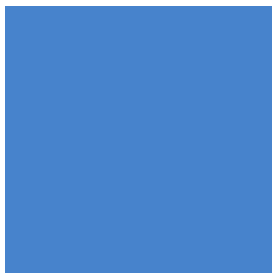




October 2005

# What Is Enterprise Service-Oriented Architecture?

Whitepaper



## Table of Contents

1. INTRODUCTION .....	1
2. BACKGROUND .....	3
3. ENTERPRISE SOA BENEFITS AND TECHNOLOGIES.....	4
4. AN ENTERPRISE SOA FRAMEWORK .....	6
5. ALIGNING IT WITH BUSINESS.....	7
6. CONCLUSION.....	8

# 1. Introduction

Service-Oriented Architecture (SOA) has received wide coverage in recent years with the proliferation of web services<sup>1</sup>. The purpose of this paper is to define both the meaning and the benefits of SOA, specifically in its enterprise usage (Enterprise Service-Oriented Architecture or ESOA) and to expand on what it means to implement an SOA in a given organisation.

## 2. Background

In recent years, the term Service-Oriented Architecture has taken on an amorphous quality, with industry pundits overloading the term to encompass many concepts. Two examples of this are:

1. **SOA as a philosophy:** "SOA is an architectural style whose goal is to achieve loose coupling among interacting software agents." – 'O'REILLY *webservices.xml.com*'
2. **As Enterprise SOA:** "A Service-oriented architecture is a style of design that guides all aspects of creating and using business services throughout their lifecycle (from conception to retirement)" – '*Understanding SOA with Web Services*' Newcomer & Lomow

The philosophical approach to SOA is too broad to be useful; therefore, we will approach SOA as Enterprise SOA giving the term a context in usage for this discussion. In examining ESOA we will cover some of the technological and practical aspects of this particular architecture discipline and then review SOA **as a means of aligning IT with business.**

---

<sup>1</sup> A common misnomer is to identify SOA with web services exclusively. In truth SOA existed long before web services, however, web services standards have helped popularise SOA.

### 3. Enterprise SOA Benefits and Technologies

Firstly, let us examine some of the technical benefits of implementing SOA as an Enterprise Architecture initiative in accordance with the Newcomer and Lomow definition:

Concept	Description	Business Benefit
<b>Promotion of loose-coupling</b>	Loosely coupled services, have no knowledge of what application will be consuming them, the advantage here is that a piece of logic can be wrapped up in a generic interface and be accessible by any number of consuming applications and hence reusable.	Business logic is not tied up in silo applications but is instead exposed and reusable to the rest of the organisation.
<b>Implementation of abstraction layers</b>	Implementing an open (or at least common) standard by providing an abstracted services layer.	Platform interoperability across heterogeneous systems.
<b>Supported advertisement, discovery and reuse.</b>	Services are only as reusable as they are visible, by promoting the visibility of such services, technical resources (at all levels) are better informed about the enterprise's assets.	Increased business agility and reduced replication across the architecture.
<b>Support for infrastructure concerns.</b>	Combining the information and business logic of multiple systems into a single interface, allows for diverse and redundant systems to be addressed through a common interface.	Providing effective controls for managing service levels.
<b>The ability to categorise and describe services</b>	Services have meta data associated with them in order to specify security, constraints and versioning concerns.	This assists with the governance of organisational assets.

The benefits listed above are technology driven, that is, the technology (and architecture) itself facilitates the benefits to the business in the forms of reuse, agility, visibility, transparency and governance.

The specifics of the technology stack have taken on a life of their own. As outlined, web services have fuelled the SOA adoption. In the past, enterprise architectures have typically dealt with the concern of standards as an implementation detail rather than entrenching it within the architecture. SOA (and especially ESOA) endeavors to do the opposite, that is, by incorporating standards as core to the initiative we have

the benefit of a baseline for tools, frameworks and extensibility. The following is a listing of some SOA concepts and their defacto standards <sup>2</sup>:

SOA Concept	Description	Technologies
<b>Contract</b>	Defines the service's interface capabilities and how it is invoked by the caller.	WSDL, WS-Policy
<b>Policy</b>	Define the (sometimes complex) rules of engagement when seeking to establish connectivity with a service.	WS-Policy
<b>Meta data</b>	(Also known as semantics) provide self describing data about the services, this assists in the ability to discover your particular service.	UDDI
<b>Security</b>	Authentication, privacy, non-repudiation etc.	SAML, WS-Security
<b>Orchestration</b>	(Sometimes called choreography). Used in linking one process to another	WS-BPEL, WS-CDL

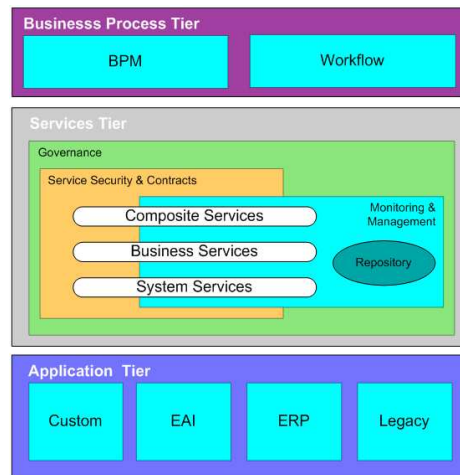
These standards are in themselves open, evolving and for the most part have been accepted by vendors for implementation.

---

<sup>2</sup> These are just some of the most common standards, several other standards exist for very niche purposes, for example; guaranteed messaging would have WS-Reliable Messaging as a standard.

## 4. An Enterprise SOA Framework

SOA usually manifests itself as a particular component within an organisation, typically referred to as a **Services Tier** (also known as a Services layer, fabric layer and Enterprise Services Bus) in enterprise software architecture. The following diagram is typical of how a services layer is conceptually placed within software architecture:



The Application Tier is usually composed of a mixed bag of systems (CRM, ERP, EAI, custom build, legacy etc). These systems play host to a wealth of intellectual property developed over many years.

The Business Process Tier is responsible for orchestration/execution of business functions (BPM).

The Services Tier contains governance, security and management functions for services that can be broken up into the following three categories (sub-layers):

- **Composite:** wraps the individual business processes (Create New account)
- **Business:** wraps business processes (Perform Credit Check)
- **System:** wraps technical processes (Get user details from Database)

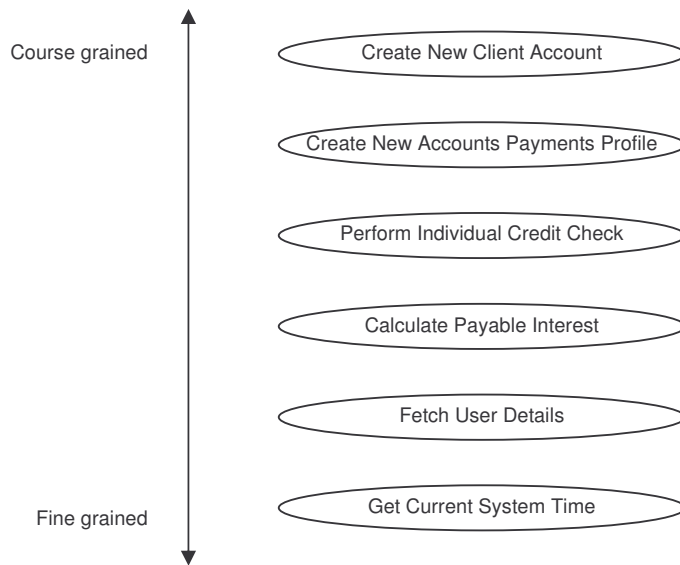
Despite the clear delineation between the Service tier with the Application tier and the Business Process tier, it is important to note that, **SOA does not equal the services tier** (even though it might be considered the heart of the SOA). Instead it should be considered as a way of leveraging and just as importantly managing work done in the other tiers.

## 5. Aligning IT with Business

In recent years the BPM space has taken on a life of its own, where 'composite applications' have advanced the idea of reuse directly. We can see the *next phase* in this reuse philosophy where BPM composite applications are themselves classed as **reusable** components in an ESOA.

A reusable service, can be placed somewhere in the spectrum of **fine-grained** (technically-oriented) to **course-grained** (business-oriented) components<sup>3</sup>. A good rule of thumb when determining the 'granularity' of a service is to consider, who its consumers might be. For instance, a fine-grained service is typically consumed as part of an automated process, whereas a course-grained service should be directly identifiable to someone who actually works for the business.

The following diagram illustrates the scale of granularity in some sample services:



Note that the course-grained services are made up of finer-grained components. This is the manner in which composite applications can become reusable.

For instance, a particular BPM model could have been built to "Create (a) New Accounts Payment Profile" this can then be leveraged as a single step in another composite application, "Create New Client Accounts".

<sup>3</sup> Typically the more course-grained a service is the more value it presents to the organisation.

It is this delineation in SOA services, that allows an SOA to better **align** IT with the business and as a consequence we find the roles of individuals within an organisation, can change to some extent:

Technologists – *Developers, System Analysts, Architects*

- Design components with the forethought of future reuse.
- Leverage existing business logic 'siloes' applications.
- Use a common (standard) interface for integration.
- Make continuous improvement of implementation code behind the service interface.
- Provide the ability to gracefully incorporate new capability over time.

Business personnel – *Business Analysts, Stakeholders, IT Managers, CIOs*

- Have visibility of business assets across the enterprise.
- Have insight into operational behavior which will in turn assist in managing service levels.
- Better understand the cost tradeoffs associated with investments in resources (such as personnel, processes and systems)

U.K. based IT analyst firm **Macehiter Ward-Dutton** had this to say about the alignment of IT to business through the use of SOA:

*"It is this "convergence" of IT domains, driven by common use of Web services technology - rather than just the simple availability of Web services - which is the real key to understanding the potential value of SOA. SOA "done right" is about using service-orientation to pull multiple different IT design, delivery, operations and change management disciplines together within an IT organization to serve business needs." – itworld.com*

## 6. Conclusion

Enterprise Service-Oriented Architecture offers some significant advantages in the overall visibility and governance of code. In adopting a 'service' based approach, true business benefits are directly applicable as not seen with other architecture disciplines.