Service Oriented Architecture for Distributed Application Architecture

Introduction

The term "Service Oriented Architecture" (SOA) refers to a style of distributed application architecture in which key business applications and data are decomposed into discrete components, or "services," that have particular relevance for various business processes. The services can then be used as building blocks to create new "composite" applications that model new or more complex business processes. An SOA can also be compatible with the existing portfolio of business applications. For example, business process-related segments of legacy applications can be exposed as services that can be accessed by client systems or other applications/services.

Service-oriented applications have specific characteristics that distinguish them from traditional distributed applications. The services are typically loosely coupled and interact with each other as peers via message-based communications over the network. "Loose coupling" means that the requester of a service doesn’t need to be aware of how the service is implemented — the basic requirement is that the service requester and the service provider can exchange and interpret messages in a common format. Each service specifies its functionality and quality of service (QoS) characteristics using a common, machine-readable format. This enables dynamic discovery of available services and provides the basis for interoperability in a heterogeneous computing environment. For widest applicability, services must be supported by access and security policies to allow operation across organizational boundaries and administrative domains.

There are many potential benefits that an enterprise can derive by adopting an SOA style of application infrastructure. These include:

- **Backward compatibility with existing applications and data:** With SOA, a legacy application can be offered as a single service, possibly incorporating multiple steps in a business process (coarse granularity). Alternatively, the application can be decomposed into fine-grained services that map to individual steps in a process. With either approach, most of the previous investment in applications and systems can be preserved.

- **Extended interoperability:** As existing applications are provided with service interfaces, both data and application functionality become far more accessible to other service-oriented applications and client systems. SOA can make diverse applications and data appear to the user as an integrated service. Users no longer have to log into multiple systems, search for relevant data, and integrate the results manually. In an ultimate SOA scenario, the user could access all the application resources on the network through a single portal.

- **Business agility and alignment of IT with business processes:** Creating application services that mirror steps in various business processes helps to bring application resources into better alignment with business practices. As an enterprise’s implementation of SOA matures, it becomes possible to support rapid changes in business processes with new applications created by rearranging existing service components. Eventually, tools may become available that will allow business people to create a model of a new business process, which can then be automatically compiled into a new service-oriented application, without necessarily involving application programmers.

- **Increased productivity and cost effectiveness:** Application programmers gain the benefit of code reuse whenever they create a new service based on legacy applications or include an existing service within a new composite application. This allows enterprise programmers to leverage their own past work, plus the services offered by SOA-enabled enterprise application software packages (such as ERP applications). Other sources of services available to enterprise programmers may include business partners and providers of SOA-compliant software-as-a-service.

Because of these significant benefits, many enterprises are in the process of adopting an SOA application environment. According to Gartner Research, SOA will be a prevailing software engineering practice by 2008, with more than 65 percent of large enterprises basing at least 35 percent of their applications on SOA by 2010.
**Elements of an SOA**

With SOA, the service is an abstracted, logical view of how a computer program performs the operations that mirror a business process. As a result, there is a strong relationship between the business process flow chart and the processing flow chart of a service-oriented application. Figure 1 provides a very simple example of a business process model and the corresponding services.

Orchestration engines can be used to allow developers to build new composite services from previously defined services. These engines can offer graphical user interfaces to business process flow chart models that allow new business processes to be defined and linked to a new composition of services. The new composite service results from the re-arrangement or "orchestration" of more basic services components.

Figure 1. Relationship between business processes and services

1. The service provider registers its service with the service broker or service registry. Services interact with each other and with the service broker using message-oriented communications. A service is described by Meta data that provides all the information needed to invoke the service. The internal structure of an agent providing the service is essentially hidden behind the service interface. Therefore, any software component or application can become a service if it is given a formal SOA service description and supports the message-handling middleware used by the SOA. Messages are sent in a platform-neutral, standardized format (e.g., using XML) and delivered through the service interfaces. The services environment, including the service interfaces, brokering, and messaging system, is sometimes described as an enterprise services bus (ESB). The ESB performs all the operations necessary to virtualize services. Services are virtualized in the sense that the service requester is unaware of the service provider's physical location, programming language, runtime environment, hardware platform, or network address.

2. The service requester discovers the service by exchanging messages with the service broker. In addition to providing a registry of services, the service broker may provide additional functions such as workload management, load balancing among several providers of the same service, or enforcing QoS policies for different SOA applications that may contend for the same services.

3. When the service-oriented program is run, the service requester communicates directly with the service provider via the messaging system to invoke the service and receive results.

Another key aspect of SOA is a suite of management tools that have been designed specifically for the various aspects of service-level management of the distributed application environment. Service-level management functions include monitoring, optimizing and controlling services, as well as ensuring that SLAs and security policies are enforced.

Figure 2 shows the basic information flows that occur when one service (the requester) invokes a second service (the provider).
Web Services and SOA

Although there are other technologies on which an SOA may be based (e.g., DCOM or CORBA), web services are expected to be the dominant implementation of SOA over the next few years. Web services offer the advantage of being based on industry standards that offer the promise of multi-vendor interoperability and long-term investment protection. According to Gartner, 75 percent of new SOA projects will be based on web services through 2008. The expected dominance of web services reflects the fact that most major systems vendors and application server ISVs are moving their software products to a web services foundation.

The World Wide Web Consortium (W3C) defines a web service as a software system designed to support interoperable machine-to-machine interaction over a network. The W3C web service definition encompasses several different styles of distributed computing. Among these, SOA-style web services are those that have their interfaces described by the web services description language (WSDL) and use simple object access protocol (SOAP) messaging systems for exchange of information in a decentralized, distributed environment.

Web services standards are currently being developed by a number of industry groups, including W3C, OASIS, IETF, and DMTF. The standards described below provide the basic foundation for web services implementations of an SOA, as depicted in Figure 3.

![Figure 3. Web services foundation for SOA](image)

**Business Process Execution Language (BPEL)** is an XML-based language that may be used to describe business processes in a way that allows steps in the process to be readily linked to web services. BPEL also supports the "orchestration" of a new composite web service that mirrors a multi-step business process.

**Web Services Description Language (WSDL)** defines an XML schema for describing a web service in an XML document. A WSDL document provides an abstract definition of the network services that operate on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then are bound to a specific network protocol and message format to define a service endpoint.

**Universal Description Discovery and Integration (UDDI)** is a protocol for publishing and discovering Meta data about available web services. UDDI is based on a set of industry standards, including HTTP, XML, XML Schema, WSDL, and SOAP.

**Simple Object Access Protocol (SOAP)** is an XML-based protocol for exchanging information in a distributed environment. SOAP provides a common message format for exchanging data between clients and services. The basic unit of transmission in SOAP is a SOAP message, which consists of a mandatory SOAP envelope, an optional SOAP header, and a mandatory SOAP body. While SOAP can potentially be used in combination with a variety of transport protocols, it is generally implemented over HTTP or HTTP extensions.

**WS-Reliable Messaging (WS-RM)** adds reliability to messaging between two web services. WS-RM provides guarantees that messages are delivered in the order sent and without duplication. In the event that the message cannot be delivered, the sender is notified to allow it to seek an alternate provider of the same service.

**Web Services Security (WSS)** is a set of SOAP extensions that can be used to implement message content integrity and confidentiality. WSS supports multiple security token formats, multiple trust domains, multiple signature formats, and multiple encryption technologies incorporated in a variety of security models including PKI, Kerberos, and SSL.

**Web Services Distributed Management (WSDM)** specifies a common messaging protocol between managed resources and management consoles. Using WSDM, management applications can themselves be implemented as web services. This allows SOA application services to be managed by SOA manage-
ment services while also allowing SOA application services to monitor each other using WSDM. As an example of the latter type of interaction, a web service requester or service broker could choose a web service provider in real-time based on quality of service metrics (e.g., availability, CPU utilization, and end-to-end latency) conveyed via WSDM. This allows services to dynamically leverage redundant service resources and thereby eliminate single points of failure and other threats to SLAs.

The Web Services Interoperability Organization (WS-I) is an industry consortium promoting interoperability for multi-vendor implementations of the various web services specifications. The current membership of WS-I includes IBM, Microsoft, BEA Systems, SAP, Oracle, Fujitsu, Hewlett-Packard, Intel, Sun Microsystems, and webMethods. WS-I has established a Basic Profile V1.1 intended to foster interoperability among vendor implementations of the following web services specifications:

- Simple Object Access Protocol (SOAP) 1.1
- RFC2616: Hypertext Transfer Protocol — HTTP/1.1
- RFC2965: HTTP State Management Mechanism
- eXtensible Markup Language (XML) 1.0 (2nd Edition)
- Namespaces in XML 1.0
- XML Schema Part 1: Structures
- XML Schema Part 2: Datatypes
- Web Services Description Language (WSDL) 1.1
- UDDI Version 2.03 Data Structure Reference, dated 19 July 2002
- UDDI Version 2 XML Schema
- RFC2818: HTTP Over TLS
- RFC2246: TLS Protocol Version 1.0
- SSL Protocol Version 3.0
- RFC2459: Internet X.509 PKI Certificate & CRL Profile

Using WS-I Basic Profile, a standards-based SOA using web services, can be implemented as illustrated in Figure 4:

Standards bodies continue to work on refinements to web services in a number of areas such as transaction processing and security. For example, OASIS is working on the security assertion markup language (SAML), which is a framework for exchanging security information. Through SAML, multiple services can exchange security information, enabling a single sign-on for authenticated access to multiple services. Building on web services security standards such as WS-Security and SAML, an industry consortium, Liberty Alliance Project, is working on standards to allow identity-based access to web services.

**Adoption of SOA**

Making a complete transition from a legacy application architecture to SOA involves a number of technical and organizational challenges. Among these are:

1. bridging the gap between business process definitions and applications service definitions
2. solving the governance issues involved when composite services that span multiple administrative domains within the enterprise or the extended enterprise
3. developing a security model that allows services to span multiple security domains in a simple and effective way
4. integrating the SOA-enabled application software, ESB, and SOA infrastructure products from a variety of vendors into a cohesive service architecture

With a gradual transition to SOA, these problems do not have to be solved all at once before implementation can begin. One example of a simple initial approach to SOA is to transition to SOA-enabled versions of enterprise applications such as ERP and CRM, and to build portals that allow critical business information to be accessed in real-time or to be distributed in an event-driven fashion. A next step might be to transition proprietary enterprise application integration (EAI) solutions to application integration based on web services, setting the stage for broader integration based on the WS standards.

As these examples illustrate, probably the best overall strategy is for the enterprise to work closely with its existing system and software vendors to understand their products portfolios for supporting transition to SOA and web services standards, the extent of their SOA ecosystem partnerships, and how these elements would fit into an eventual enterprise-wide SOA framework that can derive maximum leverage from the previous investments in distributed applications.
SOA and Virtualization

Legacy data center architectures are generally based on a modular approach in which each application is assigned dedicated resources in a point of delivery (POD) module consisting of servers, storage, switches, and other network devices. The physical and logical architecture within each POD has typically evolved to suit the needs of a particular application. For example, many enterprise applications, such as ERP, have PODs that are based on three tiers of servers: web servers, application servers, and database servers.

Dedicating physical resources to each application results in PODs, whose resources are underutilized most of the time because their capacity must be scaled to accommodate peak workloads. Under utilization can be very costly. For example, if servers are only 20 percent utilized, 80 percent of server capital investment and support cost is essentially wasted. Dedicated resources also lead to very high device counts and complex infrastructures that are difficult to manage and difficult to scale to meet growing application demand.

With SOA, the application environment will become far more dynamic because new composite applications can be created and deployed with very short lead times. Dedicating a POD of resources to each coarse-grained or fine-grained SOA service would soon become completely impractical in most large IT environments because the number of services would grow quickly to exceed greatly the number of legacy applications.

As enterprises adopt SOA as the prevailing application architecture, the breadth of services will require an underlying infrastructure that has the flexibility to automatically provision and scale resources to meet user demand for services and satisfy service level agreements (SLAs) in accordance with business policies. This requires a service-oriented infrastructure (SOI) that is flexible, dynamically scalable, highly resilient, and simple to manage.

Infrastructure virtualization, including server, storage, and network virtualization, provides the best foundational architecture for an SOI that can meet the needs of SOA in a cost-effective manner. In fact, as noted earlier, one can view SOA as simply another form of virtualization. SOA virtualizes applications by providing them with technology-neutral logical interfaces that are abstractions of the underlying programmatic and data resources. This is quite analogous to the logical interfaces that support the virtualization of the infrastructure's computing, storage, and networking resources. A more detailed discussion of infrastructure virtualization is included in the Force10 Networks white paper: http://www.force10networks.com/products/pdf/wp_datacenter_convirt.pdf

The dynamic nature of SOA brings with it a requirement for better automation of system and application administration. In the early stages of an SOA project it may be possible to manually administer the portfolio of distributed services. However, in the long run, this should be handled by workload management middleware that works in conjunction with the SOA ESB middleware and infrastructure virtualization middleware.

Workload management middleware provides the level of abstraction that enables mobility of the workload across the underlying virtualized infrastructure. This benefits SOA applications as well as other styles of applications whose underlying resources are being consolidated and virtualized.

Workload management middleware should be capable of 1) scheduling monolithic or composite batch applications, 2) provisioning resources for execution of all types of services, and 3) automating management of service execution in the face of dynamic changes in workload or resource availability within the underlying infrastructure. For example, if workload requests increase, additional service provider instances can be started automatically on reserve resources, and work can be routed to them. If either a service or its underlying resource fails, the process can be moved to a new resource and the workload rerouted without service interruption. Middleware that performs these functions is sometimes also described as grid middleware, service virtualization middleware, or "workload virtualization" middleware.

Virtual server management tools already provide many of these workload management functions at the virtual machine level, or even across the entire distributed server environment comprised of both virtual and physical servers. Using these tools, many of the benefits of workload virtualization can be gained by running SOA services on virtual machines. With virtual server management tools, the services or applications running on virtual machines can be provisioned and controlled much more easily than would be the case if the applications and services are running on dedicated physical resources.
Figure 5 shows conceptually how infrastructure virtualization can be considered as a component of the middleware layer of a software architecture that embraces SOA and other prevalent application architectures, including traditional monolithic applications without SOA interfaces, and cluster applications.

Enterprises making a major commitment to SOA are typically establishing a multi-phased process to ensure that business and technological expectations are met and there are no disruptions to business operations during the transition from the legacy applications architecture to SOA. One of the early steps in the implementation phase should be a re-assessment of the data center infrastructure to ensure that its capabilities as a service-oriented infrastructure (SOI) will be well matched to the specific approach to SOA being envisioned.

Even if the roll-out of major SOA applications is a year or two away, it is not too early to bring on-going data center consolidation and virtualization projects into line with the enterprise vision for the type of SOI required. Force10 Networks and its partners have developed a reference architecture for a virtual data center that not only consolidates and virtualizes data center resources, but also establishes the data center as the heart of an SOI that can serve as the foundation for deployment of SOA.

The virtual data center architecture encompasses infrastructure virtualization middleware together with a greatly simplified POD architectural blueprint that can be readily adapted to both legacy applications and SOA services. Figure 6 shows conceptually how the pool of computing resources may be implemented as a number of POD modules that share a number of common architectural features and physical components.
In the Force10 Networks reference architecture for the virtual data center, the PODs shown in Figure 6 have the following characteristics:

- Each POD is fully virtualized with virtual servers, virtualized storage, and with a distinct virtual LAN assigned to each application or SOA service supported within the POD. In some cases, it may be desirable to have a number of closely related fine-grained services share a common VLAN.

- Because POD resources are virtualized, individual PODs can support multiple applications or services or an application or composite service can span multiple PODs.

- The internal network within each POD uses the same basic Layer 2/Layer 3 architecture with two tiers of 10 Gigabit Ethernet/Gigabit Ethernet switches providing access and aggregation switching for servers, Ethernet-attached NAS and iSCSI storage, and application-aware appliances. In accordance with normal practice, all intra-VLAN traffic is switched at Layer 2, while all inter-VLAN traffic is switched at Layer 3.

- In spite of the common network architectural template, each POD is customized in terms of the number and type of servers and storage devices, logical configuration, and the number and type of appliances dictated by the security services required by the application/service mix of the POD. SOA-specific appliances focus on providing security, acceleration, and other functions at the SOAP/XML message level rather than at the packet level. Intelligent SOA appliances are a flexible vehicle for providing the network with the added level of application-level intelligence required for SOA.

- Virtualization is not constrained to the boundaries of the POD, but can be extended to support a pool of shared resources that spans not only a single POD, but also multiple PODs, the entire data center, or even multiple data centers. Virtualization of the entire enterprise data center infrastructure allows the PODs to be readily adapted to an SOA application model where the resource pool is called upon to respond rapidly to changes in demand for services and to new services being installed on the network.

Figure 7 shows the basic template for the virtual data center POD based on a single tier of aggregation/access switches. Multiple servers running the same application/service are placed in the same application VLAN together with Ethernet-attached storage resources and appropriate appliances furnishing packet and application level services. Enterprise applications, such as ERP, that are based on distinct, segregated sets of web, application and database servers can be implemented within a single tier of scalable L2/L3 switching using server clustering and distinct VLANs for segregation of web servers, application servers, and database servers. Alternatively, where greater scalability is required, the application could be distributed across a web server POD, an application server POD, and a database POD.

Where multiple applications or services are supported by a single POD, inter-switch links support 802.1Q VLAN trunking for Layer 2 traffic. Where an application or service may span multiple physical PODs, the VLAN is extended across the data center core. In general, the links between the aggregation/access switches and the core switches carry a combination of trunked intra-VLAN traffic switched at Layer 2 and inter-VLAN traffic switched at Layer 3.

One of the keys to server virtualization within and across PODs is a server management environment for virtual servers that automates operational procedures and optimizes availability and efficiency in utilization of the resource pool. As noted earlier, management of virtual machines is a major component of the workload management middleware that helps to integrate virtualized SOA applications with the virtual infrastructure. These server virtualization and management capabilities...
are provided via a partnership between Force10 Networks and VMware.

The VMware VirtualCenter provides the server management function for VMware Infrastructure, including ESX server, VMFS, and Virtual SMP. With VirtualCenter, virtual machines can be provisioned, configured, started, stopped, deleted, relocated, and remotely accessed. In addition, VirtualCenter supports high availability by allowing a virtual machine to automatically fail-over to another physical server in the event of host failure. All of these operations are simplified because virtual machines are completely encapsulated in virtual disk files stored centrally using shared NAS or iSCSI SAN storage. The Virtual Machine File System allows a server resource pool to concurrently access the same files to boot and run virtual machines, effectively virtualizing VM storage.

VirtualCenter also supports the organization of ESX Servers and their virtual machines into clusters, allowing multiple servers and virtual machines to be managed as a single entity. Virtual machines can be provisioned to a cluster rather than linked to a specific physical host, adding another layer of virtualization to the pool of computing resources.

VMware VMotion enables the live migration of running virtual machines from one physical server to another with zero downtime, continuous service availability, complete transaction integrity, and continuity of network connectivity via the appropriate application VLAN. Live migration of virtual machines enables hardware maintenance without scheduling downtime and resulting disruption of business operations. VMotion also allows virtual machines to be continuously and automatically optimized within resource pools for maximum hardware utilization, flexibility, and availability. In particular, virtual machine resource allocations can be established with minimum, maximum, and proportional share levels for CPU, memory, disk and network bandwidth, allowing applications to satisfy peak loads without requiring manual reallocation of fixed levels of resources.

VMware Distributed Resource Scheduler (DRS) works with VMware Infrastructure to continuously automate the balancing of virtual machine workloads across a cluster in the virtual infrastructure. When guaranteed resource allocation cannot be met on a physical server, DRS will use VMotion to migrate the virtual machine to another host in the cluster that has the needed resources.

A more complete description of the Force10 Networks Virtual Data Center is provided by a separate white paper: http://www.force10networks.com/products/10GE_virtual_data_center.asp

SOA and Ethernet Networking

10 Gigabit Ethernet is the essential networking technology in next generation data centers because it provides optimal flexibility and investment protection through multiple steps in the evolution of the data center architecture regardless of how slow or how fast the application environment is transitioned toward SOA. Figure 8 shows a possible “building block” evolutionary path for data centers that can be followed in developing an SOI.

Some of the significant advantages offered by 10 Gigabit Ethernet in conjunction with Gigabit Ethernet in the aggregation/access layer and the core of the evolving data center include:

- **Consolidation Phase:** Resilient switch/routers supporting very high-densities of 10 GbE and GbE ports greatly simplifies traditional POD designs, consolidating numerous lower density switches and supporting a transition to 2-Tier POD networks based on a collapsed aggregation/access tier plus a core tier of switching.

- **Convergence Phase:** Ethernet is the premier technology for implementing a converged (or unified) data center switching fabric that integrates general purpose LAN connectivity with storage and cluster interconnect capabilities. Recent improvements in the throughput, latency, and CPU utilization characteristics of intelligent Ethernet NICs allows data center managers to take advantage of the simplicity and cost-effectiveness of Ethernet by reducing or limiting the use of specialty switching fabrics.

Figure 8. Evolutionary steps for the next generation data center
• **Virtualization Phase**: Ethernet offers the best support for infrastructure virtualization within the PODs of the virtual data center. Popular virtual machine software, such as VMware, provides native support of Ethernet networking for both LAN and storage networking, allowing the infrastructure to be fully virtualized using Ethernet as the unified fabric for all information flows required to support virtualization.

• **Service-Oriented Infrastructure Phase**: Ethernet leverages the broadest choice of application and service-aware appliances required by legacy networked applications and SOA applications. These appliances include not only the usual load balancers and firewalls, but also intrusion prevention systems, SSL offload devices, and XML-aware firewalls, accelerators, and gateways. Ethernet at 10 Gbps also provides the performance overhead to deal with the less structured traffic patterns implicit with SOA, where new services are continually defined and demand for services can change significantly in a short time. As SOA begins to account for a significant percentage of distributed applications, more network bandwidth will be required to deal with the markup overhead of XML-based messaging. XML messages can consume more than 10 times the bandwidth of traditional binary messaging protocols.

**Summary/Conclusion**

SOA holds the promise of a wide range of significant benefits. Over the next few years, many organizations are planning to take significant strides toward implementing an SOA based on web services. One important way to prepare for SOA is to develop a service-oriented infrastructure (SOI) that provides the flexibility and adaptability required for a highly dynamic application environment. A virtual data center architecture based on a hierarchy of Ethernet switching with coordinated investments in consolidation, convergence, and virtualization can provide a highly cost-effective approach to building an SOI as a foundation for SOA.