# Centralized Policy Administration, Enforcement, and Monitoring for Runtime Policies

## IBM Redbooks Solution Guide

The SOA Policy Solution provides centralized policy administration, enforcement, and monitoring for runtime policies that enable traffic management for service level agreement (SLA) enforcement, service mediation, and other customized policies.

Business users can employ the SOA Policy Solution to help create the SLAs for their business services to deliver on promises for business performance. IT architects can use the SOA Policy Solution to create policy solution patterns that will standardize the runtime policy usage at their organization. Developers select specific policy patterns to implement the nonfunctional requirements that are associated with their projects. Operations provide information about operation needs and create a standardized monitoring policy for operational action at run time as shown in the example in Figure 1.

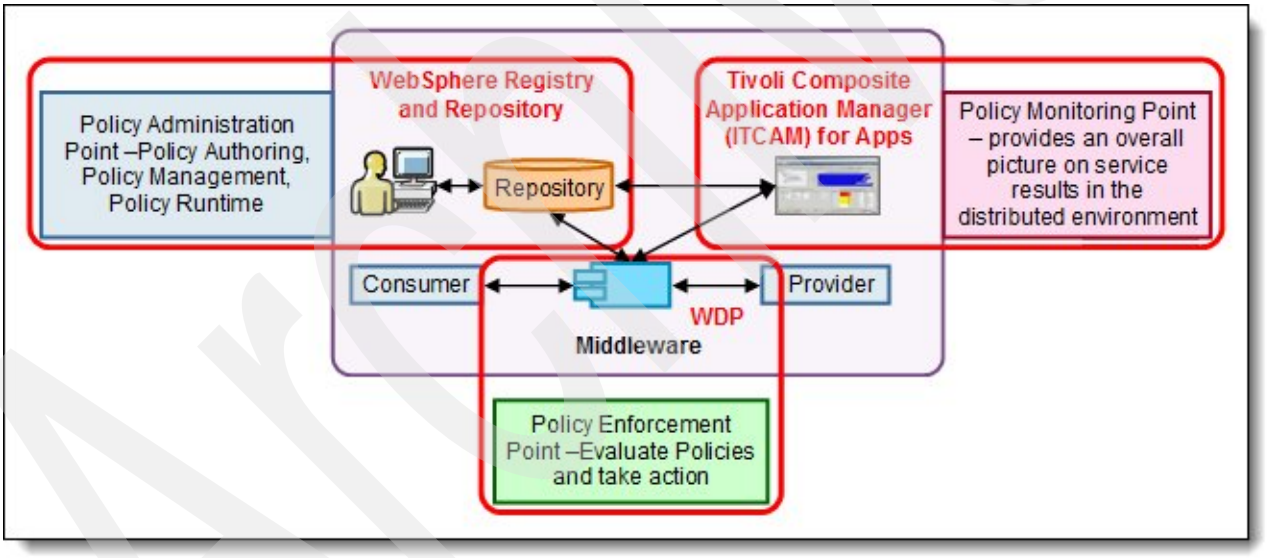


Figure 1. Example pattern for the SOA Policy Solution

You can define policies once and reuse them among multiple services, enabling a standardized, consistent approach to runtime policy. With this approach, you save time and money for implementation and maintenance of nonfunctional requirements for the enterprise and achieve faster time to market.

## Did you know?

Many organizations find that they can group their services into about a half dozen sets, where each set requires the same policies. A set of policies might include, for example, a couple of security policies, several traffic management policies, and a message transformation policy that all of the services in that set need applied to their transactions during runtime execution. By using the SOA Policy Solution, the process of managing multiple individual policies in the various middleware devices is condensed to a half dozen or so policy patterns in a centralized policy repository.

## Business value

All organizations implement policy in some form to make decisions that are important to the business and IT. For example, human decisions are a common method of implementing a policy. Your business people or finance group might ask you to prove that automating your policy decisions will provide a positive return on investment (ROI) for your organization. Proving to these decision makers that your project to automate a policy is a worthwhile endeavor always comes down to two factors:

- Saving money
- Driving incremental revenue

The business case for SOA Policy Solution largely depends on showing how its implementation will save money. The business case is derived from the premise that, by creating a set of reusable policy patterns, the organization will be able to develop and operate better, faster, and cheaper.

Specifically, three factors contribute to the business case for saving money by using policy: *dynamicity*, *reuse*, and *traceability.* For each factor, you must calculate the amount of time saved and then apply a financial factor to each.

**Dynamicity**

The ability to change policies in one place enables the changes to be implemented and deployed more quickly by IT and at lower cost with improved time to value. This capability is possible because the policy changes can be updated once instead of multiple times in multiple places as in the following examples:

- A business request to change the SLA for a specific group of customers to provide enhanced support
- A business request to change security password policies to increase security in a specific area
- An operations request to lock out a batch routine from updating a critical database during nonbusiness hours in response to a business request to improve response time

**Reuse**

Typically, there a finite grouping of policies needs to be applied across the services or resources in the runtime environment. For example, half a dozen reusable policy groupings are defined and reused among the services in an organization. After those policy groupings are defined, it is only a matter of identifying which one of those groupings to apply to that particular new service. Needless to say, the practicality of this approach saves time and money.

Using a formal set of policy domains and assertions reduces the ambiguity of the characterized policies and makes them readily available for automation. Using a standardized set of policies guarantees that two different systems are compliant with the same policy, regardless of their specific implementation. The formalization and standardization of policy domains, together with the recognition of these standards by tools, middleware, and management systems, allows the automatic and transparent configuration of these systems and the automatic enforcement of related policies.

**Traceability**

In a policy management system, traceability between and among policies and resources is an important source of information about the state of the real-time functioning of the systems. It is normally difficult to achieve this view if the policy is maintained in separate silos that do not integrate. Using a centralized policy management system overcomes this issue. It also provides a means of centralized auditing of policy actions, which is important from a regulatory and audit point of view.

Consider especially the impact of dynamicity on time to market of the business product and services that are affected. Normally, an organization should have a view as to the projects that will be funded during the next 12 months. To what degree can each product go into service faster by having a set of standardized, reusable policy groupings that exist and can be reused without development and testing of these policies? Even several weeks of faster time to market can drive significant incremental revenue.

## Solution overview

By using the SOA Policy Solution, you define policies in the SOA Policy Solution policy administration point (PAP). Then, you attach them to all of the services in the group by using a simple query. The attachment process is dynamic. As more services are created that fit the criteria of this group, they are automatically attached to this set of policies and downloaded to the policy enforcement points (PEPs) in the middleware.

The SOA Policy Solution combines IBM® WebSphere® Service Registry and Repository, WebSphere DataPower®, and IBM Tivoli® Composite Application Manager for Applications:

- *WebSphere Service Registry and Repository* is an enterprise-level registry and repository that provides scalable and automated capabilities to help organizations optimize productivity and resources in a services environment. WebSphere Service Registry and Repository performs centralized policy authoring, management, and governance within the solution.
- *WebSphere DataPower Integration Appliance XI52 (or XI50) or WebSphere DataPower Service Gateway XG45* is a purpose-built hardware platform. It delivers rapid data transformations for cloud and mobile applications, secured and scalable business integration, and edge of network security gateway in a single drop-in appliance. WebSphere DataPower performs transaction-by-transaction policy enforcement within the solution.

  WebSphere DataPower Service Gateway XG45 and WebSphere DataPower Integration Appliance XI52 are available as virtual editions to run in VMware hypervisor environments. These virtual editions provide industry-leading workload security, optimization, and integration functionality similar to the corresponding physical appliance models. Each WebSphere DataPower virtual edition appliance is powered by a purpose-built platform including an embedded, optimized DataPower Operating System.
- Tivoli Composite Application Manager for Applications offers integrated management tools for web and enterprise infrastructures to aid SOA lifecycle availability and performance. Tivoli Composite Application Manager performs service monitoring and operational monitoring control within the solution.

With the SOA Policy Solution, some initial configuration is required for WebSphere Service Registry and Repository, WebSphere DataPower, and Tivoli Composite Application Manager. After the configuration is complete, policy updates in WebSphere Service Registry and Repository result in an automatic notification to WebSphere DataPower and Tivoli Composite Application Manager to pull the policy from WebSphere Service Registry and Repository. WebSphere DataPower and Tivoli Composite Application Manager then convert the standard format WS-Policy into its own local policy format.

Figure 2 illustrates an overview of the SOA Policy Solution.

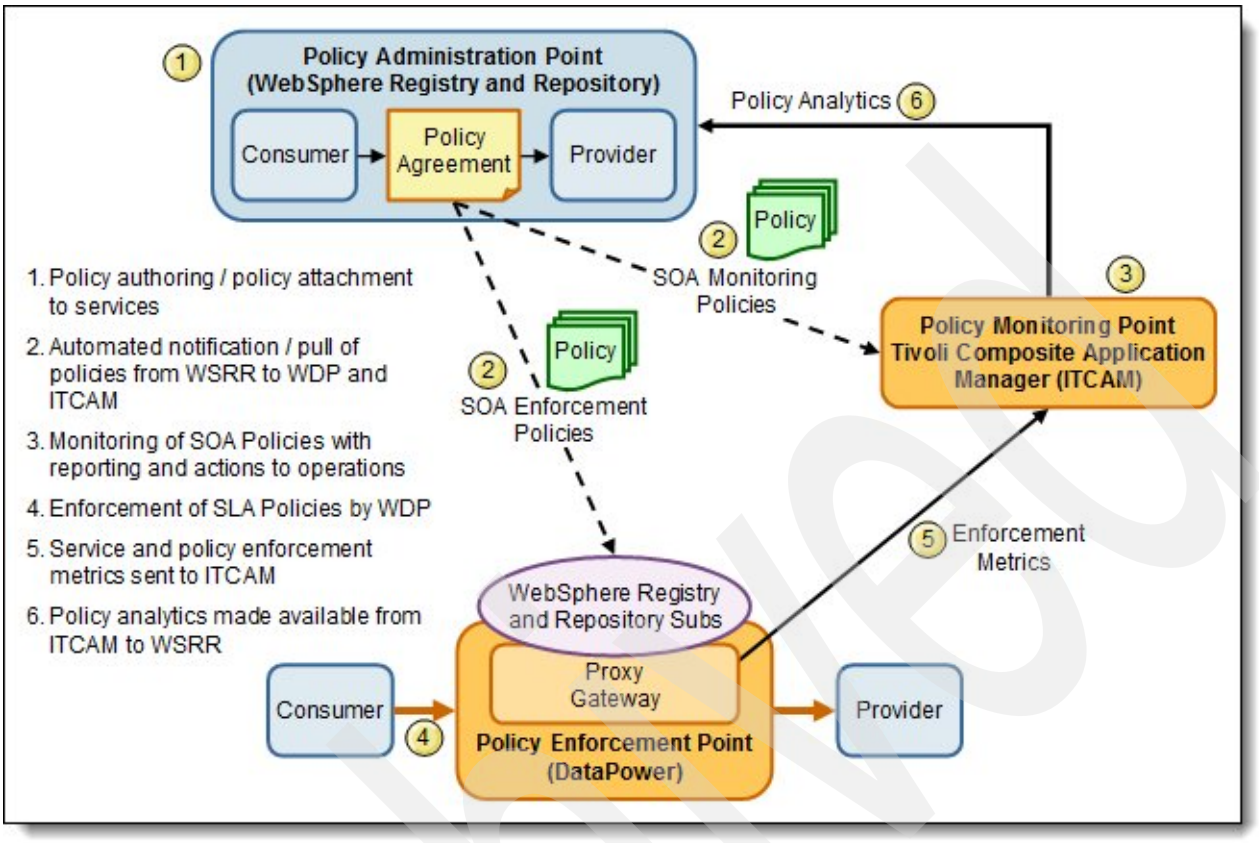


Figure 2. Overview of the SOA Policy Solution

The SOA Policy Solution process involves the following steps:

1. Policies are authored and then attached to services that need that a policy.
   a. Information about the services is loaded from external resources and then governed and managed in WebSphere Service Registry and Repository.
   b. The set of policies that is needed is created in WebSphere Service Registry and Repository or loaded from custom policies that are created in WebSphere DataPower.
   c. Policies are attached to the services that require those policies.
2. Policies are automatically published/subscribed from WebSphere Service Registry and Repository to WebSphere DataPower and Tivoli Composite Application Manager.
   a. Tivoli Composite Application Manager for SOA subscribes to the monitoring policy from WebSphere Service Registry and Repository (one time).
   b. Proxy gateways are created in each WebSphere DataPower appliance that has service transactions with policy enforcement (one time, added or changed as needed).
   c. Each proxy gateway subscribes to policies from WebSphere Service Registry and Repository for the services it is responsible for (one time, added or changed as needed).
   d. Optional: WebSphere DataPower is configured so that policies can be shared by other appliances in a cluster (one time, updated as needed).

e. Tivoli Composite Application Manager successfully downloads monitoring policies as they are published.

f. Tivoli Composite Application Manager converts the policy into an internal representation (situation policies).

g. Each WebSphere DataPower appliance can now automatically download service information (Web Services Description Language (WSDLs)) for the services that it is responsible for transacting.

h. WebSphere DataPower downloads policies for the services that it is responsible for upon notification from WebSphere Service Registry and Repository.

i. WebSphere DataPower converts the policies into internal WebSphere DataPower representation (service level management (SLM) objects).

3. SOA policies are monitored with reporting and notification of operations.
   a. Monitoring policies are active in the Tivoli Composite Application Manager Situation Policy.
   b. Tivoli Composite Application Manager receives monitoring information, periodically (default is 5 minutes) evaluates the monitoring (situation) policies, and takes action if the policies evaluate to *true*.
4. SOA policies for consumer-provider transactions are enforced.
   a. Enforcement policies are active in the various WebSphere DataPower Appliances.
   b. WebSphere DataPower receives service transactions and applies policies for a specific consumer-provider pair or for all transactions for a provider service.
5. WebSphere DataPower sends service and policy statistics to Tivoli Composite Application Manager.
6. Tivoli Composite Application Manager sends monitoring events to WebSphere Service Registry and Repository when monitoring (situation) policies that evaluate to *true*.
   a. Events are set up in WebSphere Service Registry and Repository that are desired to be monitored from Tivoli Composite Application Manager (a one-time action for each event).
   b. As situation policies evaluate to *true*, events are pushed to WebSphere Service Registry and Repository from Tivoli Composite Application Manager for display.

## Solution architecture

The power of the SOA Policy Solution lies in its capabilities as follow:

- Centralize and reuse runtime policies in one place (the PAP in the solution architecture)
- Have these policies automatically pulled by the appliances that need them for consumer-provider transactions (the PEP in the solution architecture)
- Have a monitoring capability for services (the policy monitoring point (PMP) in the solution architecture)

As shown in Figure 1, the SOA Policy Solution resembles a 3-legged stool, where the PAP, PEP, and PMP each have a set of specific duties and interact with each other in an automated fashion:

- A *policy administration point* provides the following policy capabilities that support the centralized administration of policy:
  - Creates, updates, deletes, and reads a policy
  - Manages and governs a policy
  - Assigns policies to one or more resources
  - Deploys policies to the PEP and PMP

- A *policy enforcement point* is a function component that provides enforcement of policies on a consumer-provider on a transaction by transaction basis:
    - Enforces policies
    - Receives and makes ready for usage (translates) enforcement policy updates
    - Provides service and policy enforcement metrics to the PMP
    - Sometimes uses a policy decision point (PDP), which evaluates participant requests against relevant policies/contracts and attributes to render an authorization, eligibility, or validation decision or provide calculated results
    - Sometimes uses a policy information point (PIP), which provides external information, such as results from a database with information that must be evaluated to make a policy decision
- A *policy monitoring point* is a functional component that provides an overall policy monitoring function (the big picture on services and policy in the distributed environment):
    - Receives and prepares (translates) monitoring policy updates for usage
    - Captures real-time collection and statistics on service usage for display
    - Correlates, analyzes, and visualizes the data that is fed in by the various real-time collectors including PEPs
    - Logs, aggregates, measures, and highlights significant events (as specified by the monitoring policy)
    - Provides monitoring policy analytics to the PAP

## Usage scenarios

The following runtime policy scenarios are typical of what the SOA Policy Solution can do and how a policy can help your business and IT.

### Standardized service level agreements

In this use case, to help meet IT's SLA promises to the business, a business must create standardized SLAs to automatically manage the traffic against a critical back-end service or provider service. Many of the policies that are implemented for the SOA Policy Solution are around implementation of the SLAs for a provider service regardless of who the consumers of that service are or how many there are. In this case, an SLA is created for the provider service and policies are put in place to provide the required service.

A key benefit of the SOA Policy Solution is the ability to create a set of standardized SLAs that can be applied to the service transactions in the runtime environment. The business identifies the nonfunctional requirements for their business functions, for example: "The credit authorization service must reply within 3 seconds." Ultimately, the architect must translate these requirements into one or more runtime policies that deliver on the business needs. A good way to analyze this policy is to create a *policy map* , which helps IT to translate the business SLAs into runtime SLAs that can be automated by a machine (Figure 3).

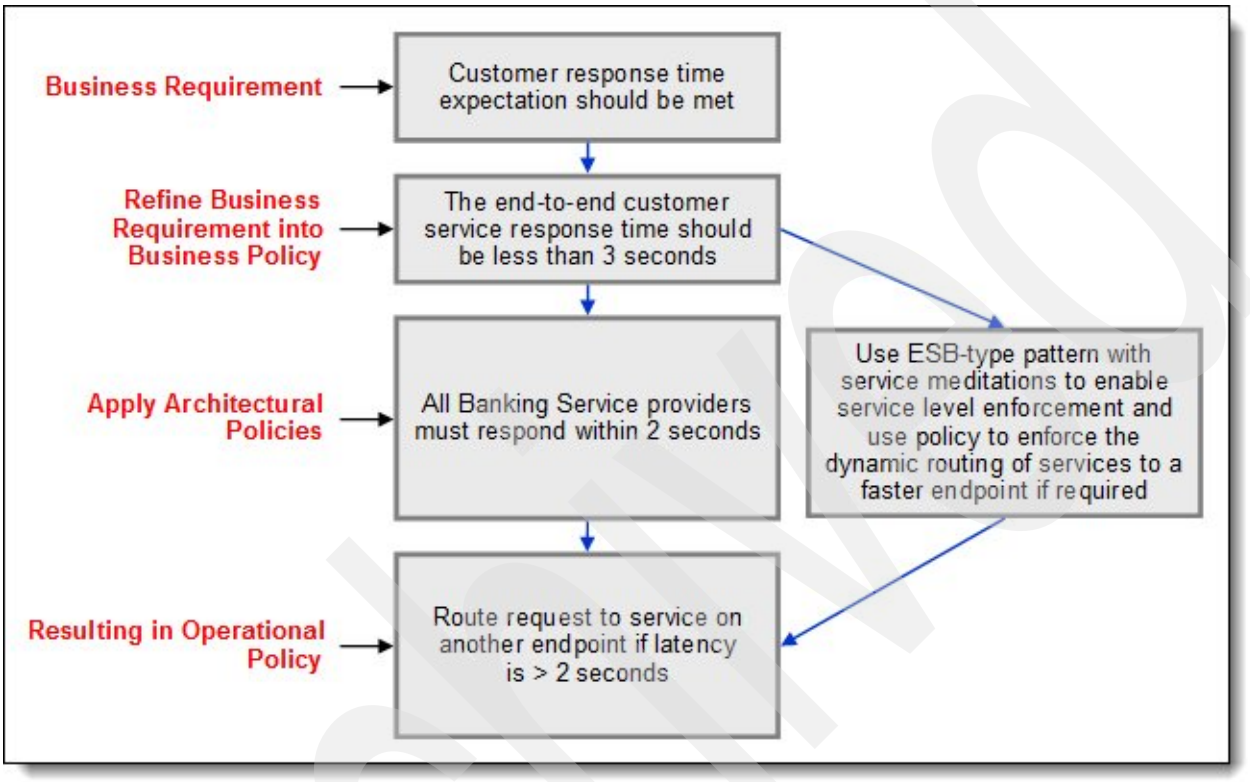


Figure 3. Creating standardized SLAs

In this case, a high-level business requirement is that a "customer response time expectation should be met". The obvious question for the analyst is: What is that response time? A conversation with the business and a customer survey elicits the answer of 3 seconds, as reflected in the refined business requirement. In terms of the overall flow, the architect determines that customer service must respond within 2 seconds or reroute the customer to a secondary endpoint. This determination is the runtime policy that is created.

Figure 4 illustrates the runtime policy actions that are available to automatically maintain the SLA that the business requires.

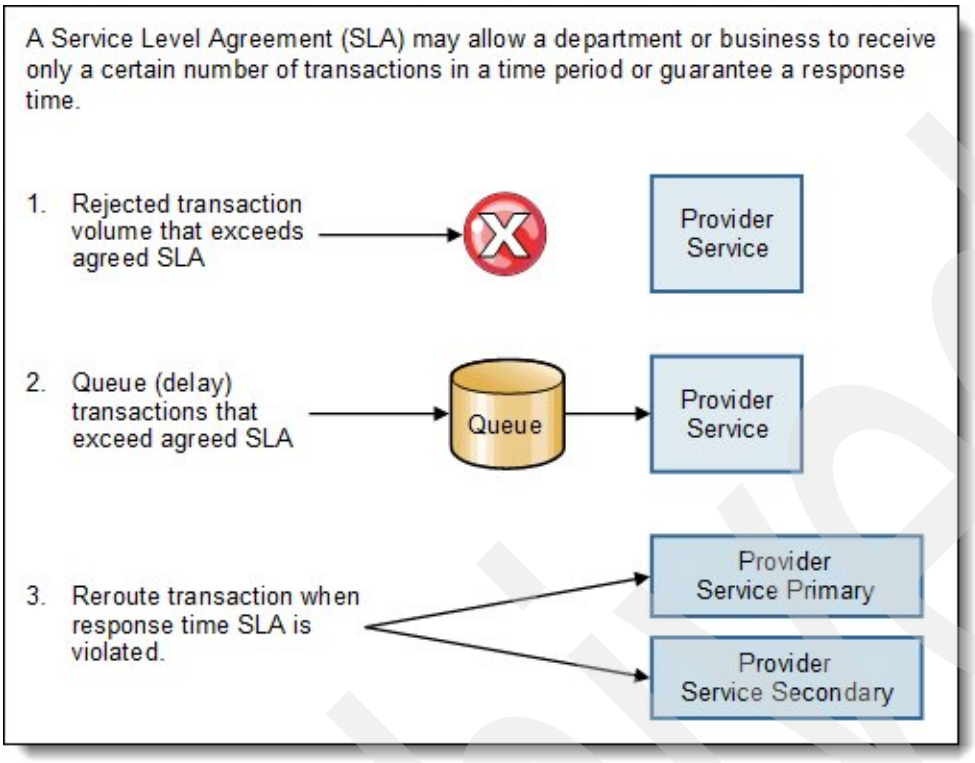


Figure 4. Traffic management SLAs

Rejection is for situations where low priority traffic (for example, batch jobs) or an agreed limit was violated that results in too many transactions for the provider (back office) service to handle. This command protects the provider service from being overwhelmed.

Queueing helps to smooth out the traffic swings from the consumers and provide a more even level of service by the providers.

Rerouting is useful in situations where traffic can be routed to an alternative provider service instance in situations where too much traffic is on the primary instance of the provider service. Rerouting helps to maintain the level of service on the primary instance and give an overall good level of service to the consumer.

### Differentiating service SLAs

In this use case, a business requires differentiating service SLAs depending on the type of consumer, as shown in Figure 5.

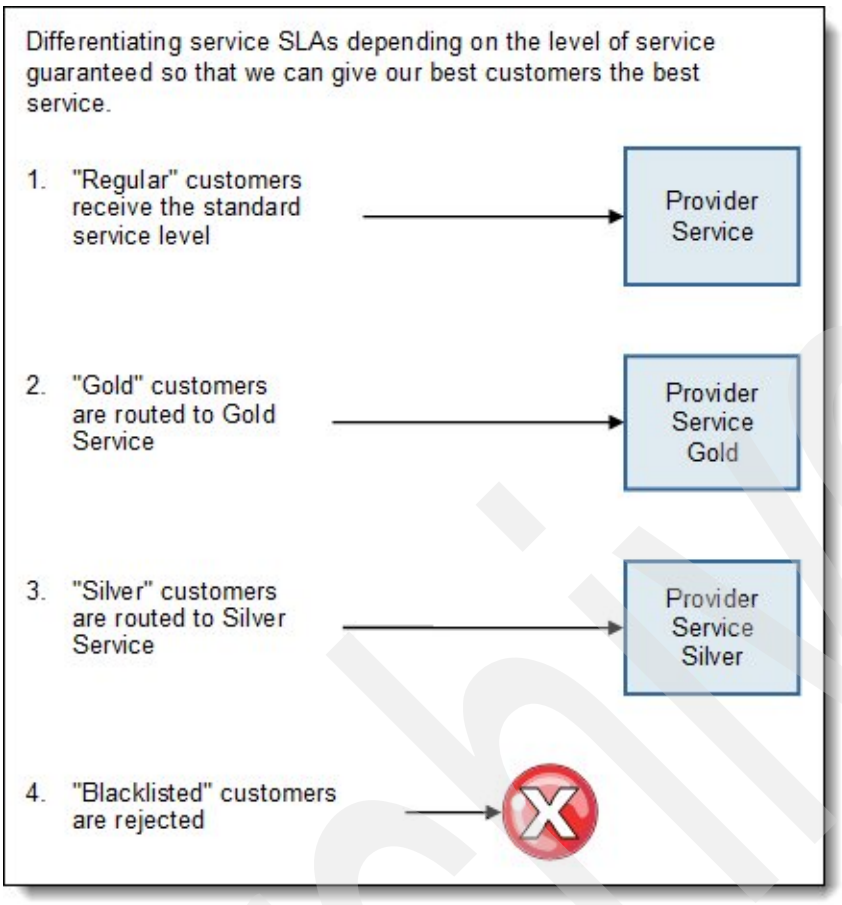


Figure 5. Differentiating service SLAs

Information about a customer can be passed in and used by the policy enforcement. A common use of such information is customer segmentation, where a different level of service is provided depending on the value of the customer. For example, you might want to give your premier gold customers excellent service response, your silver tier customers a good service response, and your regular customers a standard service response. Meanwhile, you might also reject requests from customers that you blacklisted so that their information is not presented to the back end provider service. The gold customers are routed to a lightly loaded server endpoint, silver customers are routed to a medium busy server endpoint, and regular customers are routed to a more heavily loaded endpoint (Figure 6).

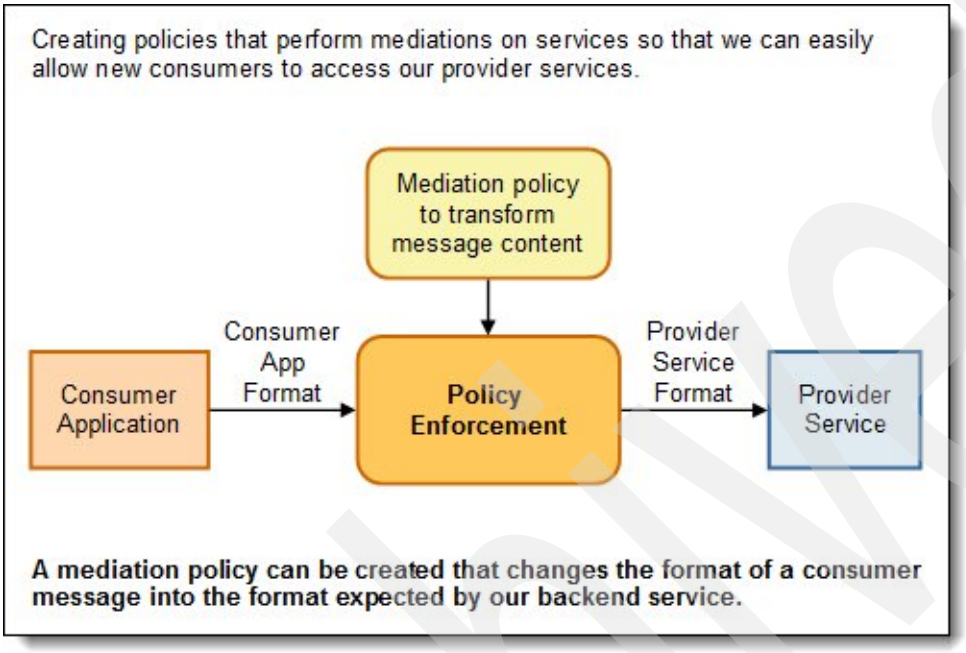


Figure 6. Allowing new customers access to provider services

In this case, because of a situation due possibly to new business opportunities or the acquisition of a new company, back-office provider services must be made available to new consumer applications. One way to ease this transition is to use a policy to transform the consumer application data format to a format that expected by the provider service. In this manner, the business can avoid expensive design, development, and testing in favor of using a policy.

### Rejecting low priority traffic during business hours

In this use case, a business requires the use of scheduling to reject low priority traffic during normal business hours, as shown in Figure 7.

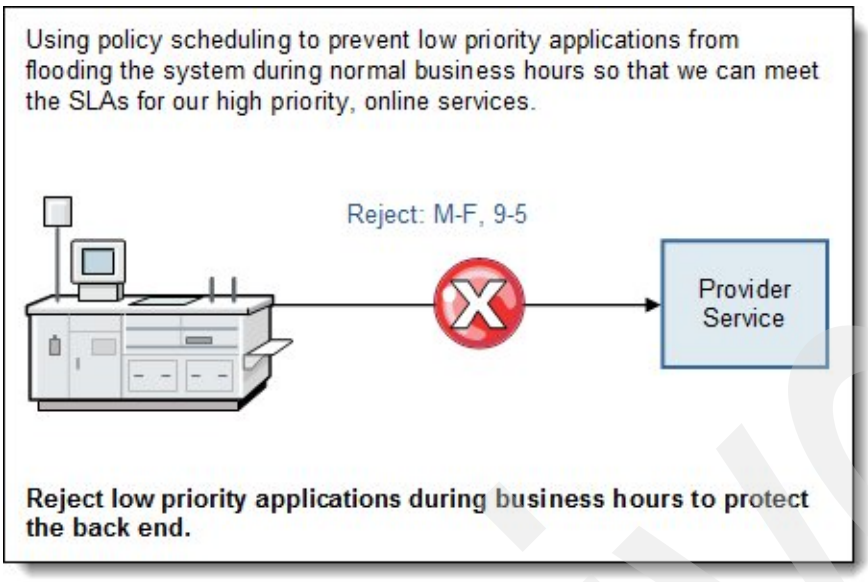


Figure 7. Rejecting low priority traffic during business hours

A policy can be scheduled by day of week and time of day. It can also be specified to start on a certain date, end on a certain date, or both. This feature is helpful for various use cases. For example, you can stop traffic from low priority transactions. For example, batch jobs might interfere with the normal processing of a provider service that has an SLA during normal business hours of 9 a.m. - 5 p.m. on Monday through Friday.

### Rerouting traffic during maintenance windows

In this use case, a business requires the rerouting of traffic during a maintenance window, as shown in Figure 8.

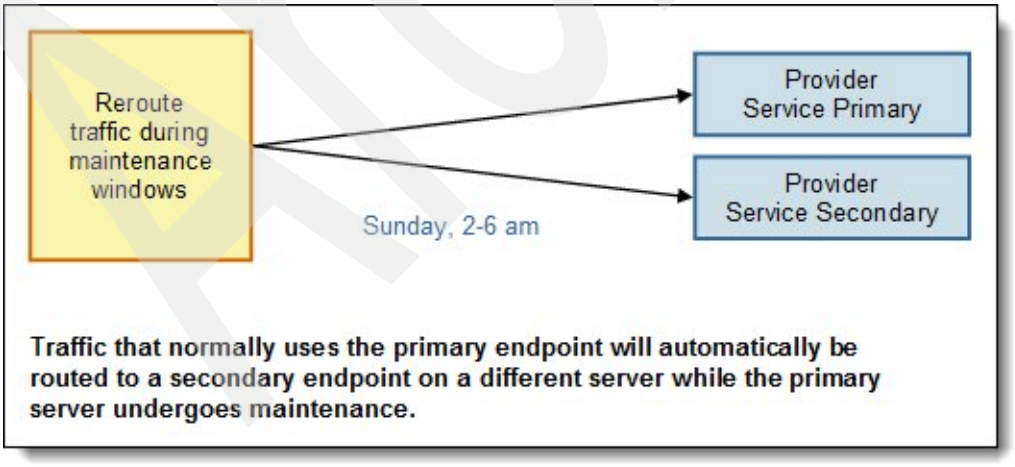


Figure 8. Rerouting traffic during maintenance windows

Another example of using the policy scheduling capability is to reroute traffic during a standard maintenance window. For example, traffic that normally routes to a primary server is instead routed to a secondary server, while the primary server undergoes maintenance.

### Denying access to rogue consumer applications

In this use case, the business requires denial of access to rogue (unknown) consumer applications, as shown in Figure 9.

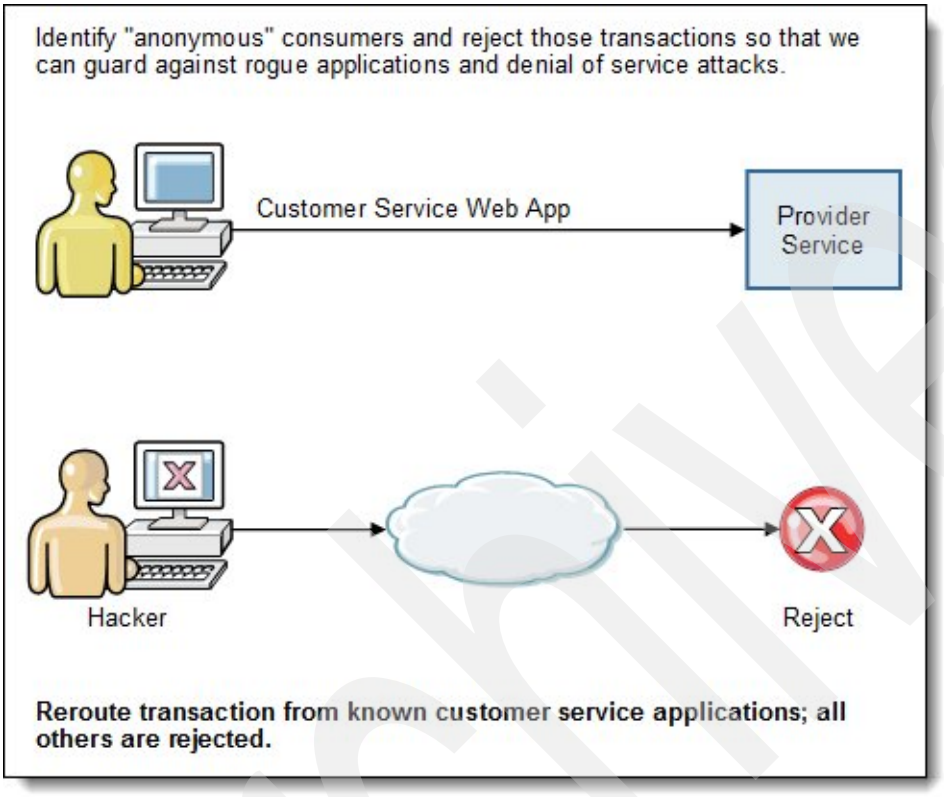


Figure 9. Denying access by rogue applications

Rogue applications are unknown to the provider service. In many cases, rogue applications are acceptable, and the provider service handles transactions regardless of the consumer it originates from. In other cases, access to the provider service might need to be locked down, which is a good way, for example, to enforce an SOA governance policy that only registered consumer applications can access the provider service. Another usage is to protect against denial-of-service attacks where hackers attempt to flood a provider service with so many transactions that it does not have time to process legitimate requests. By throttling such criminal requests with a policy, the back-end provider service is protected.

### Service versioning support

In this use case, a business requires support for deprecating old versions of a service while maintaining current consumer usage, as shown in Figure 10.

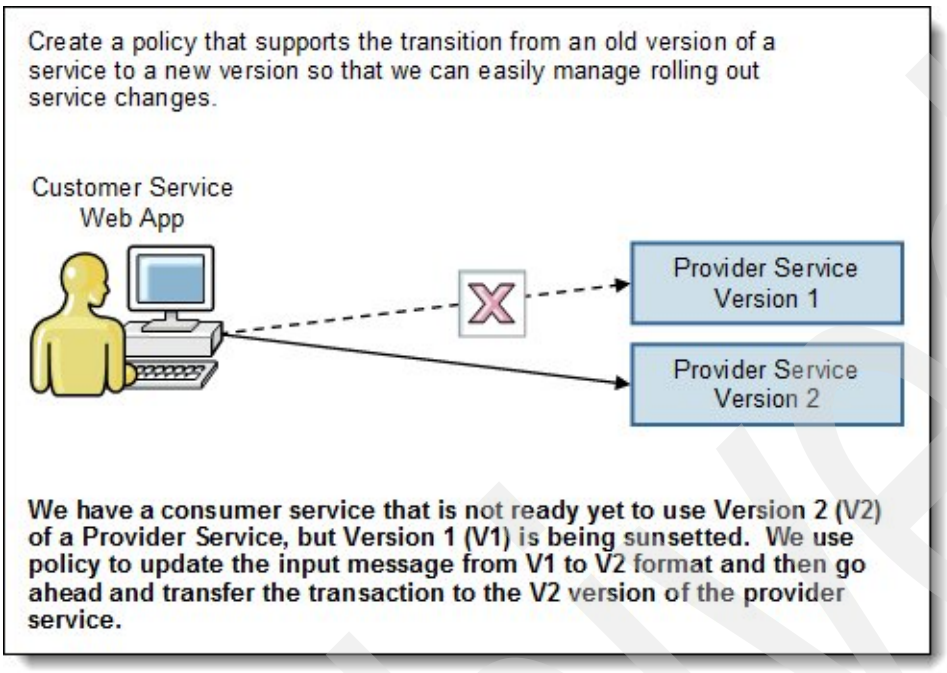


Figure 10. Service versioning support

Migrating user to a new version of a provider service can be a daunting activity if there are a large number of different consumer applications or services. Different departments might need to change, and it is not always possible to coordinate all of those changes to the same schedule. In this situation, you can use a policy to alleviate the migration issues. If the message format changed between Provider Version 1 and 2, a policy can be created to transform the incoming message to a format with default values for Version 2 of the provider service. The transaction can then be routed to Provider Service Version 2, and you can proceed with decommissioning (deprecating) the Provider Service Version 1.

### Applying standard security to access provider services

In this use case, a business requires applying a corporate standard security to all consumer to provider transactions, as shown in Figure 11.

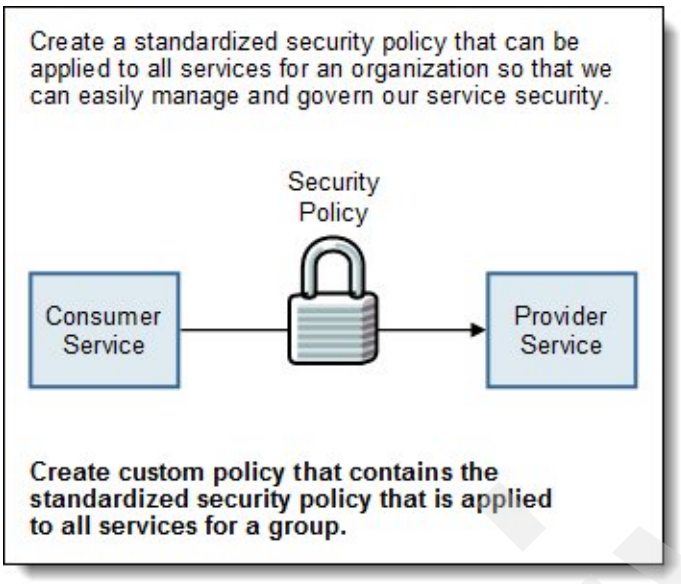


Figure 11. Standard security to all transactions

One powerful aspect of the SOA Policy Solution is the ability to create a custom policy. The custom policy can be anything that DataPower can do with policy, including the powerful security capabilities of the gateway. For example, a standard AAA policy for user ID and password validation can be specified by using a policy and then applied to all of the services for an organization. In this manner, you can easily and quickly apply a standard security policy.

### Providing operational status and alerts

In this use case, a business requires the use of a monitoring policy to create operational status and alerts, as shown in Figure 12.

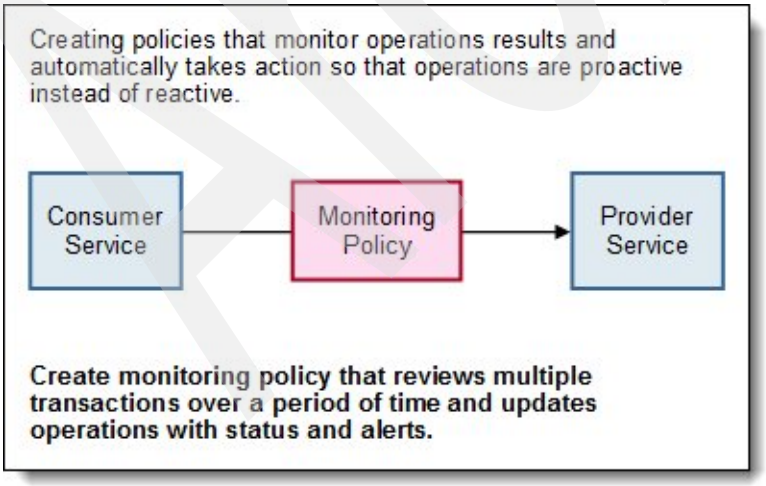


Figure 12. Operational status and alerts

A monitoring policy is used to perform traffic management and to review multiple transactions over a time, which is typically 5 minutes. In this manner, operations can be alerted when they need to review where traffic and transactions stand operationally or even to take action.

**Automatically applying a policy to new services**

In this use case, a business requires automatically applying an existing policy to new services as they are created, as shown in Figure 13.

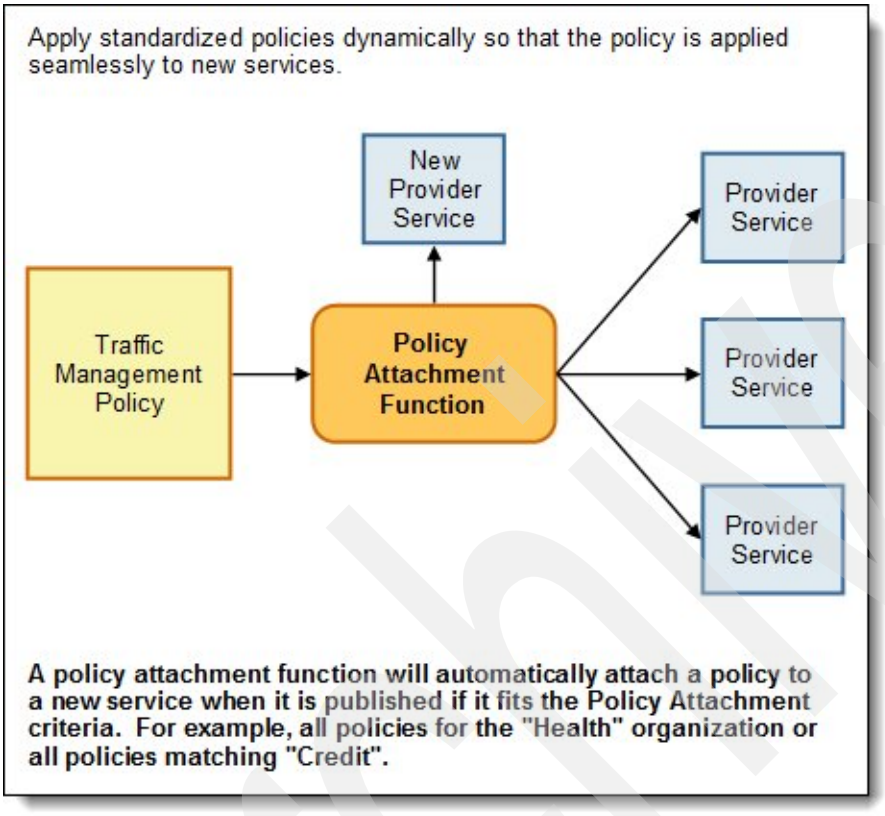


Figure 13. Automatically applying a policy to new services

A key feature of the SOA Policy Solution is the ability to create a policy attachment function query identifying all of the services that should be subject to (attached to) a policy. This way, any new service that meets the criteria specified in the policy attachment function can be attached automatically to the policy without further action on the part of the user.

**Providing a standardized policy group for services**

In this use case, a business requires a group of policies that provide a standard that all services for a certain group must follow, as shown in Figure 14.

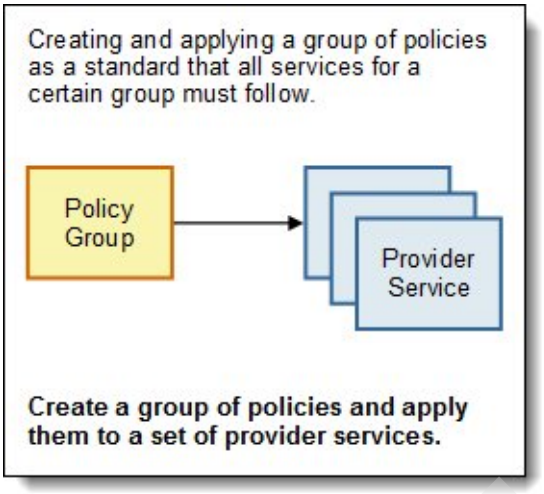


Figure 14. Standardized policy group for services

Many enterprises find that a set of provider services needs the same policies, which might include, for example, several traffic management policies, a custom security policy, and a monitoring policy. By considering these policies as part of a single policy group and applying the policies to the provider services, you can achieve significant policy reuse and standardized policy.

## Integration

The SOA Policy Solution is a solution pattern that provides automated integration of the three products to provide automated runtime policy creation, management, execution, and monitoring. The policy administration point is instantiated by the WebSphere Service Registry and Repository and provides creation and management of the policies. The policy enforcement point is instantiated by WebSphere DataPower and provides enforcement of the policies. The policy monitoring point is instantiated by Tivoli Composite Application Manager for Applications and provides for monitoring of services by using a policy.

## Supported platforms

The SOA Policy Solution consists of the following platforms:

- WebSphere DataPower. Any XG45 or XI52 model or XI50 model 9004 or later. Must run firmware release 5.0.0 or higher. See WebSphere DataPower SOA Appliances at: http://www.ibm.com/software/integration/datapower

- WebSphere Service Registry and Repository. Must run Version 7.5 or later. Version 8.0 or later is preferred. As documented for WebSphere Service Registry and Repository, see the system requirements at: http://www.ibm.com/support/docview.wss?uid=swg27010679

- Tivoli Composite Application Manager for Applications. Must run version 7.1.1.3 or later. As documented for Tivoli Composite Application Manager, see the system requirements at: http://www.ibm.com/support/docview.wss?uid=swg21409894

## Ordering information

The following tables list the ordering information.

Table 1. Part numbers and feature codes for automated installation of WebSphere Service Registry and Repository and DataPower

| Program name | PID number | Charge unit description |
|---|---|---|
| IBM SOA Policy Pattern for RedHat Linux | 5725-G73 | PVU. Available through IBM Passport Advantage® only |
| IBM SOA Policy Gateway Pattern for RedHat Linux | 5725-H60 | PVU. Available through Passport Advantage only |

Table 2. Part numbers and feature codes for individual products of the SOA Policy Solution

| Program name | PID number | Charge unit description |
|---|---|---|
| IBM WebSphere Service Registry and Repository | 5724-N72 | PVU |
| IBM WebSphere DataPower Service Gateway XG45 | 7198-32X (MTM) | Per Appliance |
| IBM WebSphere DataPower Integration Appliance XI52 V5.0.0 | 7199-42X (MTM) | Per Appliance |
| IBM WebSphere DataPower Service Gateway XG45 Virtual Edition V5.0.0 | 5725-J90 | PVU |
| IBM WebSphere DataPower Integration Appliance XI52 Virtual Edition V5.0.0 | 5725-J91 | PVU |
| IBM Tivoli Composite Application Manager for Applications v7.2 | The PID number is 5725-I45, but a customer order using a part number from Passport Advantage, not the PID number.<br>A customer can choose from two configurations, depending on the number of monitoring agents they plan to deploy to each managed server:<br>● D0V2QLL. Tivoli Composite Application Manager for Applications 3 Agent Pack<br>● D0V2WLL. Tivoli Composite Application Manager for Applications Full Agent Pack | Tivoli Composite Application Manager for Applications is priced by using the Resource Value Unit (RVU) pricing metric. The resources that are counted are the activated cores of a server Here is the formal definition of RVU:<br><br>"RVU is a unit of measure by which the program can be licensed. RVU Proofs of Entitlement (PoEs) are based on the number of units of a specific resource used or managed by the program. Licensee must obtain entitlements for this program sufficient to cover the resources managed by the program. Licensee must obtain sufficient entitlements for the number of RVUs required for licensee's environment for the specific resources as specified in the resource table found in the program's announcement or License Information document. RVU entitlements are specific to the program and the type of resource and cannot be exchanged, interchanged, or aggregated with RVU entitlements of another program or resource." |

## Related information

For more information, see the following documents:

- *WebSphere DataPower SOA Appliance: The XML Management Interface*, REDP-4446
  http://www.redbooks.ibm.com/abstracts/redp4446.html
- *IBM WebSphere DataPower SOA Appliances Part II: Authentication and Authorization*, REDP-4364
  http://www.redbooks.ibm.com/abstracts/redp4364.html
- WebSphere Service Registry and Repository product page
  http://www.ibm.com/software/integration/wsrr
- IBM Tivoli Software product page
  http://www.ibm.com/software/tivoli/?lnk=mprSO-tivo-usen
- IBM WebSphere DataPower Service Gateway XG45 product page
  http://www.ibm.com/software/integration/datapower/XG45/index.html
- IBM WebSphere DataPower Integration Appliance XI52 product page
  http://www.ibm.com/software/integration/datapower/xi52
- IBM Offering Information page
  http://www.ibm.com/common/ssi/index.wss?request_locale=en

  On this page, enter SOA Policy, select the information type (announcement letters, sales manuals, or both), and then click **Search**. On the next page, narrow your search results by geography and language.

# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you. This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

This document was created or updated on January 15, 2013.

Send us your comments in one of the following ways:

- Use the online **Contact us** review form found at:
  ibm.com/redbooks
- Send your comments in an e-mail to:
  redbook@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400 U.S.A.

This document is available online at http://www.ibm.com/redbooks/abstracts/tips0952.html .

# Trademarks