



TECHNICAL WHITE PAPER

Enabling Service-Oriented Architecture:

Aligning the enterprise for agility and competitive advantage

December 2004



Anexinet Corporation
One International Plaza, Suite 140
Philadelphia, PA 19113 USA

Phone: 610-595-1993
Fax: 610-595-2252
ANEXINET.COM

CONTENTS

Executive Summary	3
The quest for enterprise computing standards	4
XML, SOAP and the birth of SOA	4
How SOA solves enterprise computing issues	5
XML Information Modeling foundational for SOA-based enterprises	5
SOA Fabric facilitates enterprise sharing and reuse of Web services across systems, lines of business and geographies	6
Data security benefits	6
Web services management	6
Business process and change management	7
The importance of loose coupling and a new “component services” solution to achieve it	8
Required expertise for SOA implementation	9
Conclusion	10

EXECUTIVE SUMMARY

Web services change the way we can think about enterprise computing. Old issues of standardization and systems interoperability that have long suppressed returns on corporate IT investments can now be resolved. New issues of data security, information access by multiple constituencies on a global basis, responsiveness to evolving business requirements, and containment of IT costs can now be very effectively addressed.

Anexinet Corporation, a systems integrator at the forefront of engineering Service-Oriented Architecture (SOA) using Web services, occupies a unique position from which to comment. From this expert perspective, the business needs and benefits of SOA and insights for SOA enablement of enterprise computing infrastructure are explained in terms that executives with P/L responsibilities can understand.

The quest for enterprise computing standards

Confronted with ever-changing requirements, the focus of corporate IT managers across business and industry has been to make enterprise computing more efficient, agile and less costly to support by aligning capabilities with accepted standards.

Early “solutions” resulted in combinations of platform, vendor and enterprise-specific standards that remained largely proprietary to the implementing enterprise. Enforcement of these attempted standardizations within an enterprise and across lines of business and geographies led to creation of shared services organizations whose effectiveness depended on diligent management oversight.

Ultimately, such solutions for enterprise standards and enforcement by one or more shared services organizations failed to keep pace with constantly evolving business requirements. Moreover, the largely proprietary standards locked an enterprise's allegiance to specific computing platforms and supporting vendors.

When enterprise systems integration emerged as a competitive differentiator for business productivity and a catalyst for growth, IT departments turned en masse to specific products and vendors to integrate systems into more efficient, streamlined business processes. Subsequently, use of the Internet as a viable means to interact with customers and vendors led to devising front-end Web interfaces to access enterprise data for e-commerce.

Systems integration and Web enablement coupled with continuing demands from business units for greater computing functionality produced unwieldy information exchange models, insufficient data security and more needs for standardization. Java and J2EE (Java 2 Platform, Enterprise Edition) provided technologies to address some of the problems and insulate enterprises from otherwise non-conforming vendor systems and protocols. But issues of enterprise computing capabilities to manage evolving business requirements remained.

XML, SOAP and the birth of SOA

Enter the then new XML (eXtensible Markup Language) and its associated schema formats for standards enforcement. XML provided a means to describe data in different formats that could be dictated by business use. The IT industry soon recognized XML's capabilities for bringing order to the mishmash of information exchange formats that continued to plague enterprise computing models.

Using XML, Microsoft (and others) introduced SOAP (Simple Object Access Protocol) and submitted this information exchange format for industry standardization to the nascent Internet standards organizations of the time. A compelling aspect of SOAP was its design to ride atop the ubiquitous HTTP among other protocols. This capability led to a quick uptake by the IT industry, particularly application server providers and software development tool manufacturers. Everyone agreed to the viability of data standards and exchange mechanisms.

Before long, software architects began sending their submissions to standards organizations. Nearly all of the problems surrounding enterprise computing models were being addressed. SOAP interfaces became integral to new systems development and add-ons for legacy systems. The concept of Service-Oriented Architecture (SOA) was born. The IT industry trended to solving enterprise computing's information exchange problems-involving employees, customers, suppliers and partners-with technologies that aligned with overarching standards tolerant of dynamic business requirements.

How SOA solves enterprise computing issues

Enterprises are saddled with disparate systems or “point solutions” that lead to working with a mix of legacy and component-based platforms, multiple databases and silos of information. Such eclectic computing infrastructures are not only expensive to deploy and maintain, but create obstacles to standardization. SOA enables enterprises to effectively leverage their prior investments in software to perform specific business functions while reducing technology and operational costs.

SOA is vendor agnostic and solves issues of systems interoperability. It integrates and automates repeatable business processes according to well-defined standards and enables processes to be triggered by specific business events. It helps to eliminate duplicate data entry and other manual processes that inhibit employee productivity, cause data inaccuracies and slow data searches. It improves data integrity, security, tracking and reporting. It provides the means to implement standards and drive process improvements across platforms, lines of business, geographies and control boundaries.

XML Information Modeling foundational for SOA-based enterprises

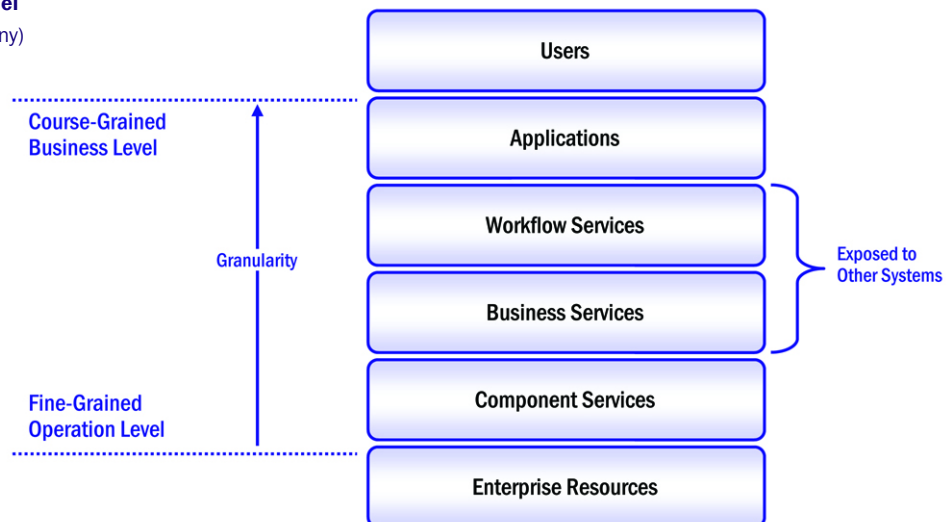
A prerequisite to SOA is an effective information model wherein all data is represented in XML. This requires enterprise data to be aggregated from a variety of sources such as data bases, data marts, data warehouses and networked operational systems supporting specific business processes.

To review, enterprises had relied on complex and inflexible ETL (Extract, Transform and Load) processes to move and synchronize data between systems and stores and across applications’ boundaries. SOA with XML-represented data (Figure 1) views and handles data differently. When data is “right-sized” for Web services-sized to the right granularity and expressed in the correct XML format-it can be sent throughout the enterprise as needed for near real-time computing regardless of application.

Once enterprise data is expressed in standard formats as dictated by XML schema, it can be used in automated workflows to support multiple business processes. When necessary, data can be transformed to other formats via XML standards to more easily support changes in business processes.

Figure 1: SOA Information Model

(Source: The Middleware Company)



SOA Fabric facilitates enterprise sharing and reuse of Web services across systems, lines of business and geographies

An SOA Fabric is the unifying software structure for security, provisioning and management of Web services applications. It provides a unified control layer for software architects to integrate, share, scale and reuse applications, define and enforce consistent operational and security policies across an enterprise’s distributed architecture, and quickly implement required changes to evolving business processes.

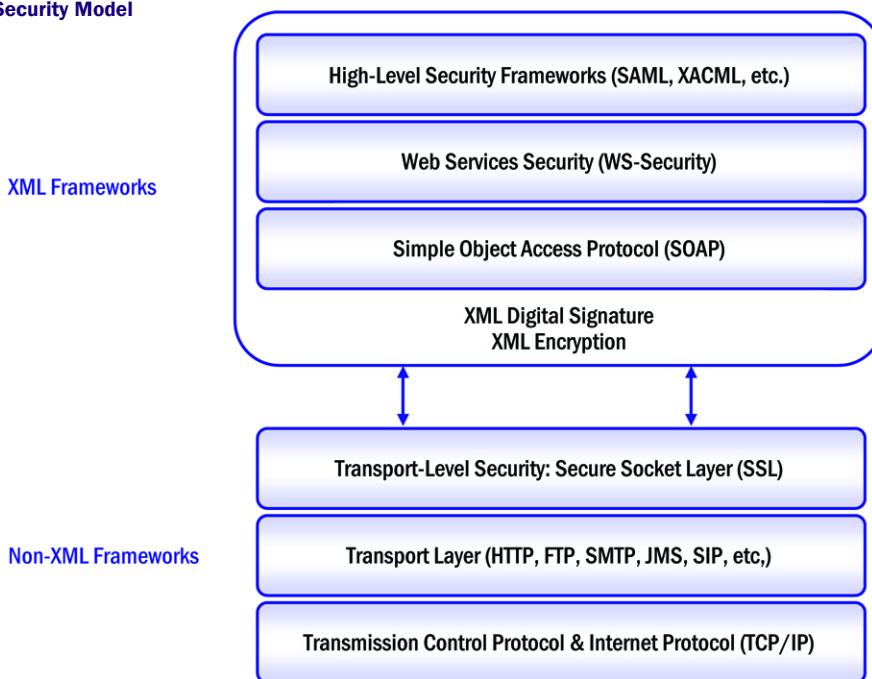
Data security benefits

No IT organization sets out to implement Web services in an insecure manner, yet current computing infrastructures in most enterprises do not support secure XML messaging (Figure 2). Such security entails safeguarding XML messages with respect to confidentiality and transmission integrity according to approved standards that only in 2004 have begun to mature for widespread implementation.

Enforcement of user identity and authorization standards becomes paramount as Web services transit the enterprise to open efficient business channels with customers, suppliers and partners. An SOA Fabric not only provides a method to achieve security enforcement objectives, but simplifies system use as well by isolating users from the complexities of policy compliance.

Figure 2: Web Services Security Model

(Source: Netegrity)



Web services management

No IT organization sets out to implement Web services in an insecure manner, yet current computing infrastructures in most enterprises do not support secure XML messaging (Figure 2). Such security entails safeguarding XML messages with respect to confidentiality and transmission integrity according to approved standards that only in 2004 have begun to mature for widespread implementation.

Enforcement of user identity and authorization standards becomes paramount as Web services transit the enterprise to open efficient business channels with customers, suppliers and partners. An SOA Fabric not only provides a method to achieve security enforcement objectives, but simplifies system use as well by isolating users from the complexities of policy compliance.

- **Recognizing contracted levels of service and response times**
- **Managing escalations and service requirements**
- **Correctly prioritizing response scheduling**
- **Distributing requests with any associated attachments and inter-departmental action items to response teams**
- **Monitoring actions taken**
- **Tracking charges not covered by contracts**

A fully integrated architecture for the example service organization will also include schema and enterprise rules for its SFA (Sales Force Automation), SCM (Supply Chain Management), business analytics and other applications deployed to drive revenue and profit.

Business process and change management

SOA delivers exceptional value by orchestrating Web services into efficient business processes and workflows. More precisely, the adoption of standards, easier engineering of a distributed enterprise architecture, and capabilities to quickly update business processes institute an environment of optimal configurability and agility. (Figure 3).

A properly implemented SOA allows non-technical business managers to configure business processes. The dynamic of keeping enterprise computing aligned with market conditions and corporate goals shifts from IT to sales, marketing and operations. With less dependence on IT to respond to evolving business requirements, an enterprise is more capable to seize opportunities and leverage its knowledge workers for competitive advantage.

Figure 3: Shift of Business Process Control from IT to Non-Technical Business Managers

(Source: Anexinet)



The importance of loose coupling and a new “component services” solution to achieve it

Through its application of Web services, SOA enables linking of software processes, even if they use otherwise incompatible technologies. “Cohesion” characterizes the degree to which a process performs a meaningful function. “Coupling” characterizes the degree of mutual interdependence between software processes. Loose coupling allows processes to be joined together on demand or disassembled into their functional components (after first establishing a semantic framework to ensure messages retain their meaning).

In object-oriented programming, software architects strive for a mix of highly cohesive but loosely coupled software processes. In a SOA, the semantic framework is engineered using WSDL (Web Services Description Language) contracts. Cohesion is orchestrated with BPEL (Business Process Execution Language). The level of cohesion that can be achieved depends on the vision and preparation of the application architecture. Loose coupling however is more complicated because of the interdependence of different technologies.

The component-level view of a business process is one of synchronous and asynchronous interaction with various EIS (Executive Information Systems) and databases—all employing various high and low level protocols and transaction heuristics. Often, a great deal of legacy code is involved. Many times with J2EE, there is native code running “under the hood”. This is the “programming in the small” or “coding” arena for software architects. Programming details are explicit, step-by-step instructions for performing relatively small-scale tasks.

The orchestration or “structuring” of software processes is termed “programming in the large.” This entails aggregating processes at the platform level with tools from platform vendors and other tools that link processes in platform-specific ways. Web logic integration uses interactions with JMS (Java Message Service) queues and topics and sometimes interactions with externally defined Web services.

SOA has led integration and platform vendors to be more sensitive to standards and accommodating software tools, which businesses are now requesting to engender reuse of software components and optimize their investments in enterprise computing. One solution, proposed by BEA Systems, Inc. and IBM Corporation, is an extension to BPEL called BPELJ (Business Process Execution Language for Java).

BPELJ seeks to unify programming “in the small” and “in the large” in a single process definition. It provides for insertion of Java snippets to allow activity and variable interaction with native J2EE constructs in a virtually seamless manner. It also allows the use of Java in BPEL expressions to support loop conditions and variants, switch conditionals, messaging preparation and business function logic.

While a step in the right direction, BPELJ overlooks a primary goal of loose coupling. The authors propose a process attribute that would allow code snippets to be served by any back-end service and language for which a preprocessor has been installed, analogous to JSP (Java Server Pages) technology. In this way, unification of large and small programming is achieved. The problem however is that this approach still ties the enterprise to a platform and impairs component reusability. For example, one can envision process definitions using Java or C++ snippets with the business running them then being tied to J2EE or .NET, respectively.

A new and proven method pioneered by Anexinet to achieve loose coupling is by surfacing low-level services, called “component services,” and integrating them into the SOA Fabric. Software architects are then able to align underlying systems, data stores and EIS with the SOA and thereby improve component availability to

business processes without locking into a platform dependency. Before the introduction of component services, the concept of loose coupling in this way could not have been considered due to the implementation time and costs using less sophisticated methodologies.

Required expertise for SOA implementation

Not all SOA solutions are as advanced as described herein. Few software developers and systems integrators committed to SOA concepts as early as Anexinet, and therefore trail in areas such as integrating point solutions into an enterprise process, implementation planning and execution expertise, XML information modeling, and working partnerships with leading XML security and SOA Fabric developers.

Most enterprise computing infrastructures encompass a mix of legacy and component-based platforms supporting discrete business processes. Interoperability generally requires a comprehensive understanding of the existing business environment; expertise with multiple technologies including J2EE, .NET, UNIX and LINUX; and experience with integrating and repurposing business systems.

Security and preservation of critical systems performance during implementation are major areas of concern. Ideally, a dedicated development environment is desirable for project teams to fully evaluate existing applications and to develop, test and implement SOA without disrupting ongoing operations.

Conclusion

SOA delivers the elusive goals of enterprise computing: vendor-agnostic systems integration, standard protocols, data security, reusable software components, and methods to quickly drive changes in evolving business processes across platforms, lines of business, and geographies. Once implemented, control of business processes shifts from IT staff to sales, marketing and operations managers. Changes can be implemented at the speed of market requirements and business opportunities.

An SOA Fabric provides the unifying software structure to integrate, share, scale and reuse applications, and define and enforce consistent operational and security policies across an enterprise's distributed architecture. Loose coupling with component services improves component availability to business processes without locking into a platform dependency. This frees an enterprise from intractable relationships that in the past led to project creep and escalated software implementation and maintenance costs.

SOA utilizes existing legacy and component-based applications, only more effectively. Return on investment is realized from multiple sources:

- **Agility and competitive advantages of enterprise computing to adapt to change**
- **Faster information management and access, increasing productivity of knowledge workers**
- **higher return on investment from prior technology investments**
- **Elimination of multiple databases, duplicate data entry and other manual procedures responsible for business delays and data inaccuracies**
- **Improved operational cost containment from new business efficiencies**
- **Reductions in IT support requirements and associated costs**

SOA implementation generally requires a comprehensive understanding of an existing business environment, expertise with multiple technologies, and experience with integrating and repurposing business systems. Security and preservation of critical systems performance during implementation are major areas of concern. Ideally, a dedicated development environment is desirable for project teams to plan and execute implementation without disrupting ongoing operations.



Anexinet Corporation
One International Plaza, Suite 140
Philadelphia, PA 19113 USA
Phone: 610-595-1993
Fax: 610-595-2252

S.A.F.E-T² and Near-Site Development Centers are trademarks of Anexinet Corporation. All other products and trademarks referred to are property of their respective owners.

ANEXINET.COM

Anexinet, a premier solutions integrator applies leading-edge technology to complex business challenges to simplify business processes, strengthen IT organizations, and provide competitive advantage. Anexinet offers application development, integration, business intelligence, technology infrastructure and enterprise program management (EPM) services to mid-size and Fortune 1000 companies, associations, and government agencies.

Anexinet's experienced consultants ensure the success of clients' solutions through the company's Program Management Office (PMO), a pragmatic approach that combines industry best practices experience with the company's innovative S.A.F.E-T² software delivery model. S.A.F.E-T² is a project execution strategy and trademark approach based on the Project Management Institute (PMI) methodology and Incremental Life Cycle Development standards.

Anexinet Near-Site™ Development Centers enable project teams to design, develop and test client solutions within minutes of a client's location, and without disrupting current systems and organization. Headquartered in Philadelphia, Pa., Anexinet has regional offices throughout the Delaware Valley and metropolitan New York City and Washington, D.C. areas.