An Introduction to SOA Governance

The Most Important Aspects of SOA Runtime Governance

Compiled by Jindřich Štumpf
Business Consultant
Progress Software Czech Republic
<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>The Four Stages of SOA Governance</strong></td>
<td>3</td>
</tr>
<tr>
<td>Runtime Governance—What is it?</td>
<td>3</td>
</tr>
<tr>
<td>The Four Stages</td>
<td>3</td>
</tr>
<tr>
<td>Process</td>
<td>3</td>
</tr>
<tr>
<td>Measurement</td>
<td>4</td>
</tr>
<tr>
<td>Enforcement</td>
<td>4</td>
</tr>
<tr>
<td>Feedback</td>
<td>4</td>
</tr>
<tr>
<td>Runtime Governance Today</td>
<td>4</td>
</tr>
<tr>
<td><strong>Provider and Consumer Discovery</strong></td>
<td>7</td>
</tr>
<tr>
<td>What is Discovery?</td>
<td>7</td>
</tr>
<tr>
<td>Visual Visibility</td>
<td>8</td>
</tr>
<tr>
<td>Service Consumer Discovery</td>
<td>9</td>
</tr>
<tr>
<td>Registry Integration</td>
<td>10</td>
</tr>
<tr>
<td><strong>Flow Mapping</strong></td>
<td>11</td>
</tr>
<tr>
<td>Sharing information about service relations and dependencies</td>
<td>11</td>
</tr>
<tr>
<td>Flow Mapping Value</td>
<td>12</td>
</tr>
<tr>
<td><strong>Other SOA Runtime Governance Elements</strong></td>
<td>13</td>
</tr>
<tr>
<td>Rogue Service Elimination</td>
<td>13</td>
</tr>
<tr>
<td>Security Policies</td>
<td>14</td>
</tr>
<tr>
<td>Business Process Visibility</td>
<td>14</td>
</tr>
<tr>
<td>Policy Enforcement</td>
<td>16</td>
</tr>
<tr>
<td><strong>Glossary of Terms</strong></td>
<td>17</td>
</tr>
<tr>
<td><strong>Resources</strong></td>
<td>18</td>
</tr>
<tr>
<td>Contact</td>
<td>18</td>
</tr>
</tbody>
</table>
The Four Stages of SOA Governance

Runtime Governance—What is it?
Interoperability and service reuse are among the major promises of service-oriented architecture (SOA). Yet interoperability and reuse can only be fully realized when everyone is working on the same page. Hence, not surprisingly, SOA has been a key driver in the increasing emphasis on, and interest in, governance in recent years. SOA has its own needs not only in pre-production phases (design, development, testing) but primarily in the production phase.

Leading the charge for governance have been enterprise architects, who know quite well that for SOA systems to deliver value, there must be control in areas ranging from service design and deployment processes, to granular items such as schemas and WSDL creation. SOA Governance was primarily focused on this pre-production phase.

Formerly, given the early stage of SOA technology and practices, it made sense that organizations implementing such systems focused primarily on these areas, especially since most companies were still in the development and design phase. Today, however, with SOA services now in production within many organizations, system architects are realizing that the most critical area for control and governance is now runtime. Data point after data point has demonstrated that many SOA implementations are just not working in production as designed or expected. Problems range from service interruptions to entire business processes failing and security and compliance risks that generate costly delays and lengthy triage cycles. As these problems continue to pile up, runtime governance is, not surprisingly, now taking center stage for companies launching and utilizing SOAs.

The Four Stages
Runtime governance can be divided into four primary areas: process, measurement, enforcement, and feedback. Process comes first because if it is compromised, circumvented, or not adhered to, there can be no effective control.

Process
While a great deal of process is employed in the SOA pre-production side, active governance kicks in when an application is migrated from development and into production. It is at this point that runtime governance can detect and report if services or consumers in production are adhering to governance guidelines. Experience indicates that violations can result not only from “rogue” service components that have somehow bypassed the development governance process, but also from services that have gone through the proper release process according to the established governance guidelines, yet somehow result in violations when in production.
The Four Stages of SOA Governance

Measurement
While significant governance work and planning occur in design and development, what is critical to governance is what occurs in the runtime environment and, more to the point, knowing what is going on across your SOA during runtime. For example:

- Are all my services in compliance now?
- Is customer data encrypted?
- Does the service have the right security policies in place?
- Are the business rules being enforced?

Enforcement
End-to-end visibility and control over business processes are critical to enforcing business and IT rules, reporting on them, and having the ability to do something about them in real time. Specifically, when governance guidelines are enforced, a runtime system can dynamically react to business opportunities or IT issues to directly impact the bottom line.

Feedback
Systematically tracking governance infractions and tracing their causes facilitate a lifecycle approach in which organizations are able to quickly fix and address breaches upstream. Runtime governance plays a crucial role as the last line of defense and is designed to protect the company and the IT system. By carefully coordinating development governance (i.e., the UDDI registry) and runtime governance, organizations can build a world-class governance initiative with each party doing its part at the proper time.

Runtime Governance Today
New SOA Governance tools are able to automatically discover services and consumers in production environments. They use software for monitoring the message transmission between services; production data in the service network can be displayed in the form of technical, infrastructure or business views. This information is also used for immediate and automatic application of governance policies based on rules defined in SLAs.

This capability represents a fundamental shift forward in SOA governance, not to mention a significant advancement in reducing the risk traditionally associated with SOA implementations. While system architects in the past had no choice but to pursue a strategy of “the services that we’ve tested compliance with are...,” they now can confidently state that “there is no service in production that does not meet the compliance requirements of our governance policy.”
The Four Stages of SOA Governance

SOA Governance system should have these features:

- Automatic discovery of service providers and service consumers
- Out-of-the-box integration with leading registries (such as Hewlett-Packard Systinet)
- Flow mapping and service dependency tracking (both upstream and downstream)
- Rogue service detection and elimination
- Separating policies from services
- Real process visualization for different user types
- Proactive policies enforcing [security, compliance],
- Changing policies independently of service (and process) changes
It goes without saying that a runtime governance strategy for SOA is not limited to Web services and HTTP running on J2EE and .NET application servers, but also includes various other protocols and platforms prevalent in real world SOAs such as RMI, EJB, JDBC, etc.

Figure 1: From the viewpoint of the whole enterprise, SOA Runtime Governance must be executed across applications.
What is Discovery?

Like the word “governance” itself, the word “discovery” can be defined a number of ways and, typically, used under a variety of situations. Some developers need a service, so they search (to discover) services that are available. When they find one in the registry, they then can use dynamic binding to “discover” the end-point (location) of the service at runtime. Think of this as “googling” for a service. So far, so good. But moving forward, things get complicated because there is no way to discover any of the following:

- What services are in production? Just because a service does not appear in the registry doesn’t mean it isn’t in use.
- What services are being used? Administrators might see load on a system or an interface, but without SOA Governance tools there is no way to tell where messages are going.
- Who are the consumers of a service? Security controls and protects access, but you still have no way of knowing which consumers are using your service without the expense of auditing every transaction/message and parsing log-files (subject to errors and bad performance).

Figure 2: “Discovered” service network
Visual Visibility

With new tools, this scenario changes completely because they allow for true runtime discovery of both service providers and service consumers. They bring end-to-end visibility into what today’s complex composite applications are actually doing and maps out each and every dependency. Yet, with these tools, this type of discovery does not require manual configuration and correlation, but constantly updates by observing the flow of real messages. The point of automatic runtime discovery, after all, is to find what you don’t know is going on within your SOA.

When searching for services, the tools visually respond by providing a list of available services and other metadata about the service (e.g., policies, security requirements, business metrics, service-level agreements, etc.).

Notice the operation-level detail provided in Figure 2. One of the challenges faced by organizations as they manage their SOA lifecycle (via versions and revisions) is that typical registry products provide only service-level visibility. The operational level richness provided dramatically increases usability.

Information learned through the governance process can also be easily shared.
with open registry or repository products for a consolidated approach to service metadata management. Imagine a search being done on a service returning actual performance statistics or current service levels. Developers will have better information, enabling a better decision-making process, increasing SOA returns across the board. Better information also aids in planning and operations, primarily in the cost savings associated with reuse. Other benefits include improved morale, lower support costs, and increased use of technology within the organization.

**Service Consumer Discovery**

Service-consumer governance is a big challenge because organizations have no way of knowing which consumers are using which services, and what SLAs they are receiving. In other words, organizations know whom they have allowed to use a service, but how do they know if unauthorized users are accessing a service? In the same way, how can they know if all critical business and security policies are being applied if they don’t know if the service or consumer even exists? The problem is the same with SLAs: there is no way to know how the service levels that customers are actually receiving compare to what they’ve been promised. SOA Governance tools solve these problems because they provide a way for IT to track and “bill” for those services in use. Moreover they provide insight into broad range of other problems connected with SOA Governance:

- If there are ten consumers of a service, how will the eleventh impact the other consumers?
- How much service capacity is available for new consumers wishing to access my service?
- A service response time averages one second (1s). Are my ten service consumers satisfied?
- I’ve been developing a service, and it has moved to production. I want to move my development server to a new project. Is anyone still using it?
- I’ve created a new service, but I’m not sure how useful it is. Who in the organization is using it and what are they using it for?
- I’ve developed a simple service, and it’s being used so much that I have no more capacity. But I don’t have budget to add capacity. How do I track and bill infrastructure and additional development to those using the service in a consistent and fair manner?

Of course, this can be looked at from the consumer perspective as well:

- I would like to use service X, but I’m not sure what response time it’s been delivering, and response time is critical to me. I know what the service provider says, but is that really the performance I’ll get?
Provider and Consumer Discovery

- I’m using a service from another part of the organization, which wants me to contribute to its budget. I know others are using the service for free, so why should I pay?
- How well has service provider A delivered on the SLAs that it has promised others? Can I trust its planning abilities?
- Group G has a less-than-desirable reliability history, and I know it is using service X. How will group G failures affect my performance?

Figure 4. Integrating runtime governance with a registry and design-time governance

Registry Integration
A registry (or repository) is often used as the central index of service artefacts for architectural (design-time) governance. Fully documented SOAP API, with full security and tiered administration, along with deeper integration with other specific registries, support extensibility to any open registry product. The general architecture of registry integration follows in Fig. 4.
Sharing information about service relations and dependencies

There are many obvious artefacts to share with a registry: for example, owner, location, security, and policy requirements. New SOA Governance tools also uniquely share service interrelationship (dependency) information based on its patent-pending Flow Mapping™ technology. Dependency information is critical on a daily basis for performing root cause analysis, for capacity planning for upgrades, for versioning services, and for scheduling maintenance windows. Flow mapping can even be used to track business processes, and with support for asynchronous messaging, policies can be triggered when things happen (events) or when they fail to happen (non-events).

A flow map is essentially an application topology map that indicates where message traffic is flowing through the network. Service interrelationships are automatically discovered and never need to be configured manually. In addition, notice the gray systems in Figure 5. These never had SOA Governance software installed, yet the SOA Governance tools can manage one node away from any node with an agent installed. Finally, even traffic lines represent information. In the case above, they show the relative traffic volume. For example, we can see that the path between OrderMgmt and DataCenterGW to logistics is heavily used.

Figure 5: A typical service flow map
Flow Mapping Value
Flow mapping is important because of the way policies are applied. Policies take advantage of the flow mapping capabilities to know when there are downstream problems that relate to service-level commitments upstream. For example, the CSR portal, communicating to the enterprise service bus via the CustomerGW, may have a response time SLA of one second (1s). Although the CustomerGW is functioning fine, perhaps the Logistics system on the backend is not. If the management system is aware of the dependency between the two and how the SLA relates to them and can generate an alert to help significantly reduce downtime and service-level violations.

Keep in mind that the SLAs are path- and process-dependent. Think of a service being used by two different consumers. In Figure 5 above, CSR Portal and Domestic Customer each use the Inventory Management service slightly differently. A CSR Portal user’s transaction goes through the CustomerGW, then Order Management, and then to Inventory Management. But a domestic customer’s transaction travels through the CustomerGW and then directly to the Inventory Management service.

The average responsiveness of the Inventory Management service is not nearly as important as the responsiveness for each type of user or the exact path travelled, which in this example is not the same. For this exact reason, the flow map and auto-discovery of the path and dependencies are all key for governance and policies based on user or transaction types.
Other SOA Runtime Governance Elements

**Rogue Service Elimination**
A rogue service is a service put into the network without any governance visibility. A rogue service adds significant risk to the viability of the SOA infrastructure. For example:

- A rogue service could expose sensitive data, thereby putting the company at risk from non-compliance with regulations and laws. Often, compliance with regulations such as HIPAA and Sarbanes-Oxley and privacy laws are explicitly runtime requirements.
- Rogue services use capacity without any accountability.
- Rogue services act under the radar of corporate compliance by circumventing the governance system and process.
- Rogue services decrease motivation for complying with the governance policies because rogue services cannot be policed.

Rogue service detection would ideally run automatically using a detection function.

**Figure 6: Rogue Service Control**
The process relies on data from registries which determines the legitimacy of the service. SOA Runtime Governance tools are then able to eliminate the service by isolating it from the environment pending the policing of the rogue service.

**Security Policies**

SOA Governance tools can provide firms with the ability to automatically initiate policy without tying policy to a particular service, eliminating the motivation to evade compliance. Once in place, the policy can be applied broadly, to all services across the network—even those that have not yet been implemented. For example, say a rogue service is discovered. The security policy of “customer data must be encrypted” is immediately and automatically applied to the rogue service, thereby protecting the company and the customer. The power of this is that anyone deploying a service will automatically inherit a base-line governance framework to which it must comply. And compliance will not be an afterthought, but will be present from development, through user acceptance testing, into production.

**Business Process Visibility**

SOA Runtime Governance tools help organizations make their business processes visible—as individual processes or the infrastructure as a whole—depending on the specific business or technological criteria. Organizations are able to identify every business process running in the service network, identify the underlying infrastructure and create process maps of message flows. Rules and policies may then be applied to the processes identified in this way. A special non-invasive software for application servers, messaging brokers, etc., are used for the business process tracking. This software runs on individual nodes or data sources and taps into and captures event information. The information is then consolidated and evaluated in a special server, and used for governance in the production environment and for both proactive and reactive evaluation of individual events in real time.

When an organization is able to define and see all components, events and systems involved in a given process, it is able to predict what should happen in the system and prevent any irregularities. It will immediately know if the process strays off track or when there is a critical event. As a result, the organization’s capability to govern and manage, ensure compliance and monitor SLAs will improve significantly.

SOA can be monitored and governed both from the IT and the business level; the IT department can track the relationships between services, the transac-
Other SOA Runtime Governance Elements

Traffic and message flows. Different business lines then can view the SOA traffic from their business perspective—for instance by individual customer segments or by order volume. Each business line may have different SLAs, use other services and be eligible for different standard of care. SOA Runtime Governance tools allow for a detailed view of individual customers and the actual speed or response time of their services. In the event of an SLA infringement, there is a warning and the system...
determines which service or relationship is liable for the infringement (where the delays are occurring and why, for instance).

**Policy Enforcement**
Visualization and control of business processes allows for the enforcement of business and IT policies, which include namely the rules for statutory compliance, service user and consumer rights for accessing the system and using its services and processes, security rules and SLAs.

Policies usually change regardless of the application development, deployment and upgrades. For this reason it is recommended to create an environment, which allows for independent implementation and enforcement of policies. Quality runtime governance tools provide just the functions for this purpose. Visualisation and control of business processes makes it easy to obtain policy information and allows for real-time changes to the policies.

Consistent application of SOA Runtime Governance principles effectively improves the capability of the corporate SOA system for responding dynamically to business opportunities and IT issues in the live environment, which reflects positively on the performance of the whole organization.
Glossary of Terms

**SOA Runtime Governance**
A set of activities spanning the governance, management, reporting and visualization of Service Oriented Architecture in a production environment. A subset of IT governance.

**Policies**
A set of data and metadata about a service during its life cycle—from design through development, testing, deployment, security, maintenance and further development, to monitoring, evaluation and visualization.

**Rogue Service**
Any service that has not progressed through the defined life cycle.

**Repository**
In addition to the service definition itself, registries and repositories store all the related metadata, i.e. all descriptions from WSDL through to XML schemas, the process model or project documentation. They may also contain various histories (service versions, various operations journals or logs), identity and role data and other information). Unlike repositories, registries also run a rule and policy engine for their enforcement.
Resources

Internet Sources
• http://www.progress.com/progress_software/products/docs/socially-oriented-architecture-ebook.pdf—guide through the socially oriented architecture
• http://www.sonicsoftware.com/solutions/soa_enterprise/service_oriented_architecture/soa_maturity_model/index.ssp—SOA maturity model
• http://www.enterpriseintegrationpatterns.com/—integration patterns

Analyses
• IDC Opinion, Western European Market Preliminary Forecast: Services Automation and Management for SOA, 2006–2010
• CurrentAnalysis, B.F.Shimmin, The Changing Role of Service-Oriented Architecture, 10/2007
• The Forrester Wave: Standalone SOA And WS Management Solutions, 10/2007
• Gartner Magic Quadrant for Integrated SOA Governance Technology Sets, 11/2007
• Gartner – J.Thompson, Management Update, Predicts 2006: The Strategic Impact of SOA Broadens
• Burton Group, A.T.Manes, Service-Oriented Architecture: Developing the Enterprise Roadmap, 06/2006

Progress Software Materials
• Datasheets for Sonic ESB, Actional and DataXtend Semantic Integrator

Books
• Thomas Erl, Service-Oriented Architecture, Prentice Hall, ISBN 0–13–185858–0

Contact
Jindřich Štumpf, Business Consultant
Progress Software Czech Republic, Michelská 60/300, 140 00 Praha 4
tel.: +420 241 095 223
e-mail: jindrich.stumpf@progress.com
Jindřich Štumpf is Business Consultant for the Czech Office of Progress Software, the leading provider of service integration and SOA Runtime Governance technology. He has many years of experience in service-oriented project development and operation in various Czech Republic/Slovakia organizations and abroad. He regularly publishes articles on the current developments in the SOA/SOI field. He lectures at local and international conferences and occasionally also at the IT Department of the School of Economics, Prague. He now specialises mainly in Service Oriented Integration, SOA Runtime Governance and data integration managed by a common data model.