Governing a Distributed SOA

IONA Technologies September 2007

White Paper

Making Software Work Together™

Executive Summary

While the advantages of a service-oriented architecture (SOA) are apparent to most organizations, the challenge of governing the SOA infrastructure still remains a concern for leading enterprises. As organizations incorporate their IT assets into networks of reusable services they are challenged with how to enforce service reuse and how to govern proper usage. Information about what services exist and how to use them are often inconsistently documented in multiple locations and formats, making it difficult to reuse services and enforce policies.

Compounding the challenge of SOA governance is the advent of the distributed architecture, which analysts predict will become the prevailing deployment model for SOA in the coming years. A distributed, microkernel design delivers a lightweight, flexible runtime that can be deployed at the endpoints, as opposed to the centralized, hub-and-spoke middleware solutions traditionally offered by most SOA vendors today. These vendors are racing to deliver microkernel-based SOA infrastructure solutions to market, proving distributed architecture is a superior SOA design model.

While a distributed architecture allows organizations to better leverage existing IT investments, adopt SOA incrementally and avoid vendor lock-in, its distributed nature creates additional governance challenges. Organizations must drive and streamline the deployment of services into a distributed SOA network across many systems and locations, as opposed to governing services in the more centralized architecture created using traditional middleware solutions.

While the challenge of SOA governance with a distributed SOA may seem daunting at first, innovative tools are available today to help leading enterprises achieve SOA success. A governance toolset may include solutions for policy management, monitoring and quality assurance; a key component is an *active* registry/ repository as it serves as the command and control center for SOA governance.

An active registry/repository automates enforcement throughout the service lifecycle. It increases developer efficiency by allowing developers to reuse existing services and avoid duplication of efforts, all while promoting consistency.

IONA leverages its experience working with some of the worlds earliest and largest SOA deployments to help organizations tackle SOA governance while experiencing the benefits of a distributed SOA. IONA offers the unique combination of an active registry/repository with a microkernel-based enterprise service bus, allowing organizations to create a dynamic environment that actively monitors, governs, and enforces all distributed endpoints. These innovative tools are helping organizations deliver on the promises of SOA, including business agility, increased ROI and lower operational costs, today.

The SOA Governance Challenge

A well-formed SOA may service-enable dozens of applications across a complex, heterogeneous IT environment, creating hundreds of services that need to be tracked, managed and measured. SOA governance offers a way to control, manage and drive a large number of services, regardless of location.

According to Gartner Research Director Frank Kenney, "the market for SOA governance technologies is a varied one with many different types of products providing support for governing the behavior of an SOA."[1] While a single approach to SOA governance has yet to emerge, the industry is taking its cues from the world around us. We are surrounded by and interact with examples of governance in the form of policies and enforcement every day. At the corporate level, investment banks forbid employees from trading stock/securities without declaring it to the bank. Other companies prohibit employees from removing sensitive documents from their offices.

At the IT level, some companies might intentionally disable all of computers' USB ports to prevent network infiltration or intellectual property theft. The email address format is first name then last name. Or maybe the CTO put a ban on Instant Messenger (IM) because he or she thought it would make support easier. All of the above are examples of establishing policies at different levels.

Just putting policies in place, however, does not ensure governance. These policies need to be enforced. For example, the previously mentioned investment bank would need to monitor employees' bank accounts to detect unauthorized transactions. Administrators might use packet monitoring tools to detect any IM traffic or disable user access rights so that IM software cannot be installed.

Gartner Research Analyst Paolo Malinverno defines SOA governance as follows:

> IT governance provides a framework in which the decisions made about IT issues are aligned with the enterprise's overall business strategy and culture. SOA governance identifies decision-making authority for defining or modifying the business processes that will be supported with SOA techniques, the service levels required, the service performance requirements, the access rights and so on. In addition, SOA governance addresses the way reusable services are defined, designed, accessed, executed and maintained. SOA governance is also an important mechanism for determining service ownership and cost allocation in a shared-service organization.[2]

SOA governance is all about declaring how services should behave, by establishing policies, and making sure those policies behave correctly through monitoring, auditing and enforcing the policies throughout the stages of the application development life cycle.

---

[1] Frank Kenney, Research Director, Gartner

[2] Paolo Malinverno, Gartner, *Toolkit Presentation: The Integration Competency Center's Role in SOA Governance*, April 24, 2007

While it's one thing establish, monitor, audit and enforce policies for services in a centralized network; it's another to do so with services enabled by a distributed architecture.

The Shift toward Distributed Architecture

With SOA, services can be deployed anywhere on a network, so its design naturally incorporates distributed computing concepts. While many SOA infrastructure solutions available today can manage these distributed services, they do so in a centralized way, usually through a hub-and-spoke integration model that requires costly application servers. These centralized hubs require additional hardware and software investments while their proprietary nature promotes vendor lock-in. Leading organizations are discovering that they do not need additional application servers to make their IT infrastructure more agile. That's why they are beginning to take a different approach to implementing and managing a SOA, leveraging a truly distributed computing architecture that is inherently flexible and agile. In fact, Gartner predicts that between 2008 and 2011, distributed, microkernel-style design will become the prevailing internal architecture offered by most vendors[3].

A distributed architecture supports a variety of service consumers, providers, and intermediaries, as illustrated in Figure 1. This approach reduces constraints on the SOA implementation and frees developers and designers to choose the most cost effective and efficient solutions for their particular environments. Quality of service policies such as security, reliability, transactionality, and so on can be added and configured as needed.

A distributed architecture offers significant advantages over SOAs built with proprietary, hub-and-spoke middleware solutions currently on the market. A distributed architecture allows enterprises to leverage existing IT assets, instead of requiring dedicated hardware. It lets companies adopt SOA incrementally, from both an economic and technical perspective, so they no longer have to pay large upfront costs or commit to a sweeping transformation of their IT environment. Finally, a distributed SOA architecture eliminates vendor lock-in and future-proofs IT investments by enabling companies to dynamically adapt to change caused by growth, mergers and acquisitions, and the introduction of new technology models.

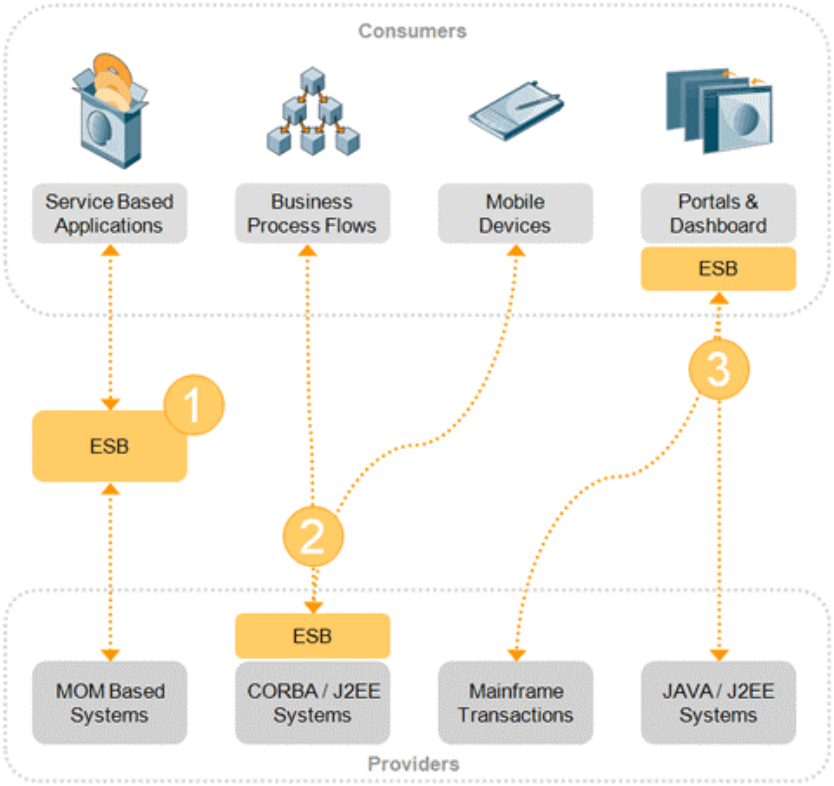

Figure 1. Architecture of a Distributed SOA Infrastructure – In a distributed architecture, an ESB be deployed to act as an intermediary between consumers and providers (1), on any provider to endpoint to service-enable it (2), or on any consumer to accept calls from any provider (3).

Governing a Distributed Architecture

While the benefits of a distributed architecture are clear, enterprises face additional governance challenges created by the inherent complexity of a de-centralized computing model. Existing governance tools such as a SOA registry/ repository begin to tackle some of this inherent complexity by providing a centralized data store for an SOA including:

- Service contracts
- Meta data (including policies)
- Design documents
- Life cycle state info (rules and workflow for migrating between states)

However, most of the registry/repository products found on the market today are static and do little more than capture information and report on it. It's critical that enterprises invest in a registry/repository tool that is ***active***. After an enterprise has defined all of the policy details associated with their services, they need to implement them, and even more importantly, enforce them.

Enforcement of policies is critical to governance of the entire SOA life cycle as it can ensure a transfer of knowledge through the organization. Many organizations are challenged with providing visibility into how various components in the IT environment work together to multiple groups within the organization. Access to training, documentation, and other materials is limited and not always shared across multiple teams. For example, the transition from development to operations can be a challenging and error prone process. The operations team must ensure that

- services are configured with the correct Qualities of Services (QoS)
- they can prevent the disabling or modification of security settings of a service by unauthorized personnel
- they can quickly fix a service when it is incorrectly changed either by an administrator's mistake or purposely by attackers

Policies should be implemented and enforced to ensure that the appropriate information is captured to enable operations to manage and support the services before they are rolled out. The registry/repository requires that such materials exist prior to deployment to ensure that groups like development and operations have everything they need to use, modify, manage and support the service.

The registry/repository must work closely with a microkernel runtime to create a dynamic environment to actively monitor, govern, and enforce all of the distributed endpoints. With a microkernel runtime that supports configuration and policy based programming, the registry/repository can easily achieve "Service Provisioning." Service Provisioning refers to the ability to configure a particular device, machine, network element, or in this case, a service endpoint, with everything it needs to provide a particular service.

A microkernel, configuration-based runtime makes it much easier to provision services because many aspects of the runtime can be dynamically altered by applying policies. A new service or modified version of an existing service can be easily deployed to a microkernel container. The package generated incorporates all elements necessary to implement the policies that define the qualities of service and other variables associated with the service.

The microkernel runtime can also allow for the addition of validation and auditing plug-ins to the container to actively perform policy enforcement while keeping the container small and agile. Unlike heavyweight, monolithic runtimes, the microkernel architecture only has to load the features an endpoint requires, thereby keeping the memory footprint of the runtime as small as possible.

Tools for Effective SOA Governance: IONA Artix

IONA offers a comprehensive suite of SOA infrastructure components including Artix Enterprise Service Bus (ESB) and Artix Registry/Repository, which represent an elegant combination of the microkernel runtime and SOA governance. With Artix Registry/Repository, users have the ability to define and store all kinds of metadata (service contracts, QoS, routes, transports, access control) for the entire distributed SOA infrastructure regardless of where endpoints are located. Organizations can also define custom scenarios for different stages of the development life cycle.

If an organization wants to roll out a deployment for any service location, or the entire infrastructure, Artix Registry/Repository will generate or retrieve the corresponding deployment artifacts (service contracts, implementations, start/stop scripts, configurations, etc.), apply the appropriate policies, and assemble them into ready-to-deploy packages. The generated packages include everything the enterprise needs to deploy the services and policies to the target service location. If the target runtime is also an Artix ESB container, thanks to the microkernel architecture, the package is composed of mainly configurations. When such package is deployed to a particular runtime (or container), it dynamically loads all the required plug-ins.

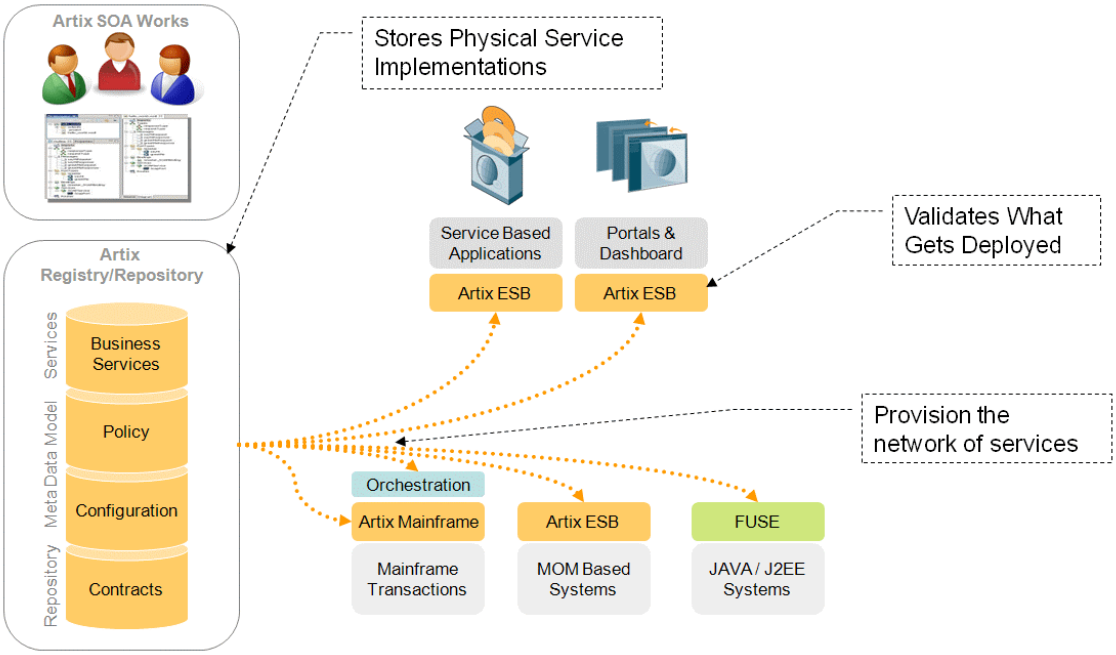

Figure 2. Artix Registry/Repository Provisions a Distributed Network of Services

Using this technology, developers who want to create a test environment simply have to define the required policies and apply them to the appropriate services to create a deployment package. Artix

Registry/Repository and Artix ESB take care of the rest. If the QoS in the test environment are different from those in production, they can simply apply different policies to the services to create a new set of deployment packages. Once the organization is ready to move the set of services to production, the packages contain all the files needed by the operations team for deployment.

The deployment package can be manually installed to the target service location, or, Artix Registry/Repository can deploy the package remotely to the target container. This service provisioning capability saves significant time and cost. In addition, Artix Registry/Repository actively provides a dynamic view of all the Artix ESB endpoints no matter where they are located. An organization can validate any of the endpoints and see what's being deployed, what policies have been configured, and whether any of the policies have been violated. If the configuration of any of the service containers has been violated or damaged, users can simply pull up the metadata saved in the repository for the problematic endpoint, and push the deployment package to the remote container.

Artix Registry/Repository and Artix ESB are components of IONA Artix, a comprehensive suite of products to streamline and modernize complex and heterogeneous IT environments.

The suite includes:

- Artix ESB
- Artix Registry/Repository
- Artix Orchestration
- Artix Data Services
- Artix Mainframe
- SOA Management provided by AmberPoint

Conclusion

A microkernel-based runtime and active registry/repository for SOA governance not only complement each other, but become a powerful combination. Organizations cannot afford to deploy a heavyweight hub-and-spoke architecture through which everything is funneled. Enterprises need services that sit close to line-of-business locations, are distributed, and decoupled with data formats, transports and protocols so their reusability is enhanced. These reusable services are integration ready so that additional hardware/software is not needed for integration. At the same time, these services must be federated together to create a dynamic, tightly governed and enforced environment. These are not experimental concepts. IONA's advanced SOA infrastructure suite, IONA Artix, is a very mature, microkernel-based product suite that allows users to build a dynamic, distributed SOA infrastructure today.

IONA Technologies PLC
The IONA Building
Shelbourne Road
Dublin 4
Ireland
Phone +353 1 637 2000
Fax +353 1 637 2888
Support: support@iona.com
WWW: www.iona.com

IONA Technologies Inc.
200 West Street
Waltham MA 02451
USA

Phone +1 781 902 8000
Fax +1 781 902 8001
Training: training@iona.com

IONA Technologies Japan Ltd
Akasaka Sanchome Building 7/F
3-21-16 Akasaka
Minato-ku Tokyo
Japan
Phone +813 3560 5611
Fax +813 3560 5612
Sales: sales@iona.com

0000KC01-1007

Making Software Work Together™