

# SOA Primer

## Service Oriented Architecture is pervasive

SOA is becoming ubiquitous as institutions strive to align their systems with their business processes to achieve improved agility and to strengthen their competitive edge. International corporations are embracing SOA concepts and major vendors such as IBM, BEA, Microsoft, Information Builders, Software A.G., Oracle, HP and SAP are on the acquisition trail, gearing up to satisfy the burgeoning market demand for SOA (and in particular, SOA using Web Services) solutions.

## Attributes of SOA

The key tenets of an SOA are that

- Systems are regarded as "Services"
- Services should be able to demand and receive service from other Services within the SOA
- A service requester should not need to know anything about the internal implementation details or platform of the service it is requesting (loose coupling)
- All services should be able to communicate with each other via a common, platform-neutral, hardware-neutral, middleware-neutral message interchange

The fastest growing form of SOA uses "Web Services" as the means of exchanging information between requesters and providers of Services. Web Services is a standard agreed between all the major platform and software vendors to ensure services interoperability.

Web Services is based on the use of:

- XML for inter-service messages
- XML messages enveloped in a SOAP (Simple Object Application Protocol) wrapper
- services definitions being written in a machine readable form – WSDL (Web Services Definition Language)
- the location of services being available for look up in a central registry, at run time

## SOA Advantages

Adoption of a Web Services/Service Oriented Architecture approach to enterprise systems yields several advantages, including:

### *Efficiency of development*

Each Web Service is self contained and can be designed and implemented independently of others.

### *Service reuse*

The high degree of interoperability and platform agnosticism intrinsic in the SOA/Web Services approach encourages the reuse of existing Web Services rather than the creation of new services i.e. inhibits reinvention of the wheel.

### *Reduced maintenance costs*

The fact that each Web Service is discrete and self contained inherently simplifies system maintenance.

### *Incremental adoption*

Modularity and loose coupling mean that Web Services can be developed in a series of small steps - functionality of existing systems can be exposed as Web Services incrementally, as and when required. New Web Services can be created and deployed as and when required.

### *Smooth evolution*

Each Web Service is accessed via a standardized interface which

requires no knowledge of the specifics of its implementation. This means that a Web Service can be replaced or modified "behind" its external interface without requiring changes to other services from which it requests service or to which it provides service.

### *Increased business agility*

Loose coupling and well defined standards based interfaces mean that it is straightforward to reflect changes in a business's environment by changing existing Web Services or adding new ones, thereby encouraging business agility.

### *Increased return from existing IT systems*

Re-engineering legacy systems so as to expose their functionality as Web Services prolongs their life and increases their utility and the chance that the functionality will be reused

### *Reduced integration costs*

Loosely coupled Web Services together with well defined, platform-neutral standards-based interfaces make integration of new functionality via addition of new Web Services straightforward, lowering integration costs.

### *Reduced vendor lock in and switching costs*

SOA future proofs an organization's IT investment by basing its architecture on platform neutral, middleware agnostic Web Services concepts. This radically decreases application, middleware and platform vendor lock in compared to conventional EAI (Enterprise Application Integration) or point-to-point connection approaches.