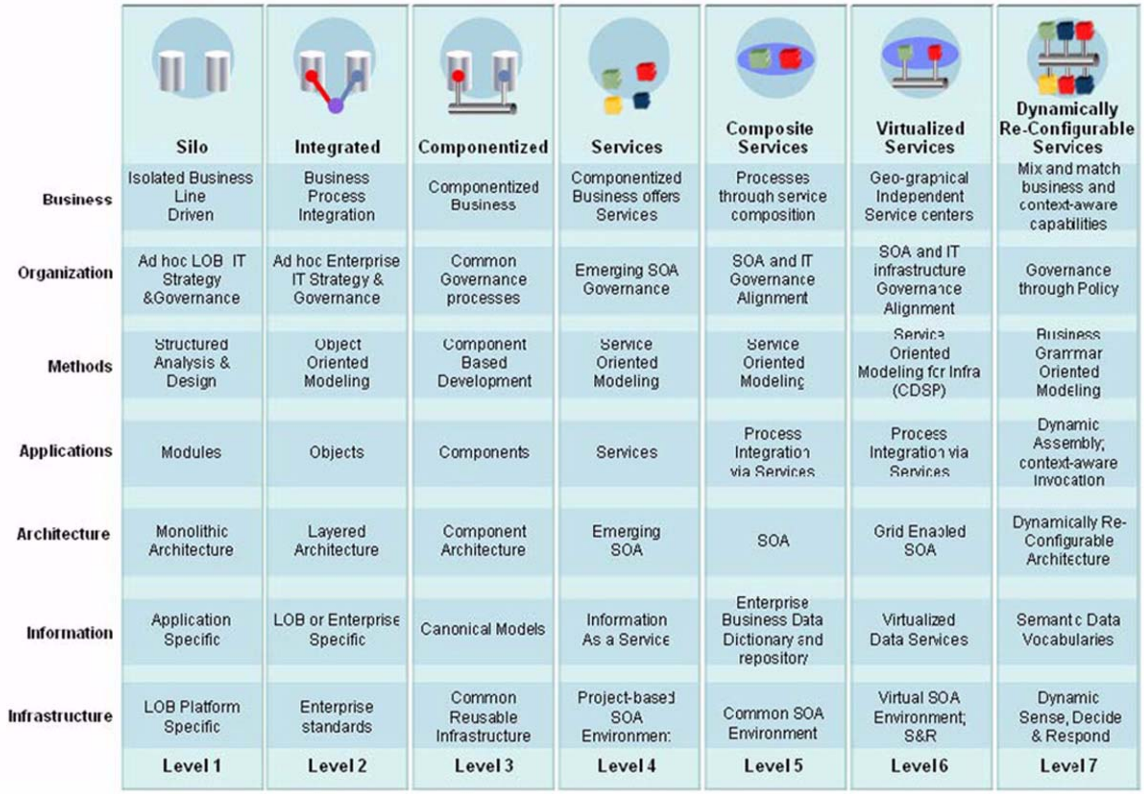
In order to enable these capabilities, services have to be exposed

Exposing services (capabilities & information) provide the foundation for a "self-service" pull-model

"Push model" of service-interfaces using SOI

Tight integration with no loose-coupling

Figure 6.1 Integration capabilities with SOA in education.



	Silo	Integrated	Componentized	Services	Composite Services	Virtualized Services	Dynamically Re-Configurable Services
Business	Isolated Business Line Driven	Business Process Integration	Componentized Business	Componentized Business offers Services	Processes through service composition	Geo-graphical Independent Service centers	Mix and match business and context-aware capabilities
Organization	Ad hoc LOB IT Strategy &Governance	Ad hoc Enterprise IT Strategy & Governance	Common Governance processes	Emerging SOA Governance	SOA and IT Governance Alignment	SOA and IT infrastructure Governance Alignment	Governance through Policy
Methods	Structured Analysis & Design	Object Oriented Modeling	Component Based Development	Service Oriented Modeling	Service Oriented Modeling	Service Oriented Modeling for Infra (CDSP)	Business Grammar Oriented Modeling
Applications	Modules	Objects	Components	Services	Process Integration via Services	Process Integration via Services	Dynamic Assembly; context-aware invocation
Architecture	Monolithic Architecture	Layered Architecture	Component Architecture	Emerging SOA	SOA	Grid Enabled SOA	Dynamically Re-Configurable Architecture
Information	Application Specific	LOB or Enterprise Specific	Canonical Models	Information As a Service	Enterprise Business Data Dictionary and repository	Virtualized Data Services	Semantic Data Vocabularies
Infrastructure	LOB Platform Specific	Enterprise standards	Common Reusable Infrastructure	Project-based SOA Environment	Common SOA Environment	Virtual SOA Environment; S&R	Dynamic Sense, Decide & Respond
	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7

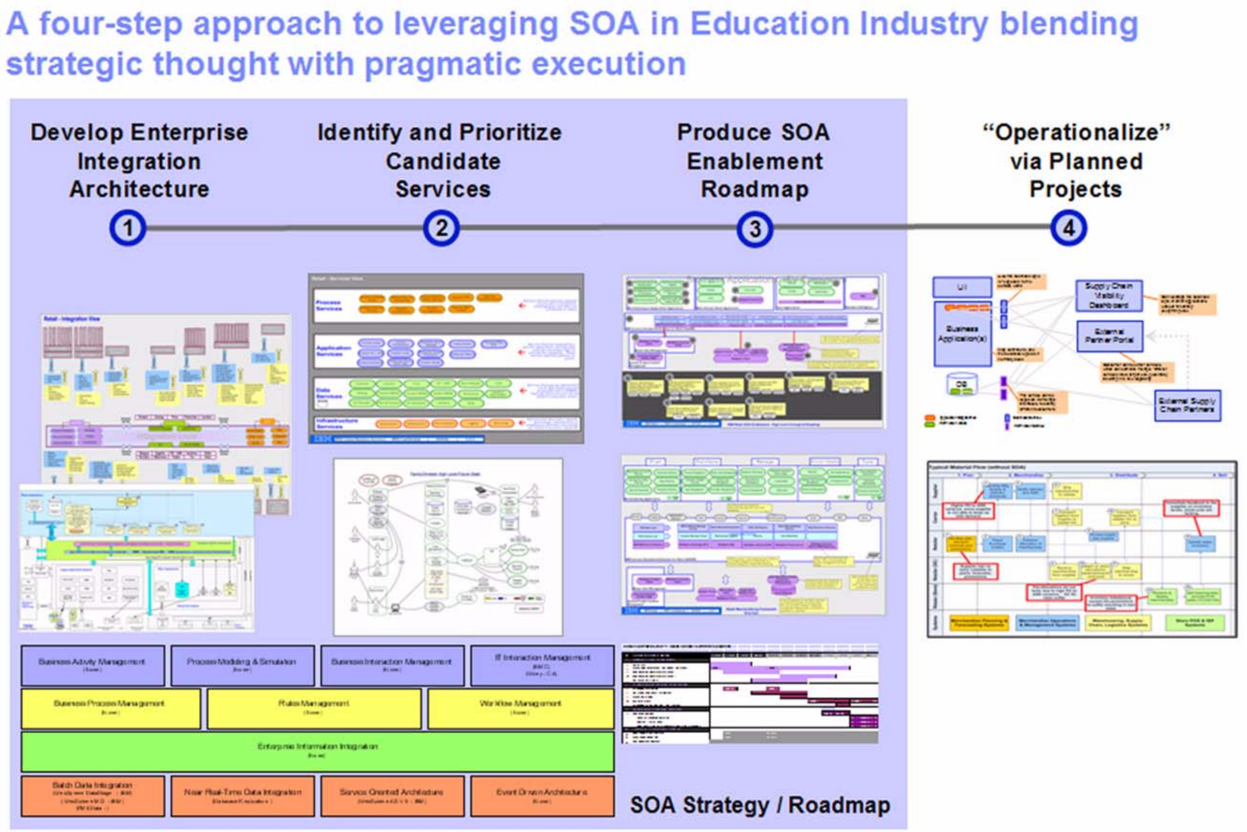
Figure 6.2 The Open Group's Service Integration Maturity Model (OSIMM).

The multiple dimensions across which maturity should be considered are also indicative of the various areas that service oriented architecture typically touches – and thus influences during an SOA project.

Typically, no given organization is at the same level of maturity in each of these dimensions. It is not atypical to find, say, an Application development maturity of Level 3, but a Business maturity of Level 1 at a given organization. The value of the table in Figure 6.1 is to not only map the current level of maturity for a given organization (or department within an organization) along various dimensions, but to also show how a given project roll-out will help in increasing the maturity of the department/organization along these various dimensions of maturity.

6.3 Operationalizing Service Oriented Integration through Planned Projects

When seeking to begin transformation of an education institution's IT systems to a more service oriented approach, it is important to take the long term, strategic perspective as well as the tactical pragmatic perspective. The strategic perspective lays the vision and roadmap, while the tactical approach ensures that your current projects deliver value while taking you towards the realization of the overall vision.



A four-step approach to leveraging SOA in Education Industry blending strategic thought with pragmatic execution

1 Develop Enterprise Integration Architecture

2 Identify and Prioritize Candidate Services

3 Produce SOA Enablement Roadmap

4 "Operationalize" via Planned Projects

SOA Strategy / Roadmap

Business Activity Management
Process Modeling & Simulation
Business Interaction Management
IT Interaction Management
Business Process Management
Rules Management
Workflow Management
Enterprise Information Integration
Batch Data Integration
Near Real-Time Data Integration
Service Oriented Architecture
Event Driven Architecture
UI
Business Application(s)
Supply Chain Visibility Dashboard
External Partner Portal
External Supply Chain Partners

Figure 6.3 Four Step approach to SOA.

Service oriented architecture always places an extraordinary level of importance on the enterprise domain and the need for alignment between the capabilities of technology and the services being delivered by the enterprise. Furthermore, given the high number of packaged applications in Education, it is imperative to take that first, holistic view of the organization and its IT architecture and lay down an *integration strategy* for the Enterprise. In other words, understanding the IT systems that need to integrate with each other coupled with an understanding of the data elements that flow between them greatly simplifies the creation of a flexible and agile service oriented architecture.

Enterprise Integration architecture will help highlight the many areas where exposing enterprise capabilities as an IT service would be beneficial and will feed into a candidate list of IT services. Furthermore, by analyzing specific pain points in the organization through business process analysis, several more candidate services would get added to the candidate list of services. Identifying and prioritizing these services would help create the SOA enablement roadmap for the organization itself.

The first step on that roadmap would be to pick individual projects with a high return on investment through which specific enterprise and IT services would be realized and the organization would move further along the road to service oriented maturity across multiple dimensions e.g. Information architecture, Application architecture, etc. Individual projects that provide definite value allows for a step-wise roll-out of SOA transformation of the overall enterprise and avoids the “big-bang” approach, and allows the organization to adjust and adapt to the changed paradigm that service oriented architecture requires.

Furthermore, these projects build on previously completed projects, leveraging the previously installed tools and middleware, consistently reducing cost of the newer projects, and moving the organization to a higher level of effectiveness and agility.

The following chapters detail out specific scenarios in Education and undertake a high-level SOA analysis of the same, identifying and prioritizing services and recommending select SOA solution patterns that are applicable.

7. Integrating Enterprise Applications in an SOA Context in Education – An Integration Scenario

7.1 Flexible integration of Student Information Systems (SIS) with Learning Management Systems (LMS)

In most education institutions, the process of populating courses within a learning management system is typically cumbersome, frequently requiring intervention on the part of system administrators to complete the process. This causes stress and delay for faculty and students who require instant access to their virtual course environment, resulting in poor user experience for all concerned. Furthermore, there is a measurable cost in time spent maintaining the process as well as responding to the errors introduced through this semi-automatic methodology. Redefining this process in a service-oriented manner will help the institution provide a more flexible and timely system integration that is less error prone and cost efficient, resulting in superior user experience.

7.1.1 Scenario Details

Our scenario for this category “Integrating Enterprise Applications in a Service Oriented World” states:

> *Learning institutions have traditionally structured their administrative and academic systems to mirror the hierarchy of the brick and mortar institution. Online learning environments require more flexibility in the way course rosters are populated because instructors tend to teach their students outside of the restrictions imposed by the institutional hierarchy. There should be a common methodology to populate course rosters from student system automatically into LMS. Specifically with:*

- *The flexibility to match multiple student system courses to one LMS course;*
- *The ability to add/drop students from online classes triggered from student system;*
- *The ability to create course sections automatically within the LMS when the section is created in the student system;*
- *The ability to transfer grades from the LMS into the SIS.*

7.2 SOA Analysis of the Scenario

The concept of integrating enterprise systems is certainly not new and most organizations have a method for accomplishing this integration. However, as stated above, the method used is frequently problematic: inflexible, costly, risky and often requiring manual intervention. A service-oriented approach is appropriate for this scenario because the principles of SOA provide a framework to overcome these challenges and move towards an integration solution that is reusable, flexible and standards based.

This scenario lends itself to an iterative and incremental approach to adopting a SOA, which is the recommended approach for an organization beginning to adopt a SOA. The following is a discussion of the iterations an organization can take towards incremental adoption of a SOA for this integration scenario.

7.2.1 First Iteration SOA for the SIS to LMS Integration: Adopt Open Enterprise Standards

At the most basic level of SOA adoption, we begin to use open standards as enterprise standards. For this scenario the relevant standard is provided by the LIS specification [LIS, 09] from IMS GLC. SIS and LMS vendors support and conform to the LIS standard to ease the integration burden on the education institutions. By utilizing the LIS standard an organization is able to facilitate data exchange between the SIS and LMS with a common language and common methodology. The benefit for the institution is that once they have learned the syntax to work with data in the LIS format they are able to work with that data in any system that supports LIS. If the organization decides to change from one LMS vendor to another, or to add an additional LMS system that supports the LIS standard, the work of exchanging data with the new system is minimal because there is no new proprietary data format to support. The use of open enterprise standards such as LIS helps to build a foundation for adopting a SOA and supports SOA principles such as reusability and potential for increased vendor diversification options.

7.2.2 Second Iteration SOA for the SIS to LMS Integration: Expose Information as a Service

If the goal of integration is limited to connecting two systems such as the LMS and SIS, then a point-to-point integration as indicated in the first iteration is sufficient to solve the given problem. However, if there are additional systems that need access to the information provided by the LMS, there is value in considering a move towards a more services based approach based upon a “pull” model of information sharing. The information available from the SIS is presented as an abstracted service that is more flexible, sustainable and available to be used by multiple systems. These characteristics of a service allow an organization to integrate more systems than a single SIS and a single LMS. With the LIS enterprise standard as the foundation for our SOA there are defined ways to expose information as a service from the SIS. One way the LIS specification is implemented is through the use of Web Services.

As a particular instance example, consider a common data exchange between SIS and LMS such as the addition of a new student to the LMS who has registered in the SIS. This transaction can be implemented with Web Services based on the LIS Person Model. The LIS Person Model describes an XML format for the data transfer from the SIS to the LMS. One can utilize this XML model to format and standardize the data to expose it as a web service that can be accessed by multiple systems.

7.2.3 Third Iteration SOA for the SIS to LMS Integration: Expand Use of Exposed Service

Now that we have exposed the necessary integration information in a standards based way and begun to utilize SOA principles for exposing that data we can begin to use the data in more flexible ways. For instance: in an institution that supports more than one LMS system that requires person data from the SIS we can now facilitate that data transfer without building point-to-point integrations for each LMS. We expose the person data one time and simply map it and mediate it for each LMS as needed. This is a standard SOA based model that provides numerous benefits:

- Flexible integration – we can drop in a different LMS to get the person data it needs;
- Routing – the information can be routed from the SIS to LMS or numerous LMS systems;
- Sustainable integration – when any of our applications change in our integration picture, as long as they continue to support the standard we have adopted (LIS in this case) our integration(s) will continue to function;
- Reusable – the work invested in building the integration can be repurposed and reused as many times as necessary to provide the same information to other systems whether they are other LMS systems or simply other institution business and academic systems.

7.2.4 Fourth Iteration SOA for the SIS to LMS Integration: Choreograph and Transform to Expand Service Usage

Additional capabilities can be realized from the above architecture such as:

- Choreographing LIS Create/Read/Update/Delete (CRUD) operations – each atomic update that is provided by the LIS specification from our SIS can be transmitted as a standalone event. However, by aggregating multiple changes, one can be considerably more economical and potentially more flexible with the solution;
- Mapping LIS CRUD operations – another example is the case where we have a person associated with a specific course section within the SIS but we want to map that person to a different course section within the LMS. This is a common occurrence because many times a professor will teach multiple sections of a course from one virtual course section to facilitate cross section communication, collaboration or simply to facilitate easier course administration for the professor. The mapping of sections from SIS to LIS can be done using appropriate mediation/mapping code;
- Routing of updates – users can be mapped from the SIS to multiple LMS systems as part of the routing/mediation mechanism put in place as part of the integration. There could be, for instance, the case where one department is utilizing one LMS and another department is utilizing another LMS. In that case we can create the mappings at the abstracted mediation layer can be created.

The above iterative approach incrementally adds capabilities to the LMS/SIS type enterprise integration using a services approach. Other capabilities, out of scope for the scenario highlighted here, include but are not limited to:

- An exploration of governance around this scenario to include securing the Web Service(s), maintaining a repository of information and code about this Web Service and publishing this Web Service in a way that it is available to a larger community of users. For example, the service that provides the information that a course has new enrolments could be utilized by the campus bookstore to re-order textbooks as a class enrolment grows;

- A further exploration of utilizing the information delivered through the work in our integration to systems that do not today support LIS standards. For example, the SOA tools at the mediation layer can be utilized to map attributes that are presented from the SIS to attributes recognized by a system that does not understand the LIS standard;
- A more extensive service orchestration scenario where we enrich information provided by the SIS with information provided by another system. For example, in the United States when Family Educational Rights and Privacy Act (FERPA) protected data is transmitted from the SIS out to the larger user community the privacy of students who have elected to close their record of FERPA protected data must be confirmed. In our service enabled scenario we are able to build out a more extensive business process that includes a check for FERPA data protection.

7.3 How the solution helps increase SOA Maturity & Organization Effectiveness

Through incremental adoption of SOA via this integration scenario, an organization can increase its overall IT maturity and effectiveness. Using the SOA Maturity Model (introduced in “Service Integration Maturity Model” , Section 6.2) we can highlight the specific dimensions along which the organization has become more effective.

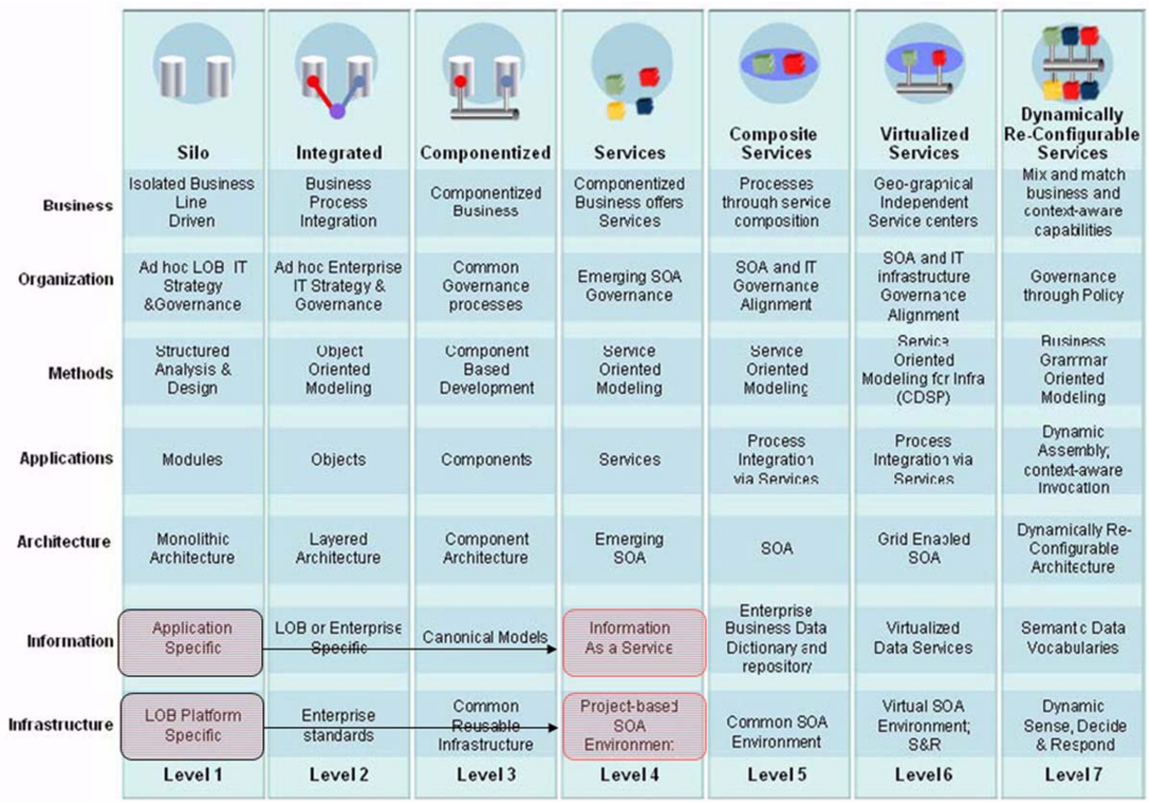
As shown in Figure 7.1, the fundamental areas along which the solution increases the maturity and effectiveness of the IT environment are:

- Information – prior to applying the SOA approach, information sharing was restricted to the set of applications that made a point-to-point connection with each other. By applying the selected solution to the scenario, information is now available as a service. New consumers of the service simply need to follow a standardized way of accessing the service, doing away with point transformations of data and protocol;
- Infrastructure – by introducing elements of SOA, enterprise standards such as LIS, and SOA tools, the IT environment has begun supporting project-based SOA infrastructure. These investments can be fully leveraged for future projects, reducing the associated cost of the solution.

7.4 Applying SOA Patterns to Solution Realization

There are several established SOA design patterns to reference that help simplify and expedite the design of service oriented integration. Based on the patterns established by the SOA community and presented at http://www.soapatterns.org some example patterns include:

- Enterprise Service Bus/Internal Connectivity pattern;
- Service Broker;
- Canonical Schema Bus.



	Silo	Integrated	Componentized	Services	Composite Services	Virtualized Services	Dynamically Re-Configurable Services
Business	Isolated Business Line Driven	Business Process Integration	Componentized Business	Componentized Business offers Services	Processes through service composition	Geo-graphical Independent Service centers	Mix and match business and context-aware capabilities
Organization	Ad hoc LOB IT Strategy &Governance	Ad hoc Enterprise IT Strategy & Governance	Common Governance processes	Emerging SOA Governance	SOA and IT Governance Alignment	SOA and IT infrastructure Governance Alignment	Governance through Policy
Methods	Structured Analysis & Design	Object Oriented Modeling	Component Based Development	Service Oriented Modeling	Service Oriented Modeling	Service Oriented Modeling for Infra (CDSP)	Business Grammar Oriented Modeling
Applications	Modules	Objects	Components	Services	Process Integration via Services	Process Integration via Services	Dynamic Assembly; context-aware Invocation
Architecture	Monolithic Architecture	Layered Architecture	Component Architecture	Emerging SOA	SOA	Grid Enabled SOA	Dynamically Re-Configurable Architecture
Information	Application Specific	LOB or Enterprise Specific	Canonical Models	Information As a Service	Enterprise Business Data Dictionary and repository	Virtualized Data Services	Semantic Data Vocabularies
Infrastructure	LOB Platform Specific	Enterprise standards	Common Reusable Infrastructure	Project-based SOA Environment	Common SOA Environment	Virtual SOA Environment; S&R	Dynamic Sense, Decide & Respond
	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7

Figure 7.1 A mapping of the scenario to the SoA maturity model.

8. Enabling Shared Services & Cloud with Service Oriented Architecture – A Virtual Desktop Shared Service Scenario

Shared Services is a collaborative model of operation where multiple organizations or multiple departments within an organization come together to fund, maintain and operate a common service that is accessible to all. Shared services, as detailed here, may be provisioned within the organization or could be provisioned by a third-party – with the net benefit that no one user has to bear the entire cost of the service.

The business models of Software as a Service or Infrastructure as a Service begin adding virtualization, hardware efficiency and a pay-as-you-go culture to the mix, which has collectively matured into what is referred to as Cloud computing – and Cloud is now quickly becoming the de-facto model for shared services.

Service oriented architecture relies on the foundation of providing well-characterized, platform-independent, standards based and easily accessible business services – all of which support the creation and management of shared services. Having a well-defined SOA in an organization directly supports and promotes the establishment of shared services, provisioned locally or at the Cloud.

8.1 The Imperative for Shared Services

Shared services have been attractive to Education institutions for at least three key reasons:

a) Cost Reduction – shared services can lower Total Cost of Ownership even as provisioning, maintenance and upgrade costs are now either shared or handed off to a third-party provider;

b) Quality Improvement – services provided by a specialist in the field, such as Tax services from specialized providers (either internal or external to the organization) can be considerably superior in quality to home grown solutions or even commercial off the shelf products;

c) Shorter Time-to-market – upfront capital layout is reduced considerably, and if the service already exists, it can be incorporated into the organization’s business processes very rapidly, and thus makes this option very attractive for offering new features and functions, especially when the value of the offering has not been vetted.

8.2 Challenges of Shared Services and Third Party Provisioning

Shared services and third-party provisioning are also associated with significant concerns:

a) Security – potential for losing data, or having an unwanted party get access to sensitive data are ongoing concerns around shared services and third-party provisioning;

b) Danger of Losing Control – if certain key functions are not available at a crucial time, the entire organization may be held hostage. This type of single point of failure with widespread consequences has to be appropriately managed;

c) Danger of Losing Critical Know-How – over time, especially with third-party provisioning of services, loss of critical knowledge in specialized areas of the business is possible with this approach;

d) Cost of transitioning existing functionality – getting existing systems to switch their integration to a new system or service provider is always expensive. Furthermore, existing business processes may be affected as well and will need to be upgraded along with retraining of personnel to support the new processes;

e) Hidden, ongoing costs of managing third-party relationships (internal or external), responsiveness to change, overall support costs and so on can quickly add up.

8.3 How does an SOA enable Shared Services?

Service oriented architecture seeks to deploy the technology of an organization as a series of re-usable, standardized business services that can be choreographed into business processes and higher level composite services. The net result is a highly reactive, agile and flexible architecture with a high degree of reuse, resulting in lowered maintenance and new development costs.

Services, which are the key building blocks of a service oriented architecture, can be deployed anywhere and still be accessible through architectural patterns such as the ESB. Moreover, using established communication standards such as Web Services, these services are platform independent and accessible over the Internet. Furthermore, good SOA governance ensures that service lifecycle management, service availability and service level agreement are all in place to ensure that the service delivers on its promised capabilities.

In brief, SOA makes it significantly easier to share common services and work with third-party providers of services, as many of the issues and concerns around provisioning – security, service level agreement, integrating third-party service with rest of Enterprise IT – are issues that SOA handles directly.

8.4 The Promise of Cloud Computing for Shared Services

Cloud computing is an emerging computing model by which users can gain access to their applications from anywhere, through any connected device, making the infrastructure transparent to the user. The services themselves reside in data centers where computational resources can be dynamically provisioned and shared to achieve significant economies of scale. Furthermore, due to efficient service management and automated provisioning, the costs of adding IT resources to the cloud is significantly lower than doing so through traditional mechanisms. In other words, Cloud Computing focuses on:

a) Virtualization of Infrastructure and Services;

b) Automated Provisioning of Services;

c) Increased Availability and Connectivity with End Users.

Figure 8.1 highlights the various roles around cloud computing including, creator of services who creates and publishes services to the cloud, service administrator/manager at the cloud who manages the services at the cloud and finally the service consumer, who consumes the services provided.

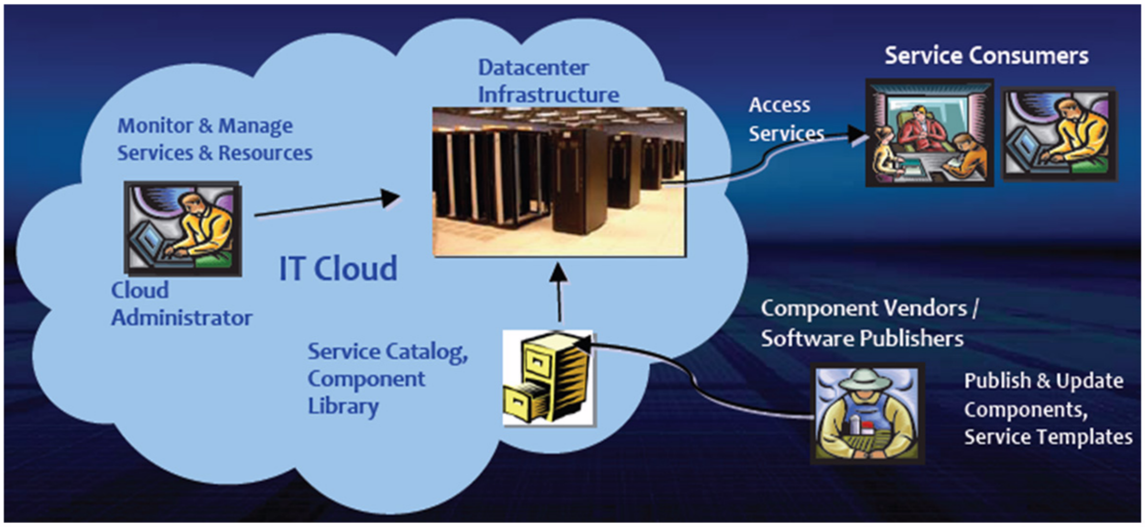


Figure 8.1 Cloud computing.

8.4.1 SAAS, PAAS, IAAS and the Cloud

Services from the Cloud frequently take the form of:

a) Software as a Service – business applications such as Accounts Payable, Tax Calculation as well as development and test tools are made available in a virtualized fashion to the end user. Typically, the notion of Shared Services as described here falls in this category;

b) Platform as a Service – Web Services, Web 2.0 applications, middleware can all be offered as a platform on top of which applications can be built, assembled and run;

c) Infrastructure as a Service – servers, storage, networking and other infrastructure can be offered as a service through a virtualized interface.

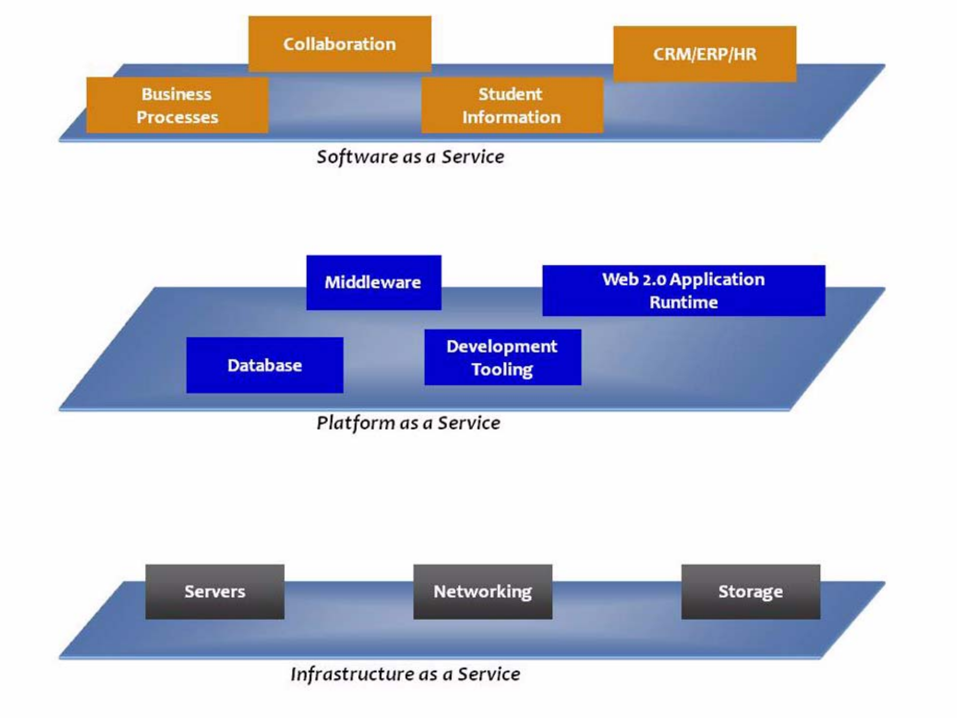


Figure 8.2 Forms of service from a cloud.

8.4.2 Public versus Private Clouds

In the general industry, two forms of Cloud Computing models are beginning to develop – Private Clouds and Public Clouds. Typically private clouds are enhancements of traditional enterprise data centres that have always been a part of an organization’s architecture and provisioning, while public clouds are the same data centres, but hosted and managed by a third-party, offering services frequently on a subscription basis. In other words, Public Clouds are:

- Service provider owned and managed;
- Accessed by subscription;
- Deliver select set of standardized business process, application and/or infrastructure services on a flexible pay per use basis.

While Private Clouds are:

- Owned and managed by the Education institution itself;
- Limit access to student, staff and faculty of the organization and other partners in its network;
- Drive efficiency, standardization and best practices in the services provided while *retaining greater customization and control* than public clouds would permit.

8.4.3 SOA and the Cloud

As highlighted in the previous sub-sections, the focus of Cloud is in the provisioning and delivery of shared services by virtualizing the applications, platform and infrastructure such that the user of the service gets a seamless experience without being aware of the underlying provisioning methodology.

If we take a look at the overall set of elements that go on to realize SOA, as shown in Figure 8.3, the key focus of the Cloud is in the Infrastructure Management and Services Management space to provide the virtualization and automated provisioning capabilities.

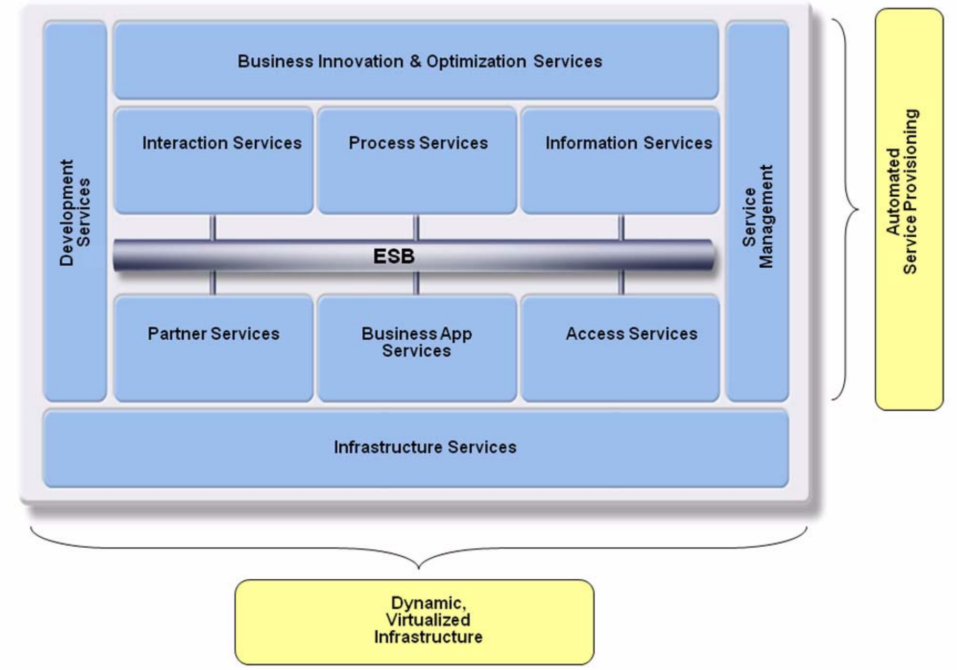


Figure 8.3 Cloud and SOA.

8.5 Virtual Computer Labs – Moving the Lab to the Cloud as a Shared Service

Cloud computing and Shared Services are helping education and Education Institutions move towards an open and collaborative environment while reducing the costs of delivering and managing the needed services. *Virtual Computer Laboratories* is one such instance where cloud computing and shared, pooled services are making a significant difference in the cost structure, service delivery and user ecosystem satisfaction for Education organizations.

8.5.1 Computer Labs – Big Budget Expense

Computer laboratories have been a fixture at most major Higher Education Institutions across the world, providing easy access to students, faculty and staff to the latest computing facilities, complete with productive applications such as word-processors, spreadsheets, Internet access, Web 2.0 applications, and peripherals such as web-camera, scanners, printers and so forth. In addition to the initial capital layout for such facilities, there is an ongoing cost of maintaining the hardware, upgrading and installing new software and ensuring security and authorization access to the facilities.

Furthermore, the arrival of mobile technologies including laptops, cell-phones, PDAs has meant that the user ecosystem is now demanding access to an institution’s computing facilities from new, remote access points while expecting it to be delivered with the same security and ease of use that they expect when inside a laboratory.

Therefore, the challenge in front of Education Institutions in this instance is two fold:

- Enhanced User Experience – remote and mobile access, access to latest applications, secure;
- Easier Maintenance and Lower Cost – efficient utilization of hardware and software resources, easier management and provisioning of services, secure and authorized access.

8.5.2 Enhancing the User Experience – the Virtualized Desktop

From a user ecosystem experience point of view, the need is for the user to be able to log onto a system, requiring only their browser and Internet connectivity, and be given access to a desktop that they can configure based on their current needs. For example, if a user requires a Circuit Simulator application and a spreadsheet, that request can be made after logging in and getting authorized and the system would automatically provision the desktop with these selected applications (if available at that time).

For the user, the virtual desktop appears as any desktop that they would have used if they were logging onto a physical machine at the institution’s laboratories directly. Furthermore, given the ease of access, it greatly enhances productivity because the user is not physically required to be at a certain place to get their work completed.

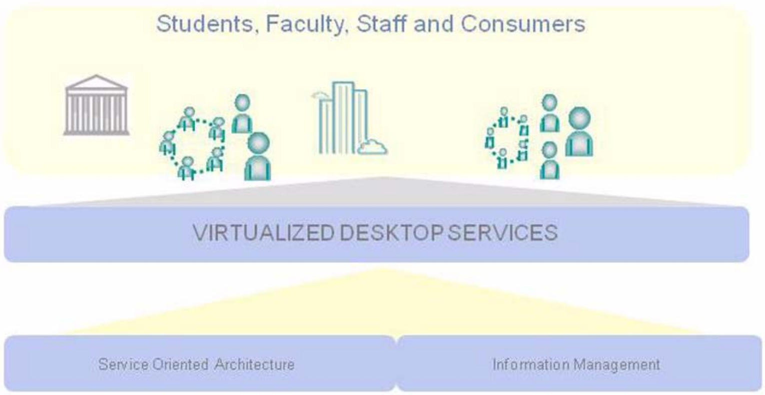

Students, Faculty, Staff and Consumers
VIRTUALIZED DESKTOP SERVICES
Service Oriented Architecture
Information Management

Figure 8.4 Desktop – anytime, anyplace.

8.5.3 Provisioning Virtual Desktops at the Cloud

The best way to manage the desktop services is by having an underlying SOA that makes the applications available as a service, ready for use based on user request.

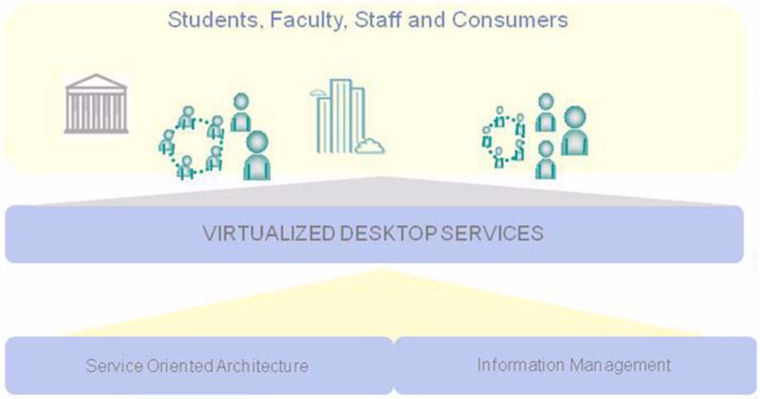

Students, Faculty, Staff and Consumers
VIRTUALIZED DESKTOP SERVICES
Service Oriented Architecture
Information Management

Figure 8.5 Service at the cloud is also delivered on SOA principles.

The SOA supporting the virtualized desktop services is created, run and managed on a virtualized infrastructure, platform and software services. Cloud computing brings in provisioning software and hardware applications that manage the availability of services on a per-request basis, as well as hide the hardware resources being used – thus providing the efficiency in hardware resource consumption.

Furthermore, since images are efficiently provisioned on servers that have available resource, and powered on only when needed, there is significant energy cost savings for power and cooling. Also, since the virtual desktop is running on a centralized server, usually in a datacenter with other connected applications, bandwidth is usually saved and applications perform much better and respond more quickly.

Security and authorization are centrally managed to all services on the cloud, thus providing a common point of control for those aspects of the user and management experience.

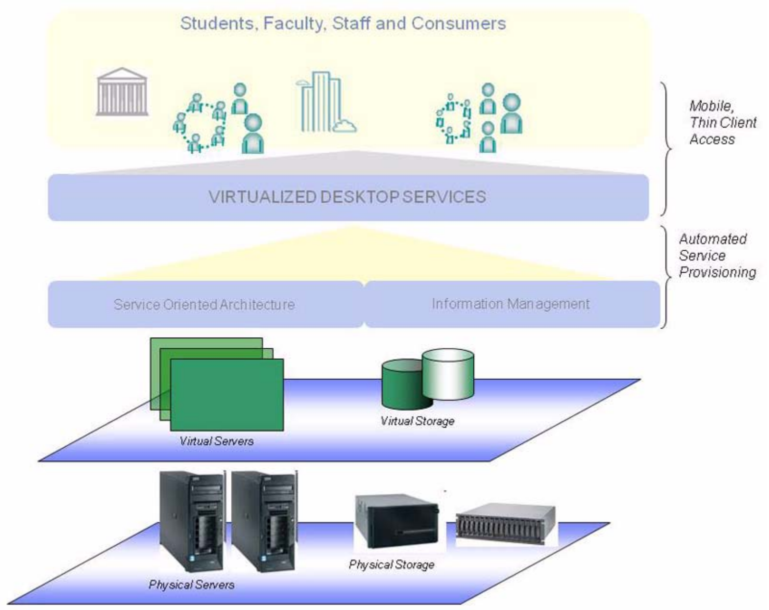

Students, Faculty, Staff and Consumers
Mobile, Thin Client Access
VIRTUALIZED DESKTOP SERVICES
Automated Service Provisioning
Service Oriented Architecture
Information Management
Virtual Servers
Virtual Storage
Physical Servers
Physical Storage

Figure 8.6 Cloud Computing and Virtualized Desktop Services

Generalizing the requirements for a Virtual Desktop:

- Reduce hardware footprint;
- Constantly maintain and upgrade multiple application software;
- Deliver software as a service when and where needed;
- Automatically provision services based on user request.

It is easy to turn to cloud computing as a possible solution given the value it delivers by:

- Virtualizing hardware and thus making more efficient use of hardware resources;
- Automating provisioning of software services on a per request basis;
- Simplifying and centralizing management of security and authorization;
- Potentially reducing licensing cost of software by more efficient utilization of available applications.

8.5.4 Virtualized Desktops - Private or Public Cloud?

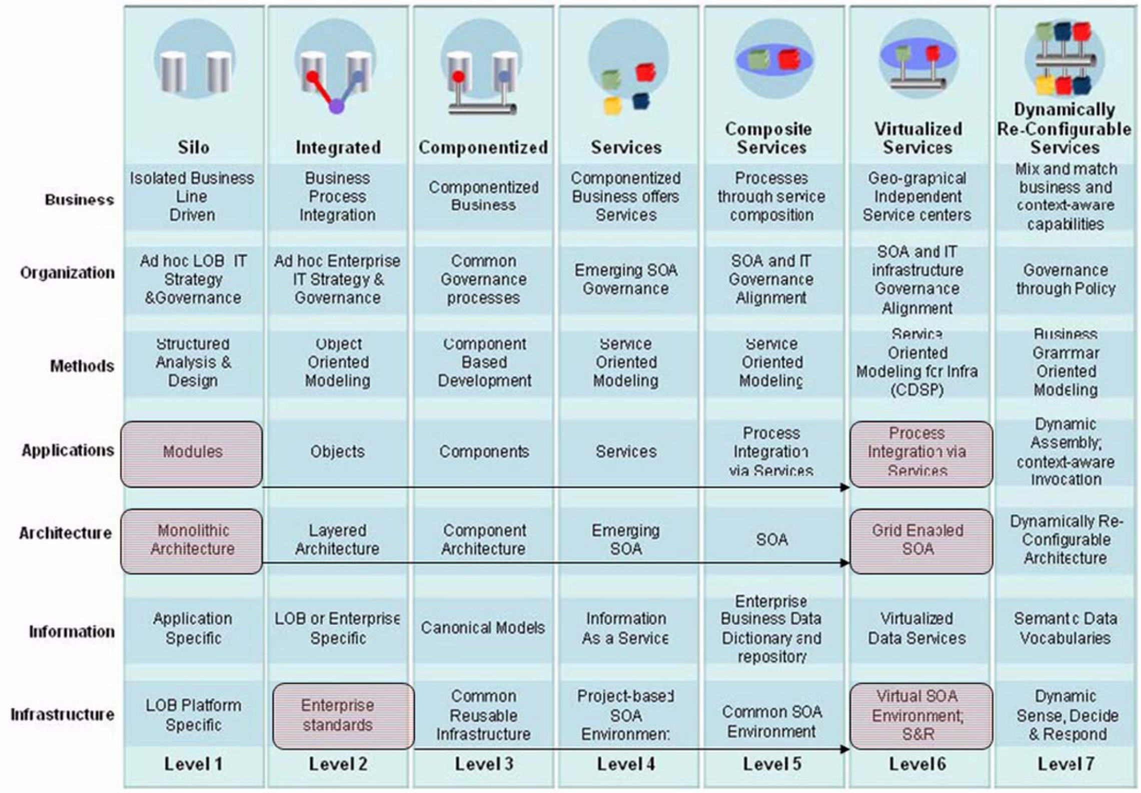
Virtualized desktops hosted in a cloud environment end up saving Education Institutions significant money in terms of capital outlay for hardware and software purchases and maintenance. Institutions however do have another choice to make – do they host this cloud themselves, in the form of a “Campus Cloud”, or do they completely rely on third-party provisioning of a cloud, and make payments on a per use/per user basis? While every organization has to make this decision independently, there are cost savings and benefits in either form as indicated in "Public versus Private Clouds" , Section 8.4.2.

Private clouds offer greater control and customization ability while still becoming more efficient in hardware and software resource use, and reducing the cost of management of the solution considerably. Public clouds for virtualized desktops will tend to be generic in nature and might be useful for gaining access to certain commodity services – such as word processing, spreadsheets, calculators and so forth. It is also possible that over time, public cloud services catering to niche requirements (such as high-end bioinformatics services, high speed circuit simulators and so forth) may emerge and begin to weigh the decision in favor of public clouds. At the same time, the control and customization offered by private clouds may be something that individual institutions may continue to value very highly as a differentiator in the services it provides to its user ecosystem.

Adoption of Shared Services and Cloud Computing is going to continue to gain momentum in the world of Education even as the available capital outlay for providing services through technology keeps declining and the demand on the systems keeps rising. The ability to share the cost of services, make more efficient use of hardware and software resources, and simplified, centralized management makes shared services and cloud computing a welcome innovation for organizations and institutions serving Education.

8.6 How the Solution Helps Increase SOA Maturity

The solution improves SOA maturity significantly in areas of Application, Architecture and Infrastructure, even as shared services and a virtual infrastructure significantly improve the accessibility and availability of services.



	Silo	Integrated	Componentized	Services	Composite Services	Virtualized Services	Dynamically Re-Configurable Services
Business	Isolated Business Line Driven	Business Process Integration	Componentized Business	Componentized Business offers Services	Processes through service composition	Geo-graphical Independent Service centers	Mix and match business and context-aware capabilities
Organization	Ad hoc LOB IT Strategy &Governance	Ad hoc Enterprise IT Strategy & Governance	Common Governance processes	Emerging SOA Governance	SOA and IT Governance Alignment	SOA and IT infrastructure Governance Alignment	Governance through Policy
Methods	Structured Analysis & Design	Object Oriented Modeling	Component Based Development	Service Oriented Modeling	Service Oriented Modeling	Service Oriented Modeling for Infra (CDSP)	Business Grammar Oriented Modeling
Applications	Modules	Objects	Components	Services	Process Integration via Services	Process Integration via Services	Dynamic Assembly; context-aware invocation
Architecture	Monolithic Architecture	Layered Architecture	Component Architecture	Emerging SOA	SOA	Grid Enabled SOA	Dynamically Re-Configurable Architecture
Information	Application Specific	LOB or Enterprise Specific	Canonical Models	Information As a Service	Enterprise Business Data Dictionary and repository	Virtualized Data Services	Semantic Data Vocabularies
Infrastructure	LOB Platform Specific	Enterprise standards	Common Reusable Infrastructure	Project-based SOA Environment	Common SOA Environment	Virtual SOA Environment; S&R	Dynamic Sense, Decide & Respond
	Level 1	Level 2	Level 3	Level 4	Level 5	Level6	Level 7

Figure 8.7 Increased maturity.

As shown in Figure 8.7 the improvement in maturity is at the:

a) Application Architecture – the new application architecture is a choreography of an inventory of services in support of business processes. Provisioning on the cloud does not fundamentally change the benefits introduced by this approach which are flexibility and ability to use of best of breed services;

b) IT Architecture – the IT architecture at the Enterprise is now supporting a service oriented approach applying select SOA patterns to achieve its goals. Furthermore, the IT provisioning is now at the cloud, which supports a virtualized approach and takes care of service availability, bandwidth and storage concerns and other unpredictable demands on the system;

c) Infrastructure – the cloud environment makes a significant impact to the infrastructure of the environment in which the virtual desktop solution has been deployed. By offering virtualized servers, storage and network, it takes the overall environment at the site to that of a virtual SOA environment where capacity can be increased on demand and the infrastructure resources are utilized in a highly efficient manner.

9. Bringing on New Systems with a Services Approach – A Financial Aid Scenario

9.1 Complexity in Financial Loan Processing

The particular scenario under consideration in this chapter deals with the delivery of financial aid to student applicants. There are multiple business processes and systems typically involved in the delivery of financial aid, and the use of a service-oriented approach can help improve the flexibility of the processes and make them responsive to change.

9.1.1 Scenario Details

A brief description of the scenario is:

> *Every year, hundreds of thousands of students and families, in the USA, apply for and receive in excess of $80 billion dollars of government and private student loans for higher education. In a complex and heavily regulated process, loan applications, credit reports, tax forms, student aid applications, loan guarantee services, all must be processed in a timely, accurate, and secure system. With emerging standards, common applications, and systems moving to a SOA approach, new services offerings will become available to schools, families, lenders, guarantors and third party agencies.*

In the specific situation under consideration, an educational institution wishes to replace its existing SIS because it has become outdated (over 20 years in operation) and inflexible, having been built on an older architectural style and unsuited to the newer demands on the system. The organization now wishes to install a new system and is considering a Services-based approach because of the built in flexibility and readiness for change with this approach. Specifically, the organization wishes to revamp its financial aid processing processes using a service-oriented approach.

9.2 SOA Analysis of the Scenario

9.2.1 Entry Point

The organization is looking to upgrade its financial aid processes and is looking to replace an aging legacy application with a services based approach. The first step in an SOA analysis always starts with business requirements, and in this case, it is imperative to fully understand and model the business processes associated with financial aid processing. Therefore, this is the "Business Process Entry Point" for further SOA analysis for this organization.

9.2.2 Business Process

Business Process analysis starts with laying out the business process under consideration using a standard notation. Business Process Modelling Notation (BPMN) specification is managed by the Object Management Group (OMG) and tutorials can be found at http://www.bpmn.org/) has emerged as an industry standard for depicting business processes.

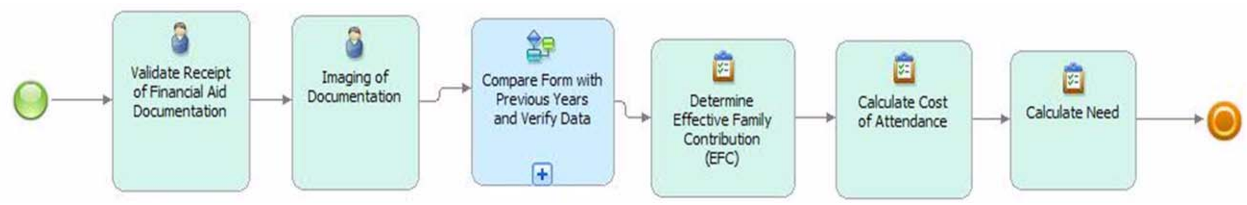

Validate Receipt of Financial Aid Documentation
Imaging of Documentation
Compare Form with Previous Years and Verify Data
Determine Effective Family Contribution (EFC)
Calculate Cost of Attendance
Calculate Need

Figure 9.1 Needs analysis for the Financial Aid scenario.

Figure 9.1 depicts the business process for the Needs Analysis portion of financial aid processing. This shows the activities and tasks associated in making the need determination for the particular applicant. Some activities are manual in nature while others can be performed by a system. One sub-process has been identified “Compare Form with Previous Years’ and Verify Data”.

9.2.3 Identifying Candidate Services

Service is the most fundamental unit of a service-oriented solution that is packaged as a reusable component for use in a business process. The fundamental benefits of well-defined services are to facilitate interoperability of services and to enable flexibility and reuse of business functionality within an enterprise and with partners. This establishes an environment wherein Services produced by different projects at different times can be repeatedly assembled together to help automate a range of business tasks for addressing new business opportunities or changing business priorities.

Looking at the Activities and Tasks in the Business Process diagram (Figure 9.1), some of these tasks and activities must be performed by a human actor while others could be performed by a computer system. In this process, the following activities could thus be modelled as a Service, frequently referred to as Candidate Services at this stage.

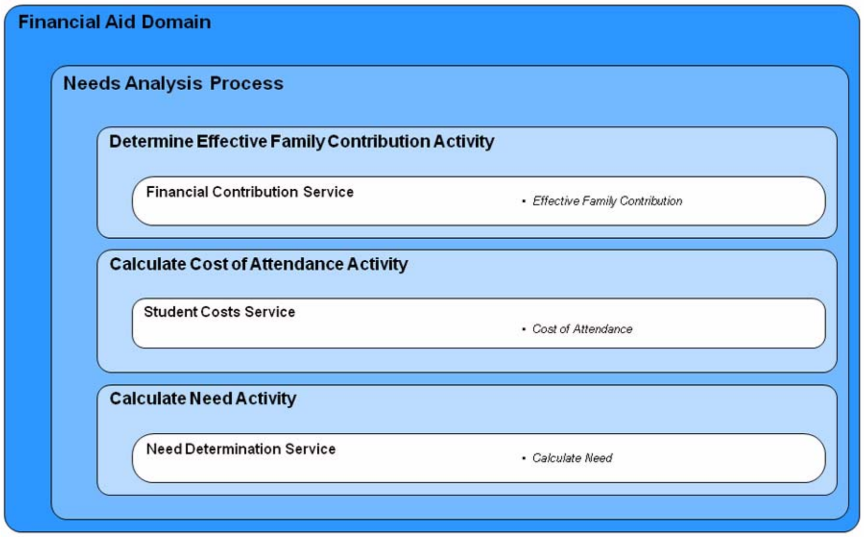

Financial Aid Domain
Needs Analysis Process
Determine Effective Family Contribution Activity
Financial Contribution Service
• Effective Family Contribution
Calculate Cost of Attendance Activity
Student Costs Service
• Cost of Attendance
Calculate Need Activity
Need Determination Service
• Calculate Need

Figure 9.2 Select candidate services.

Figure 9.2 is not a full list of Candidate Services for the Needs Analysis process, but it captures some of the interesting possibilities. In this particular scenario, each activity maps to a single underlying business service. This is not always the case, especially, when dealing with pre-existing systems. In this scenario, the organization wishes to replace an aging system with a services approach; it is then possible to create new business services that map directly to a business activity.

9.2.4 Prioritizing Services

It is not necessary to realize every candidate service identified in the previous step at the same time. Instead, it is better to look at some of the characteristics of these services to help prioritize where the focus should be first. As an example, looking at the Student Costs Service (shown in Figure 9.1), one can test it across some select criteria:

Table 9.1 Prioritizing criteria for services.

Multiple Consumers	**High**	Multiple systems such as Financial Aid, Loan Offering and Cost Tracking are all potential consumers of this service.
Spanning Domains	**High**	The service is useful for systems not only in financial aid, but also general cost tracking systems that belong to a different domain.
Business Timely Information	**Low**	The demand on this service is not real-time or time-sensitive performance.
Business Alignment / Cost-effective	**Medium**	The service is highly business aligned as Student Cost is a very important component of multiple business processes and service offering.

Additional criteria that could help prioritize services include:

- Leveraging Existing Capability – if this service or service capability is already available through an existing system, it may help prioritize the usage of the service in support of the business process(es) under consideration;
- External consumers – if a service is meant for consumption external to an organization, special security and data considerations need to be taken into account. This can affect whether a service should be created or not.

Performing this analysis across all the Candidate Services will help zero-in on the right set of Services to help realize first. Additional dimensions for prioritizing services can be added to the above list to help pick the right set for realization, including criteria such as:

- Is the Service replacing existing application function or introducing brand new capabilities?
- What is the overall impact on Return on Investment (value versus cost)?

9.2.5 Service Specification and Realization

After identifying services to be developed, the next step is to start specifying those services. The first step, like in any good software engineering methodology, is to analyze the service that needs to be developed. This analysis primarily consists of defining various types of requirements:

- Functional Requirements – as already defined through Business & Use Case descriptions;
- Non Functional Requirements – such as logging, security, availability and composability.

In addition to the traditional requirements, it is important to think about the consumers for these services and the type of demands they will place on the service on an ongoing basis. This type of analysis helps significantly when designing, realizing and ultimately deploying these services. Service specification typically involves identification of:

a) Service Interface – preferably based on an open standard;

b) Service Context – understanding all the consumers of the service;

c) Service Dependencies – other services that this service depends upon;

d) Service Composability – can this service be composed as part of another service?

e) Service Message and Data Mapping.

Furthermore, realization of a Service involves activities including:

a) Mapping Services to existing systems/sub-systems;

b) Laying out SLA for the Service;

c) Identifying Non-Functional Requirements including availability and security.

Services are typically realized from existing systems or created from the ground-up afresh, and in either situation, they are slowly rolled out to an increasing set of consumers until it can meet the service-level-agreement for all consumers of the service.

9.3 Applying SOA Patterns to Solution Realization

Realizing the said business process is simplified by applying specific SOA patterns that yield significant business value. Some of the key patterns applicable to this process are mentioned below.

9.3.1 Application of Consolidation Pattern

Looking at the data and information side of the Needs Analysis business process, it is important to realize that data such as Financial Aid Application from Previous Years and Student Record must be available in a consolidated fashion from a single source. This does not always happen because data is often spread across multiple databases and hidden behind specific applications.

An applicable SOA pattern that makes such disparately housed data available as a service is the ‘Consolidation’ pattern, which introduces *Information as a Service* (see Figure 9.3). In this particular scenario, the institution is replacing its existing student information system with a more service oriented approach and so it is appropriate to apply the above pattern for consolidating student information and presenting it as a service. One could seek to realize the Needs Analysis business process without necessarily consolidating student data as a service, though the long-term value of the solution is considerably enhanced if this data is consolidated and presented as a service.

Without consistent and complete data, a service-oriented approach simply makes bad or incomplete information more widely available and easily accessible. Therefore data consolidation and availability of information as a service is a very important consideration when considering a service-oriented approach, as is the case with this organization.

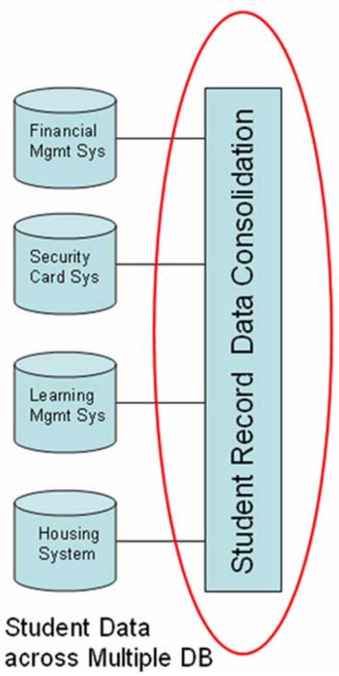


Figure 9.3 Consolidation pattern.

9.3.2 Application of Internal Connectivity Pattern (Enterprise Service Bus)

Internal connectivity describes how multiple internal clients can access services within the organization. For example, this SOA atomic pattern can be used to describe how remote offices of an organization access headquarters systems using, for example, Web Services standards. In this example, shown in Figure 9.4, it is important for the business capabilities and data of each of the services to be available to participate in the Needs Analysis business process, which is realized by the application of this pattern.

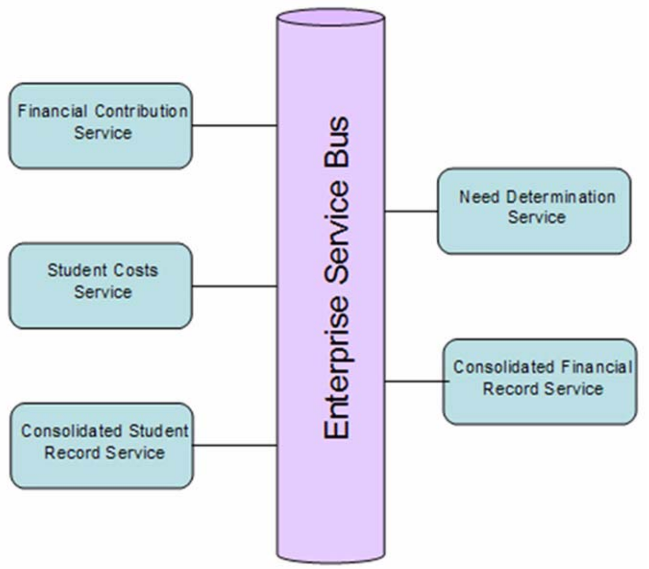

Financial Contribution Service
Student Costs Service
Consolidated Student Record Service
Enterprise Service Bus
Need Determination Service
Consolidated Financial Record Service

Figure 9.4 Internal connectivity pattern.

By adopting internal connectivity through an ESB, the organization now has access to identified services irrespective of their deployment location, allowing flexibility to change processes in response to changing requirements.

The alternative to an Internal Connectivity pattern using ESB is to have point-to-point connectivity, which may work well to support a single business process, but does not make the service generally available and reusable across multiple consumers. Services such as Student Costs and Financial Contribution are good candidates for reuse and thus are best made available generally using the Internal Connectivity pattern.

9.3.3 Application of Process Automation Pattern

The process automation pattern, shown in Figure 9.5, addresses the problem of how to automate workflows including the integration of automated human tasks. It also addresses the ability to automate integration across multiple applications and back-end repositories. By realizing the Needs Analysis business process through choreography of services and automating the same, allows the organization to respond to changes quickly and efficiently.

Applying the three patterns begins to provide the organization with the architectural flexibility it needs and the alignment between its IT systems and business objectives it desires (as shown in Figure 9.6). Particularly, it improves the effectiveness of the organization in fulfilling new requirements from its user ecosystem.

The following section particularly highlights the different dimensions along which the organization improves its maturity and effectiveness.

9.4 How the solution helps increase SOA Maturity & Organization Effectiveness

By adopting the solution laid out in this Section, the organization has increased its overall IT maturity and effectiveness and is now more aligned to the business and business services provided by the organization and thus ready for future challenges and requirements coming from its users. Using the SOA Maturity Table (introduced in Section 6.2) we can highlight the specific dimensions along which the organization has become more effective.

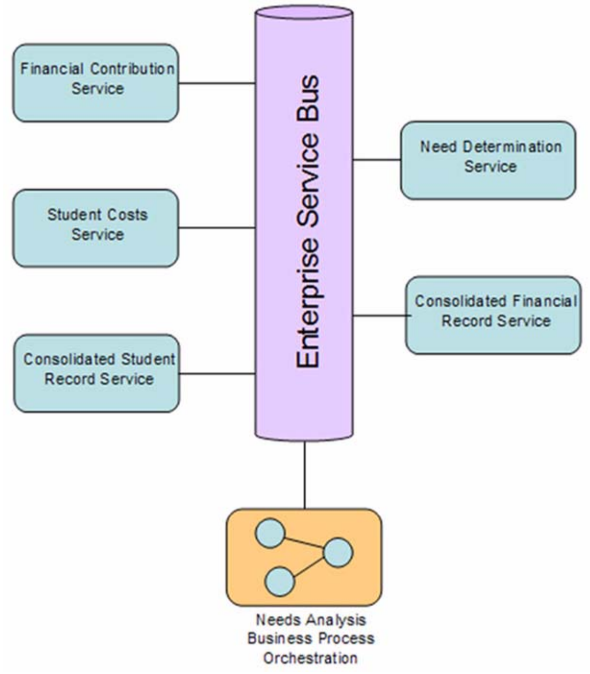

Financial Contribution Service
Student Costs Service
Consolidated Student Record Service
Enterprise Service Bus
Need Determination Service
Consolidated Financial Record Service
Needs Analysis Business Process Orchestration

Figure 9.5 Process automation pattern.

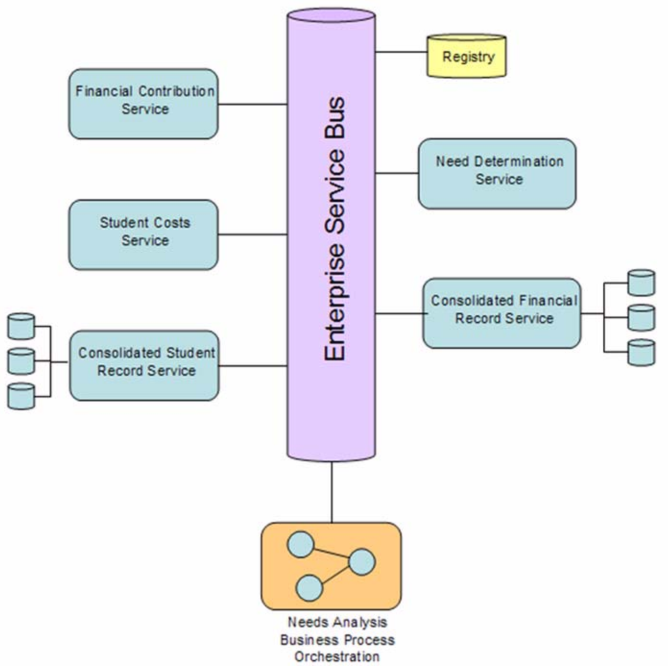
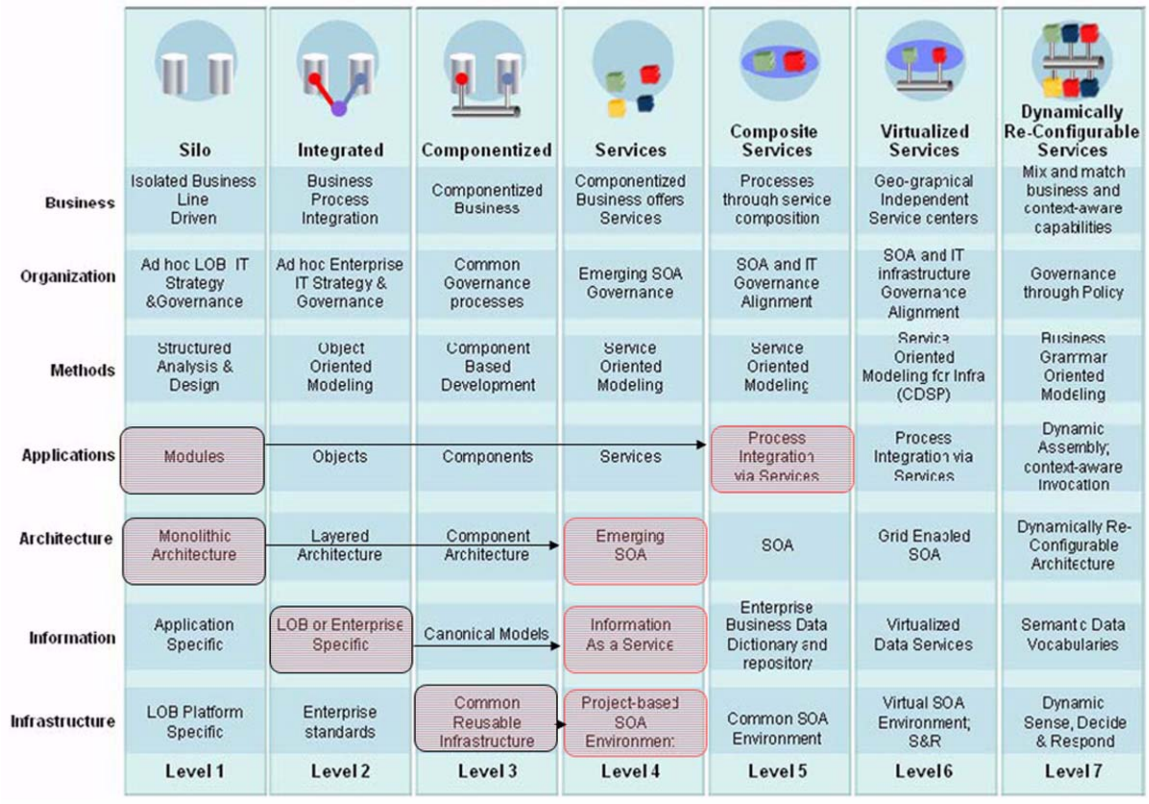

Registry
Financial Contribution Service
Student Costs Service
Consolidated Student Record Service
Enterprise Service Bus
Need Determination Service
Consolidated Financial Record Service
Needs Analysis Business Process Orchestration

Figure 9.6 Integrated view of logical deployment pattern.



	Silo	Integrated	Componentized	Services	Composite Services	Virtualized Services	Dynamically Re-Configurable Services
Business	Isolated Business Line Driven	Business Process Integration	Componentized Business	Componentized Business offers Services	Processes through service composition	Geo-graphical Independent Service centers	Mix and match business and context-aware capabilities
Organization	Ad hoc LOB IT Strategy &Governance	Ad hoc Enterprise IT Strategy & Governance	Common Governance processes	Emerging SOA Governance	SOA and IT Governance Alignment	SOA and IT infrastructure Governance Alignment	Governance through Policy
Methods	Structured Analysis & Design	Object Oriented Modeling	Component Based Development	Service Oriented Modeling	Service Oriented Modeling	Service Oriented Modeling for Infra (CDSP)	Business Grammar Oriented Modeling
Applications	Modules	Objects	Components	Services	Process Integration via Services	Process Integration via Services	Dynamic Assembly; context-aware invocation
Architecture	Monolithic Architecture	Layered Architecture	Component Architecture	Emerging SOA	SOA	Grid Enabled SOA	Dynamically Re-Configurable Architecture
Information	Application Specific	LOB or Enterprise Specific	Canonical Models	Information As a Service	Enterprise Business Data Dictionary and repository	Virtualized Data Services	Semantic Data Vocabularies
Infrastructure	LOB Platform Specific	Enterprise standards	Common Reusable Infrastructure	Project-based SOA Environment	Common SOA Environment	Virtual SOA Environment; S&R	Dynamic Sense, Decide & Respond
	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7

Figure 9.7 Increase in SOA maturity.

As Figure 9.7 illustrates, the fundamental areas along which the solution increases the maturity and effectiveness of the IT environment are:

a) Application Architecture – the new application architecture is now a choreography of an inventory of services in support of business processes. This introduces deep flexibility in changing business processes in response to changing business needs. Furthermore, this encourages the use of best of breed services further improving efficiency of the supported business processes. Finally, re-use of services becomes a routine matter as the services are created at the lowest level of business use;

b) IT Architecture – the IT architecture at the Enterprise is now supporting a service oriented approach applying select SOA patterns to achieve its goals. This approach makes services easily accessible and widely available resulting in increased reuse and availability of information;

c) Information – prior to applying the SOA approach, information sharing was restricted to the set of applications that made a point-to-point connection with each other. By applying the selected solution to the scenario, information is now available as a service backed with consolidated data to ensure accuracy and completeness of information. New consumers of the service simply need to follow a standardized way of accessing the service, doing away with point transformations of data and protocol;

d) Infrastructure – by introducing elements of SOA and SOA tools, the IT environment has begun supporting project-based SOA infrastructure. These investments can be fully leveraged for future projects, reducing the associated cost of the solution.

10. Cross Institution Records – A Learner-centered ePortfolio Scenario

10.1 Context

Education institutions are facing growing demands to provide more flexible access to information about student learning. This information may be required for reporting purposes, cross-institutional enrollment or, as in this scenario, it may be required by the students. Students may wish to aggregate information about their learning for a variety of purposes such as applying for employment and further study or to gain formal recognition of existing skills. In this scenario, a student will use an ePortfolio to aggregate information and evidence of learning across different organizations. Although the term ePortfolio is applied to a range of different contexts in education, for the purposes of this scenario an ePortfolio is a student-controlled aggregation of information and evidence of that student's learning. ePortfolio information may be derived from a wide range of sources; from a Web 2.0 viewpoint an ePortfolio could be considered a type of mash-up. This scenario is aspirational and illustrates a transition towards an environment where learners are able to more easily access digital information about their learning and experiences to support their lifelong education and career goals.

10.1.1 Scenario Details

The scenario being addressed is:

> *To allow representative work for a student, from several institutions that the student has attended, to be aggregated for presentation externally. Work is drawn from a variety of LMSs and content applications, in a variety of formats and from several institutions. Wherever available, SOA can be used to transform the work into a consistent presentation, arrange for the requisite authentication and authorization, allow content to be aggregated on the fly (including changes in release policy and presentation) and deliver the aggregation securely where appropriate. This is useful to expose a single's institutions content in an ePortfolio, but even more effective when SOA solutions are in place between organizations, so that a learner-centered ePortfolio can be mashed up across institutions.*

This scenario specifically positions the ePortfolio system as a consumer of services that provide information about student learning. Although focused on ePortfolios in this instance, the services described could also serve a variety of other purposes. For example, in some regions or countries information from these services may be aggregated in a centralized learner record of a learner's education qualifications. In both cases, the requirements for the service provider systems would be broadly similar.

10.2 SOA Analysis of the Scenario

10.2.1 Entry Point

There is an increasing pressure for education institutions to provide more timely, secure and flexible access to digital information about learners and their learning progress and outcomes. With learners often studying at multiple schools, colleges and universities (sometimes concurrently), sharing information about students across organizations is also becoming increasingly important. There are a number of potential ways to approach this scenario. For example, we could start by considering the requirements for student information as a service. However, in this scenario, the SOA analysis begins with an attempt at understanding the sources of this information and the final consumers who need this information. This understanding of source and consumer will be clarified by mapping out the business process where these services are used.

10.2.2 Business Processes

The following business process illustrates some of the possible ways a student may use an ePortfolio to access and use services that provide information about their learning. This business process assumes a student-centered approach to the use of services in an ePortfolio. Although the ePortfolio software may interpret, contextualize or render the content derived from a service, the student is the actor that determines how that information is used in the ePortfolio and under what conditions other actors (people and systems) may access it.

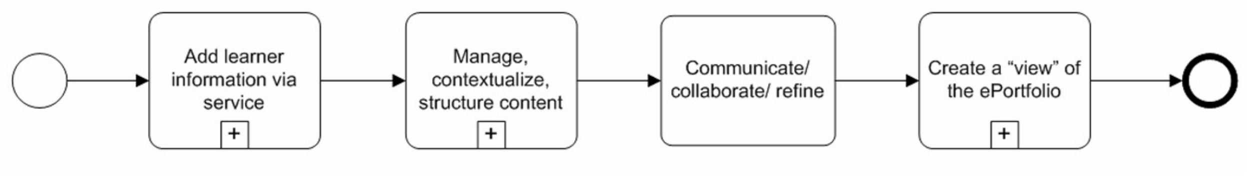

Add learner information via service
+
Manage, contextualize, structure content
+
Communicate/ collaborate/ refine
Create a “view” of the ePortfolio
+

Figure 10.1 Student adds information about their learning to their ePortfolio

In Figure 10.1 the “learner information” added to the ePortfolio in the first activity could come from a very wide range of sources including education organization systems, the WWW, e-government services or Human Resources (HR) systems. The ePortfolio system may also standardize the formatting of student information from disparate sources to enable a cohesive presentation.

Two common education organization systems that contain learner information that a student may wish to access is SIS and LMS. These sources may be considered service providers, while the ePortfolio system (under the student’s authorization) is a service consumer.

In some circumstances (for example K-12/Schools) a third party such as parent/guardian or teacher/trainer or lecturer may also be part of this business process.

10.2.3 Identifying Candidate Services

In order to add learner information into an ePortfolio, services for systems holding this type of information are required. As mentioned in the previous Section, two very common education organization enterprise systems are the LMS and SIS. Most education organizations would consider building services for each of these systems. In the case of the SIS service, this could be broken down further into different types of specific services providing particular types of data including for example:

- Evidence of enrollment service;
- Transcript of results service;
- LMS content service.

The consumer (in this case an ePortfolio system) may require real-time data or content or there may be a one-off process of transferring information into the ePortfolio. Security, access and authentication are important considerations for any service developed for these purposes.

10.2.4 Prioritizing Services

As previously discussed, a particular system may potentially provide multiple services where different types of data may be supplied. The SIS is a good example. Typically, an organization may not have the resources to expose every identified candidate as a service and thus needs to prioritize the services exposed. Some of the criteria for prioritization are highlighted in Table 10.1. Additional criteria that could help prioritize services include:

- Leveraging Existing Capability – as the system already provides transcript of results data for reporting purposes, this existing functionality may be repurposed for this service;
- External consumers – as the service will often be used by systems external to the organization, special security and data considerations need to be taken into account. As this is sensitive personal information it is vital that the student is able to effectively control the conditions of access to this information.

Additional dimensions for prioritizing services can be added to the above list to help pick the right set for realization including criteria such as:

- Is the Service replacing existing application function or introducing brand new capabilities?
- What is the overall impact on Return on Investment (value versus cost)?

Table 10.1 Prioritizing criteria for services (Transcription of Results data Service).

Multiple Consumers	**High**	A large number of past and present students and authorized third party actors will utilize this service (usually via an e-portfolio tool or similar).
Spanning Domains	**High**	This service will be used by systems and organizations outside the education organization.
Business Timely Information	**Medium**	Depending on implementation, the service consumer may require real-time access to data.
Business Alignment / Cost-effective	**Medium**	With planning, this service could also be derived from information on student outcomes provided for reporting purposes.

10.3 Applying SOA Patterns to Solution Realization

As many of the issues of access, authentication and data security are common to other scenarios; existing SOA patterns may be utilized to facilitate implementation. A list of SOA patterns that are of relevance may include the following:

Access control and authentication

- Brokered authentication pattern – the ePortfolio system will require access to services across multiple organizations in an environment where the service provider and service consumer will not necessarily be known to each other. Establishing trust relationships between service providers and consumers will be necessary before personal information can be accessed;
- Data confidentiality pattern – encryption of messages between service provider and the ePortfolio system (the service consumer) will help to protect personal data;
- Trusted subsystem pattern – service providers can utilize this pattern to ensure that an ePortfolio system or other service consumers use the service as intended and do not access resources directly from the data source;
- Service perimeter guard pattern – this pattern is used to ensure that ePortfolio systems are not able to access unauthorized data and content from a service provider.

Verification of content

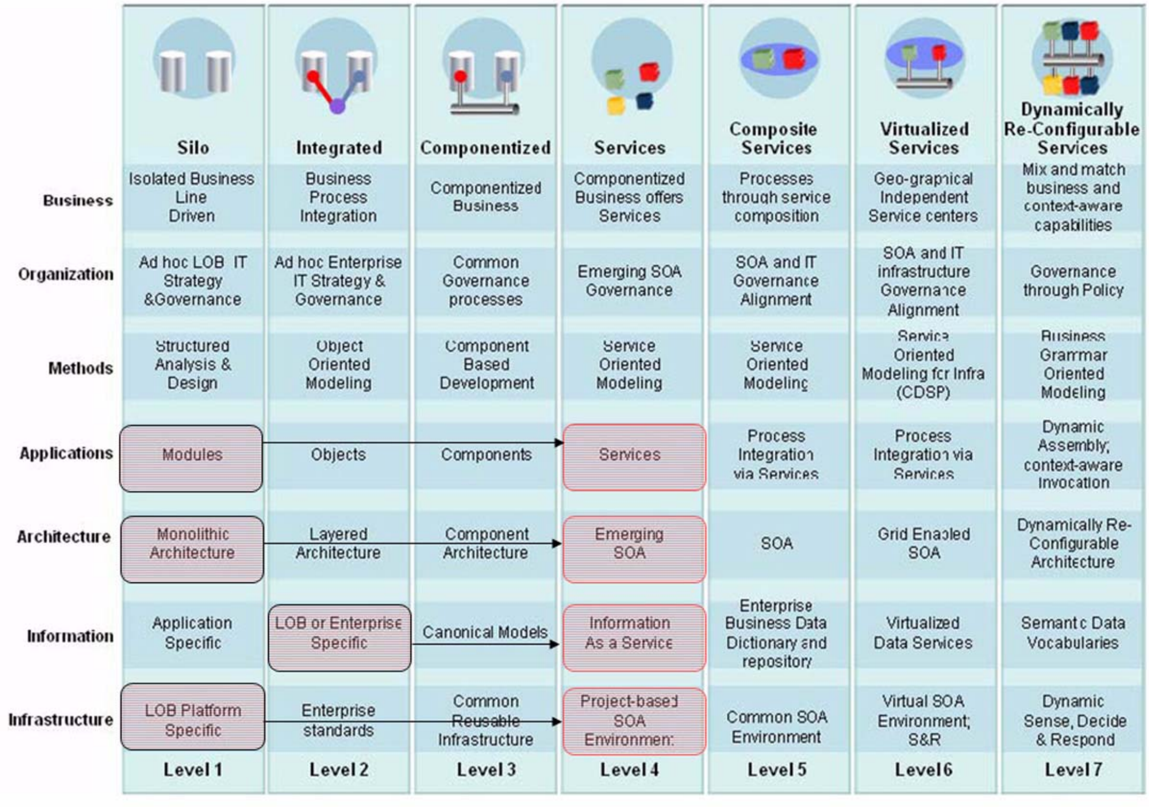
- Data origin verification – verification of the source of the content and/or data can be very important for confirming education qualifications and other personal data derived from a service provider.

Governance and good practice

- Metadata centralization – this pattern provides a method of describing a service so that ePortfolio system developers are able to find and utilize available services. This is particularly important as services are being used across organizations;
- Process abstraction – separating task specific logic can support good practice and make functionality more reusable and easier to manage. It may be possible to develop a number of similar but distinct services based on the end user requirements such as education organization or employer.

10.4 How the Solution Helps Increase SOA Maturity & Organization Effectiveness

Figure 10.2 maps the progress of this scenario against the SOA maturity model.



	Silo	Integrated	Componentized	Services	Composite Services	Virtualized Services	Dynamically Re-Configurable Services
Business	Isolated Business Line Driven	Business Process Integration	Componentized Business	Componentized Business offers Services	Processes through service composition	Geo-graphical Independent Service centers	Mix and match business and context-aware capabilities
Organization	Ad hoc LOB IT Strategy &Governance	Ad hoc Enterprise IT Strategy & Governance	Common Governance processes	Emerging SOA Governance	SOA and IT Governance Alignment	SOA and IT infrastructure Governance Alignment	Governance through Policy
Methods	Structured Analysis & Design	Object Oriented Modeling	Component Based Development	Service Oriented Modeling	Service Oriented Modeling	Service Oriented Modeling for Infra (CDSP)	Business Grammar Oriented Modeling
Applications	Modules	Objects	Components	Services	Process Integration via Services	Process Integration via Services	Dynamic Assembly; context-aware invocation
Architecture	Monolithic Architecture	Layered Architecture	Component Architecture	Emerging SOA	SOA	Grid Enabled SOA	Dynamically Re-Configurable Architecture
Information	Application Specific	LOB or Enterprise Specific	Canonical Models	Information As a Service	Enterprise Business Data Dictionary and repository	Virtualized Data Services	Semantic Data Vocabularies
Infrastructure	LOB Platform Specific	Enterprise standards	Common Reusable Infrastructure	Project-based SOA Environment	Common SOA Environment	Virtual SOA Environment; S&R	Dynamic Sense, Decide & Respond
	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7

Figure 10.2 Increase in SOA maturity for the scenario.

In taking a view of existing systems as service providers, the organization is able to develop and deploy a number of services of value to their learners. In addition to these services having relevance to ePortfolio systems, in the future these services may also be utilized by students in different ways to benefit their future education or career prospects.

11. Accelerators to Adopting a Service Oriented Approach – Getting a Quicker Return on Investment

Adopting a service oriented approach holds the promise of achieving alignment between the services provided by an organization and the technology supporting it such that changes in one are easily reflected in the other without complex re-engineering or re-integration. Furthermore, with technological capabilities now being made available as a set of re-usable services, it permits organizations to combine and re-combine these into more complex services as needed to suit the needs of the organization or department. As an additional benefit, these services could now be provided by best-of-breed providers and combined in a seamless fashion.

This document has highlighted the elements of SOA, demonstrating with the help of select scenarios how this approach would help solve some of the known issues that an education organization faces. This chapter focuses on some practices that help accelerate the adoption of a services approach in an education organization.

11.1 Organization Level Accelerators

Organization leaders and IT leaders should both understand the services approach. Since the goal is to align the technology being provisioned in the form of services that an organization renders, both leaders should grasp the implications of such a move:

- **Ownership** – service provided in one part of the organization could be re-used elsewhere. Who bears ownership and responsibility for maintaining such a service?
- **Security and Service Level Agreement** – should the service be provided beyond the organization to say other collaborating institutions? What should the terms and conditions be (service level agreement)? How is the service maintenance to be shared?
- **Implications of change –** while a services approach makes changes simpler, the greater number of consumers of the service also creates a ‘drag’ to maintain the status quo. How best can this be addressed?
- **Greater role for Open Source Applications** – while not a direct feature of a services approach, SOA will permit an organization to use services from a variety of different sources, including from the open source community. Leveraging this opportunity appropriately would need an understanding from an organizational perspective in terms of how best to incorporate services from the open source community while guaranteeing the quality of service to its user ecosystem.

11.2 Organizational Application Integration Accelerators

SOA not only makes technological capabilities available as reusable services, it also integrates many of these services into useful processes and higher-level services and thus benefits from available information on:

- **Availability of an Application Integration Map** – if a picture or analysis of the existing integration of an organization’s application resources exist, this can prove very helpful in accelerating the creation of a service oriented architecture as it already begins to detail the data flowing between different applications and the functional dependencies that exist between these applications;
- **Availability of a Reference Architecture –** a reference architecture for the organization or department that demonstrates how the architecture should behave ideally is often very helpful in guiding the set of services to be realized and then integrated to create higher level services.

11.3 Service Design & Implementation Accelerators

Service Design & Implementation takes into account several factors, covering both functional and non-functional aspects of a service (such as availability, security, auditing and logging). A few factors that help accelerate design and implementation are:

- **Prototyping one service design and implementation** completely, and providing it as an example for other developers to use. This is useful in sharing even with vendors who may be developing services for the organization as sometimes services reflect unique requirements specific to that particular organization;
- **Clearly understanding consumers of the service** and the type of load each will bring to bear on the service. This is one of the “blind spots” of service design and realization, and knowing more about the consumers of a service helps in the design and deployment of long-lasting service;
- **Incorporating Open Standards** – service design is the stage to include open standards as a core part of a SOA strategy. Open standards provide a ready made message and service interface, allowing the organization to leverage industry best practices while improving integration and interoperability;
- **Realizing services using Service Component Architecture** – another accelerator that helps with service realization is the use of the open standard specification of Service Component Architecture (SCA) and Service Data Objects (SDOs) that simplifies the creation of services and access to various types of data objects.

11.4 Project Roll-out Accelerators

From an SOA roll-out perspective the most important accelerators to success are:

- **Picking the right-sized roll-out project** – preferably at a contained departmental level and working the project till it yields a positive ROI. First projects typically involve significant investment in new software, tooling and integration, and thus need to be contained, yet valuable for the organization/department;
- **Planned leveraging of previous investments** – both from an existing applications as well as middleware perspective in order to continue to keep the cost of the project low;
- **Applying governance and service management policies incrementally** – as size of project grows as opposed to taking a big-bang approach.

These are some of the factors that would help an education organization accelerate the roll-out of a service oriented architecture.

References

[Erl, 07] *SOA: Principles of Service Design*, Thomas Erl, Prentice Hall, 2007, ISBN -13-234482-3.

[GWS, 05] *IMS GLC General Web Services Basic Profile v1.0 Final Specification*, C.Schroeder, J.Smon and C.Smythe, IMS Global Learning Consortium, December 2005.

[IAF, 03c] *IMS GLC Abstract Framework: White Paper v1.0*, Ed. C.Smythe, IMS Global Learning Consortium, July 2003.

[LIS, 09] *IMS GLC Learning Information Services Specification v2.0*, CM/DN Release, L.Feng, B.Lee and C.Smythe, IMS Global Learning Consortium Inc., May 2009.

[OASIS, 06] *Web Services Distributed Management: Management Using Web Services (MUWS 1.1) Part 1*, OASIS Standard, 1st August 2006. [http://docs.oasis-open.org/wsdm/wsdm-muws1-1.1-spec-os-01.htm].

[OASIS, 07] *Web Services Business Process Execution Language Version 2.0*, OASIS Standard, 11th April 2007. [http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html].

[OpenGroup, 06] *The Open Group Service Integration Maturity Model (OSIMM)*, Draft, The Open Group, 2006.

[W3C, 06] *Web Services Choreography Description Language: Primer*, W3C Working Draft, W3C, 19th June 2006. [http://www.w3.org/TR/2006/WD-ws-cdl-10-primer-20060619/].

Appendix A – Abbreviations

BAM	Business Activity Monitoring
BMM	Business Motivation Modelling
BPE	Business Process Execution
BPEL	Business Process Execution Language
BPMM	Business Process Maturity Model
BPMN	Business Process Modelling Notation
CEP	Complex Event Processing
CIM	Common Information Model
CRUD	Create, Read, Update, Delete
DODAF	Department of Defense Architecture Framework
EAI	Enterprise Application Integration
EMP	Event Metamodel and Profile
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
FERPA	Family Educational Rights and Privacy Act
GWS	General Web Services
HE	Higher Education
HEI	Higher Education Institution
HR	Human Resources
IAAS	Infrastructure As A Service
IAF	IMS GLC Abstract Framework
ICT	Information & Communication Technology
IDE	Integrated Development Environment
IMS GLC	IMS Global Learning Consortium Inc.
IT	Information Technology
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LCMS	Learning Content Management System
LIS	Learning Information Services
LMS	Learning Management System
MODAF	Ministry of Defence Architecture Framework
MTOM	Message transmission Optimization Mechanism
OASIS	Organization for the Advancement of Structured Information Standards
OMG	Object Management Group
OO	Object-Oriented

BAM	Business Activity Monitoring
OSIMM	Open Group Service Integration Maturity Model
PAAS	Platform As A Service
PC	Personal Computer
PDA	Personal Digital Assistant
PESC	Postsecondary Electronic Standards Council
QoS	Quality of Service
REST	Representational State Transfer
RIF	Rules Interchange Format
ROI	Return on Investment
SAAS	Software As A Service
SAML	Security Assertion Mark-up Language
SCA	Service Component Architecture
SDO	Service Data Object
SIS	Student Information System
SLA	Service Level Agreement
SML	Service Modelling Language
SOA	Service Oriented Architecture
SoaML	SOA Mark-up Language
TOGAF	The Open Group Architecture Framework
UDDI	Universal Description, Discovery, and Integration
UML	Unified Modelling Language
URL	Uniform Resource Locator
WS-BPEL	Web Service Business Process Execution Language
WS-CDL	Web Services Choreography Description Language
WSDL	Web Services Description Language
WS-DM	Web Services Distributed Management
WS-I	Web Services Interoperability Organization
WS-RM	Web Services Reliable Messaging
XAMCL	Extensible Access Control Mark-up Language
XML	Extensible Mark-up Language
XMLSIG	XML Signature

Appendix B – Glossary of Terms

Business Process Execution Language (BPEL)

BPEL defines a notation for specifying business process behavior based on Web Services. Processes in BPEL export and import functionality by using Web Service interfaces exclusively. Business processes can be described in two ways. Executable business processes model actual behavior of a participant in a business interaction. Business protocols, in contrast, use process descriptions that specify the mutually visible message exchange behavior of each of the parties involved in the protocol, without revealing their internal behavior. The process descriptions for business protocols are called abstract processes. BPEL is intended to model the behavior of both executable and abstract processes. BPEL provides a language for the formal specification of business processes and business interaction protocols. By doing so, it extends the Web Services interaction model and enables it to support business transactions. BPEL defines an interoperable integration model that should facilitate the expansion of automated process integration in both the intra-corporate and the business-to-business spaces.

Business Process Management

This is the field of management focused on aligning organizations with the wants and needs of the clients. It is a holistic management approach that promotes business effectiveness and efficiency while striving for innovation, flexibility and integration with technology. Business process management attempts to continuously improve processes. It could therefore be described as a “business optimization process”. [From Wikipedia].

Business Process Modelling Notation (BPMN)

BPMN is a graphical notation that depicts the steps in a business process. BPMN depicts the end-to-end flow of a business process. The notation has been specifically designed to coordinate the sequence of processes and the messages that flow between different process participants in a related set of activities. BPMN is targeted at a high level for business users and at a lower level for process implementers. The business users should be able to easily read and understand a BPMN business process diagram. The process implementer should be able to adorn a business process diagram with further detail in order to represent the process in a physical implementation. BPMN is targeted at users, vendors and service providers that need to communicate business processes in a standard manner.

Cloud Computing

Cloud computing is a style of computing in which dynamically scalable and often virtualized resources are provided as a service over the Internet. Users need not have knowledge or expertise in, or control over the technology “in the cloud” that supports them. The concept incorporates infrastructure as a service (IAAS), platform as a service (PAAS) and software as a service (SAAS) as well as Web 2.0 and other recent technology trends that have the common theme of reliance on the Internet for satisfying the computing needs of the users.
[From Wikipedia].

Componentized Level (OSIMM level 3)

The third lowest level (3) of maturity in the OSIMM. At this level of maturity the IT systems in the silos have been analyzed and broken down into component parts, with a framework in which they can be developed into new configurations and systems. There may also be some limited analysis of the business functionality into components. Although components interact through defined interfaces, the way that these components interact together is not loosely coupled, which limits the interoperability between systems in different parts of the organization (or even different organizations within the business “eco-system”), and causes difficulties in development of business processes that can be constructed across the parts of the organization.

Composite Services Level (OSIMM level 5)

The third highest level (5) of maturity in the OSIMM. At this level of maturity it is now possible to construct a business process for a set of interacting services, not just by bespoke development, but by the use of a composition language, such as BPEL, to define the flow of information and control through the individual services. This permits the assembly of business services into composite business processes, which may be short or long running, without significant construction of code. Thus the design and development of business services is agile, and may be performed by developers under the close guidance of business analysts.

Dynamically Re-configurable Services (OSIMM level 7)

The highest level (7) of maturity in the OSIMM. Prior to this level, the business process assembly, although agile, is performed at design time by developers (under the guidance of business analysis and product managers) using suitable tooling. Now this assembly may be performed at "runtime", either assisted by the business analysts via suitable tooling, or by the system itself. This requires the ability to access a repository of services and to query this repository via the characteristics of the required services. In its simplest form, these characteristics may have been defined in advance, restricting the system to selecting and locating specific instances of services.

Enterprise Service Bus (ESB)

ESB refers to a software architecture construct. This construct is typically implemented by technologies found in a category of middleware infrastructure products, usually based on recognized standards, which provide foundational services for more complex architectures via an event-driven and standards-based messaging engine (the bus). An ESB generally provides an abstraction layer on top of an implementation of an enterprise messaging system that allows integration architects to exploit the value of messaging without writing code. Contrary to the more classical Enterprise Application Integration (EAI) approach of a monolithic stack in a hub and spoke architecture, the foundation of an ESB is built from base functions broken up into their constituent parts, with distributed deployment where needed, working in harmony as necessary. ESB does not implement SOA but provides the features with which one may be implemented. ESB should be standards-based and flexible, supporting many transport mediums. Based on EAI rather than SOA patterns, it tries to remove the coupling between the service called and the transport medium.

Entity Services

These are services that have their functional context scoped by the entities that they manage. For example, granularity of a Student Management Service is defined by the Student entity. In this case a Student Management Service would include all the capabilities that are necessary to maintain the Student entity.

ePortfolio

An electronic portfolio, also known as an ePortfolio or digital portfolio, is a collection of electronic evidence assembled and managed by a user, usually on the Web. Such electronic evidence may include inputted text, electronic files, images, multimedia, blogs, etc. ePortfolios are both demonstrations of the user's abilities and platforms for self-expression, and, if they are online, they can be maintained dynamically over time. Some ePortfolio applications permit varying degrees of audience access, so the same portfolio might be used for multiple purposes. An ePortfolio can be seen as a type of learning record that provides actual evidence of achievement. Learning records are closely related to the Personal Learning Plan.

Integrated Level (OSIMM level 2)

The second lowest level (2) of maturity in the OSIMM. At this level of maturity, technologies have been put in place to communicate between the silos and to integrate the data and interconnections. The construction of an IT system that integrates across different parts of the organization becomes possible. However integration does not extend to common standards in data or business processes. Therefore to connect two systems, it requires a, possibly complex, conversion of the data, operations and protocols used by these systems. Each such connection may require bespoke code and adapters, leading to a proliferation of software that is difficult to manage and complex to code. Therefore, it is not easy to develop new business processes.

IT Governance

Governance means the set of practices, policies and processes that are put in place to increase the likelihood that an entity functions as intended. Corporate governance is the set of practices and policies that dictate the way a corporation is controlled and directed, while IT Governance is the set of practices and policies that ensure that IT investments are generating the intended value for the business.

Process Services

These are services that usually deal with a larger functional context defined by the encapsulated business process. For example, Financial Aid Management Service would include all the capabilities necessary to manage financial aid.

Service

A service is a discrete element of functionality, used to support a business process or function i.e. a service is a piece of computer code used to do something useful, typically a logical step in an overall business process of an enterprise (the workflow or set of operations). In a service-oriented computing environment, services can be thought of as the building blocks used in constructing an overall computing environment or ICT system to solve the overarching business problem. To be useful, a service must provide a defined capability to do something. The service must be implemented in operational code and must be deployed and managed in some way. Services may be used in combination to support the needs of the business, each service providing one part of the overall operation.

Service Coupling

Services need to interact to solve business problems, a service may be defined to be independent of any other service and can function without the knowledge of how other services work, or it may have explicit knowledge about not only what other services are available, but how they perform their operations. Services that are not dependent on other services are *loosely coupled*, while those that know about other services are (more) *tightly coupled*.

Service Granularity

The complexity of the business process provided by the service may range in scale. The service may define a small process (or many small processes) where each request performs a simple operation and many requests are needed to complete a business process i.e. *fine grained* or the service may provide a substantial process that maps directly to a business-level step i.e. *coarse grained*.

Service Level (OSIMM level 4)

The middle level (4) of maturity in the OSIMM. At this level of maturity, composite applications can now be built from loosely-coupled business services. The way that services may be invoked is based upon open standards and independent of the underling application technology, and running on an IT infrastructure that supports the services with suitable protocols, security mechanisms, data transformation and service management capabilities. The services may therefore interoperate across all of the parts of the organization and even across different organizations within the eco-system, and may be managed by assigning responsibilities for SLAs to relevant parts of the organization. However the flow of control within a composite application is still defined by bespoke programming, rather than by a declarative flow language. The business functionality has been analyzed in detail and is broken down into business services residing within a business architecture that ensures that business services will interoperate at the business level. In addition, it is possible to define the services via a specification language that unambiguously defines the operations performed by the service, permitting the construction of a catalogue of services. The combination of IT and business service architectures permits the construction of systems based upon these services, operating right across the organizations in the ecosystem. However at this stage the composition of services is still performed by developers writing bespoke code, thus limiting the agility of the development of new business processes.

Service Oriented Architecture (SOA)

Service Oriented Architecture defines how two or more entities interact in such a way as to enable one entity to perform a unit of work on behalf of another entity. The unit of work is referred to as a service and the service interactions are described using a well-defined description language. Each interaction is self-contained and loosely coupled, so that each interaction remains independent of any other interaction. While SOAP-enabled Web Services is the most common implementation of SOA, Web Services are not necessarily required to define a SOA. The protocol independence of SOA means that different consumers can use services by communicating with the service in different ways. Service orientation is a method of architecting systems of autonomous services. Services are built to last, with good availability and reliability, and systems are built to change, with new service topologies evolving over time.

Silo Level (OSIMM level 1)

The lowest level (1) of maturity in the OSIMM. At this level of maturity individual parts of the organization are developing their own software independently, with no integration of data, processes, standards or technologies. This severely limits the ability of the organization to implement business processes that require co-operation between the different parts, and the IT systems cannot be integrated without significant manual intervention, such as re-keying and re-interpretation of data.

SOA Governance

This is the set of practices and policies that ensure Service Oriented Architecture and its component services represent and conform to the objectives and requirements of the business. It involves not only IT support processes to ensure regulatory compliance and quality of service, but also organizational processes to ensure that stakeholders are receiving value from the architecture.

SOA Maturity Model

Integration maturity is a progressive realization in a service-oriented world, going from tight-integration and tight-coupling to loose-coupling and availability of Information and Business Capabilities as an IT Service. There are multiple dimensions along which service integration maturity can be measured or captured, and it also provides a roadmap for where future efforts should be focused on in order to fully realize the power and benefits of a SOA. The framework by which this integration progress can be analyzed is termed a SOA Maturity Model. The Open Group's Service Integration Maturity Model is one such framework.

Task Services

These are services that typically contain a group of capabilities related to the same business task, for example Tax Calculation Service. Task services tend to be fairly granular.

Utility Services (Infrastructure Services)

These services are defined by grouping together infrastructural capabilities with a common purpose, for example Logging Service, and thus granularity is directly defined by the utility functions supported.

Virtualization

This is a broad term that refers to the abstraction of computing resources. This includes: platform virtualization, resource virtualization, computer clusters and grid computing, application virtualization and desktop virtualization.
[From Wikipedia].

Virtualized Services Level (OSIMM level 6)

The second highest level (6) of maturity in the OSIMM. At this level of maturity the business and IT services are now provided through a façade, a level of indirection. The service consumer does not invoke the service directly, but through the invocation of a “virtual service”. The infrastructure performs the work of converting the virtual invocation into a physical call of the real service, and may as part of this conversion change the address, the network, the protocol, the data and the synchronization pattern that is contained in the call. Such conversions may be a complex service in their own right, such as the transformation of data from one data model to another. The virtual service thereby becomes more loosely coupled from the infrastructure on which it is running, permitting more opportunities for the composition of business services. This is in contrast to the lower levels of service maturity where the service is more closely coupled to the infrastructure. Although virtualization has been used in non-SOA systems, this level extends the concept (and advantages) of virtualization to business services.

Appendix C – Overview of Relevant Standards & Specifications

Archimate: *Frameworks* Modelling language for enterprise architecture (http://www.archimate.org).

ATOM: *REST* Specifies a syndication format, and a simple protocol for creating and updating resources. Often used in conjunction with the REST paradigm (http://www.ietf.org/rfc/rfc4287.txt, http://tools.ietf.org/html/rfc5023).

BMM: *Modeling:* Business Motivation Modelling. Defining business plans, including business processes, business rules, and organizational units (http://www.omg.org/spec/BMM/1.0/).

BPEL/WS-BPEL: *Choreography* Business Process Execution Language – An executable language for describing business processes via interactions with web services. (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel).

BPEL4People/WS-HumanTask: *Choreography* Extensions to WS-BPEL for human interactions (http://www.oasis-open.org/committees/bpel4people/charter.php).

BPMM: *Methodology* Business Process Maturity Model (http://www.omg.org/spec/BPMM/1.0/).

BPMN: *Modeling* Business Process Modelling Notation, used to represent a business process graphically (http://www.bpmn.org).

CIM: *Management* Common Information Model. Defines how managed IT resources are represented as objects (http://www.dmtf.org/standards/cim/).

DODAF/MODAF: Department of Defense (and Ministry of Defense) Architecture Framework. A reference model for enterprise and systems architecture (http://jitc.fhu.disa.mil/jitc_dri/pdfs/dodaf_v1v2.pdf, http://www.modaf.org.uk/).

EMP: *Modeling* Event Metamodel and Profile. (http://www.omgwiki.org/soaeda/doku.php).

GWS: *Education* IMS General Web Services. Profile promoting web services interoperability for education web services (http://www.imsglobal.org/gws/index.html).

LIS: *Education* IMS Learning Information Services. Interactions and data exchange between learning systems and administrative, student, or human resource systems (http://www.imsglobal.org/enterprise.cfm).

JSON: *REST* JavaScript Object Notation. Data exchange format frequently used with REST (http://tools.ietf.org/html/rfc4627).

MTOM: *XML* Message Transmission Optimization Mechanism. Standard for optimizing binary data transmission over XML (http://www.w3.org/TR/soap12-mtom/).

REST: *REST* Representational State Transfer. Not a standard or specification, but a set of principles on how to define and address resources. Usually seen as complimentary to WS-*/SOAP as an SOA approach (http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm).

RIF: *Rules* Rules Interchange Format (http://www.w3.org/2005/rules/wiki/RIF_Working_Group)

SAML: *Security* Security Assertion Mark-up Language, XML-based protocol and data specification, to exchange authentication data between security domains (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security).

SCA: *ESB* Service Component Architecture, a specification that enables technology-agnostic service invocations (http://www.osoa.org/display/Main/Service+Component+Architecture+Home).

SDO: *ESB* Service Data Objects, the data specification for SCA (http://www.osoa.org/display/Main/Service+Data+Objects+Home).

SML: *Management* Service Modeling Language. XML extensions for modelling and constraining IT resources (http://www.w3.org/XML/SML/).

SoaML: *Modeling* Service Oriented Architecture Mark-up Language. UML Profile and Metamodel for design of services (http://www.omg.org/docs/ad/08-08-04.pdf).

TOGAF 9: *Methodology* The Open Group Architecture Framework Version 9. Detailed method and resources for developing Enterprise Architectures (http://www.opengroup.org/togaf/).

UDDI: *Registry* Universal Description, Discovery and Integration, registry API and protocol specification (http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm).

WS-I Basic Profile: *Connectivity* A specification combining and restricting use of various mainstream web service specifications, including XML, WSDL, SOAP, and WS-Addressing, in order to minimize interoperability problems between independent web service implementations. (http://www.ws-i.org/Profiles/BasicProfile-1.1.html).

WS-I Basic Security Profile: *Connectivity* Security Profile for interoperability (http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html).

WS-CDL: *Choreography* Choreography Description Language – A language for describing collaboration between peers – there is no central control (as opposed to BPEL). (http://www.w3.org/TR/ws-cdl-10/).

WS-Discovery *Registry/Discovery* Dynamic service discovery (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-dd).

WS-DM: *Governance* Web Services Distributed Management, a specification for the managing and monitoring services (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsdm).

WS-Federation: *Security* A specification for allowing distinct security domains to collect and broker identity and authentication information (http://www.oasis-open.org/committees/documents.php?wg_abbrev=wsfed).

WS-Glossary: *General* Standard definitions of web service terms (http://www.w3.org/TR/ws-gloss/).

WS-Notification: *Infrastructure* Publish/subscribe functionality for web services (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsn).

WS-Policy: *Governance* XML-based specification for specifying and advertising policies (http://www.w3.org/TR/ws-policy/).

WS-I Reliable Secure Profile: *Connectivity* An interoperability profile dealing with secure, reliable messaging capabilities for Web services (http://www.ws-i.org/deliverables/workinggroup.aspx?wg=reliablesecure).

WS-RM: *Infrastructure* Protocol specification for ensuring message delivery in the event of software, system, or network failures. (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrm).

WS-Security: *Security* A specification for including security information in SOAP headers – uses XMLENC and XMLSIG. XMLENC, XML encryption ensures confidentiality of one or more parts of the SOAP body using the included security information. XMLSIG, XML signature is used to verify message integrity in the SOAP body using the included security information. (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss).

WS-SecureConversation: *Security* Extensions to WS-Security for requesting and issuing security tokens, as well as brokering trust relationships (http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/ws-secureconversation-1.3-os.html).

WS-SecurityPolicy: *Security* Definitions of WS-Policy assertions for WS-Security (http://docs.oasis-open.org/ws-sx/ws-securitypolicy/v1.2/ws-securitypolicy.html).

WS-Trust: *Security* An extension to WS-Security for issuing, validating, and renewing security tokens, and establishing trust relationships for a message exchange. (http://docs.oasis-open.org/ws-sx/ws-trust/v1.3/ws-trust.html).

XACML: *Security* eXtensible Access Control Markup Language, XML-based data specification and processing model, to set access control policies (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml).

XPATH: *XML* Language for specifying nodes in an XML document. (http://www.w3.org/TR/xpath20/).

XQuery: *XML* Language for querying XML data (http://www.w3.org/XML/Query/).

Appendix D – The Original Scenarios

Scenario Title:	*Integrating LMS/LMS and LMS/Content Applications*
Scenario Local ID:	SOA-01
Brief Description:	Learning institutions are challenged to create a seamless, contextual and interactive environment for online learners. At the same time these institutions are challenged to create an environment for online teachers and course designers that balances flexibility with usability. To enable such an environment there should be a common methodology for integrating Learning Management Systems (LMS) with other academic systems and a variety of other sources of content. Examples of such integrations include: • Expose content from various sources to LMS (or as the case may be multiple LMS across campus); • Integrate LMS with 3rd party "Web 2.0" applications, e.g., Facebook; • Mash-up type examples: wiki and blog systems presented in courses in LMS.
Level:	Summary
Stakeholders & Interest:	LMS Vendors, 3rd Part LMS Tool Vendors
	Educational Institutions e.g. HEI, Schools, FE, Community Colleges

Scenario Title:	Integrating ERP/SIS with LMS
Scenario Local ID:	SOA-02
Brief Description:	Learning institutions have traditionally structured their administrative and academic systems to mirror the hierarchy of the brick and mortar institution. Online learning environments require more flexibility in the way course rosters are populated because instructors tend to teach their students outside of the restrictions imposed by the institutional hierarchy. There should be a common methodology to populate course rosters from student system automatically into LMS. Specifically with: • The flexibility to match multiple student system courses to one LMS course; • Add/drop students from online classes triggered from student system; • Create course sections automatically within LMS when section is created in student system; • Transfer grades from LMS into student system.
Level:	Summary
Stakeholders & Interest:	LMS Vendors,
	Educational Institutions e.g. HEI, Schools, FE, Community Colleges

Scenario Title:	*Integrating ERP/SIS with Library Systems*
Scenario Local ID:	SOA-03
Brief Description:	To enable students who register for a particular class to check out books on the reserve list for that class.
Level:	Summary
Stakeholders & Interest:	LMS Vendors, Library System Vendors
	Educational Institutions e.g. HEI, Schools, FE, Community Colleges

Scenario Title:	*Exposure of ERP Data*
Scenario Local ID:	SOA-04
Brief Description:	Learning institutions are challenged to support a model of centralized computing and yet meet the growing demand for personalized, contextual, flexible access to institutional data. There should be a standard methodology to expose data within ERP systems to be presented in whatever presentation tool or presentation layer that consumers of the data prefer. Examples of data that could be exposed this way include: student course schedules, grades, biographical data etc. Examples of presentation layers or tools could be: portal, web page, composite application etc.
Level:	Summary
Stakeholders & Interest:	ERP System Vendors
	Educational Institutions e.g. HEI, Schools, FE, Community Colleges

Scenario Title:	*Applying for College – A Services Approach*
Scenario Local ID:	SOA-05
Brief Description:	College bound high school seniors applying to a prestigious selective private university in the Midwest are instructed to send any printed materials related to their admissions application to an address in Boston. Even in this digital age, letters of recommendation, high school transcripts, teacher referral documents and a myriad of paper documents still flow through the college admissions process. The ability to receive, scan, store and then provision these documents automatically, through a Services enabled infrastructure, to dozens of college and university admissions offices by a third party provider in the New England area simplifies workflow for the Universities and provides excellent business for the outsourcing outfit. Local campus admissions systems also deploy workflow and SOA technologies to access these services from the said third party provider.
Level:	Summary
Stakeholders & Interest:	Admissions Systems Vendors
	Educational Institutions e.g. HEI, Schools, FE, Community Colleges

Scenario Title:	*Paying for College – Student Financial Services*
Scenario Local ID:	SOA-06
Brief Description:	With the rapid development of e-commerce and web-based solutions, students and families at universities across the country demand and now receive electronic invoices for their college tuition and fee charges, which they routinely review and pay online. While running their local student financial and accounting systems, hundreds of colleges and universities now outsource (or net-source) their billing and payment processing to third party, web services, SOA based providers. In a complex process of secured, PCI compliant and FERPA regulated financial transaction routing, third party providers provide standardized, loosely coupled services to schools and their consumers, students and parents alike.
Level:	Summary
Stakeholders & Interest:	SIS and Financial System Vendors
	Educational Institutions e.g. HEI, Schools, FE, Community Colleges

Scenario Title:	*Loan Processing Scenario – Financial Aid Delivery*
Scenario Local ID:	SOA-07
Brief Description:	Every year, hundreds of thousands of students and families apply for and receive in excess of $80 billion dollars of federal and private student loans for higher education. In a complex and heavily regulated process, loan applications, credit reports, federal tax forms, student aid applications, loan guarantee services, all must be processed in a timely, accurate and secure system. With emerging standards, common applications, and systems moving to a Service Oriented Architecture approach (at the Federal Department of Education), new services offerings will become available to schools, families, lenders, guarantors, and third party agencies.
Level:	Summary
Stakeholders & Interest:	SIS and Financial System Vendors
	Educational Institutions e.g. HEI, Schools, FE, Community Colleges

Scenario Title:	*The Business of Education – Managing Accounts Payable*
Scenario Local ID:	SOA-08
Brief Description:	A major consortium of colleges has for years been studying and implementing different models of providing shared services solutions to drive down operational costs. With the advancement of SOA technologies, these schools are now able to explore decoupling core business processes. They are exploring creating a centralized AP processing center that will provide common, standards based processing in a scalable, replicable, and reliable services model. Through cloud technologies and enabling SOA infrastructure, built upon innovative, cost effective, and quality service standards, these schools strive to manage their costs and improve operational efficiencies.
Level:	Summary
Stakeholders & Interest:	SIS and Financial System Vendors
	Educational Institutions e.g. HEI, Schools, FE, Community Colleges

Scenario Title:	*A Community of Learners – Collaborative Education Initiatives*
Scenario Local ID:	SOA-09
Brief Description:	As our campus boundaries continue to blur, the virtual higher education learning enterprise emerges. Students at hundreds of institutions are now able, often required, to work across institutional boundaries, conducting research at other campus libraries, working with students at other institutions on joint scholarly projects, conducting social and scientific research collaboratively. Likewise in the school sector, students can work on joint projects with students at other schools, and access external resources; this is of particular interest to small schools in remote locations. A complex array of services to manage the directory access systems, knowledge object repositories and collaborative learning environments, while ensuring security, privacy and protection, all rely on SOA and technologies. When a student in New Mexico signs into a research facility in New England, and “checks out” digital resources, then uses them with a team of peers across the country, complex services solutions enable an effective, reliable, and secure infrastructure.
Level:	Summary
Stakeholders & Interest:	LMS, LCMS and Library System Vendors
	Educational Institutions e.g. HEI, Schools, FE, Community Colleges

Scenario Title:	*Academic Records and the Virtual Student – Cross Institutional Records*
Scenario Local ID:	SOA-10
Brief Description:	It has increasingly become the norm for students to attend approximately 2.4 post secondary institutions as they pursue their undergraduate education. No longer do students simply enroll in one school and complete their program of study in four years. Managing the academic records, including course transcripts from high schools and colleges, becomes a new challenge in the digital world. Requirements for processing academic records, maintaining academic portfolios, ensuring transferability of records and enabling increasingly rigorous outcomes reporting and performance management assessments create new demands that can only be met with new SOA based services, systems and infrastructures. Standards must be developed to ensure interoperability.
Level:	Summary
Stakeholders & Interest:	SIS, Transcript and ePortfolio System Vendors
	Educational Institutions e.g. HEI, Schools, FE, Community Colleges

About This Document

Title	Adoption of Service Oriented Architecture (SOA) for Enterprise Systems in Education: Recommended Practices
Co-chairs	Viswanath Srikanth (IBM) and Austin Laird (Oracle)
Editor	Colin Smythe (IMS GLC)
Version	v1.0
Version Date	14 September 2009
Status	**Final Release**
Revision Information	Original White Paper
Purpose	This document is for public release. Please provide feedback to the Project Group via IMS GLC SOA Forum at http://www.imsglobal.org/community/forum/categories.cfm?catid=54
Document Location	http://www.imsglobal.org/soa/index.html

List of Authors

The following individuals contributed to the authoring of this document:

Kerry Blinco	DEEWR (Australia)
Travis Grisby	IBM (USA)
Austin Laird	Oracle (USA)
Owen O’Neill	DEEWR (Australia)
Viswanath Srikanth	IBM (USA)
Colin Smythe	IMS GLC (UK)

Revision History

Version No.	Release Date	Comments
Final Release v1.0	14 September 2009	This document is for public use. Please provide feedback via the IMS GLC SOA Forum at http://www.imsglobal.org/community/forum/categories.cfm?catid=54

Index

IPR and Distribution Notices

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the specification set forth in this document, and to provide supporting documentation.

IMS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on IMS's procedures with respect to rights in IMS specifications can be found at the IMS Intellectual Property Rights web page: http://www.imsglobal.org/ipr/imsipr_policyFinal.pdf.

Copyright © 2009 IMS Global Learning Consortium. All Rights Reserved.

If you wish to copy or distribute this document, you must complete a valid Registered User license registration with IMS and receive an email from IMS granting the license to distribute the specification. To register, follow the instructions on the IMS website: http://www.imsglobal.org/specificationdownload.cfm.

This document may be copied and furnished to others by Registered Users who have registered on the IMS website provided that the above copyright notice and this paragraph are included on all such copies. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to IMS, except as needed for the purpose of developing IMS specifications, under the auspices of a chartered IMS project group.

Use of this specification to develop products or services is governed by the license with IMS found on the IMS website: http://www.imsglobal.org/license.html.

The limited permissions granted above are perpetual and will not be revoked by IMS or its successors or assigns.

THIS SPECIFICATION IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NONINFRINGEMENT IS EXPRESSLY DISCLAIMED. ANY USE OF THIS SPECIFICATION SHALL BE MADE ENTIRELY AT THE IMPLEMENTER'S OWN RISK, AND NEITHER THE CONSORTIUM, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY IMPLEMENTER OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER, DIRECTLY OR INDIRECTLY, ARISING FROM THE USE OF THIS SPECIFICATION.

IMS Global Learning Consortium, Inc. (“IMS GLC”) is publishing the information contained in this Adoption of Service Oriented Architecture for Enterprise Systems in Education: Recommended Practices *(“Specification”) for purposes of scientific, experimental, and scholarly collaboration only.*

IMS GLC makes no warranty or representation regarding the accuracy or completeness of the Specification.

This material is provided on an “As Is” and “As Available” basis.

The Specification is at all times subject to change and revision without notice.

It is your sole responsibility to evaluate the usefulness, accuracy, and completeness of the Specification as it relates to you.

IMS GLC would appreciate receiving your comments and suggestions.

Please contact IMS GLC through our website at http://www.imsglobal.org

Please refer to Document Name: Adoption of Service Oriented Architecture for Enterprise Systems in Education: Recommended Practices

Date: 14 September 2009