An Oracle White Paper
April 2010

An Agentless Approach to Runtime Governance: Leveraging the Platform to Enforce Policies

ORACLE

Service-Oriented Architecture Runtime Governance Overview

The principles of a service-oriented architecture (SOA) call for active governance of SOA-based systems at runtime to ensure the meeting of quality of service expectations. To accomplish this, organizations need runtime visibility across a heterogeneous network of services and must be able to maintain control as their networks evolve. Without adequate runtime governance, a SOA-based system is unable to deliver the desired results.

In the Gartner SOA Business Case Framework, industry analyst firm Gartner recommends a policy-based approach to runtime governance. In the context of SOA, Gartner defines *policy* as a set of guidelines, rules, regulations, or requirements to be enforced on services. Policies capture the declarative specification of as many characteristics of the system as possible. They represent a range of system characteristics—from the business process and specific business rules that define the functional behavior of the system to security, performance, and robustness requirements to configuration settings for the infrastructure on which the system executes.

The policies pertaining to the runtime behavior of a SOA system are runtime policies. Gartner cites many examples of runtime policy: security policies, such as access and authentication; management policies, such as performance, monitoring, and availability; routing policies, such as content-based routing; and transformation policies, based on document types or partner profiles.

Runtime Policy

The process of declaring runtime policy and enforcing it consistently across various constituents of a SOA network is daunting. However, a runtime governance system offloads this responsibility from an organization and dramatically reduces the level of effort and cost required.

Policy Definition, Enforcement, and Execution

A runtime governance system allows organizations to establish policies that govern the quality of a SOA system at runtime and executes all the activities necessary to enforce these policies. Architecturally, you can think of runtime governance as comprising three distinct phases:

- Policy definition
- Policy enforcement
- Policy execution

To use a simple illustration, policy definition is similar to creating automobile traffic laws. In this scenario, policy enforcement is analogous to the role played by the police as they ensure that drivers adhere to these traffic laws. It is the drivers themselves who execute traffic laws, making sure to stop at red lights, stay under the speed limit, and stop for pedestrians.

It is important to understand these concepts because they are often confused: policy enforcement and policy execution are often mistaken for one another.

Policy Definition

Policy definition is the process of capturing policies that express the runtime constraints enterprises would like to impose on their running SOA systems. The definition of runtime policies takes place throughout the SOA lifecycle, as a service progresses from design to development, quality assurance (QA), staging, and production. Policies are defined using tools available within the runtime governance system as well as those within other policy definition integrated development environments. The runtime governance system collects policies from various sources and manages these policies in a single facility, thereby greatly simplifying policy administration for the enterprise.

Policy Enforcement

As soon as a policy has been bound to a service, the runtime governance system begins the ongoing activity of policy enforcement. The system must ensure compliance with the constraints specified by the runtime policies. To accomplish this task, the governance system distributes the policy to the infrastructure that can implement the runtime constraints, and then monitors the policy execution for successes and failures. It decides what policies to apply, how to apply them, and how to execute them. It gathers operational data generated by the policies, analyzes this data, and then reports the results to the end users. The runtime governance system must continuously evaluate the policy bindings and enforce them across the SOA landscape.

Policy Execution

Policy execution focuses on running policies in the most optimal manner. Examples of policy execution include load-balancing, encrypting, and routing messages. Policy execution might require some decision-making capabilities. The scope of the decision-making is normally restricted to the message executing the policy. Thus, you could think of policy execution–related decisions as being "local" to the service (which is the original target of the message).

The runtime governance system automates policy enforcement and execution and ensures uniform enforcement of runtime policies across the SOA landscape.

Runtime Governance Lifecycle and Architecture

It is important to understand that a runtime environment exists at each stage of the service lifecycle. Runtime is not equal to production, as is often assumed. As soon you code the business logic for a service, you "run" it, and that means runtime. Every stage of the service's lifecycle—development, QA, production, or maintenance—that hosts a running service has a service runtime environment associated with it. Runtime policies govern the behavior of a service in any of its runtime environments and at each stage of the lifecycle. The process of runtime governance executes in parallel to that of design time governance, not after. One does not supplant the other.

The SOA landscape has seen various architectures for runtime governance—from monolithic, broker-based architectures to distributed, agent-based approaches. The most successful architectures use a multitiered approach to runtime governance and comprise these main components:

- Tooling for defining runtime policy
- A collection of policy enforcement services that manage policy distribution, data collection, and data analysis
- A collection of policy execution points (PEPs), often a set of distributed software intermediaries or agents that can execute runtime policy
- A repository for collecting and storing runtime metadata and operational information

This multitiered architecture aligns well with the distributed nature of SOA itself. It scales and federates across large networks of services. However, the rapidly evolving SOA stack presents a unique opportunity to further optimize the process and architecture for runtime governance.

Of particular interest in the SOA stack is the runtime infrastructure for SOA development and deployment, where the greatest evolution of capabilities relevant to runtime governance is taking place.

Policy-Capable Service-Oriented Architecture Infrastructure

Traditionally only capable of hosting and executing the service logic, SOA development and deployment platforms are becoming increasingly powerful PEPs for SOA runtime policies. For example, the newest versions of application server platforms such as Oracle WebLogic, IBM WebSphere and Microsoft Windows Communication Foundation all support the execution of certain

dialects of security and reliable messaging policies. Enterprise service bus technology excels in implementing data integration policies, such as the routing and transformation of XML messages.

That means various layers of SOA runtime infrastructure can now have embedded runtime policy execution capabilities—not just in software agents supplied by the runtime governance system. Infrastructure that can natively execute some form of SOA runtime policies is "policy capable." Not only is SOA infrastructure becoming policy capable, but various products are also beginning to specialize in the execution of particular types of runtime policy. For example, initiatives such as Cisco's Application-Oriented Networking and technologies such as XML firewalls, security, and integration appliances seek to embed policy execution capabilities into the physical network. They specialize in policies that are executable at accelerated speeds using hardware; XML processing and XML security are primary examples of this. Due to specialization and focus, policy-capable SOA runtime infrastructure can be more efficient at executing the policies they support than any third-party technology. Not surprisingly, the ability to execute policy has quickly become a significant competitive advantage for SOA runtime infrastructure technologies.

Agentless Runtime Governance

So what does policy-capable infrastructure mean to runtime governance? Is a policy-capable infrastructure the same as a runtime governance system? Not quite.

Because runtime infrastructure can natively execute runtime policies, its intrinsic capabilities can enforce runtime governance, rather than employing a (separate) layer of PEPs. Oracle, the industry's leading provider of SOA runtime governance software, recognized the opportunities presented by policy-capable infrastructure as it relates to the runtime governance of complete SOA systems. Oracle has an innovative agentless architecture for enforcing runtime policies to help organizations more-fully leverage their policy-capable infrastructure, as shown in Figure 1. (See the section titled "Oracle's Agentless Architecture for Runtime Policy Enforcement" for more on this).

Agentless runtime governance enforces runtime policy by leveraging as much of the native policy execution capability of the runtime infrastructure underlying a SOA as possible. It manages runtime visibility and control by delegating runtime policy execution to the SOA infrastructure whenever policy execution capabilities are available, rather than relying solely on its own agents.

This approach fundamentally decouples policy definition, enforcement, and execution—three distinct activities typically combined into a monolithic task by most runtime governance systems. Agentless runtime governance exploits this decoupling to provide enterprises with a mechanism to leverage their existing stacks for runtime governance. The runtime infrastructure is no longer an adjunct to the process of governance; it is now an integral component.

Agentless runtime governance brings many benefits, including the optimization of runtime governance, the ability to insulate policy enforcement from infrastructural decisions and changes, streamlined administration, and higher ROI.

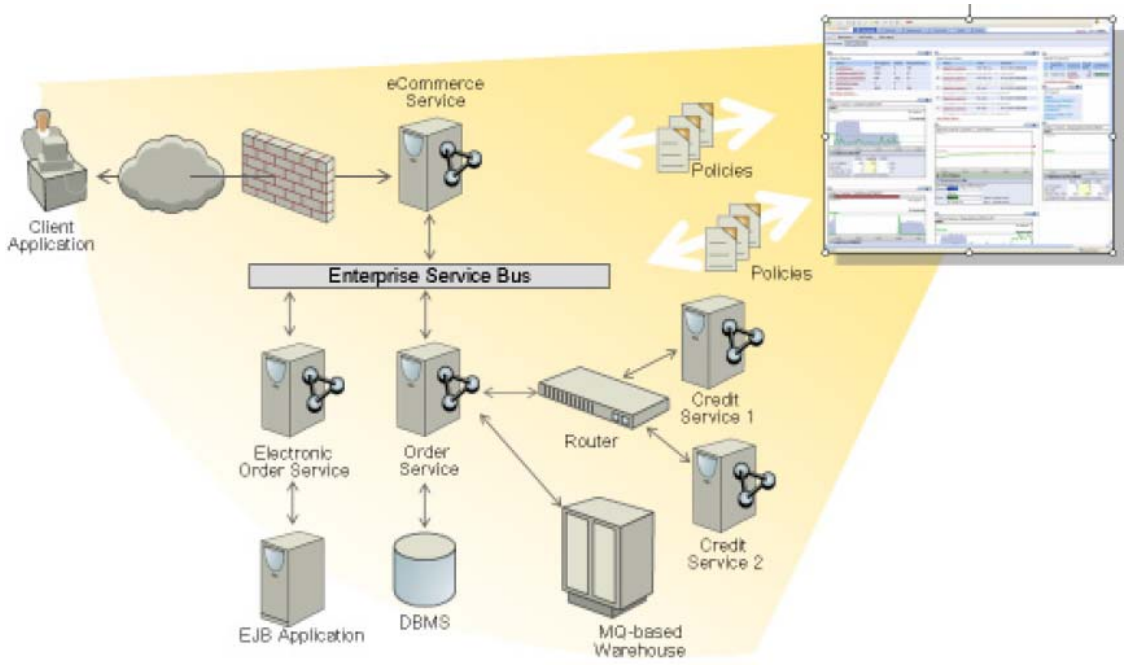

Figure 1. An agentless runtime governance system enforces runtime policies to help organizations more-fully leverage policy-capable infrastructures.

Leveraging Intrinsic Platform Capabilities

Policy-capable infrastructure specializes in the types of policies they execute using hardware acceleration, special algorithms, and other techniques. They can deliver substantial gains in performance and reliability of execution for each policy assigned to them.

Agentless runtime governance delegates the task of policy execution directly to the SOA runtime infrastructure. The governance system can, therefore, optimize the overall execution of policies in a SOA system by assigning policies intelligently, matching them to the intrinsic capabilities of each piece of infrastructure. This brings significant benefits to the SOA runtime governance and to the enterprise itself.

Business-Centric Policy Decisions

An agentless governance system gives enterprises the ability to make policy decisions independent of their runtime infrastructures. Organizations can focus on expressing business needs as governance policies, rather than allowing their selection of infrastructure components or the ongoing evolution of their IT environments to limit them. The runtime governance system manages the assignment of policies to various infrastructure components, taking into account the capabilities of each and using agents to fill in the gaps. Such infrastructure-independent policy decisions dramatically improve the overall manageability of the SOA environment.

Insulation from Infrastructure Changes

Agentless runtime governance also insulates the enterprise from changes to the underlying infrastructure. Because policies are decoupled from the infrastructure, the enterprise can be assured that runtime policies are enforced through the course of changes—and without additional manual effort. An organization does not have to recreate policies with each change to the system.

Moreover, the governance system can actually facilitate changes to the IT landscape by automating the rollout of new infrastructure within the enterprise. Existing policies are transparently provisioned to the new infrastructure (assuming the requisite policy capabilities match those available in the new infrastructure)—again without manual effort.

Such governance automation is invaluable, especially in light of the ever-evolving nature of IT environments in today's business climate, where mergers, acquisitions, and other business events drive frequent and unpredictable change.

Streamlined Administration of Runtime Policies

An agentless runtime governance system offers a holistic view of the runtime policies enforced across the infrastructure. The system's ability to enforce policies across a variety of runtime infrastructures allows an enterprise to get at-a-glance visibility into all existing policies. This provides valuable operational information, such as the status of each policy, the runtime infrastructure that is enforcing the policy, and any policy failures.

Such comprehensive insight augments the reuse of policies. Rather than continually devising new policies, enterprises can leverage existing policies, tweaking them to meet new needs and provisioning them to new infrastructure—all from a single vantage point.

To better understand this benefit, contrast this approach to one that requires inspection of each piece of infrastructure to know which policies it is executing. It is clear that a single, comprehensive view of runtime policies dramatically streamlines the administrative overhead and, therefore, makes SOA runtime governance much more efficient.

Increased Return on Investment from Infrastructure Investments

The agentless scheme allows enterprises to exploit their existing infrastructure in new ways. Policy-capable infrastructure becomes integral to runtime governance of the system, enabling organizations to gain more from their investments in this technology.

In addition, an enterprise more-directly benefits from the ongoing investments infrastructure vendors make in their platforms. If a new piece of infrastructure or an upgrade to an existing infrastructure comes into the environment, the agentless governance system will allow the enterprise to leverage that new investment to its full capability and in a seamless fashion. As a result, enterprises achieve much-higher overall returns on their SOA infrastructure investments.

Agentless Runtime Governance: Requirements, Challenges, and Issues

The agentless enforcement of runtime policies has set into motion the next wave of innovation in the SOA runtime governance marketplace. However, challenges abound when implementing an agentless governance scheme. The primary challenges arise from the reality of the runtime infrastructure stack:

- Each infrastructure has its own communication protocol, if any at all.
- Each infrastructure has its own policy dialect.
- Each infrastructure differs in its policy execution capabilities.
- Each infrastructure can be independently controlled (federation).
- Infrastructure change within an enterprise is inevitable.
- Business needs are very dynamic.

These practical realities make agentless runtime governance a difficult task. Insulating users from this complexity allows organizations to reap the rewards of this approach.

It is likely that other vendors will eventually adopt a model similar to Oracle's approach to agentless runtime governance. Therefore, it is important to get an in-depth understanding of the issues underlying a successful agentless architecture. Enterprises should evaluate their technology decisions against all these requirements.

Operate in Present-Day Environments

The agentless approach mandates that the runtime governance system communicates with a wide variety of SOA runtime infrastructures to distribute policy and collect the results of policy execution. However, each infrastructure differs in communication mechanisms, as well as policy dialects and data formats for exchanging information. Standards exist in some areas, but they are still evolving.

Consider, for example, the standards for expressing policy. The Web Services Policy Framework offers a uniform syntax for expressing policy. However, it provides only an envelope for describing policies in a standard form. The actual formats are still undefined. The Web Services Security Policy Language is the closest to a mature standard for representing some flavors of security policies, but it does not offer a mechanism to express all the security policy needs of an enterprise.

Unique capabilities in each runtime infrastructure also create additional, proprietary dialects. If standards do not provide a means of representing a policy that a platform can uniquely implement, the platform has no choice but to invent its own dialect. Even standards leave room for such extensibility. As a result, every runtime infrastructure expresses the same policy in a different manner. For example, the syntax of a load-balancing policy expressed in a particular appliance is very different from that expressed in an enterprise service bus.

Similarly, efforts such as OASIS Web Services Distributed Management and Web Services for Management, aimed at facilitating a standards-based dialog between governance systems and the

infrastructure, are still evolving. Most contemporary runtime infrastructures do not even have documented APIs to facilitate a digital dialog, let alone support for such standards-based dialogs. Under such conditions, it is fruitless to wait for standards to materialize and settle. It is imperative that the runtime governance system operate in a mixed environment. A more-pragmatic approach is necessary.

- The agentless system must be able to use infrastructure-specific mechanisms—be it by raw interfaces or by brute forcing the system to exchange policies with such infrastructure.
- The agentless system must be able to handle multiple descriptions for the same runtime policy, and the system must insulate end users from variations in policy syntax.

Avoid a Least Common Denominator

As previously discussed, not all runtime infrastructures are equal: they differ in their policy execution capabilities. Some infrastructures even specialize in certain types of policy execution, and the degree of specialization becomes their hallmarks. Therefore, the runtime governance system cannot make assumptions about the baseline capabilities and functions available in the infrastructure.

For example, application traffic management routers focus on delivering extremely efficient application-layer load balancing and failover. They provide these capabilities using purpose-built ASICs and other technology optimized for such routing scenarios.

However, they do not offer many capabilities for executing security-centric policies—for example, encryption or decryption. Other devices focus on such security aspects of SOA systems. Some use hardware to accelerate encryption, whereas others are software-based and tout price and flexibility advantages. As a result, there are many instances when specific runtime governance policies are not executable by a single piece of infrastructure.

Consider a versioning policy that requires message transformation and routing to the appropriate service endpoint. Load-balancing routers are often ill equipped for message transformation. A message transformation service is, therefore, required alongside the router to implement this scenario.

Runtime infrastructure components also differ in performance, the number of policies they can simultaneously execute, their robustness, and their resilience. The list goes on. Furthermore, their capabilities evolve as products mature. Customers request more functions, and in response, vendors introduce new features.

An agentless policy enforcement architecture must be able to support such diverse runtime infrastructures.

- It must not assume that any given infrastructure will always provide a common basic set of functions.
- It must be able to model the unique capabilities of each infrastructure and leverage them accordingly for policy execution.

Only after meeting these requirements will the runtime governance system be able to fully accommodate the diversity of infrastructure commonly found in enterprise runtime stacks.

Fire-and-Forget Policy Rollout

Business needs usually drive governance requirements. That means these requirements are very dynamic. An enterprise must be able to introduce policy changes to the system anytime it requires. The runtime governance system must immediately communicate these changes to the policy execution infrastructure.

However, it can be costly to assume that a piece of runtime infrastructure is up and available at all times, especially when rolling out policy changes. In the real world, SOA environments merge. Different groups and individuals are responsible for maintaining different portions of infrastructure.

System failures occur across the infrastructure, often as unrelated events. It is wrong to expect that all maintenance teams work in lockstep with the policy rollout process.

In such an environment, the governance system cannot assume that a policy execution point is available when dispatching a policy to that entity. If that particular piece of infrastructure is unavailable to receive the policy, the governance system must recognize this temporary failure in communication and attempt to submit the policy again later.

Of course, a relatively quick fix to this issue would be manual intervention into the policy rollout process. Yet, placing the burden of policy rollout on end users is an impractical approach, especially given the federated nature of these systems.

Therefore, it is critical that the agentless architecture insulate users against such dynamism.

- An agentless architecture for policy enforcement insulates the user against uncertainty about system uptime and guarantees the delivery of each policy to its appropriate execution point—in an automated fashion.
- Because various runtime infrastructures become available independent of each other, the governance system must track each component and facilitate the management of the SOA system as a whole.

Support Bidirectional Information Exchange

Federation within an enterprise also results in distributed control of the infrastructure. Enterprise runtime infrastructure is not typically owned or controlled by a single entity—an individual or an organization. Different individuals administer it, possibly across different organizations, especially if different groups leverage the same resources simultaneously.

Consider the example of a security appliance. A SOA architect might set a certain group of security policies when developing the SOA system. However, when the SOA system goes into production, a security officer might decide to adjust some of the policies to meet specific compliance regulations. The security officer might apply this change via the appliance's native user interface or via another governance system that can delegate policy execution to this appliance. Thus, different individuals via different governance systems manage the same runtime infrastructure.

An agentless policy enforcement architecture must allow for such federation. The runtime governance system cannot assume that it is the sole entity controlling any particular runtime infrastructure or the policies executed by that infrastructure.

To support this scenario, the runtime governance system must meet the following requirements:

- It must understand changes introduced into the runtime infrastructure via interaction through channels other than the governance system.
- It must reconcile these changes with the view of the world that has been described to it directly.

Therefore, the system must exchange information bidirectionally with the runtime infrastructure. Not only must it provision policy changes to the infrastructure, but it must also discover changes introduced by other entities participating in the federation—human beings or other systems.

Enable Uninterrupted Runtime Governance

Another truism is that an enterprise's infrastructure environment constantly evolves. Change comes from many sources—some technological, some business related, and some political.

Much of this discussion assumes that the infrastructure is still evolving. Even after maturity, the infrastructure continues to evolve, adding new features and capabilities over time. This drives change in the enterprise environment. As vendors continue to improve their products, enterprises upgrade their stacks in search of greater efficiency and improved performance. With each upgrade come new features and capabilities that the enterprise must leverage to gain ROI of time and expense in the upgrade.

Mergers and acquisitions are yet another source of change to an enterprise stack. The enterprise must be able to reap the benefits of the new infrastructure, while incorporating the new technologies into the environment. Again, the runtime governance system must allow enterprises to leverage these new investments fully.

Change can occur for purely political reasons as well. Federation within an enterprise drives the adoption of certain types of technology over others. It might also lead to the replacement of an existing stack. The agentless runtime governance system must consistently and transparently deliver its value in lieu of any change.

- The agentless system must loosely couple runtime policies with the infrastructure executing those policies. When the infrastructure executing a particular policy changes or is replaced, the governance system must enable the policies to be easily migrated to the replacing stack.
- The agentless system must use a dynamic approach to binding policies to runtime infrastructure. For example, if a router becomes equipped with encryption capabilities, the system must dynamically take into account this capability, while making decisions about policy provisioning to the runtime infrastructure.

Thus, an agentless policy enforcement architecture must accommodate enterprise change and evolution. It must continuously track and record the latest capabilities of the runtime environment, and

it must keep manual intervention to a minimum. Otherwise, the runtime governance system will become an impediment to the natural growth within an enterprise.

User-Friendly Runtime Governance

The issues covered so far combine to create yet another requirement, perhaps the most important, for the agentless runtime governance of SOA: simplicity. It must be simple for end users to manage, administer, and implement the runtime governance system.

It is necessary to insulate users from the complexity of the runtime environment. The lack of standards for policy exchange, the lack of standards for policy dialects, the need for a wide variety of policies, the dynamic nature of the environment—all need to be abstracted away to create an effective user experience.

- The agentless governance system should make it easy to express different kinds of policies, while preserving enough flexibility to cover the gamut of capabilities provided by various infrastructure components.
- Agentless governance should hide infrastructure-specific policy and connectivity details from the end user.
- The system should present policies as they relate to the user's view of their SOA—not the infrastructure's view.
- Users should be able to administer the system from a single vantage point.
- The system should automate as much of governance as possible.

Policy requirements are diverse. No single policy meets the needs of all enterprises, or even all groups within an enterprise; therefore, the system must handle a large number of policies. The agentless system must be able to account for and represent all runtime policy types executed by and pertinent to the governance problem at hand. Not only should the data model support such extensibility, but the user interface must also accommodate multiple types of policy.

Bringing such flexibility to the interface can create an arduous user experience. Hand editing a policy in a scheme as laborious as raw XML is the only option offered by some tools. In this case, the usability burden introduced by the governance system negates the benefits of leveraging the infrastructure.

There is also a need to abstract away the disparities in the runtime environment. As previously discussed, the lack of standards exacerbates the situation. Every policy-capable infrastructure available today uses its proprietary representation of any given policy. Similarly, various means connect infrastructure and exchange information.

This presents a significant challenge. Organizations require a system that is independent of policy dialects, yet intelligent enough to represent various policies and runtime infrastructures in a user-friendly fashion. This is beyond the scope of tools that call for hand-editing infrastructure-specific XML dialects.

It is also important that the infrastructure's view of the SOA be translatable into an intuitive mental picture for the user. Different infrastructure components use different concepts to represent SOA artifacts. A service might appear as a Web service in one system and as simply a network address (URL) in another. Some infrastructure components do not even recognize the SOA-specific artifacts of an IT environment. For example, load balancers might model machines, while ignoring the service containers hosted on those machine and the services hosted within these containers. This variance in data models must map to a uniform view, one that facilitates the user's understanding of the SOA environment. Before being presented to the user, non-SOA semantics must become standard SOA concepts.

The user interface should make it easy to administer and manipulate runtime policies across a heterogeneous infrastructure from a central vantage point. Without this capability, an end user will need to use the individual user interfaces of each infrastructure to manage the runtime environment. This approach is unlikely to scale.

The agentless governance system must keep human involvement to a minimum. The system cannot rely on humans to manually enter all the information necessary to transform the data models, policies, and semantics across various infrastructure components. Such an approach would be destined for failure. As much as possible, the system should learn from the environment on its own. The automated discovery of the runtime environment and its characteristics and capabilities is critical to the success of agentless runtime governance.

Oracle's Agentless Architecture for Runtime Policy Enforcement

As is clear from the long list of requirements previously discussed, agentless runtime governance is a complex effort. Built from the ground up, Oracle Business Transaction Management is an agentless runtime governance solution that meets these requirements. Oracle Business Transaction Management utilizes a policy-based approach to governance in order to manage the health and well-being of SOA applications. Furthermore, Oracle uses SOA to deliver SOA runtime governance.

Oracle Business Transaction Management is a feature of the following products:

- Oracle Management Pack for WebLogic Server, Enterprise Edition
- Oracle Management Pack for SOA, Enterprise Edition
- Oracle Diagnostics Pack Plus for Non Oracle Middleware.

Leveraging Policy-Capable Infrastructure to Enforce Runtime Policy

An extensible, loosely coupled policy definition scheme and a flexible data model that accurately captures the policy execution capabilities of the runtime environment are at the core of Oracle's agentless architecture. In addition, there is a powerful portfolio of policy enforcement services that intelligently enforce policies using a policy execution layer—including policy-capable infrastructure and Oracle's policy execution agents.

Oracle runtime governance allows organizations to achieve better end-to-end control of services-based applications by eliminating ad hoc runtime policy definitions as well as policy definitions tightly

coupled with any specific infrastructure. Using extensible policy templates, Oracle reduces costs by minimizing the time and skill required to build and set new policies. Policies are reusable across the SOA environment and authored without being bound to any one piece of infrastructure. Enterprise SOA architects, developers, and managers can easily understand infrastructure-specific concepts rationalized into SOA-specific models. This allows enterprises to focus on the runtime governance policies themselves, rather than concerning themselves with the underlying technology.

One of the primary constituents of the policy enforcement portfolio of services is an industrial-strength policy provisioning service, which is responsible for selecting the appropriate policy-capable infrastructure given a runtime policy, and then submitting the policy to the selected entity.

Oracle's runtime data model describes the services in a SOA and their supporting infrastructure. It automatically catalogs the policy capabilities of the runtime infrastructure using a combination of discovery techniques as well as user-directed inputs. This wealth of metadata about the infrastructure matches policies with their target execution points.

For example, three infrastructure elements might support a service (see Figure 2):

- Application server on which the service is hosted
- Security appliance that delivers requests to the service
- Agent that intercepts requests addressed to the service

.

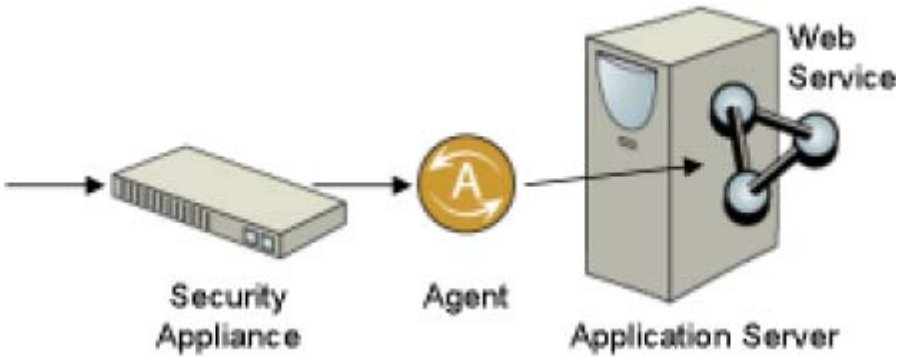

Figure 2. There are three possible supporting infrastructure elements of a SOA runtime infrastructure.

In the Figure 2 example, it is likely that all three infrastructure elements could implement a security policy. The policy provisioning service is responsible for determining the optimal infrastructure for implementing a policy. The provisioning system uses the metadata captured by the runtime data model to determine the best infrastructure for hosting each policy. It then provisions that infrastructure element to execute the runtime policy.

While the policy execution takes place, Oracle's policy enforcement services assume the role of a compliance-monitoring agency for these policies. They provide a rich layer of analytics, collecting operational information from various runtime sources and processing them in real time to understand whether the SOA complies with the various runtime policies mandated by the enterprise.

Continuous monitoring of the environment and enforcement of runtime policies ensure that service levels are always met, errors are quickly identified and resolved, security is tightly controlled, and any change in one portion of the environment doesn't disrupt dependent services or consumers.

Oracle enforces runtime policies at each stage of the SOA lifecycle. The system allows users to tweak policies at each of these stages to meet the needs of the enterprise in the best way possible. Because the policy enforcement is infrastructure agnostic, users can choose which portions of the runtime infrastructure to use for the execution of various policies.

Oracle's runtime governance architecture ensures the delivery of policies to their target execution points and enforcement as required. The agentless enforcement services track the policy execution infrastructure for their availability. If any entity is unavailable when a policy change is introduced into the environment, then the system will automatically attempt a redelivery at a later time, and repeat this until the policy has been successfully submitted to the target.

Data-driven selection of the policy execution infrastructure ensures that the system can accommodate an evolving or dynamic SOA environment. As various pieces of infrastructure mature or replacement of infrastructure elements occurs, the system can reprovision the policy to optimize policy execution continually across the environment.

The user interface of Oracle Business Transaction Management ensures that the implementation of runtime governance can be easy and done in a straightforward way. A Web-based user experience allows enterprises to remotely manage their entire SOA and to do so from a central vantage point. The rich user experience does not sacrifice the architecture's underlying flexibility and extensibility.

Agentlessness Is a Roadmap

The SOA runtime infrastructure continues to evolve. While the infrastructure matures, most SOA policy execution environments will remain a combination of agents and agentless infrastructure. The proliferation of the agentless approach will be proportionate to the advancements in policy execution capabilities into the SOA runtime stack. The more the infrastructure becomes policy capable, the more the enterprises will benefit from an agentless architecture. In the meantime, policy execution agents will remain necessary to fill in the gaps in existing runtime infrastructure from a policy-execution standpoint. Over a period of time, enterprises will see an increasing portion of their runtime governance functionality implemented in an agentless manner.

Thus, the notion of agentlessness does not mean the demise of policy execution agents. Rather, agentlessness is a roadmap to the fully optimized SOA runtime governance of the future.

Conclusion

The runtime stack for SOA is rapidly evolving. It is becoming more powerful and more capable—so much so that it can now execute some of the runtime policies required for governing SOA.

In lieu of this transition, runtime governance systems must also evolve to fully leverage the native capabilities in each layer of SOA infrastructure. The runtime governance system must become

intelligent about distributing the workload between its built-in policy execution capabilities and those available natively in the SOA environment.

Agentless refers to a runtime policy enforcement architecture that leverages the underlying infrastructure to execute policies. Agentless runtime governance brings many benefits to an enterprise, including more-comprehensive governance and increased ROI from investments in SOA infrastructure.

However, as enterprises embrace this next generation of runtime governance, they must ensure that the architecture they choose meets the long list of requirements brought forth by this agentless approach. Oracle is leading the way in delivering against these requirements with its innovative approach to agentless runtime governance and policy enforcement.

Oracle is committed to developing practices and products that help protect the environment

An Agentless Approach to Runtime Governance: Leveraging the Platform to Enforce Policies
April 2010

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

SOFTWARE. HARDWARE. COMPLETE.