Whitepaper

SOA Infrastructure Reference Model

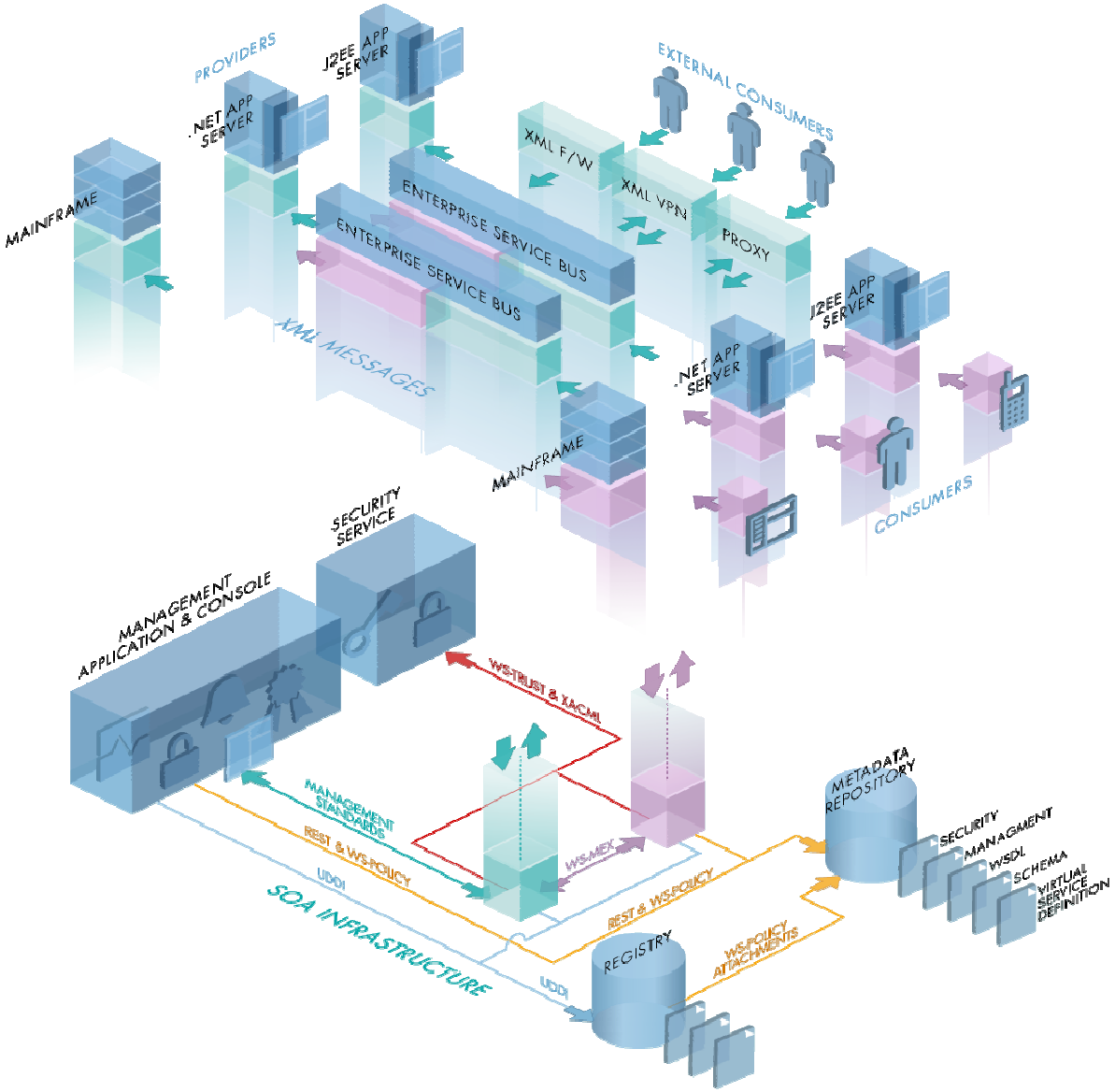
SOA Software, Inc.

12100 Wilshire Blvd, Suite 1800
Los Angeles, CA 90025
866-SOA-9876
www.soa.com
info@soa.com

Table of Contents

1 Introduction

SOA Infrastructure is the set of tools and technologies that an organization deploys to secure and manage services and service-oriented business applications. SOA Infrastructure has two main goals, to facilitate and promote reuse for enterprise agility and cost efficiency, and to provide visibility into, and ensure the security and reliability of the services and applications it deploys using the principals and concepts of service-oriented architecture.



The SOA Infrastructure reference model is published by SOA Software, the leading provider of SOA Infrastructure software products. It provides a product and vendor agnostic view of the concepts, components and standards that make up a successful SOA Infrastructure.

2 Reference Model Overview

This reference model focuses on the concepts, components and standards that are required to build effective SOA Infrastructure. It provides a conceptual breakdown of an enterprise SOA into two layers; an application and messaging layer, and an infrastructure layer. In reality these two layers are tightly integrated, although their focus and role is considerable different.

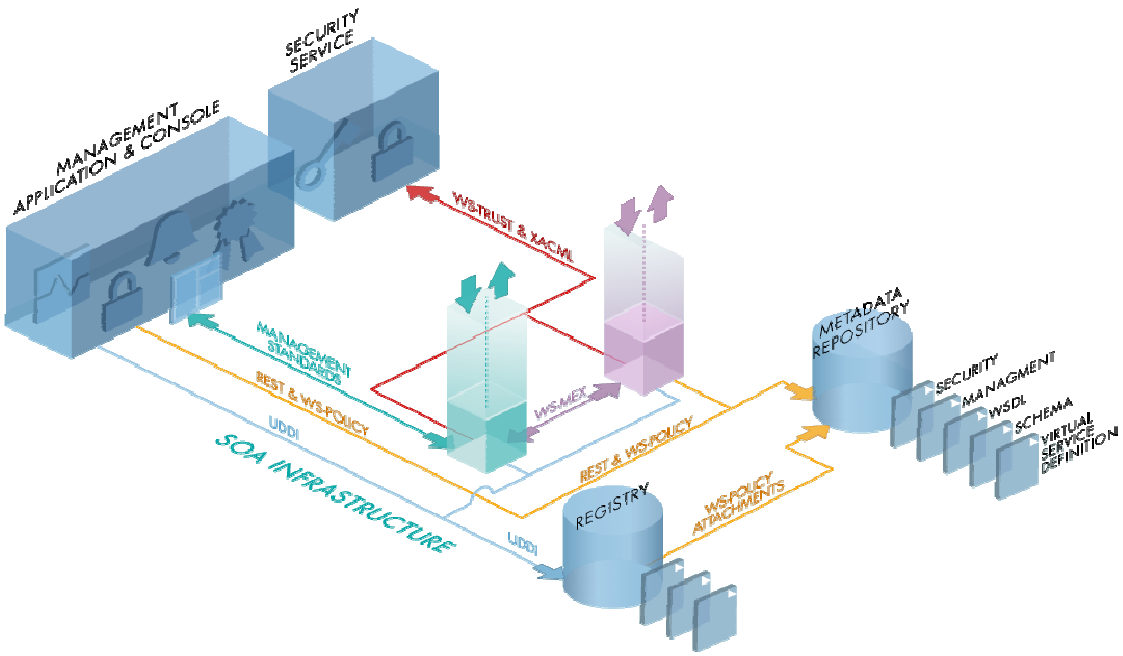
The application and messaging layer is where services, applications, and messaging platforms – such as Application Servers, Enterprise Service Bus(es), and Business Process Management engines – reside. In this layer, applications and services expose interfaces that other applications and services consume focusing only on the business logic, and business interface specifications.

The infrastructure layer provides security, management, governance and monitoring services to the application and messaging layer. It ensures that appropriate policies are enforced by services as they receive messages, and that applications send message that comply with the policies that will be enforced by the receiving service.

This separation between these layers is critical to ensure the true loose-coupling of services and applications that is required to achieve the efficiency and agility benefits of SOA. The infrastructure layer provides agents, proxies, and delegates to ensure that the application and messaging layer can access and use the services it delivers.

3 Infrastructure Components

The infrastructure components deliver the core infrastructure services to the applications and entities at the messaging and application layer.



3.1 Registry

A registry is a simple entity that provides a system of record for services in the SOA, and a catalog for finding these services. Most solutions have now standardized on using UDDI for the core registry, in the same way that most identity and access management solutions standardized on LDAP as a system of record for user objects. Registry products range from simple UDDI server implementations to complete asset management repositories. This reference model recommends the use of a UDDI server as a central system of record for services, but does not mandate anything beyond support for essential UDDI APIs. Note, a registry is a distinct functional entity from a repository, although many registry servers also provide fully functional repositories.

3.2 Metadata server

The Metadata server is a simple repository service that works in conjunction with the registry to provide applications with essential information about the services they either consume or provide. At its core a metadata server is a database that provides simple access to metadata artifacts (most of which are standards defined XML documents) the likely standard for this access is a simple http based protocol such as REST. This metadata will most often include; WSDL documents, security policy assertions, management policy assertions, schema documents, and virtual service definitions.

A metadata server can also go beyond this simple data publishing model to provide a comprehensive environment for controlling and managing access rights and workflow for the creation and management of metadata and service versions.

3.3 Security service

In this reference model, the security service serves two purposes. It is both a token server, and an authorization server. These two functions do not have to be provided by the same server, although most security architectures do consolidate these functions into a single product, or product family. Common examples of consolidated security product families are CA SiteMinder, IBM Tivoli Access Manager, and Oracle CoreId.

A security token server provides both authentication and token exchange services. It can consume a credential, and return a token of some description, most likely a SAML assertion in a Web services environment. A common use case for both authentication and token exchange in Web services is for the security token server to work in conjunction with a portal to request a username and password from a Web browser user, and provide the browser with an http session cookie. When the portal needs to request access to a Web service, it should then contact the security token service and exchange the cookie for a SAML assertion. There are a number of emerging standards for security token servers, the most popular of which are WS-Trust for requesting tokens, and SAML as a token format.

An authorization service is exactly what its name suggests. It provides a service for making decisions about whether a particular request is authorized or not based on a number or factors including user, role, or other sender identifying characteristics, request content, request destination, and environmental factors such as destination real-time performance. Most authorization servers still implement proprietary APIs, although XACML remains the most commonly discussed and implemented authorization standard.

3.4 Management Application

Management application is a term taken from the WS-Distributed Management standard (see standards for more details). It refers to a server that monitors the performance, throughput, and usage of services and applications, and consolidates this information to provide valuable services such as SLA reporting, performance charting and trend analysis, and alert and exception management.

3.5 Management Console

The management console provides an interface for monitoring the entities in both the application and infrastructure layer, it also provides the central console for policy definition, metadata management, and can deliver a comprehensive workbench for service lifecycle management. The management console will interact with each of the infrastructure services, ideally using the standard

interfaces they define and implement (UDDI, REST, WS-Trust, XACML, and "management standards").

3.6 Intermediaries

Intermediaries provide the connection between the applications and entities that exist at the application and messaging layer, and the services offered by the infrastructure layer. Over time the application servers and other entities at the application and messaging layer will evolve to be able to directly use the infrastructure services, but until this time a wide range of intermediary types continue to be critical.

3.6.1 Endpoint (Agent)

An endpoint intermediary exists inside the provider application as an agent. There are a wide range of different agent types, although they should ideally be deployed as handlers in a standard programming model like AXIS or JAXRPC for Java applications, and should use published APIs for .NET and other programming models. The agent intermediary model offers a number of advantages, mainly last-mile security – because there is no way to physically access the service without passing through the handler, and distributed processing – there is no central bottleneck for XML and cryptographic operation, this load is delegated to the application server. The disadvantage of the agent model is that an agent cannot practically route traffic, once the agent has received a message, it has already been routed, and it does not make sense to try to use an agent model for load-balancing or failover type scenarios.

3.6.2 Proxy

A proxy intermediary exists as a standalone entity in the network. There are several different proxy types, some that deploy into an application server container, others that are standalone, some that provide their own hardware as a network appliance, and others that are software only. Proxies have the advantage that they can route traffic and so can deliver load-balancing and high-availability capabilities, and can even deliver complex mediation and routing services. Good proxy architecture provides proxies that are stateless and clustered for high-availability and load-balancing themselves. The proxies should expose "virtual services" – see sidebar. The main disadvantage of a proxy is that it cannot ensure the security of the service to the last mile, there is no way to ensure that consumers cannot send message directly to services bypassing the proxy.

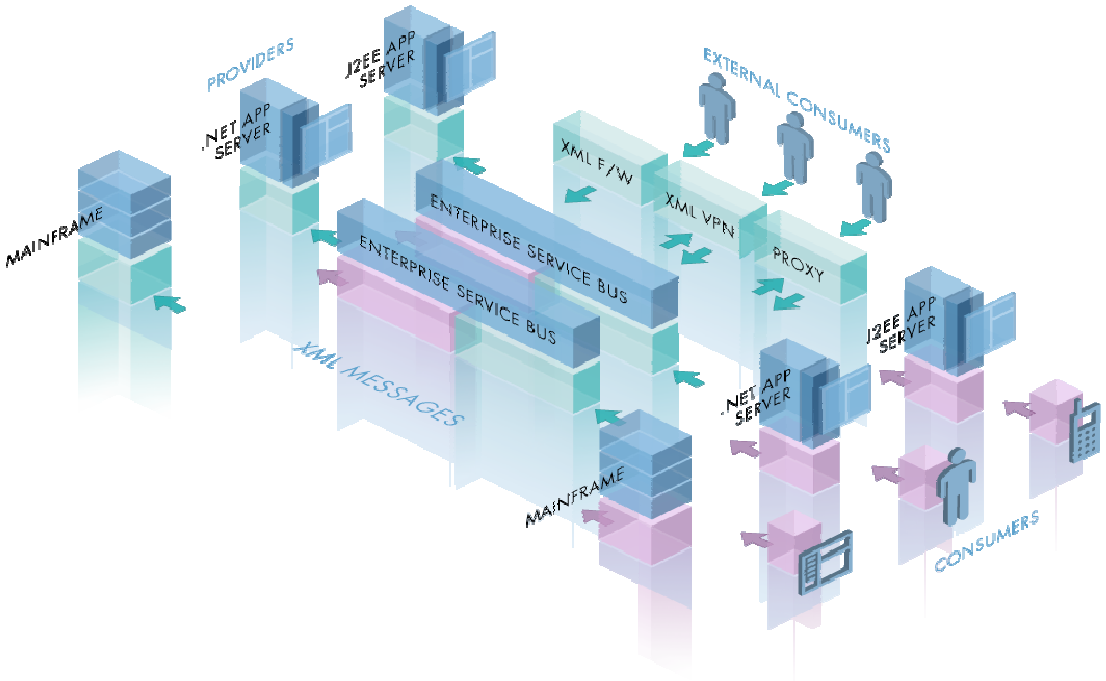
3.6.3 Delegate

The delegate is a critical component of an SOA, but is less often considered by infrastructure providers. The delegate is a client side intermediary, most often deployed declaratively as a client-side handler, or used as a full SDK. It abstracts the consumer developer from any direct knowledge of the location, policies, or even the messaging style requirements of the service itself. A good delegate provides automatic runtime discovery of service location, transport and policy requirements, and ensures that any message the client sends is formatted

correctly, implements the right security policies – including XML-Signature, XML-Encryption, and credential insertion, before sending the message to right location.

4 Application and Messaging Components

The application and messaging components are the entities that implement and consume business services using XML messages to interact with each other.



4.1 Application Server

Application servers are an accepted and well understood part of enterprise architecture. More and more applications are deployed in either Java or .NET application server platforms and modern development tools assume that the programs they create will run in one of these platforms. A large majority of newly built Web services applications will be deployed inside an application server of one form or another. One advantage of working with application servers for SOA deployment is that there are a large range of tools and standards to ensure interoperability of the more complex messaging requirements.

4.2 Mainframe

It may seem out of place in an SOA reference model, but Web services and SOA are partly responsible for the resurgence of the mainframe as a mainstream enterprise computing platform. Web services allow reliable, high-performance business applications running on the mainframe to participate in modern architectures. Large enterprises are now using tools to expose mainframe applications as Web services rather than trying to rebuild business logic in their new applications. As mainframe Web services usage grows, so does the need to ensure that the mainframe can participate in the enterprise SOA, leveraging the service provides by the SOA infrastructure.

4.3 Business Application

Ranging from deep back office systems like ERP and financials, to desktop applications like email, word processing and spreadsheets, business applications are the heart and soul of any IT organization. Web services and SOA provide an efficient, agile way to integrate back office systems, and to connect desktop applications to these systems to ensure that information technology can keep up with the fast changing business. The SOA infrastructure provides the underlying security, management, monitoring and governance services to ensure true agility, performance, and compliance.

4.4 Enterprise Service Bus

Enterprise Service Bus (ESB) is an increasingly popular concept. Originally conceived as the evolution of both message-oriented middleware and EAI (enterprise application integration) solutions, the ESB means very different things to different organizations. As analysts, vendors, and customers come to terms with the idea of an ESB, it appears that an ESB encapsulates 3 functional concepts; adapters taken from an EAI tool to ensure connectivity with legacy applications, messaging services taken from a message-oriented middleware platform to ensure reliable delivery, and process orchestration to build agile applications. The consensus amongst analysts is that most organizations will end up with multiple ESB platforms from multiple vendors, providing services in their own context. In this environment it is critical that the ESBs use a central SOA infrastructure to ensure consistent compliance with security and messaging policies for the services they expose and consume.

4.5 Business Process Management

Sometimes a standalone server or service, sometimes part of an ESB, business process management solutions provide tools for the definition and editing of business processes, and a server environment for executing the defined processes. Business process management solutions often implement a standard called BPEL – Business Process Execution Language, as a way to define sharable interoperable business process documents.

5 Standards

This reference model does not attempt to define the standards used at the application and messaging layer, it focuses on the infrastructure standards, specifically the standards that provide the interfaces between the two layers, and also between the various components of the infrastructure layer itself.

5.1 WS-Policy

WS-Policy refers to a set of standards used to define, and share policy documents. It includes a set of assertions, although at the time of publishing, the only published assertions are WS-Security Policy, and WS-Reliable Messaging Policy. It also defines a model for associating policies with services, (WS-Policy Attachments). While, WS-Policy is seeing increased adoption through the industry, and is the policy standard chosen for this reference model, it does have some deficiencies that the standards community is working hard to address.

5.2 WS-Distributed Management

The WS-Distributed Management standard is currently in a state of flux, as many of the protocols that underlie it have been impacted by the harmonization initiative driven by HP and IBM. At the time of publication, there is no immediate successor to this standard, so the reference model simply refers to "management standards" as the interfaces to the management application.

5.3 UDDI

Universal Discovery, Description, and Integration (UDDI) is the most commonly used standard for registry services. It provides a system of record and catalog of services. The commonly used APIs are inquiry, publish, and subscription. Inquiry is used to find services and service details from the registry, publish provides an interface for adding or changing services and service details, and subscriptions allows registry consumers to keep abreast of changes to services in the registry.

UDDI was conceived as a central, Internet-based service that business would use to publish their services, and find services published by other businesses. For a few years, some registry vendors created and maintained the UBR, but this initiative was terminated between 2005 and 2006. Today, UDDI is the accepted standard for internal service registries for enterprise SOA.

5.4 REST

Rest (REpresentational State Transfer), is a simple http-based mechanism for providing access to documents (often, but not always XML documents). REST is often thought of as an alternative to SOAP-based Web services, but it really serves a different purpose, and is a valuable component of the reference model, providing an ideal protocol - along with other standards like WS-Metadata

Exchange (WS-MEX) – for a metadata repository to use to serve up policy and other metadata.

5.5 WS-Trust

WS-Trust is a standard designed to provide interfaces for security token servers. It provides authentication and token exchange services, and is most commonly associated with SAML authorities.

5.6 XACML

The XML Access Control Markup Language (XACML) is an authorization protocol, providing a mechanism for defining authorization requests in a standard form. It has not yet been widely adopted, but does seem to have widespread acceptance.

6 Concepts

6.1 Virtual Services

A virtual service is a service exposed by a proxy on behalf of one or more physical services. Virtual services have their own WSDL, implement their own policies, and provide a variety of value added capabilities to physical services. Virtual services can be used for many things including; simple service monitoring, load-balancing and failover, content and SLA-based routing, central security policy enforcement (see notes on last mile security), service versioning, and many others. One simple, but powerful use case for virtual services is to provide a service that delivers a consistent interface that can send requests to operations taken from multiple different services implemented in multiple different technologies.

6.2 Policy Definition

Policy Definition is the process of defining and managing policy assertions that can be associated either by copy or reference with service and method entities in the UDDI registry. SOA Software's Service Manager provides a centralized policy definition point for SOA Infrastructure as well as a complete monitoring solution and management application.

6.3 Policy Enforcement

Policy enforcement is the process of ensuring that Web services and XML messages and transactions comply with local, corporate and global security and operational policies. SOA Software's products offer three types of policy, security, management and monitoring:

- Policy-based security enforcement includes access-control list management, identity management, authentication and authorization policies
- Policy-based management simplifies service provisioning with a powerful template mechanism
- A powerful contract and SLA template mechanism ensures consistent policy-based monitoring of services

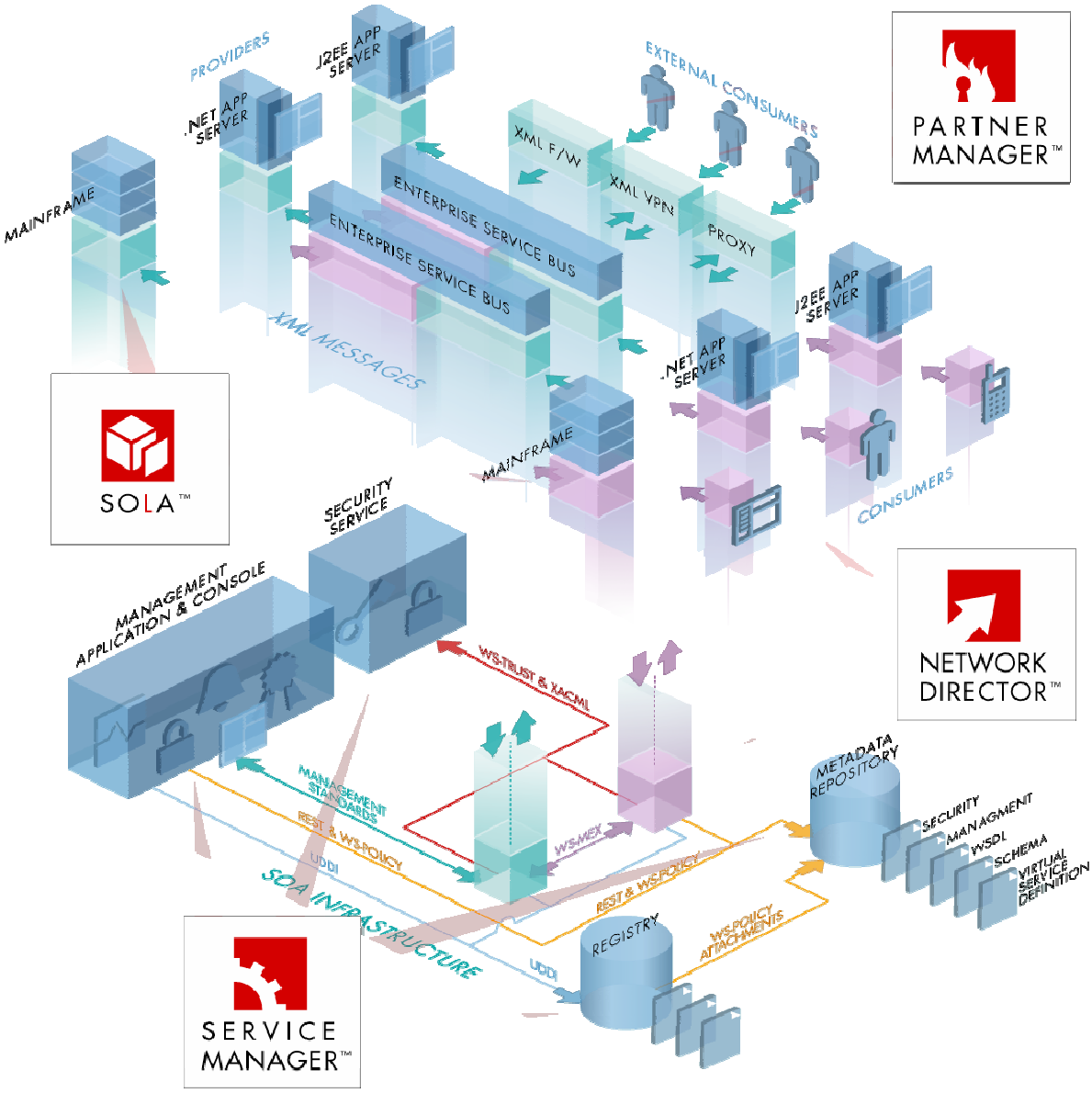
6.4 Policy Implementation

Policy enablement goes beyond policy enforcement by providing technology to allow consumers and providers to readily leverage centralized policy.

Through the sharing of policy meta data, SOA Software's products can act as proxies for end users to encrypt and sign messages, authenticate users using one authentication method and transform outbound messages to another authentication type, e.g. basic user ID/password authorization inbound, signed SAML assertion outbound. It also leverages shared policy to deliver reliability,

7 SOA Software Products

SOA Software's products provide a strong implementation of this SOA infrastructure reference model, leveraging the standards, and providing intermediaries to deliver core infrastructure services to XML applications.



SOA Software's products include:

7.1 Service Manager

Service Manager delivers a complete set of infrastructure applications providing comprehensive implementations of each of the infrastructure components described above.

It includes:

- Policy Manager – provides the security services
- Management Server – provides the management application monitoring functions
- Alert Manager – provides alerting services to all the other subsytems and delegates
- Registry Manager – provides a UDDIv3 registry combined with a comprehensive metadata repository

7.2 Network Director

Network Director provides the intermediaries that enforce and implement policy for Service Manager. It includes a standalone proxy, in-container agents for most Java and .NET application servers, and ESBs, and delegates in the form of client handlers and SDKs. It delivers comprehensive governance, security, monitoring, management, and mediation of Web services.

7.3 Partner Manager

Partner Manager is an extension to Service Manager, providing a complete trading partner management, and service provisioning workflow solution. It uses service virtualization to securely extend internal services into partners' networks.

7.4 SOLA

SOLA is a Mainframe Web services tool that allows CICS applications to easily, securely, and reliably expose and consume Web services without impacting performance. It acts as a policy enforcement and implementation point for CICS Web services.

8 About SOA Software

SOA Software is the leading provider of comprehensive enterprise-class SOA management, security, and governance solutions. SOA Software's products include the award winning Service Manager™, Network Director™, Partner Manager™, and SOLA™. Service Manager provides a high-performance, scalable SOA Management solution. Network Director provides mediation and tolerance capabilities, ensuring that services can be consumed by the widest possible range of applications. Partner Manager makes it easy for companies to securely publish B2B Web services for their partners to consume. And SOLA is the only production proven mainframe Web services solution for CICS programmers. These enterprise-class products combine to create the only complete SOA Infrastructure solution available today. SOA Software is a privately held company backed by leading investors including Draper Fisher Jurvetson, Redpoint, Mellon Ventures, Palisades Ventures Fund and Paladin Capital Group. For more information, please visit http://www.soa.com.