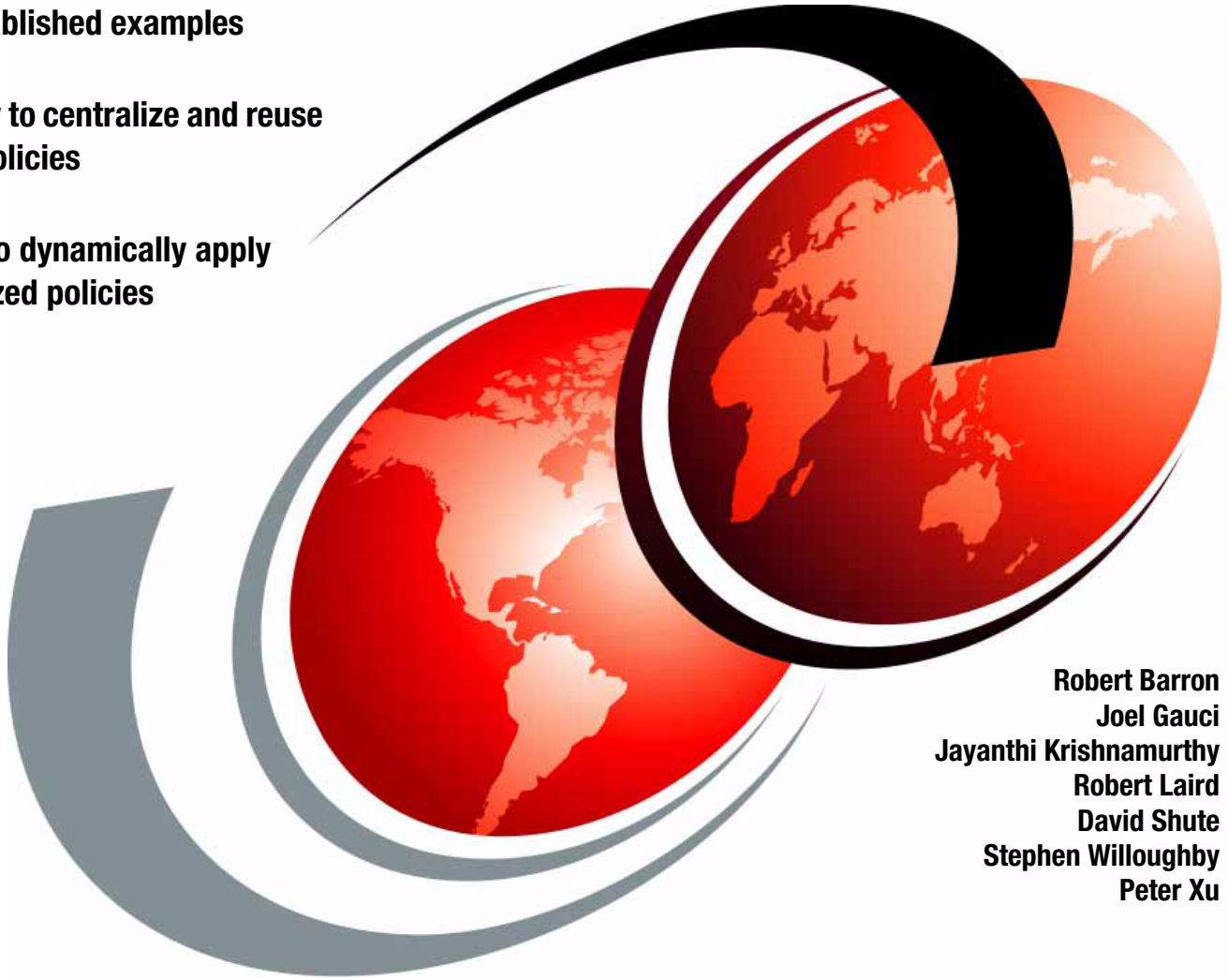
IBM

SOA Policy, Service Gateway, and SLA Management

- Explore the SOA Policy Solution by using established examples
- Learn how to centralize and reuse runtime policies
- See how to dynamically apply standardized policies



Robert Barron
Joel Gauci
Jayanthi Krishnamurthy
Robert Laird
David Shute
Stephen Willoughby
Peter Xu

Redbooks

ibm.com/redbooks

International Technical Support Organization

SOA Policy, Service Gateway, and SLA Management

April 2013

SG24-8101-00

Note: Before using this information and the product it supports, read the information in “Notices” on page ix.

First Edition (April 2013)

This edition applies to WebSphere Service Registry and Repository V8.0, IBM Tivoli Composite Application manager V7.1.1.3 and WebSphere DataPower Integration Appliance XI52.

© Copyright International Business Machines Corporation 2013. All rights reserved.
Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices ix
Trademarks x

Preface xi
The team who wrote this book xi
Now you can become a published author, too! xiii
Comments welcome. xiv
Stay connected to IBM Redbooks xiv

Part 1. Introduction 1

Chapter 1. The SOA Policy Solution 3
1.1 Information about the SOA Policy Solution 4
1.1.1 What is the solution. 4
1.1.2 What products are involved 4
1.1.3 Who uses the solution. 5
1.1.4 Did you know 6
1.2 Business value: The business aspects of the solution 6
1.3 Solution overview 8
1.4 Solution architecture 10
1.4.1 How the components of this solution fit together 10
1.4.2 Hardware platform 11
1.4.3 Software requirements 11
1.5 Scenarios of use 12
1.5.1 Standardized SLAs 12
1.5.2 Differentiating Service SLAs 15
1.5.3 Easily allow new consumers to access back office services. 16
1.5.4 Reject low-priority traffic during business hours 16
1.5.5 Reroute traffic during maintenance windows 17
1.5.6 Deny access to rogue consumer applications. 18
1.5.7 Service versioning support 19
1.5.8 Apply standard security to access provider services 20
1.5.9 Provide operational status and alerts 20
1.5.10 Automatically apply policy to new services 21
1.5.11 Provide a standardized policy group for services 22
1.6 How to use this book. 22

Chapter 2. Business case for using the SOA Policy Solution. 25
2.1 Introducing the ITSO Redbooks Travel Company 26
2.1.1 History of the company 26
2.1.2 The IT environment. 27
2.1.3 The business goals and objectives 27
2.2 Creating the business case. 28
2.3 Business case results 29
2.4 Instructions for constructing your own business case. 30

© Copyright IBM Corp. 2013. All rights reserved.

Part 2. Policy examples 33

Chapter 3. Policy traffic management, provider only, with operations 35
3.1 Implement the provider services scenario 36
3.1.1 Verify product installations 37
3.1.2 Validate existing services 37
3.1.3 Integrate products in the pattern 39
3.2 Service governance 54
3.2.1 Sample files 56
3.2.2 Governing the services yourself 56
3.2.3 Using your own services 56
3.2.4 Creating your business spaces 56
3.3 Govern existing services 56
3.3.1 Creating a business service and service version 57
3.3.2 Creating service level definitions 60
3.3.3 Creating policies 62
3.3.4 Attaching policies to SLDs for services you want 62
3.3.5 Setting up a service gateway in DataPower 63
3.3.6 Promoting the services to WSRR run times 68
3.3.7 Synchronization with DataPower 71
3.4 Creating policies 73
3.4.1 Creating mediation policy for queuing 73
3.4.2 Creating mediation policy for throttling 75
3.4.3 Creating Mediation Policy for Notify 77
3.4.4 Running client transactions to validate provider services 78
3.5 Attaching policies 78
3.5.1 Attaching a policy to a specific service and operations 78
3.5.2 Attaching the policy to all services matching a name 80
3.5.3 Attaching policy to all services for an organization 82
3.6 Reporting on services and policies applied to services 84

Chapter 4. Policy traffic management and consumer provider pairs 87
4.1 Implement the consumer-to-provider services scenario 88
4.1.1 Create a business application with capability application version 88
4.1.2 Govern business application 91
4.1.3 Create a service level agreement 91
4.1.4 Create policies 92
4.1.5 Add the SLD as an agreed endpoint to the SLA 93
4.1.6 Attach policies to an SLA 94
4.1.7 Promote the services to the WSRR run time 95
4.1.8 Synchronization with DataPower 96
4.1.9 Validate the services 98
4.2 Determining customer priority 99
4.3 Customer priority policy example 100
4.3.1 Defining a Gold customer 102
4.3.2 Silver customer 107
4.3.3 Blacklist customer 111
4.3.4 Rogue customer 114

Chapter 5. Versioning with custom policy 119
5.1 General description of the versioning use case 120
5.2 Implementing the versioning pattern with DataPower and WSRR 122
5.3 Custom versioning policy enforcement details 125
5.4 Pricing service versions 128

5.4.1 Details of the Pricing service versions 129
5.4.2 Data mediation between versions of the Pricing service. 129
5.4.3 Pricing services in WSRR 133
5.5 Creating custom policy domain and assertions for versioning 134
5.6 Creating custom policy XSL style sheet for the custom versioning policy. 139
5.6.1 Creating the DataPower configuration artifacts. 140
5.6.2 Exporting the DataPower configuration artifacts 146
5.6.3 Writing the custom policy style sheet for versioning 150
5.7 Attaching the custom versioning policy to a specific service. 159
5.7.1 Attaching the custom versioning policy in WSRR at design-time 159
5.7.2 Creating a saved search in WSRR at design-time 164
5.7.3 Creating a Web Service Proxy for versioning in DataPower. 166
5.7.4 Demonstration of the custom versioning policy enforcement 180

Chapter 6. Security using custom policy 187
6.1 Service security. 188
6.2 Creating a custom policy for security 191
6.2.1 DataPower AAA policy 191
6.2.2 Prerequisites 192
6.2.3 Policy decision point (PDP) versus policy administration point (PAP) 192
6.3 Creating custom policy domain and assertion for security 194
6.4 Creating a custom policy XSL style sheet for the custom security policy 197
6.4.1 Creating the DataPower configuration artifacts. 198
6.4.2 Exporting the DataPower configuration artifacts 203
6.4.3 Writing the custom policy style sheet for security 210
6.5 Attach custom policy to all services for an organization 220
6.5.1 Attaching the custom security policy in WSRR at design-time 220
6.5.2 Attaching a custom security policy loaded on the DataPower device 234

Chapter 7. Policy monitoring 235
7.1 Monitoring services 236
7.2 Displaying services in ITCAM 237
7.3 Monitoring examples 240
7.3.1 Steps to create a policy. 240
7.3.2 Attaching to an SLD and creating a situation 243
7.3.3 Various situation tables 251

Part 3. Policy administration point 261

Chapter 8. WebSphere Service Registry and Repository for traffic management . . 263
8.1 Overview of WSRR as the PAP 264
8.2 Creating a mediation policy. 264
8.2.1 Attribute conditions 266
8.2.2 Schedule conditions 268
8.2.3 Actions 269
8.3 Viewing a mediation policy 273
8.4 Updating a mediation policy 273
8.5 Deleting a mediation policy 274
8.6 Using Consumer ID and Context ID 276

Chapter 9. WebSphere Service Registry and Repository for monitoring policy. . . . 277
9.1 Why monitor 278
9.1.1 Monitoring or IBM Service Management. 278
9.1.2 Terminology 278

9.2 Monitoring options in WSRR . . . 279
9.2.1 Basic ITCAM architecture . . . 279
9.2.2 Creating a monitoring policy . . . 282
9.2.3 Modifying a Policy . . . 294
9.3 When a threshold is crossed . . . 296
9.3.1 Operator's monitor . . . 296
9.3.2 Automatic response by ITCAM . . . 298
9.3.3 Automatic response to WSRR (eventing) . . . 301
9.4 Advanced monitoring with ITCAM . . . 308
9.4.1 Creating situations without WSRR . . . 309
9.4.2 Creating situations in ITCAM . . . 309
9.4.3 Advanced situation functions . . . 310

Chapter 10. Attaching a policy to a service . . . 313
10.1 Attaching a policy to a service . . . 314
10.2 Viewing and editing a policy attachment by using WSRR . . . 314
10.2.1 Dynamic attachment of a policy to a set of services . . . 314
10.2.2 Attaching a policy from a specific item . . . 323
10.3 Attaching a policy to a provider service using an SLD . . . 325
10.3.1 Service . . . 325
10.3.2 Operation . . . 325
10.3.3 Endpoint . . . 326
10.4 Attaching policy to consumer-provider pair by using SLA . . . 327
10.4.1 Service . . . 327
10.4.2 Operation . . . 327
10.4.3 Endpoint . . . 329
10.5 Specifying policies from unknown consumers . . . 330

Chapter 11. Policy administration point utilities . . . 333
11.1 Understanding the policy lifecycle . . . 334
11.2 Viewing policies attached to a service . . . 336
11.3 Viewing services attached to a policy . . . 338
11.4 Restricting policy access . . . 339
11.5 Viewing activity log for a policy . . . 342
11.6 Using policy events for management of monitoring policy results . . . 343

Part 4. Policy enforcement point . . . 345

Chapter 12. DataPower policy enforcement point configuration . . . 347
12.1 Overview . . . 348
12.2 WSRR server object . . . 349
12.3 Web Service Proxy object . . . 350
12.4 Front Side Handler . . . 353
12.5 WSRR subscriptions and saved search subscription . . . 354
12.5.1 WSRR subscription . . . 354
12.5.2 WSRR saved search subscription . . . 355
12.6 Policy Parameter Set object . . . 356
12.7 SLA policy details . . . 357
12.7.1 SLD Definition Files . . . 357
12.7.2 SLA policy . . . 358
12.7.3 DataPower Rules grid . . . 360
12.8 Scope of a policy on one or more DataPower appliances . . . 362
12.8.1 Applying policy to a single appliance . . . 362
12.8.2 Applying policy to several appliances . . . 362

12.8.3 Applying policy to a DataPower cluster 362

Chapter 13. Creating and using custom policies 367
13.1 When to use a custom policy 368
13.2 Creating a custom policy 368
13.2.1 Custom policy governed in WSRR 369
13.2.2 Custom policy loaded in DataPower 369
13.2.3 Actors and components 369
13.2.4 Steps to create a custom policy 371
13.2.5 Custom policy domain 372
13.2.6 Custom policy XSL style sheet 372
13.3 Importing a custom policy in WSRR 382
13.4 Custom policy attachment in WSRR 383
13.5 Importing a custom policy in DataPower 385
13.6 Custom policy attachment in DataPower 386
13.7 Custom policy enforcement 388

Chapter 14. ITCAM as policy monitoring point 389
14.1 Overview of ITCAM as the PMP 390
14.2 Overview of Tivoli architecture 391
14.2.1 IBM Tivoli Monitoring Architecture 391
14.2.2 ITCAM for SOA architecture 392
14.3 ITCAM for Applications processes policy and management updates from WSRR . . 393
14.4 Configuring for integration of WSRR and ITCAM for Applications 394
14.5 Troubleshooting the installation 396
14.6 Troubleshooting and tracing the integration 396
14.6.1 Situation verification 396
14.6.2 Trace levels in WSRR 398
14.6.3 Trace levels in ITCAM 399
14.6.4 Example traces 400
14.7 Operations notification and upstream integration with other Tivoli products 409
14.8 ITCAM monitoring agent for DataPower policies 410

Part 5. Appendixes 413

Appendix A. Implementing a SOA Policy Solution flow of work 415
A.1 Installing the samples from this book 416
A.2 Validating installation of each individual product in the SOA Policy Solution 417
A.2.1 IBM Tivoli Composite Application Manager 417
A.2.2 IBM WebSphere Service Registry and Repository 418
A.2.3 IBM WebSphere DataPower 418
A.3 Integrating products in the SOA Policy Solution 418
A.3.1 ITCAM with WSRR 418
A.3.2 IBM WebSphere DataPower with ITCAM and WSRR 419
A.4 Governing of services 419
A.4.1 IBM WebSphere Service Registry and Repository 419
A.4.2 IBM WebSphere DataPower 419
A.4.3 IBM Tivoli Composite Application Manager 419
A.5 Creating Policies 420
A.6 Attaching policies 420
A.7 Promotion within WSRR 420
A.8 Running the client to validate the services 420

Appendix B. ITCAM monitoring attribute tables 421
B.1 Attribute table descriptions 422
B.2 Endpoint Inventory attributes 423
B.3 Fault Log_610 attributes 423
B.4 Message Arrival Threshold_610 attributes 424
B.5 Services Inventory_610 attributes 425
B.6 Services Inventory Requester Identity_610 attributes 426

Appendix C. Additional material 427
C.1 Locating the web material 427
C.2 Using the web material 427
C.2.1 Downloading and extracting the web material 428
C.2.2 Deploy sample web services into WebSphere Application Server 428
C.2.3 Import objects into WebSphere Service Registry and Repository 433
C.3 Set up and use the Sample SOA Policy Pattern Business Case ROI spreadsheet . . 434

Related publications 435
IBM Redbooks 435
Online resources 435
Help from IBM 436

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 grant 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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites 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.

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 measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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.

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.

© Copyright IBM Corp. 2013. All rights reserved.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

CICS®	Global Technology Services®	Tivoli®
DataPower®	IBM®	WebSphere®
DB2®	Redbooks®	
developerWorks®	Redbooks (logo) ®	

The following terms are trademarks of other companies:

ITIL is a registered trademark, and a registered community trademark of The Minister for the Cabinet Office, and is registered in the U.S. Patent and Trademark Office.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbooks® publication teaches you how to automate your runtime policy by using a centralized policy management system. The SOA Policy Solution provides a centralized policy administration, enforcement, and monitoring for runtime policies that enable traffic management for service level agreement enforcement, service mediation, and other customized policies. Policies can be defined once and reused among multiple services, thus enabling a standardized, consistent approach to a runtime policy that saves time and money for implementation and maintenance of non-functional requirements for the enterprise and assists with faster time to market.

Business users can use the SOA Policy Solution to help create the service level agreements for their business services to deliver on promises for business performance. IT Architects can use the SOA Policy Solution to architect the policy solution patterns that standardize the runtime policy usage at their organization. Developers select specific policy patterns to implement the non-functional requirements that are associated with their projects. Operations groups provide information about operation needs and create standardized monitoring policy for operational action at run time.

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Robert Barron is an IBM Tivoli® Technical Specialist And Architect in the IBM Global Technology Services® group in Israel. Since 1997, he has worked in a variety of IT disciplines, covering application development, QA and testing, configuration management, customer services, and more. He has worked on Enterprise Service Management with IBM for over seven years. Robert holds a degree in the Classics and History of Science and Technology.

Joel Gauci is a Certified IT Specialist in the IBM WebSphere® Software group in France. Since 2006, Joel has worked for leading European firms on projects including IBM DataPower® appliances, mainly in the telecommunications, industry, energy, banking, and transportation sectors. As a Client Technical Professional, Joel mainly works on DataPower and API Management selling opportunities. He assists potential customers from basic presentation to complex architecture definition. Joel has authored several IBM Redbooks and articles related to DataPower SOA appliances. Joel holds a master's degree in computer science and a master's degree in mechanics from the University Paris 6 in France.

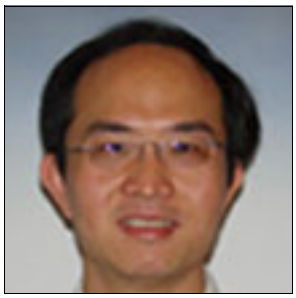
© Copyright IBM Corp. 2013. All rights reserved.

Jayanthi Krishnamurthy is an Architect at Royal Caribbean Cruises in Miami, USA. She has 17 years of IT experience in travel, government, and manufacturing sectors, architecting and implementing infrastructure for SOA, Virtualization and WebSphere suite of products. She also has an application development background and worked with a wide range of products from multiple vendors. Jayanthi holds an Master of Science degree in Computer Engineering from FAU, Florida and Bachelor's degree in Electronics and Communication Engineering from University of Madras, India. She holds certifications in WebSphere products, ITIL. and Java. She has previously written Redbooks on WebSphere Application Server, Portal, and IBM DB2®.

Robert Laird is an Enterprise Architect in the IBM SOA Foundation group in the US. He leads the software product architecture for SOA, SOA Governance, and SOA Policy at IBM. He has consulted on SOA for a wide variety of industries and customers while at IBM. Robert has 20 years of experience in the telecommunications industry where he had a variety of development and architect roles, rising to Chief Architect at MCI Communications, where he led the SOA and SOA Governance effort. Robert has written several books: *Executing SOA*; *SOA Governance: Achieving and Sustaining Business and IT Agility;* and also *SOA Governance: Governing Shared Services On-Premise and in the Cloud* with Thomas Erl and the Prentice Hall Service-Oriented Computer Series.

David Shute is Technical Enablement Program Manager for DataPower. He has eight years of experience with DataPower and more than 20 years of experience producing and delivering learning materials on high technology subjects. Mr. Shute has authored several other IBM Redbooks and developerWorks® articles, and also numerous classroom and online courses. He has taught classes in SOA, XML, security and application-level firewalls worldwide.

Stephen Willoughby is a Solution Test Specialist for the WSRR development team in the United Kingdom. He has over 10 years of experience in solution testing. He holds a degree in Computer Science from the University of York. His areas of expertise include WSRR and its integration with DataPower and ITCAM for SOA. He has written other Redbooks and many developerWorks articles.



Peter Xu is currently a Portfolio Architect and Retail Lead in the US with the IBM Worldwide WebSphere Business Agility "Tiger" sales team. He had been a Senior Managing Consultant for 10 years in the IBM Software Services for WebSphere organization. Peter has extensive software development, consulting, and sales experience. He helped many retail, retail banking, and public sector clients achieve business agility by using solutions based on Business Process Management and SOA in an industry context. Peter publishes extensively on process modeling and service integration, and is a frequent speaker at events and conferences. Peter holds TOGAF 9 certification and is also an OMG-Certified Expert in BPM.

Thanks to the following people for their contributions to this project:

Margaret Ticknor, Shari Deiana, Tamikia Lee, Linda Robinson, Diane Sherman, Stephen Smith, Debbie Willmschen
International Technical Support Organization, Raleigh Center

Mario De Armas, Thomas Burke, Ian Heritage, Oswaldo Gago
IBM US

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@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

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

Part 1

Introduction

This part contains the following chapters:

- Chapter 1, “The SOA Policy Solution” on page 3
- Chapter 2, “Business case for using the SOA Policy Solution” on page 25

© Copyright IBM Corp. 2013. All rights reserved.

1

The SOA Policy Solution

This chapter introduces the SOA Policy Solution by describing what it is, what it accomplishes, who uses it, and the business benefits it provides. The chapter examines the solution details, including an overview of the capabilities and the architecture, and several use cases that demonstrate how the SOA Policy Solution might typically be used.

This chapter contains the following topics:

- 1.1, “Information about the SOA Policy Solution” on page 4
- 1.2, “Business value: The business aspects of the solution” on page 6
- 1.3, “Solution overview” on page 8
- 1.4, “Solution architecture” on page 10
- 1.5, “Scenarios of use” on page 12
- 1.6, “How to use this book” on page 22

© Copyright IBM Corp. 2013. All rights reserved.

1.1 Information about the SOA Policy Solution

As the Open Group states: “A policy is a statement of direction that a human actor may intend to follow or may intend that another human actor should follow. Knowing the policies that apply to something makes it easier and more transparent to interact with that something.”[1]

The SOA Policy Solution deals with the automation of that policy at run time. It provides a centralized policy administration, enforcement, and monitoring for runtime policies. It enables traffic management for service level agreement enforcement, service mediation, and customized policies including being able to create standardized security, optimization, and integration policy. Policies can be defined once and reused among multiple services, thus enabling a standardized, consistent approach to runtime policy that saves time and money for implementation and maintenance of non-functional requirements for the enterprise, and assists with faster time to market.

1.1.1 What is the solution

The SOA Policy Solution offers an integrated solution architecture that is ready to use and that enables centralized management and governance of policies, transaction by transaction policy enforcement, and monitoring of those services with monitoring and operational policy capabilities.

1.1.2 What products are involved

The policy administration point (PAP) is instantiated by the WebSphere Service Registry and Repository (WSRR) and provides creation and management of the policies. The policy enforcement point (PEP) is instantiated by WebSphere DataPower and provides enforcement of the policies. The policy monitoring point (PMP) is instantiated by ITCAM for Applications and provides for monitoring of services with policy. WSRR is an enterprise-level registry and repository providing scalable and automated capabilities to help organizations optimize productivity and resources in an SOA environment. WSRR performs the function of centralized policy authoring, management, and governance within the solution.

The WebSphere DataPower Integration Appliance XI52, XI50, and XG45 is a purpose-built hardware platform for delivering 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 the function of transaction-by-transaction policy enforcement within the solution. In addition to using the purpose built hardware appliances mentioned previously, one can also use the WebSphere DataPower Service Gateway XG45 and WebSphere DataPower Integration Appliance XI52 as virtual editions to run in VMware hypervisor environments. These virtual editions are designed to 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.

IBM Tivoli Composite Application Manager (ITCAM) for Applications offers integrated management tools for Web and enterprise infrastructures to aid SOA life-cycle availability and performance. ITCAM performs the function of service monitoring and operational monitoring control within the solution.

[1] See the Open Group website for more information: http://www.opengroup.org/soa/source-book/ontology/policy.htm

1.1.3 Who uses the solution

The types of users and how they use the solution are as follows:

- Business Management
  - Understand the benefits of centralized runtime policy and how it can benefit this user's business by saving money and providing faster time to market.
  - Lead in creating the business case for centralized runtime policy and determine if there is a positive return on investment (ROI) for using a centralized policy.
- Business Analyst or Business User
  - Understand policy capabilities so that they can define the business plans and objectives for their business services.
  - Refine centralized policy strategy with the SOA Architect and create the requirements necessary for business performance.
- Enterprise Architect
  - Understand the capabilities of the SOA Policy Solution to be able to define the policy patterns to be used at their organization.
  - Define and apply how runtime policy domains and standards will be used.
  - Work to apply the solutions with the SOA Architect, Infrastructure Architect and the Chief Risk Officer.
- IT Management
  - Work with Business Management to implement standardized policies to save money and have a faster time to market.
  - Provide leadership within the IT organization to realize the benefits of centralized policy.
- SOA Architect
  - Work with the Enterprise Architect to understand the SOA Policy Solution and implement the policy solution patterns to be used at the organization.
  - Work with the Business Analyst or Business User to create the requirements necessary for business performance
  - Work with development to choose policies required for a particular development project and help to implement for that project.
  - Work with operations to understand operation needs and create standardized monitoring policy for operations.
- Developer
  - Understand the runtime policy capabilities.
  - Work with the SOA Architect to reuse existing policies or create new or modify existing policies for the services that SOA Architects are responsible for.
- IT Infrastructure Architect
  - Create the SOA Policy Solution deployment models for the overall SOA infrastructure in consultation with the Enterprise Architect.

- Operations
  - Understand the SOA Policy Solution monitoring capabilities.
  - Work with the SOA Architect to define the monitoring policies that are required.
- Chief Risk Officer
  - Is responsible for understanding and articulating external compliance requirements and identifying internal business and operational policies that are necessary to comply with the law and manage corporate risk. This also includes policies that are needed to govern the creation and updating of policy within the organization (policy for policy).

1.1.4 Did you know

Many organizations find that they can group their services into approximately six sets, where each individual set requires the same policies. A set of policies, for example, might include 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. After those policies are defined in the SOA Policy Solution policy administration point (PAP), they are easily attached to all of the services in the group through a simple query. The attachment process is dynamic; as additional 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.

In this manner, the process of managing multiple individual policies in the various middleware devices is now simplified into creating and attaching approximately six policy patterns in a centralized policy repository. This saves time and money.

1.2 Business value: The business aspects of the solution

This book teaches you how to automate your runtime policy by using a centralized policy management system. But why should you automate?

All organizations implement policy in some form (for example, human decisions are a common method of implementing policy) to make decisions that are important to the business and IT. Your business 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 the decision-makers that your project to automate policy is a worthwhile endeavor always involves one or both of these two factors:

- Saving money
- Driving incremental revenue

Every benefit you contemplate will ultimately fit into one of those two categories. If it does not, you have not yet applied sufficient thought and analysis to the source and origination of the benefits that will be realized from your efforts in implementing policy automation. Consequently, the discussion in this section and the sample business case that is presented focuses on the typical benefits that can be realized in these two areas.

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

Specifically, three factors contribute to the business case for saving money by using policy. For each factor, you calculate the amount of time that is saved and apply a financial factor to each. In addition, the first, *dynamicity*, helps an organization drive incremental revenue because of faster time to market.

- 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 factor is possible because the policy changes can be updated once instead of multiple times in multiple places. Consider the following examples:

  - Business request to change the service level agreement (SLA) for a specific group of customers to provide enhanced support
  - Business request to change security password policies to increase security in a specific area
  - Operations request to lock out a batch routine from updating a critical database during non-business hours in response to business request to improve response time

- Reuse

  Typically, a finite grouping of policies must be applied across the services or resources in the runtime environment. For example, if several reusable policy groupings are defined and reused among the services in an organization, simply identify which one of those groupings should be applied to each new service. No new policy work needs to be done so saving time and money is possible. 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 separate services 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. Achieving this view is normally difficult if policy is maintained in separate silos that do not integrate. Using a centralized policy management system overcomes this difficulty. It also provides a means of centralized auditing of policy actions, important from a regulatory and audit perspective.

As mentioned previously, one must consider the impact of dynamicity on time to market of the business product and services that are affected. Normally, an organization should be aware of the projects that will be funded during the next 12 months. To what degree is each product able to go into service faster by having a set of standardized, reusable policy groupings that already 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 and should be considered.

Chapter 2, “Business case for using the SOA Policy Solution” on page 25 contains a business case example that you can use as a starting point for your ROI calculations on the adoption of the SOA Policy Solution. That chapter demonstrates how to use the corresponding spreadsheet that is included with this book.

1.3 Solution overview

The SOA Policy Solution was created in way that requires some initial configuration for WSRR, WebSphere DataPower, and ITCAM. After the configuration is complete, policy updates in WSRR result in an automatic notification to WebSphere DataPower and ITCAM from WSRR to pull the policy (unless you opt for manual synchronization). WebSphere DataPower and ITCAM then convert the standard format WS-Policy into its own local policy format. Figure 1-1 shows this relationship.

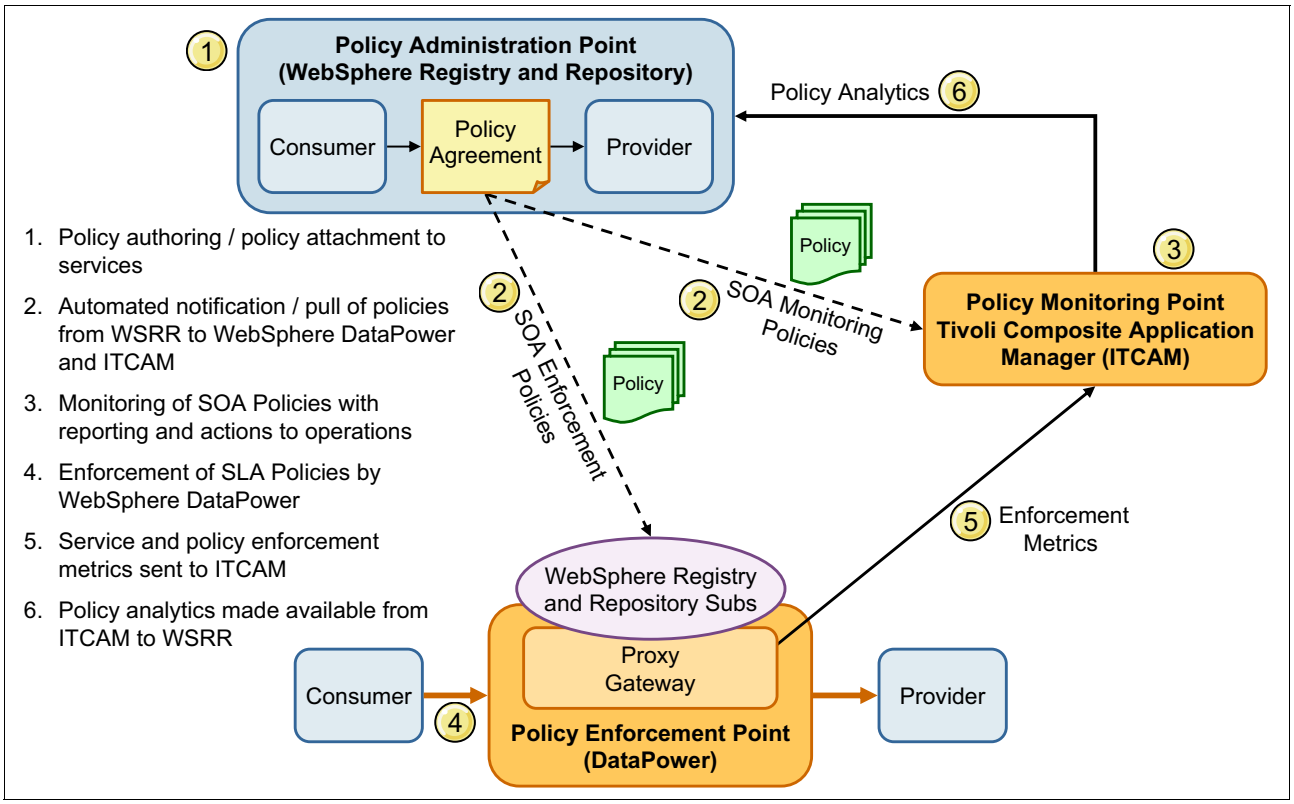


Figure 1-1 Solution overview

The following steps characterize the SOA Policy Solution process:

1. Policies are authored and then attached to services that need that policy, as appropriate.
   a. Information about the services and associated metadata including their governance is created in WSRR or loaded from external sources into WSRR.
   b. The set of policies needed are either created in WSRR, loaded from third-party tooling, or loaded from custom policies created in WebSphere DataPower.
   c. Policies are attached to the services as appropriate.
2. The automated publish/subscribe process of policies is done from WSRR to WebSphere DataPower and IBM Tivoli Composite Application Manager (ITCAM) for Applications.
   a. As part of the setup, one or more proxy gateways are created in each WebSphere DataPower appliance that defines the services handled by that proxy.

      Policies that are attached to those services automatically gathered and downloaded in the publish/subscribe process (one time, added or changed as needed).
   b. Optional: As part of the setup, WebSphere DataPower is configured so that policies may be shared by other appliances in a cluster (one time, and updated as needed).

c. Each WebSphere DataPower appliance can now automatically download service information (WSDL files) for services it is responsible for transacting.

d. WebSphere DataPower downloads policies for services it is responsible for upon notification from WSRR.

e. WebSphere DataPower converts policies into internal WebSphere DataPower representation (service level management, objects).

f. As part of the setup, IBM Tivoli Composite Application Manager (ITCAM) for Applications subscribes to monitoring policy from WSRR (one time).

g. ITCAM for Applications successfully downloads monitoring policies as they are published.

h. ITCAM for Applications converts policy into internal representation (situation policies).

3. Monitoring of SOA policies with reporting and notification of operations occurs.

   a. Monitoring policies are active in the ITCAM for Applications Situation Policy.

   b. ITCAM for Applications receives monitoring information from WebSphere DataPower and monitors the service itself on the application server.

   c. Periodically (the default is five minutes), ITCAM for Applications evaluates the monitoring (situation) policies and takes action if the policies evaluate to true.

4. Enforcement of SOA policies for consumer-provider transactions. Monitoring (step 3) and enforcement are done in parallel during run time.

   a. Enforcement policies are active in the various WebSphere DataPower appliances.

   b. WebSphere DataPower receives service transactions and applies policies for that consumer service and provider service.

5. WebSphere DataPower sends service and policy statistics to ITCAM.

6. ITCAM sends monitoring events to WSRR when the monitoring (situation) policies evaluate to true.

   a. Set up events in WSRR that are to be monitored from IITCAM for Applications (one-time action for each event).

   b. As situation policies evaluate to true, events are pushed to WSRR from ITCAM for Applications for display.

1.4 Solution architecture

The power of the SOA Policy Solution lies in its ability to centralize and reuse runtime policies in one place (the policy administration point in the solution architecture) and then have these policies automatically be pulled by the appliances that need them for consumer-provider transactions (the policy enforcement point in the solution architecture) and also have a monitoring capability for services (the policy monitoring point in the solution architecture).

1.4.1 How the components of this solution fit together

The solution architecture, is shown in Figure 1-2.

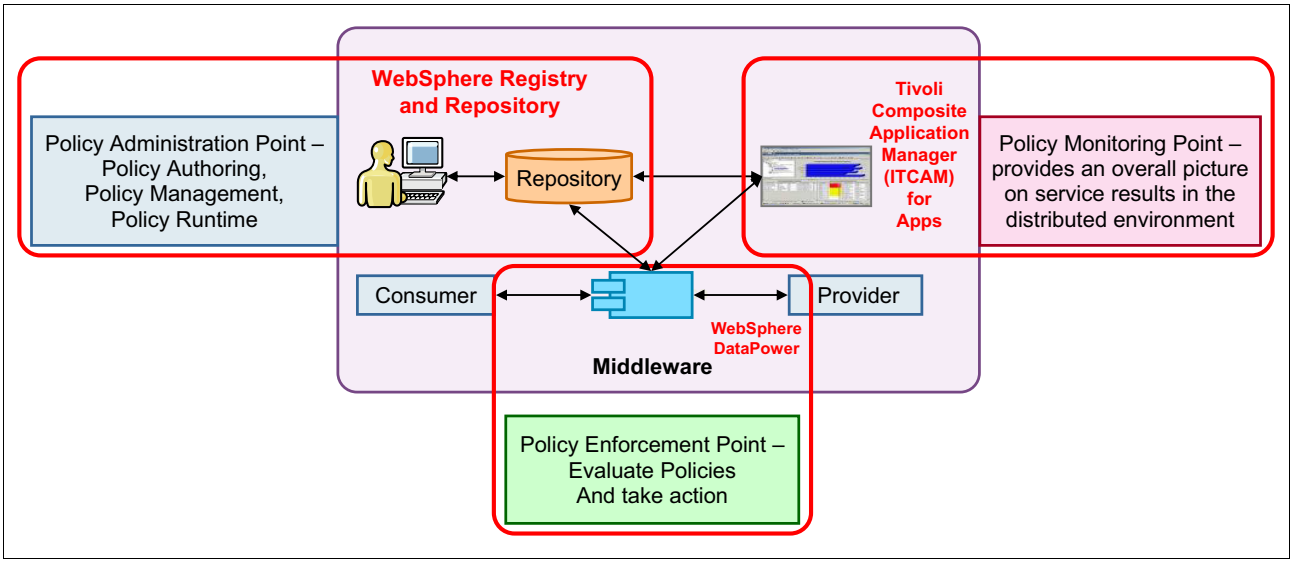


Figure 1-2 Pattern for SOA Policy Solution

The following list describes the duties of each of the functional components of the solution architecture (the SOA Policy Solution architectural pattern):

- Policy administration point (PAP)

  A policy administration point provides policy capabilities that support the centralized administration of policy:

  - Create, update, delete, and read a policy
  - Management and governance of the policy
  - Assignment of policies to one or more resources
  - Deployment of policy to the PEP and PMP

- Policy enforcement point (PEP)

  A policy enforcement point is a function component that provides enforcement of policies on a consumer-provider on a transaction by transaction basis. This includes:

  - Takes action to enforce policies.
  - Receives and makes ready for usage (translates) enforcement policy updates.
  - Provides service and policy enforcement metrics to the PMP.
  - Sometimes makes use of a policy decision point (PDP). A PDP evaluates participant requests against relevant policies/contracts and attributes to render an authorization, eligibility, or validation decision or provide calculated results.

- A PEP will sometimes make use of a Policy Information Point (PIP). A Policy Information Point (PIP) provides external information, such as results from a database with information that must be evaluated to make a policy decision.

- Policy monitoring point (PMP)

  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 makes ready for usage (translates) monitoring policy updates.
  - Captures real time collection and statistics on service usage for display.
  - Correlates, analyzes, and visualizes the data fed in by the various real time collectors including policy enforcement points.
  - Logs, aggregates, measures, and highlights significant events (as specified by monitoring policy).
  - Provides monitoring policy analytics to PAP.

1.4.2 Hardware platform

The hardware platform consists of the following components:

- WebSphere DataPower
  - Any WebSphere DataPower Service Gateway XG45 and WebSphere DataPower Integration Appliance XI52 models plus a WebSphere DataPower Integration ApplianceXI50 model 9004 or later.
  - Also, WebSphere DataPower Service Gateway XG45 and WebSphere DataPower Integration Appliance XI52 as virtual editions to run in VMware hypervisor environments. These virtual editions are designed to 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.
- WebSphere Service Registry and Repository (WSRR), as documented for WSRR system requirements:

  http://www.ibm.com/support/docview.wss?uid=swg27010679
- Detailed System Requirements for ITCAM for SOA

  http://www.ibm.com/support/docview.wss?uid=swg21409894

1.4.3 Software requirements

The following software is required:

- WSRR 8.0 fix pack 1 with iFix IFIV31285 applied or later, installed.
- WebSphere DataPower v5.0.0.0. All Fix packs for DataPower Integration appliances version 5.0 should be applied. In particular v5.0.0.5 resolves an issue with custom policies. See the following location for details:

  http://www.ibm.com/support/docview.wss?uid=swg24032620
- ITCAM for Application 7.1.1.3 can work, but 7.2 is preferred.

1.5 Scenarios of use

Part 2, “Policy examples” on page 33 offers a set of examples that explain the mechanics of how to create policy and attach that policy to the services or set of services that must be subject to the policy. The purpose of this section is to help you think about the types of solutions that can be provided by the SOA Policy Solution.

The following use cases are discussed:

- Standardizing SLAs to be applied to provider services
- Differentiating service SLAs for provider services depending on the guaranteed level of service to be able to give the best customers the best service
- Creating policies that perform mediations on services so that new consumers are easily allowed access to provider services
- Using policy scheduling to prevent low priority applications from flooding the system during normal business hours so that the SLAs can be met for high priority, online services
- Rerouting traffic during maintenance windows
- Identifying anonymous consumers and rejecting (throttle) those transactions, so that you can guard against rogue applications and denial of service attacks
- Creating policy that supports the transition from an old version of a service to a new version so that you can easily manage rolling out service changes
- Creating a standardized security policy that can be applied to all services for an organization so that you can easily manage and govern our service security
- Creating policies that monitor runtime results and automatically take action so that the operations group can be proactive instead of reactive
- Applying standardized policies dynamically so that policy is applied seamlessly to new services
- Creating and applying a group of policies as a standard that all services for a certain group must follow

1.5.1 Standardized SLAs

In this use case, the requirement is *create standardized service level agreements (SLAs) to automatically manage the traffic against critical back end or provider service so that you can meet IT's SLA promises to the business*.

Many policies that are implemented for the SOA Policy Solution are about implementation of the SLAs for a provider service no matter how many or who the consumers of that service are. In this case, an SLA was created for the provider service and policies are put in place to take action to provide the required service level.

One 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 should identify the non-functional requirements for their business functions such as *the credit authorization service must reply within three 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 situation is to have the SOA Architect or Enterprise Architect create a *Policy Tree* that helps IT to translate the business SLAs into runtime SLAs that can be automated, as shown in Figure 1-3.

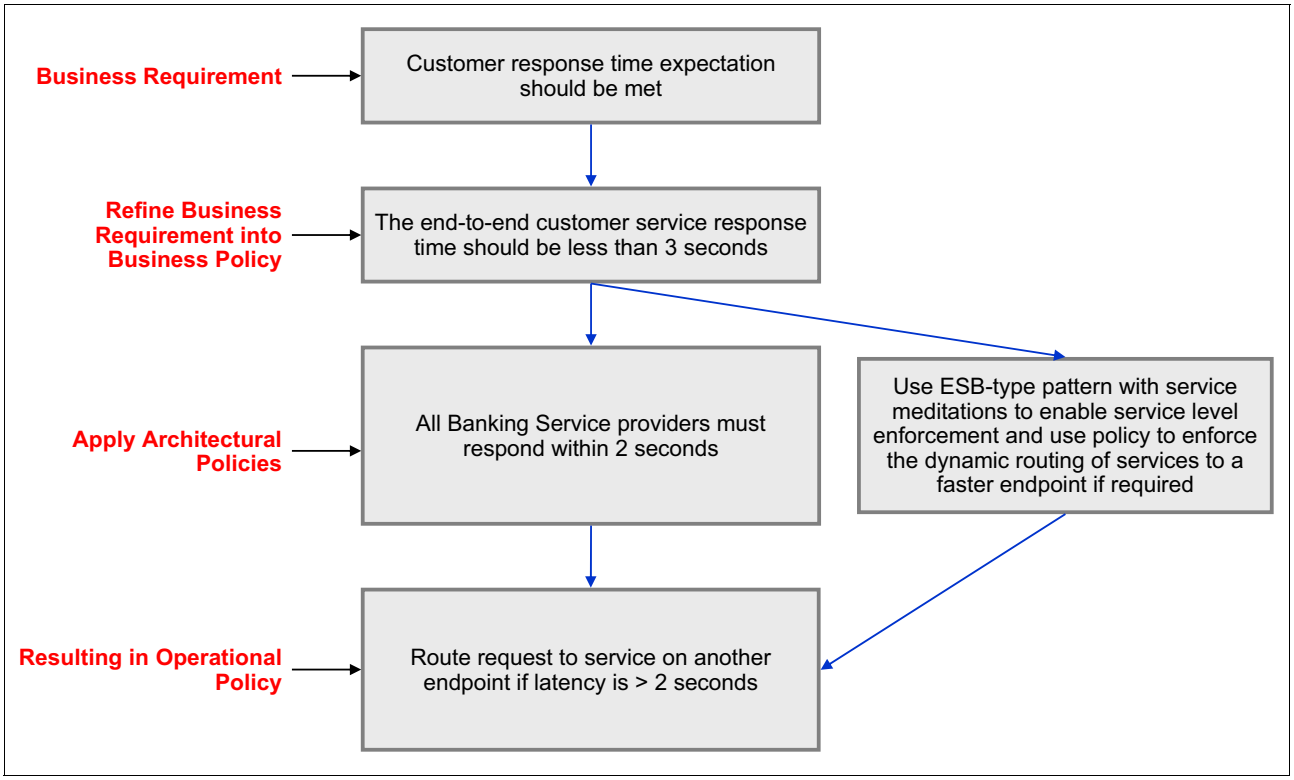


Figure 1-3 Creating standardized SLAs

In this case, a high-level business requirement is stated: *Customer response time that is expected should be met*. The obvious question for the analyst now is what exactly is that response time? A conversation with the business and a customer survey elicits the answer of three seconds, as reflected in the refined business requirement. In terms of the overall flow, the architect determines that customer service must respond within two seconds or be rerouted to a secondary endpoint and this is the runtime policy that is created.

Figure 1-4 shows sample runtime policy actions that are available to automatically maintain the SLA that the business requires.

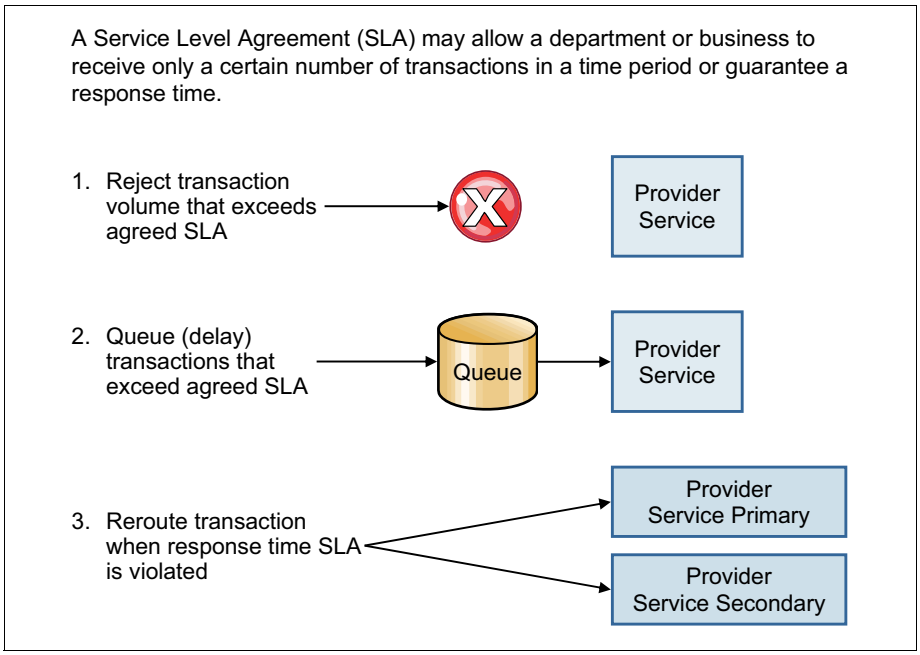


Figure 1-4 Traffic management SLAs

Use rejection of transactions for situations where low-priority traffic (for example, batch jobs) or traffic from a variety of sources are sending too many transactions for the provider (back-office) service to handle. This type of policy protects the provider service from being overwhelmed while enforcing an SLA.

Queueing helps to smooth 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 alternate provider service instance. This helps to maintain the level of service on the primary instance and give an overall good level of service to the consumer.

1.5.2 Differentiating Service SLAs

In this use case, the requirement is for d*ifferentiating service SLAs depending on the level of service guaranteed, to give the best customers the best service*. See Figure 1-5.

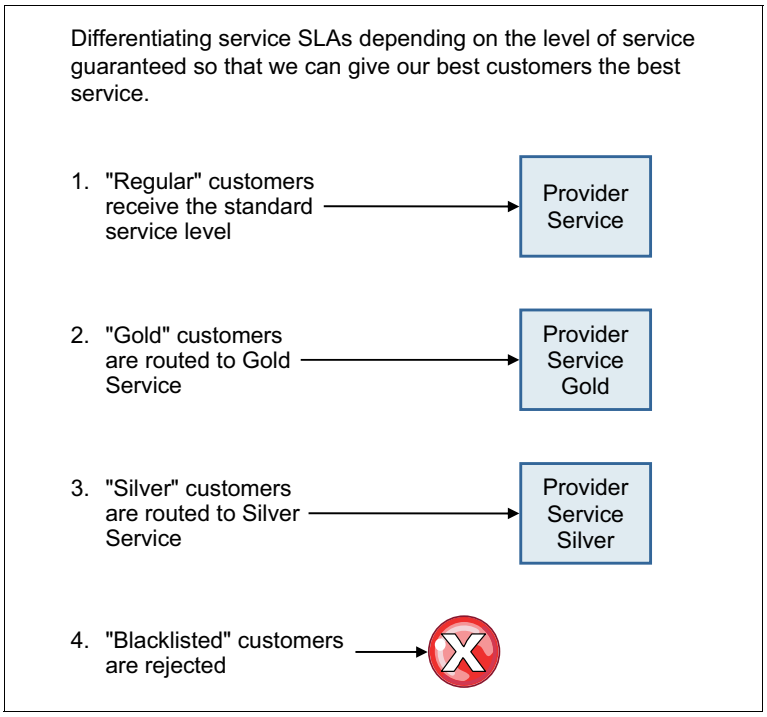


Figure 1-5 Differentiating service SLAs

Information about a consumer “type” can be passed in and used by the policy enforcement. A common use of such information is customer *segmentation* (differentiation), where a different level of service is provided, depending on the value of this consumer type. For example, you might have several tiers:

- Premier *gold* consumers receive an excellent service response.
- Next tier *silver* consumers receive a good service response.
- Next tier *regular* consumers receive standard service.
- Blacklist consumer requests are rejected so that their information is not presented to the back-end provider service.

The gold consumers are routed to a lightly loaded server endpoint; silver consumers are routed to a medium busy server endpoint; and regular consumers are routed to a more heavily loaded endpoint.

1.5.3 Easily allow new consumers to access back office services

In this use case, the requirement is for *creating policies that perform mediations on services, to easily allow new consumers to access the back-office services*. See Figure 1-6.

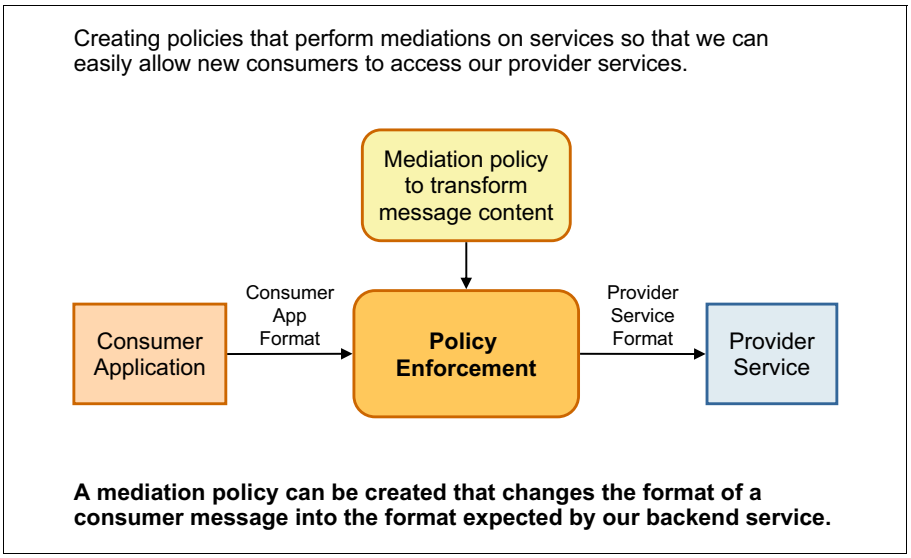


Figure 1-6 Allow new customers access to provider services

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

1.5.4 Reject low-priority traffic during business hours

In this use case, the requirement is for *using policy scheduling to prevent low priority applications from flooding the system during normal business hours, to meet the SLA's for the high priority, online services*. See Figure 1-7.

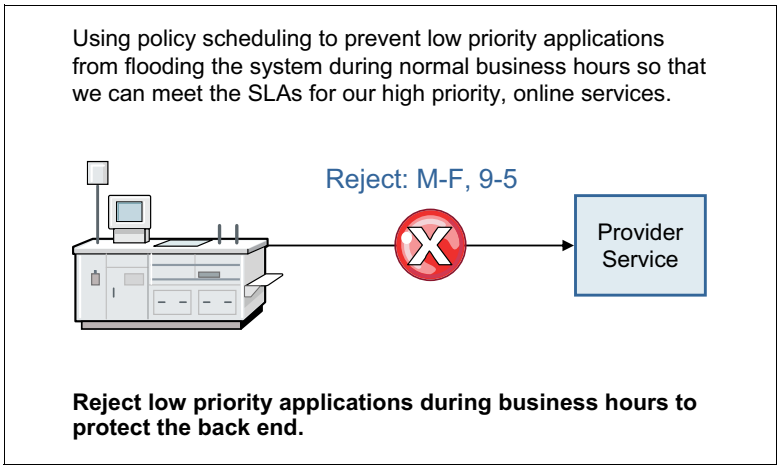


Figure 1-7 Reject low priority traffic during business hours

Policy can be scheduled by day-of-week and time-of-day, and also specified to start at a certain date or end on a certain date, or both start and end on a certain date. This feature can be used for a wide variety of use cases. One is stop traffic from low priority transactions such as batch jobs from interfering with the normal processing of provider service that has an SLA during normal business hours of Monday through Friday, 9 a.m. to 5 p.m.

1.5.5 Reroute traffic during maintenance windows

In this use case, the requirement is *reroute traffic during maintenance windows*. See in Figure 1-8.

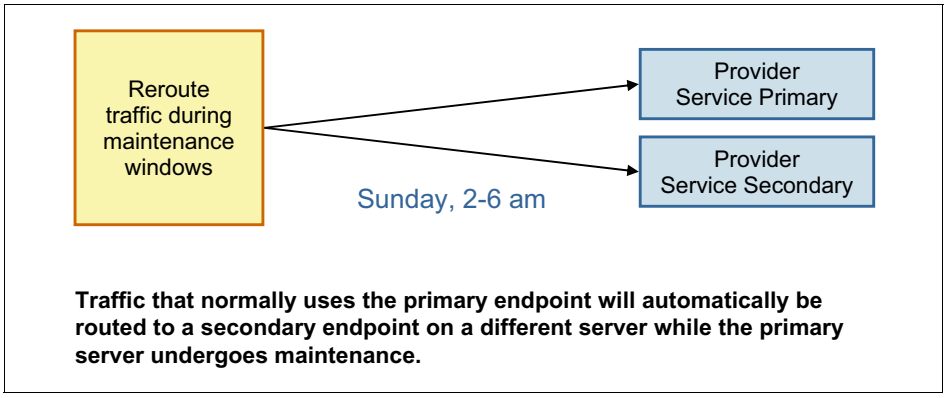


Figure 1-8 Reroute traffic during maintenance windows

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 that primary server is undergoing maintenance.

1.5.6 Deny access to rogue consumer applications

In this use case, the requirement is *identify anonymous consumers and reject those transactions to guard against rogue applications and denial of service attacks*. See Figure 1-9.

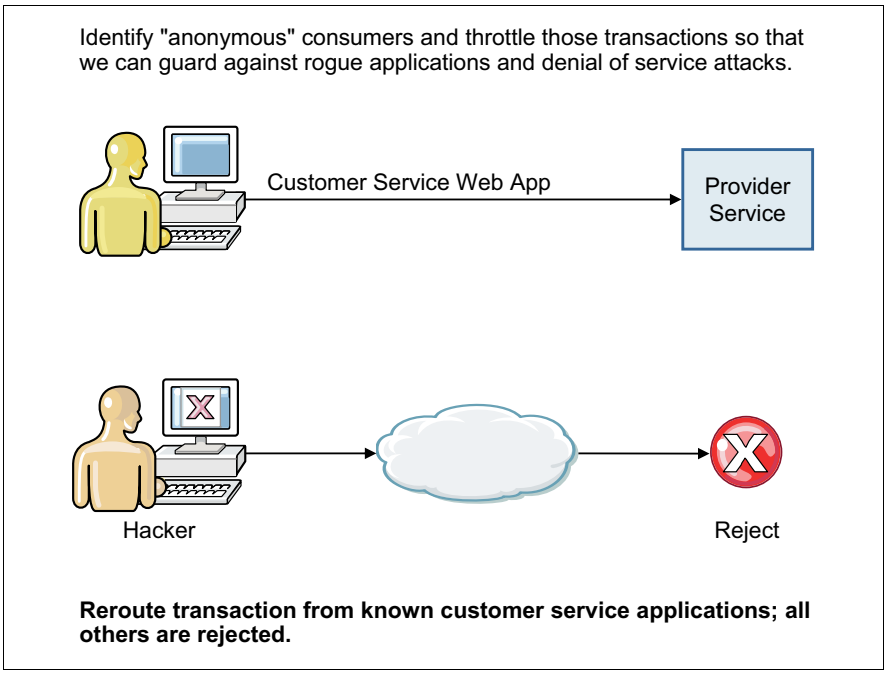


Figure 1-9 Deny access by rogue applications

Rogue applications are those that are unknown to the provider service. In many cases, that is acceptable, and the provider service can handle transactions no matter what consumer it originates from. In other cases, access to the provider service might need to be “locked down.” This is a good way, for example, to enforce an SOA governance policy that indicates only registered consumer applications may access the provider service. Another usage is to protect against denial of service attacks where criminals attempt to flood a provider service with so many transactions that the provider does no have time to process the legitimate requests. By throttling such criminal requests with policy, the back-end provider service is protected.

1.5.7 Service versioning support

In this use case, the requirement is c*reate policy that supports the transition from an old version of a service to a new version, to easily manage rolling out service changes*. See Figure 1-10.

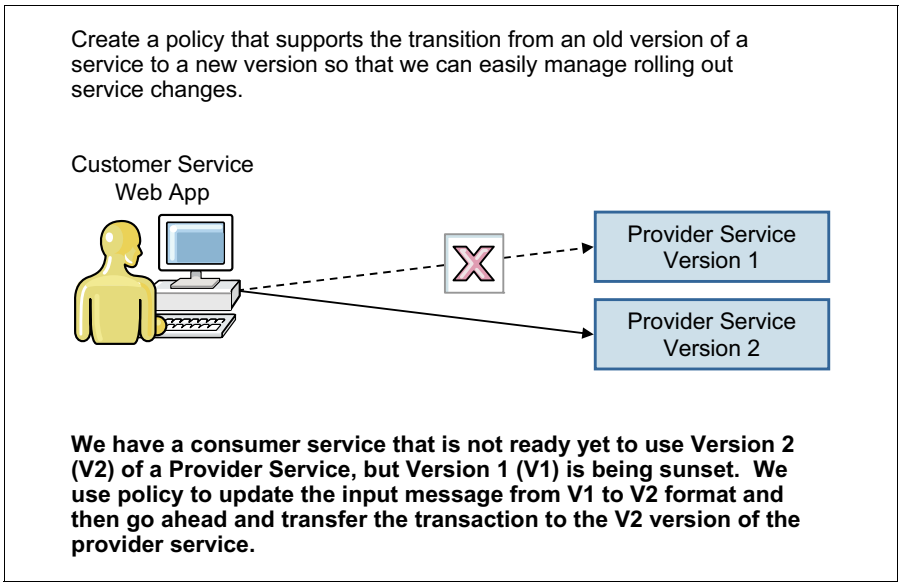


Figure 1-10 Service versioning support

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

1.5.8 Apply standard security to access provider services

In this use case, the requirement is *create a standardized security policy that can be applied to all services for an organization to easily manage and govern the service security.* See Figure 1-11.

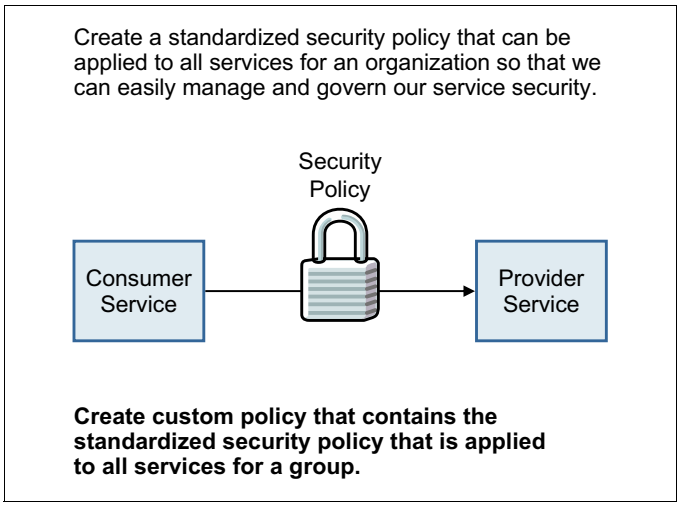


Figure 1-11 Standard security to all transactions

One of the powerful aspects of the SOA Policy Solution is the ability to create custom policy. The custom policy can be anything that WebSphere DataPower can do with policy, including its powerful security capabilities. For example, a standard authentication, authorization and audit (AAA) policy for user ID and password validation may be specified through 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.

1.5.9 Provide operational status and alerts

In this use case, the requirement is for *creating policies that monitor operations results and automatically take action so that operations is proactive instead of reactive*. See Figure 1-12.

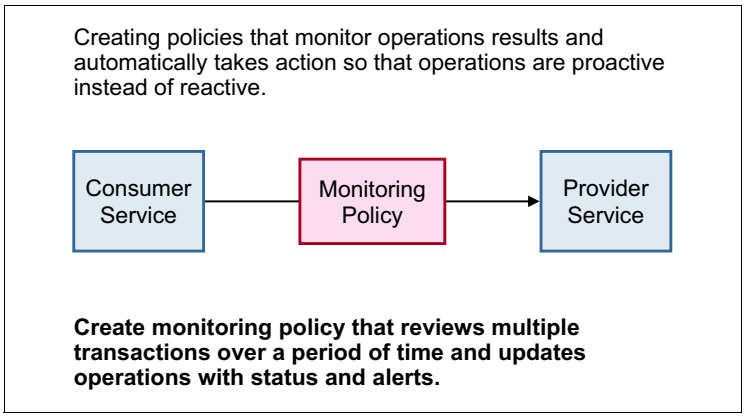


Figure 1-12 Operational status and alerts

Monitoring policy is used to perform traffic management while reviewing multiple transactions over a time period, typically five minutes. In this manner, operations can be alerted when they need to review where things stand operationally or even take action.

1.5.10 Automatically apply policy to new services

In this use case, the requirement is for *applying standardized policies dynamically so that policy is applied seamlessly to new services*. See Figure 1-13.

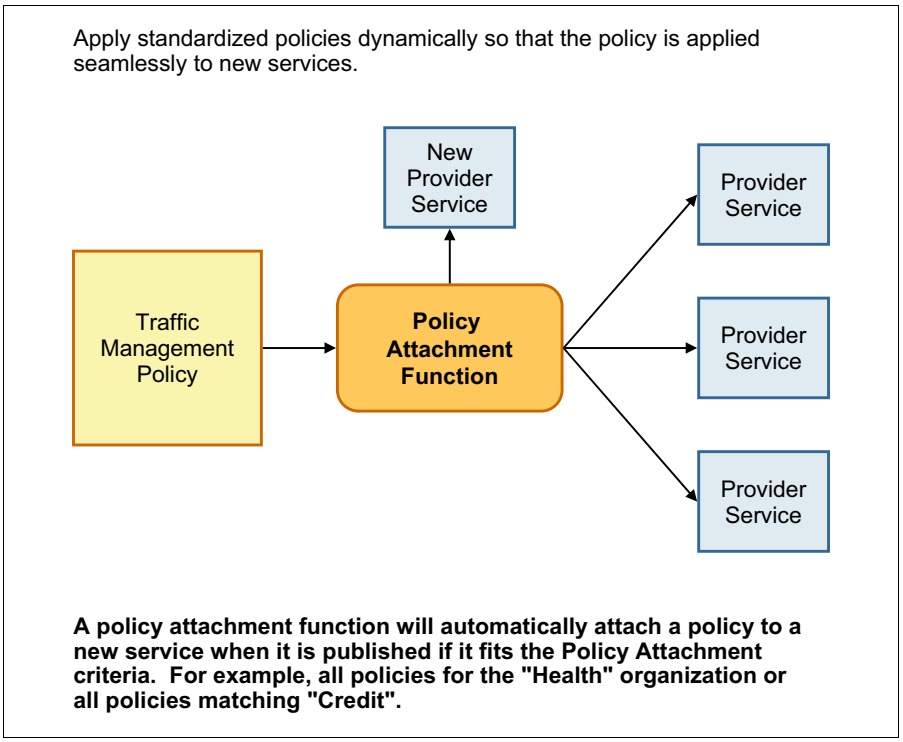


Figure 1-13 Automatically apply policy to new services

One of the capabilities of the SOA Policy Solution is the ability to create a policy attachment function query that identify all of the services that should be subject to (attached) to a policy. This allows any new service that meets the criteria specified in the policy attachment function to automatically and dynamically to be attached to the policy without further action on the part of the user.

1.5.11 Provide a standardized policy group for services

In this use case, the requirement is for *creating and applying a group of policies as a standard that all services for a certain group must follow.* See Figure 1-14.

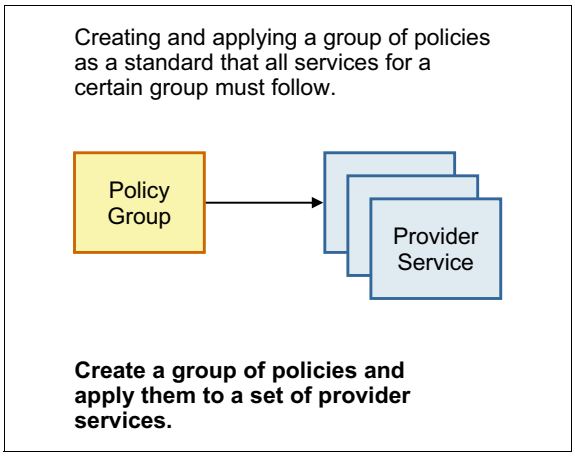


Figure 1-14 Standardized policy group for services

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

1.6 How to use this book

This book is organized in a way that offers you benefits of reading it from beginning to end (preferred), but it is also useful to read the sections on an as-needed basis.

- Part 1, “Introduction” on page 1 introduces the SOA Policy Solution and business case for using it:
  - Chapter 1, “The SOA Policy Solution” on page 3
  - Chapter 2., “Business case for using the SOA Policy Solution” on page 25. It is useful for all readers, but especially those who are interested in learning the overall capabilities of the solution and need to justify their usage with a business case.
- Part 2, “Policy examples” on page 33 contains examples and is the best section to read to gain an understanding of how to implement policies and get them attached to the services that need them.
  - Chapter 3, “Policy traffic management, provider only, with operations” on page 35 and Chapter 4, “Policy traffic management and consumer provider pairs” on page 87 describe how to govern, author, and manage the policies and the associated services for traffic management enforcement.
  - Chapter 5, “Versioning with custom policy” on page 119 and Chapter 6, “Security using custom policy” on page 187 delve into the realm of creating custom policy from WebSphere DataPower and uses service versioning and security as the examples, for those chapters.
  - Chapter 7, “Policy monitoring” on page 235 explains how to correctly use Monitoring Policy for use with Tivoli ITCAM for Applications.

- Part 3, “Policy administration point” on page 261 and Part 4, “Policy enforcement point” on page 345 should be used as reference information. When you want to read about how to do certain tasks for the SOA Policy Solution, these parts are helpful.

  Part 3, “Policy administration point” on page 261 is focused on WSRR, which functions as the policy administration point. As such, it is the centralized policy repository and also the service repository.

  - Chapter 8, “WebSphere Service Registry and Repository for traffic management” on page 263 focuses on policy authoring for policies that are enforced by WebSphere DataPower.
  - Chapter 9, “WebSphere Service Registry and Repository for monitoring policy” on page 277 describes policy authoring for monitoring policies for ITCAM for Applications.
  - Chapter 10, “Attaching a policy to a service” on page 313 specializes in the area of attaching policies to services.
  - Chapter 11, “Policy administration point utilities” on page 333 focuses on various utilities for the PAP, including restricting user access to policy functions, listing policies that are attached to services and vice versa.

  Part 4, “Policy enforcement point” on page 345 is focuses on WebSphere DataPower functions as the PEP, and ITCAM for Applications functions as the PMP.

  - Chapter 12, “DataPower policy enforcement point configuration” on page 347 tells how to set up and configure WebSphere DataPower as the PEP for the SOA Policy Solution.
  - Chapter 13, “Creating and using custom policies” on page 367 tells how to create custom policies in WebSphere DataPower and then put those policies into WSRR so that they can be attached to services.
  - Chapter 14, “ITCAM as policy monitoring point” on page 389 focuses on how to set up and configure ITCAM for Applications as the PMP.

- Part 5, “Appendixes” on page 413
  - Appendix A, “Implementing a SOA Policy Solution flow of work” on page 415 is best used by practitioners who complete the book through Part 2, “Policy examples” on page 33 of this book and want to make use of those examples in their own environment to gain experience in the solution.
  - Appendix B, “ITCAM monitoring attribute tables” on page 421 provides information about the attributes that can be considered for use in a SOA Policy Solution monitoring policy.
  - Appendix C, “Additional material” on page 427 describes the web material that accompanies this book. The material contains the examples (from Part 2, “Policy examples” on page 33) that you can download. This publication also has a business case setup in a Microsoft Excel spreadsheet that you can download.

    Download the files (examples and spreadsheet) from the following website before reading through this book:

    `ftp://www.redbooks.ibm.com/redbooks/SG248101/`

2

Business case for using the SOA Policy Solution

This chapter introduces the “Fictional IBM ITSO Redbooks Travel Company.” It explains the history, IT environment, and business goals and objectives for this company. The business case in this chapter uses this fictitious company, and the examples in Part 2 of this book are built upon this business case.

For an organization to receive permission from executive management to engage in an initiative, such as the SOA Policy Solution, it is often necessary to create a business case. This chapter explains the factors that are used in the construction of the business case and then presents the business case results. A file that includes the spreadsheet of the business case that is created is available in Appendix C, “Additional material” on page 427. This example might be useful in constructing a business case for your company or organization.

This chapter contains the following topics:

- 2.1, “Introducing the ITSO Redbooks Travel Company” on page 26
- 2.2, “Creating the business case” on page 28
- 2.3, “Business case results” on page 29
- 2.4, “Instructions for constructing your own business case” on page 30

© Copyright IBM Corp. 2013. All rights reserved.

2.1 Introducing the ITSO Redbooks Travel Company

This book covers a range of SOA policy examples that can arise based on an organization's maturity level, structure, and readiness for policy automation. Numerous use cases are provided in Part 2, "Policy examples" on page 33 to supplement the understanding of the SOA policy capabilities that are explained in detail in Part 3, "Policy administration point" on page 261 and Part 4, "Policy enforcement point" on page 345 of the book.

For most readers, a better understanding of the capabilities of SOA policy can be obtained by using the solutions and approaches shown with real-world context in Part 2. All of these examples relate to the background that is established in this chapter. Each chapter in Part 2, "Policy examples" on page 33 includes a step-by-step approach to supplementing the existing SOA policy implementation with additional capabilities.

2.1.1 History of the company

The ITSO Redbooks Travel Company is a diversified cruise ship operator with worldwide operations. The relevant history of the ITSO Redbooks Travel Company is as follows:

- The company started operations in the United States in 1953 in response to increased disposable income and leisure time demand from the American public.
- Since 1953, the company has expanded through organic growth and through mergers and acquisitions into a provider of cruise and related services in the Americas and Europe.
- The company recently concluded an agreement to merge with a medium-sized cruise operator in the Asia Pacific (AsiaPac) region. As a result, additional cruise itineraries and shore excursions will be entering the travel portfolio.
- Shore excursions are operated by third-party providers that are sub-contracted to the ITSO Redbooks Travel Company. These excursions are optional for customers on a cruise but are popular. Excursions benefit the company because they are high margin. They change rapidly based upon the ability of the company to negotiate more favorable margins with cheaper tour operators and in response to customer feedback. As a result, the company offers many changes to the itineraries, and the IT systems must be able to support these business needs.
- Based on demand for a specific boat sailing, cruise pricing can change often, sometimes several times a day.
- Reservations are made both by travel agents and directly by customers. Because there are many cruise competitors, response time must be reasonably good. In fact, the business now specifies the service level agreements (SLAs) in their non-functional requirements.

The ITSO Redbooks Travel Company is dedicated to making the leisure experience of customers an enjoyable experience. The company expects to differentiate itself based on outstanding service and to use this outstanding service to grow the business in the future. The growth strategy of the Chief Executive Officer (CEO) is recently focused on acquisition, and the company acquired a cruise operator that operates in Tahiti, Australia, New Zealand, the Philippines, Singapore, Thailand, Malaysia, Indonesia, Vietnam, China, South Korea, and Japan. This region is expected to be high growth and, therefore, is critical to the company's plans to grow revenue and profit. The common processes and functions that are core to the growth mission within IT are mandated to become agile and flexible, including being able to integrate new assets quickly into the ITSO Redbooks Travel Company operations.

In response to the agility mandate, the Chief Information Officer (CIO) initiated an effort to standardize and automate the runtime policies that are applied to services. The effort was

slow to start because of the company's size and, within the development and operations group, an attitude of "we've always done it this way." The CIO prefers to persuade first and command second. Therefore, the CIO initiated an SOA Policy Business Case to prove to the business and operations that the SOA policy initiative can save time and money. This case is described in 2.2, "Creating the business case" on page 28.

The economic downturn significantly decreased revenues in 2008 and consequently, in future years, put pressure on the company to contain costs and accomplish its mission better, faster, and cheaper.

2.1.2 The IT environment

The ITSO Redbooks Travel Company has a sizeable and diversified IT environment that must respond quickly to the needs of the business. Its technology stack includes a diverse set of platforms, hardware, and software. The IT group started to establish governance processes to ensure compliance with any standards that it decides to put in place. The company wants to establish a standard set of runtime policies that it can reuse for its services.

2.1.3 The business goals and objectives

The primary objective of the company is to become an agile organization that can quickly respond to competitive pressures. Cost pressure became more important as overall margins contracted, and the company is seeking to reduce its own costs. Because the organization is diverse and consists of many types of operations, governance and adherence to standards have gained a major emphasis, though is by no means the only area the company is seeking to make more efficient.

The IT group must handle resistance to its enterprise-wide initiatives, especially because many of the groups in the company do not like to change. The transformation must be made as easy as possible and must be able to demonstrate the benefits.

A number of business goals and objectives for the company were put in place by the business and IT leaders:

- Create standardized SLAs to manage the traffic against critical "reservations" services, to meet IT's promises to the business.
- Apply the standardized policies, which are agreed on, to all of the services that should be subject to those policies, so that as the mix of services changes, the standards policies still apply.
- Differentiate the service SLAs, depending on the level of service guaranteed for separate customer tiers, to provide the best customers with the best service.
- Determine the best actions to take when the SLAs are violated to be more proactive instead of reactive.
- Create policies that perform mediations on services, including schema validation and message transformation, so that new services can be added easily to the portfolio, no matter where they are from.
- Filter transactions based upon the consumer transaction to provide better service to high priority transactions, such as customer and travel agent web transactions.
- Identify "anonymous" consumers and take action based on that anonymity to guard against rogue transactions.
- Handle policy for different versions of a service to be agile and to change easily when needed.

- Create a special, customized policy that can be governed and attached to any service to use policy to accomplish other policies that should be standardized, such as authentication, authorization and audit (AAA) security.
- Monitor and report on policy execution results on multiple enforcement devices to check overall monitoring policy conformance.
- Use monitoring policies to let operations know that action must be taken to keep production running smoothly so that operations are proactive instead of reactive.

2.2 Creating the business case

The CIO asked the IT finance group to create a business case to demonstrate a positive or negative return on investment (ROI) for the implementation of the SOA Policy Solution. The finance group gathered the following information that is relevant for the business case:

- The cost of capital for the company is 8%. The company uses the standard financial metric of net present value (NPV), which is calculated as a timed series of cash flows. NPV is calculated each year as follows, where the letter “i” is cost of capital and the letter “t” is the number of years from the present:

  ```
  (benefits – costs) / (1+i)**t
  ```

  In other words, the present value of cash flow from the current year is more valuable than cash flow from the future.
- All initiatives of this type must show a payback (break even) period of three or fewer years and demonstrate an ROI of 30% or greater by the end of year five, by using conservative estimates.
- The Chief Financial Officer (CFO) must agree with and be prepared to instruct the finance group to measure all metrics that are used in the business case in future years to validate that those benefits are actually realized.
- There is a total initial setup cost for SOA policy to implement the SOA Policy Solution infrastructure. This includes any incremental cost for hardware, software, and labor hours to get the SOA Policy Solution initiated and ready for policy creation. The IBM Redbooks Travel Company needed to add a DataPower XI52 to its middleware portfolio so that the company could have the processing power to perform policy enforcement. IBM WebSphere Service Registry and Repository (WSRR) and IBM Tivoli Composite Application Manager (ITCAM) for SOA are already implemented, but an additional processor and license expense for WSRR is required. In addition, staff training to write policies and enterprise architecture hours to set standards is included. Total budgeted cost is $350,000 (US dollars).
- There is an Annual Maintenance and Overhead for SOA Policy cost to maintain the SOA Policy Solution infrastructure. Some of this is allocation of data center overhead costs, but there is also maintenance licensing for the XI52 and additional WSRR licensing. In addition, enterprise architecture taxed this project with an annual charge to continue to update policy standards. Annual cost is budgeted at $50,000.
- There is an SOA Policy Solution cost to add and attach policies to services. This is a labor cost to create and test each new policy and attach to the services needed. This task is much easier to do with the SOA Policy Solution than writing the policy in the middleware. In the proof of concept (PoC) that the company engaged in last month, it was determined that a policy that was created by using the SOA Policy Solution cost $2,500 to write, test and attach to the services, which needed it. The same work for the cost of creating the policy the “old” way, by writing the policy directly in the middleware, was determined to cost $5,000. The difference in cost was partly because of the reduced hours to write the policy in a centralized policy administration point, instead of creating policy directly in the

middleware. Part of the savings also occurred because of the higher cost of engaging additional middleware engineers in the direct middleware method.

- There is a savings because of the policy reuse percentage (`Policy reuse%`). Policies may be reused and attached to multiple services. In fact, the SOA Policy Solution allows policies to be dynamically attached to a set of services based on criteria specified in the policy administration point (PAP). For the travel company, the Enterprise Architecture group was consulted and agreed to have zero reuse in the first year and 25% reuse in subsequent years. The group accepted that these are conservative figures, and that reuse will probably be higher.
- The `Cost of maintaining each policy the "old" way` acts as a credit per policy for yearly maintenance. It Is simply an average cost for the middleware engineer to maintain the policies directly in the middleware. Another way to think about this cost is how much will be saved by freeing the DataPower engineer from policy work to be able to do other work. This worked out to $500 per policy for the company.
- The figures relate to cost savings of creating policies and applying them in an efficient manner with the SOA Policy Solution. An additional savings is associated with the ability of the company to get to market faster, therefore driving incremental revenue sooner. In addition, customer satisfaction will be higher because of having SLAs that protect the speed of the user experience. For the business case, this benefit is known as the *benefit of enterprise agility* created by SOA Policy Solution. The CFO agreed that this benefit was a “soft” number, but asked the finance group to calculate a reasonable result. The finance group also decided to be conservative and estimated that there was a yearly benefit of $25,000 that is realized in incremental revenue from getting new products and enhancement to market faster, and $25,000 that is realized in retained customer revenue because of higher customer satisfaction from the better maintained response times. The CIO argued that the $50,000 total number was low, but decided to accept the ruling of the finance group.

2.3 Business case results

The finance group created a spreadsheet named *SOA Policy Solution Business Case ROI*. The results are shown in Table 2-1and Table 2-2 on page 30.

Spreadsheet available for download: You can download this spreadsheet in XLS format:

`ftp://www.redbooks.ibm.com/redbooks/SG248101/`

Table 2-1 Assumptions

Assumption	Input
Cost of capital equals	8%
Total initial setup cost for SOA policy	350,000
Annual maintenance and overhead SOA policy	50,000
SOA Policy Solution cost to add and attach policies to a service	2,500
Policy reuse%	25%
Cost of creating policy the “old” way (savings)	5,000
Cost of maintaining each policy the “old” way (savings)	500
Benefit of enterprise agility created by SOA Policy Solution	50,000

In Table 2-2, consider the following information:

- NPV is net present value: All adjusted amounts use cost of capital for time adjustment
- All costs are shown as negative number (in parenthesis and the color red)

Table 2-2 IBM Redbooks travel company business case

Year	New policies needed per year	Policies created with reuse	SOA policy development cost	Service development savings	Service maintenance savings	Incremental revenue because of agility	Service savings + revenue - cost	NPV savings + revenue - cost	Cumulative NPV benefit	SOA policy accumulated cost	NPV accumulated cost	ROI
0; start	0	0	(350,000)	0	0	0	(350,000)	(350,000)	(350,000)	(350,000)	(350,000)	0%
1	60	60	(162,500)	300,000	30,000	50,000	217,500	201,389	(148,611)	(512,500)	(474,537)	-31%
2	20	15	(78,125)	100,000	40,000	50,000	111,875	95,915	(52,696)	(590,625)	(506,366)	-10%
3	20	15	(78,125)	100,000	50,000	50,000	121,875	96,748	44,052	(668,750)	(530,875)	8%
4	20	15	(78,125)	100,000	60,000	50,000	131,875	96,932	140,984	(746,875)	(548,975)	26%
5	10	8	(64,063)	50,000	65,000	50,000	100,938	68,696	209,680	(810,938)	(551,910)	38%
6	10	8	(64,063)	50,000	70,000	50,000	105,938	66,759	276,439	(875,000)	(551,398)	50%
7	10	8	(64,063)	50,000	75,000	50,000	110,938	64,731	341,170	(939,063)	(547,934)	62%
8	10	8	(64,063)	50,000	80,000	50,000	115,938	62,637	403,807	(1,003,125)	(541,957)	75%
9	10	8	(64,063)	50,000	85,000	50,000	120,938	60,499	464,306	(1,067,188)	(533,859)	87%
10	10	8	(64,063)	50,000	90,000	50,000	125,938	58,333	522,640	(1,131,250)	(523,988)	100%
Totals			(1,131,250)	900,000	645,000	500,000	913,750					

The business case proved breaking even at 2.5 years, and a 38% ROI after five years. Further, the ROI improves to 100% after 10 years.

The CFO reviewed and accepted the results and released $350,000 in funding to the CIO to initiate and complete the project.

2.4 Instructions for constructing your own business case

The following steps summarizes the necessary tasks for creating the business case for SOA Policy Solution usage at your organization. Of course, your finance group might have its own process, but think how impressed they will be when you have your own business case as a starting point and know what net present value means.

1. Download the SOA Policy Solution business case spreadsheet (see 2.3, “Business case results” on page 29 and the instructions in Appendix C, “Additional material” on page 427).
2. Determine the `Cost of Capital` that is used at your organization. Your IT or corporate finance group will know, but if not or if you cannot communicate with them, use the default of 8%, which is reasonable in these days of low interest rates. Enter this value in cell E6 as either a number with a percent sign at the end (for example 6.5%) or as a decimal value (for example 065).
3. Determine the `Total Initial Setup Cost for SOA Policy`. To do this step, work with your infrastructure group and determine any incremental cost for hardware, software, and labor hours to get the solution setup, but not any cost associated with actual creation of the policy. As the example shows, you must consider any cash that is necessary for additional

hardware (a DataPower XI52, for example) or budgetary output that is necessary to use existing hardware for this project. Software costs are licensing fees, but those might already be expended for previous projects. Be sure to budget labor hours for configuring and setting up the SOA Policy Solution. Enter the resultant total cost in cell E8 as an integer number.

4. Determine the `Annual Maintenance and Overhead for SOA Policy` cost to maintain the SOA Policy Solution infrastructure on a yearly basis. Several factors are probably part of this calculation. You might have some budgeted overhead costs, although one can make the argument that these costs occur anyway and incremental overhead is zero. However, cash costs must be accounted for, including any maintenance licensing costs. You might also want to consider amortizing future hardware needs. For example, you might need an additional DataPower in three years and should divide that cost by 3 and add it into this value. Enter the resultant total cost in cell E9 as an integer number.
5. Determine the `SOA Policy Solution cost to add & attach policies to services` information. This is a labor cost per policy to create and test each new policy and attach to the services needed. A necessary task is to identify the amount of time necessary to write a policy, get it through the policy lifestyle, and then attach that policy to the services to which it needs to be applied. You might be able to do this task in your sandbox or equivalent environment. Multiply this number by the loaded labor rate for the types of personnel that will do this work (this “loaded labor rate” includes salary, benefits, and other corporate overhead). Usually the finance group can tell you what the load rate is. Enter the resultant cost in cell E10 as an integer number.
6. Determine the `Policy reuse%` value. There is a savings because of the policy reuse percentage. Policies may be reused and attached to multiple services. This percentage will decrease the total cost of the previous item because it implies a lesser number of policies to be added. This metric can be determined only with experience; but an estimate will have to suffice. The best approach is for the architects who are involved to plan and consider the types of policies and how the policies might be used. Enter this value in cell E11 as either a number with a percent sign at the end (for example 25%) or as a decimal value (for example 25).
7. The `Cost of creating policy the "old" way` acts as a *credit per policy add* compared to the cost of adding policy with the SOA Policy Solution. The same work for the cost of creating the policy the “old” way (by writing the policy directly in the middleware) will cost more. The loaded labor rate for a DataPower engineer will usually be higher than for someone who will write policy in WSRR. Multiply the loaded labor rate by the number of hours per policy and enter the result as an integer in cell E12.
8. The `Cost of maintaining each policy the "old" way` also acts as a credit per policy for yearly maintenance. It is an average cost for the middleware engineer to maintain the policies directly in the middleware. Another way to think about this is how much will be saved by freeing the DataPower engineer from policy work to be able to do other work. Enter this figure as an integer in cell E13.
9. The `Benefit of enterprise agility created by SOA Policy Pattern` measures the additional savings associated with the ability of to get to market faster, therefore driving incremental revenue sooner. In addition, it can measure the customer satisfaction from faster service. The best approach here is to work with the finance group to find business projects that will benefit from faster implementation and calculate a percentage of the incremental revenue. If that approach is not possible, enter 0 (zero) here. Enter this figure as an integer in cell E14.
10. Enter the `New Policies needed per Year` in cells C22 - C31 for years 1 - 10. The best way is to calculate this number based on working with all stakeholders, especially the Business Users and Enterprise Architect.

Part 2

Policy examples

This part contains the following chapters:

- Chapter 3, “Policy traffic management, provider only, with operations” on page 35
- Chapter 4, “Policy traffic management and consumer provider pairs” on page 87
- Chapter 5, “Versioning with custom policy” on page 119
- Chapter 6, “Security using custom policy” on page 187
- Chapter 7, “Policy monitoring” on page 235

© Copyright IBM Corp. 2013. All rights reserved.

3

Policy traffic management, provider only, with operations

This chapter focuses on policy management only for the case where policy must be applied to provider services. The case where policy needs to be applied to a consumer-provider service pair is the subject of the next chapter. The intent of this chapter, therefore, is to provide a base set of information to help you register and govern provider services, author policies, attach policies to the provider services, and report on policies.

This chapter contains the following topics:

- 3.1, “Implement the provider services scenario” on page 36
- 3.2, “Service governance” on page 54
- 3.3, “Govern existing services” on page 56
- 3.4, “Creating policies” on page 73
- 3.5, “Attaching policies” on page 78
- 3.6, “Reporting on services and policies applied to services” on page 84

© Copyright IBM Corp. 2013. All rights reserved.

3.1 Implement the provider services scenario

This book uses “Fictional IBM ITSO Redbooks Travel Company” as a scenario. The company offers four services, which are shown in Figure 3-1 on page 38 and described in Table 3-1 on page 38:

- Itinerary Reservation Business Service
- Itinerary Availability Business Service
- Pricing Business Service
- Itineraries Batch Update Business Service

The company purchased and installed IBM WebSphere Service Registry and Repository (WSRR), WebSphere DataPower, and IBM Tivoli Composite Application Manager (ITCAM) for SOA. This section outlines the sequence of steps for the fictional company to act as a service provider, implementing policies, service gateway, policy enforcement and management, and govern them appropriately.

This chapter explains how to do the following tasks:

- Validate product installation
- Validate services backend
- Integrate products in pattern (WSRR, DataPower, and ITCAM)
- Govern existing services
- Create a business service with capability service version
- Create service level definitions (SLDs)
- Create policies
- Attach policies to SLDs for desired services
- Set up a service gateway in DataPower
- Promote the services to WSRR run times
- Synchronization with DataPower
- Run the client transactions to validate the provider services

For the purposes of the scenario, the assumption in this book is that all required software is installed. The scenario provides details for ensuring that the installation is working correctly and configures the various products to interact securely with each other.

3.1.1 Verify product installations

Verify the installations of WSRR, DataPower, and ITCAM for SOA in your environment. Sample URLs that we used in the IBM lab environment are provided in this section. Login URLs are provided as a reference after product installation. These URLs are the main access links (Business Space, Service Registry, and administrative console) that are used to do the various activities that are described in this book.

WSRR 8.0

Complete the following steps to verify the WSRR installation; in these steps, *wsrrhost* is `govmasterwsrr`, `stagingwsrr`, or `productionwsrr`:

1. Log in to WSRR Business Space:

   `http://`*wsrrhost*`:9080/BusinessSpace`

2. Log in to Service Registry:

   `http://`*wsrrhost*`:9080/ServiceRegistry`

3. Log in to WSRR WebSphere Application Server administrative console:

   `http://`*wsrrhost*`:9060/admin`

For detailed product installation and verification steps, see the following information center:

`http://pic.dhe.ibm.com/infocenter/sr/v8r0/index.jsp`

DataPower 5.0

Log in to the DataPower administrative console:

`https://datapower:9090`

For more information, see the information center:

`http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/index.jsp`

ITCAM for SOA 7.1.1

Access ITCAM for SOA:

`http://itcam:1920 -> IBM Tivoli Enterprise Portal Webstart Client`

For more information, see the information center:

`http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=%2Fcom.ibm.itcama.doc_7.1%2Fwelcome_itcamfapps71.html`

3.1.2 Validate existing services

IBM Redbooks Travel Company offers four business services. These services are deployed to WebSphere Application Server on Linux. Validate that your services back end is functional. You can use existing client applications or SOAPUI to validate the proper response from the services.

Figure 3-1 shows services that are currently offered by the company.

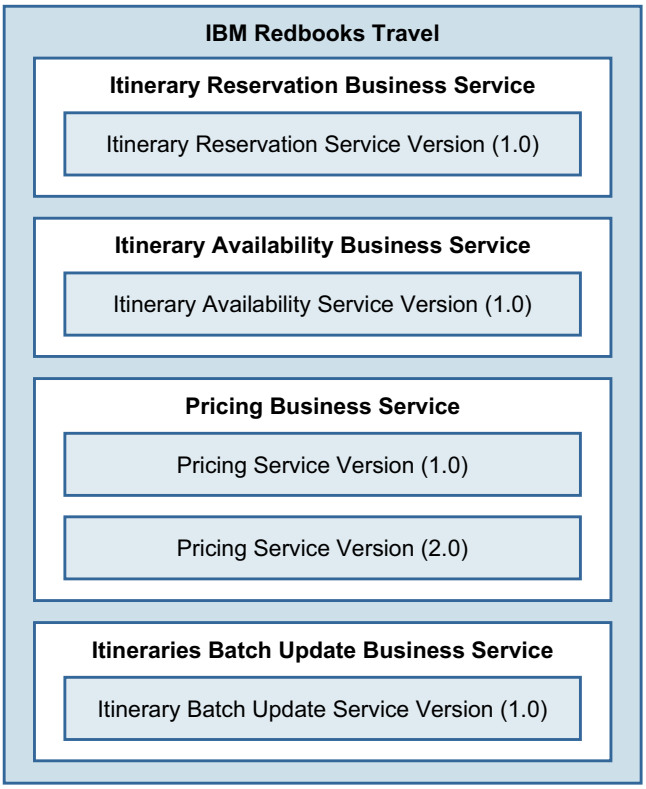


Figure 3-1 Provider Services

Table 3-1 describes the functionalities of each service.

Table 3-1 Service functionality

Provider services	Description
Itinerary Reservation Service	Is built with CRUD (create, retrieve, update, delete) operations to manage itineraries. This service also invokes ItineraryAvailability Service to check the inventory before making a reservation.
Itinerary Availability Service	Has read operation (available method) to check the inventory availability.
Pricing Service (Versions 1.0 and 2.0)	Has read operation (getPrice method) to retrieve price.
Itineraries Batch Update Service	Has update operation to process batch itineraries coming from business partners.

You can download the sample services that are used in this book or use your own services. Appendix C, “Additional material” on page 427 has information about downloading and setting up the sample application.

3.1.3 Integrate products in the pattern

Figure 3-2 show an overview of the topology that is used by IBM Redbooks Travel Company.

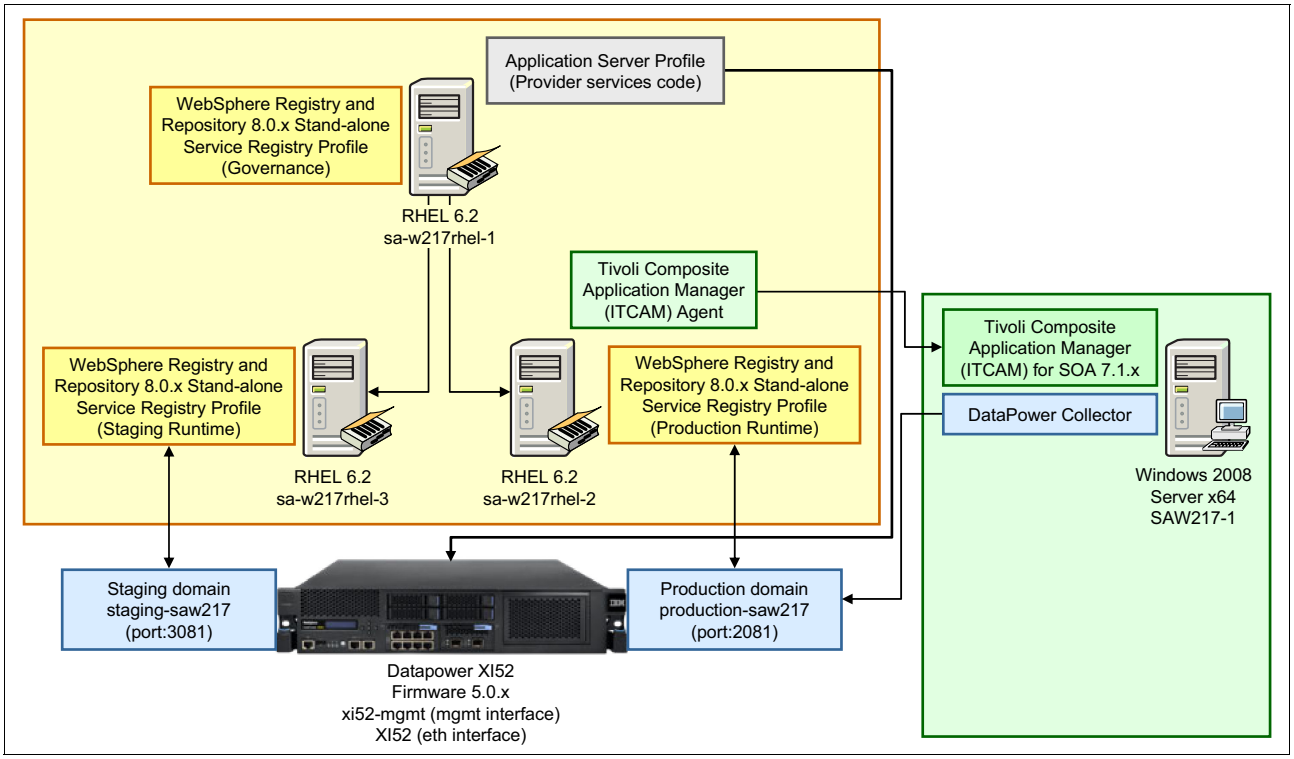


Figure 3-2 Infrastructure topology for the provider

The provider infrastructure is built with the following components:

- WSRR as the policy administration point (PAP)
  - One server for Governance Master
  - One server for Staging run time
  - One server for Production run time and ITCAM agent

  In our topology, we monitored only the production runtime server. In customer environments, the ITCAM agent can be installed on the servers that the customer wants to monitor. We are demonstrating the topology we used.
- DataPower as the policy enforcement point (PEP)
  - One staging domain
  - One production domain

  One DataPower appliance was used. Two separate domains were created, each with one HTTP Front Side Handler (FSH) listening on separate ports. Staging FSH listens on port 3081 and production FSH listens on port 2081.
- ITCAM for SOA as the policy monitoring point
  - One server with ITCAM and DataPower collector was used.
- WebSphere Application Server
  - Web services are deployed. A separate WebSphere Application Server profile was created on the same server as Governance Master and deployed the services WAR files. Typically, provider services reside on a separate WebSphere Application Server cluster.

Integrating WSRR Governance Master and WSRR run times

The Governance Master registry is the administration point of services, policies, and other artifacts. Governance Master and WSRR run times must be integrated to promote any configuration changes from Governance Master to the runtime environments. Assuming that you installed WSRR and created WSRR Profiles for each of the three instances, but did no other configuration, complete the following steps.

1. The Load WSRR Governance Enablement Profile action must be completed on all three WSRR servers after installation to activate the governance enablement profile. The easiest way to do this step is to select **Load WSRR Governance Enablement Profile** from the IBM WebSphere Service Registry FirstSteps wizard, which opens automatically after profile creation, as shown in Figure 3-3. This process can take several minutes, during which time you should refrain from using the registry.

Figure 3-3 Load the governance enablement profile

2. Log in to the WSRR Governance Master Business Space at a URL such as in the following line:

   `https://productionwsrr:9443/BusinessSpace/`

   For example, in the environment that is used for writing this scenario, the URL is as follows:

   `https://sa-w217rhel-1.itso.ral.ibm.com:9443/BusinessSpace/`

3. Create business spaces for each of the templates that are shown in Figure 3-14 on page 56:
   a. Click **Manage Spaces** at the top left.
   b. In the pop-up window, click **Create Space**.
   c. Enter a suitable name for each space, for example, `IBM Redbooks Travel Service Registry for Business`. Under Create a new space using a template, select the corresponding template, for example, **Service Registry for Business**.
   d. Optional: Set a style and icon.
   e. Click **Save**.
4. Repeat these steps 1 - 3 for Service Registry for Development, Service Registry for Operations and Service Registry for SOA Governance.
5. In this scenario, the following servers are used:
   – WSRR Governance Master (Host: `govmasterwsrr`)
   – WSRR Staging run time (Host: `stagingwsrr`)
   – WSRR Production run time (Host: `productionwsrr`)

   Establish the trust relation between the Governance Master and run times:
   a. Log in to the Governance Master WebSphere Application Server administrative console (`http://govmasterwsrr:9060/admin`) and load the staging WSRR runtime SSL certificate. Use the following selection sequence to navigate:

      Security → SSL Certificate and Key management → Keystores and Certificates → NodeDefaultTrustStore → Signer Certificates → Retrieve from port
   b. Specify the following values:
      - Host name: `stagingwsrr`
      - Port: `9443`
      - Alias: `stagingwsrr`
   c. Click **OK**.
   d. Click **Retrieve from port** and specify the following values:
      - Host name: `productionwsrr`
      - Port: `9443`
      - Alias: `productionwsrr`
   e. Click **OK** and **Save**.
   f. Log in to the WSRR Staging WebSphere Application Server administrative console (`http://stagingwsrr:9060/admin`) and load the WSRR Governance Master server SSL certificate. Use the following selection sequence to navigate:

      Security → SSL Certificate and Key management → Keystores and Certificates → NodeDefaultTrustStore → Signer Certificates → Retrieve from port
   g. Specify the following values:
      - Host name: `govmasterwsrr`
      - Port: `9443`
      - Alias: `govmasterwsrr`
   h. Click **OK** and **Save**.
   i. Log in into WSRR production WebSphere Application Server administrative console (`http://productionwsrr:9060/admin`) and load the WSRR Governance Master server SSL certificate.

Use the following selection path to navigate:

Security → SSL Certificate and Key management → Keystores and Certificates → NodeDefaultTrustStore → Signer Certificates → Retrieve from port

j. Specify the following values:

- Host name: `govmasterwsrr`
- Port: `9443`
- Alias: `govmasterwsrr`

k. Click **OK** and **Save**.

6. Set up the promotion properties in WSRR Governance Master:

a. Log in to WSRR Governance Master Service Registry:

`http://govmasterwsrr:9080/ServiceRegistry`

b. If necessary, switch to the Configuration perspective.

Select **Active Profile → Promotion → PromotionProperties**.

c. Erase the sample environments and replace them with those in Example 3-1, and correct for your environment.

Example 3-1 WSRR Promotion Properties

```
<environment
name="http://www.ibm.com/xmlns/prod/serviceregistry/6/1/GovernanceProfileTaxonomy#St
aging">
    <servers>
        <server name="stagingwsrr" port="2809"/>
    </servers>
    <promotion>
        <type>sync-optimized</type>
    </promotion>
    <security enabled="true">
        <wsrrUser>wasadmin</wsrrUser>
        <wsrrRealm>defaultWIMFileBasedRealm3</wsrrRealm>
        <wsrrPassword>yourPassword</wsrrPassword>
    </security>
</environment>

<environment
name="http://www.ibm.com/xmlns/prod/serviceregistry/6/1/GovernanceProfileTaxonomy#Pr
oduction">
    <servers>
        <server name="productionwsrr" port="2809"/>
    </servers>
    <promotion>
        <type>sync-optimized</type>
    </promotion>
    <security enabled="true">
        <wsrrUser>wasadmin</wsrrUser>
        <wsrrRealm>defaultWIMFileBasedRealm2</wsrrRealm>
        <wsrrPassword>yourPassword</wsrrPassword>
    </security>
</environment>
```

d. By default, the transitions in the Promotion Properties are commented out. Remove the comment delimiters (`<!--` and `-->`) from around the `<transition>` tags.

The three WSRR servers are now sucessfuly configured for promotion. This information can be verified by the creating a concept and transitioning it through the SOA Lifecycle. The result is that the concept is promoted to the Staging and Promotion registries.

Use the following steps to verify:

1. Log in to WSRR Governance Master Service Registry (In this scenario, `http://govmasterwsrr:9080/ServiceRegistry`) and switch to the Administrator perspective.
2. Navigate to **View** → **Concepts,** and then click **New.**
3. Give the Object an artibary name, for example, `Test Concept` and click **Finish**.
4. Click the **Governance** tab.
5. Select **Initiate SOA Lifecycle**, and then click **Govern**.
6. Select the following transitions in order, and click **Transition** after each selection.
   a. Propose Scope
   b. Approve Scope
   c. Propose Plan
   d. Approve Plan
   e. Propose Specification
   f. Approve Specification
   g. Propose Realization
   h. Approve Realization
   i. Propose Staging Deployment
   j. Approve Staging Deployment (this will promote to the Staging registry)
   k. Propose Certification
   l. Approve Certification
   m. Propose Production Deployment
   n. Approve Production Deployment (this selection promotes to the Production registry)

Figure 3-4 shows the activity log of the test concept.

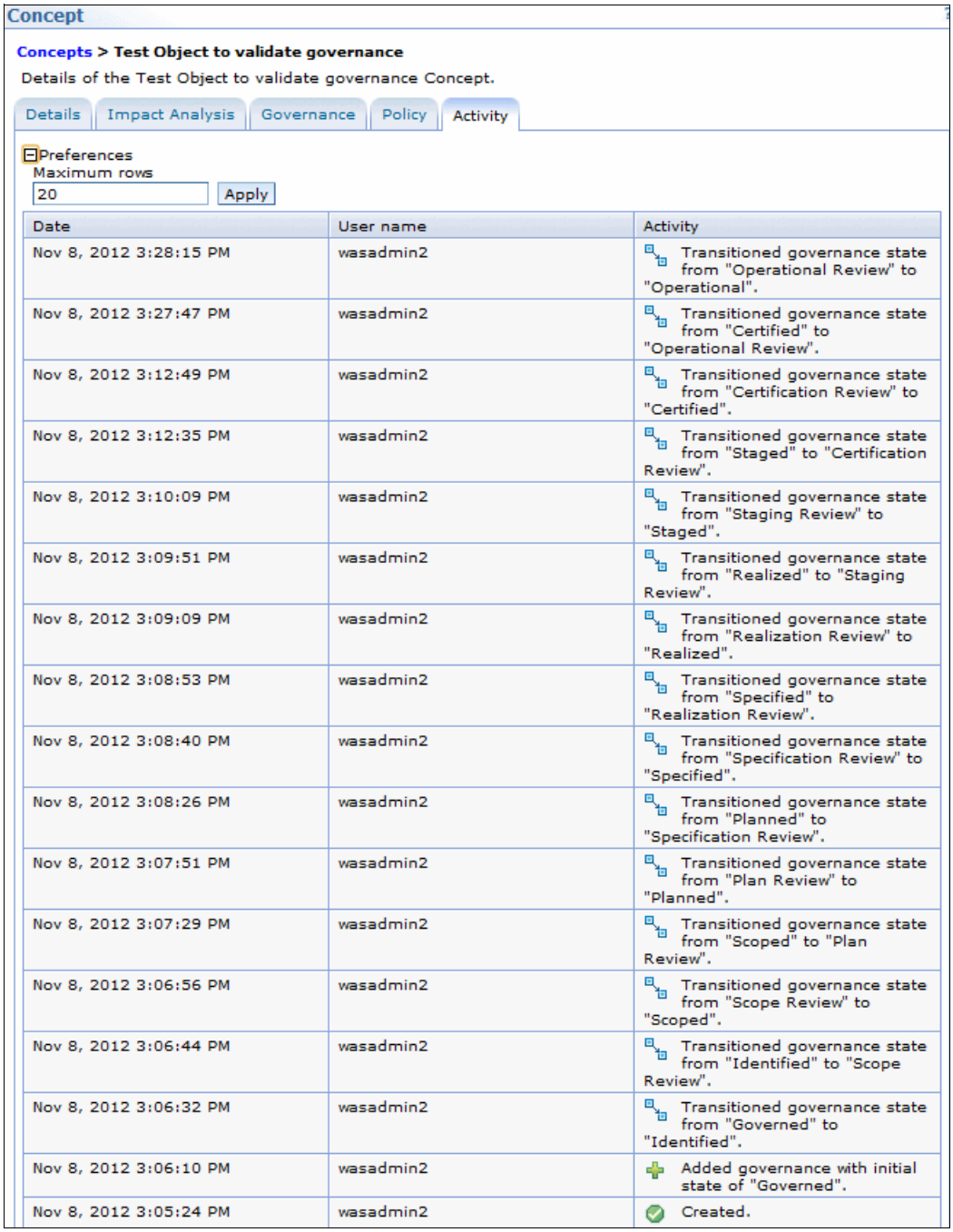


Figure 3-4 Successful governance transitions of the test object

Integrating WSRR run time and DataPower

WSRR runtime servers must be integrated with DataPower for successful policy enforcement. No integration is required between the Governance Master and DataPower.

The WSRR promotion process deploys artifacts from Governance Master to WSRR runtime environments. To propagate the artifacts (WSDL, policies, and so on) from the WSRR run time to DataPower, both components must be appropriately configured.

The Staging WSRR should be integrated with staging domain in DataPower and the Production WSRR should be integrated with production domain in DataPower. These domains can, but do not need to, reside on the same DataPower appliance. Steps are listed here to integrate with the production environment. You must repeat the steps in the staging environment.

DataPower configuration

To configure DataPower to communicate with a secure instance of WSRR, the SSL certificate from the Application Server that hosts WSRR, or a web server if a webserver is being used, must be imported correctly in DataPower, and the WSRR server must be defined in DataPower. Complete the following steps:

1. Acquire the certificate from WSRR. Log in to the WebSphere Application Server administrative console for the Production WSRR. The URL should be something like the following example:

   `https://productionwsrr:9043/ibm/console/`

2. Navigate to **Security** → **SSL certificate and key management** → **Key stores and certificates** → **NodeDefaultKeyStore** (typically) → **Personal certificates**.
3. Select the check box and click **Extract**.
4. Enter a file name and path that are suitable for the operating system that is running the WSRR server. and select **Binary DER data** in the Data Type drop-down menu, as shown in Figure 3-5.

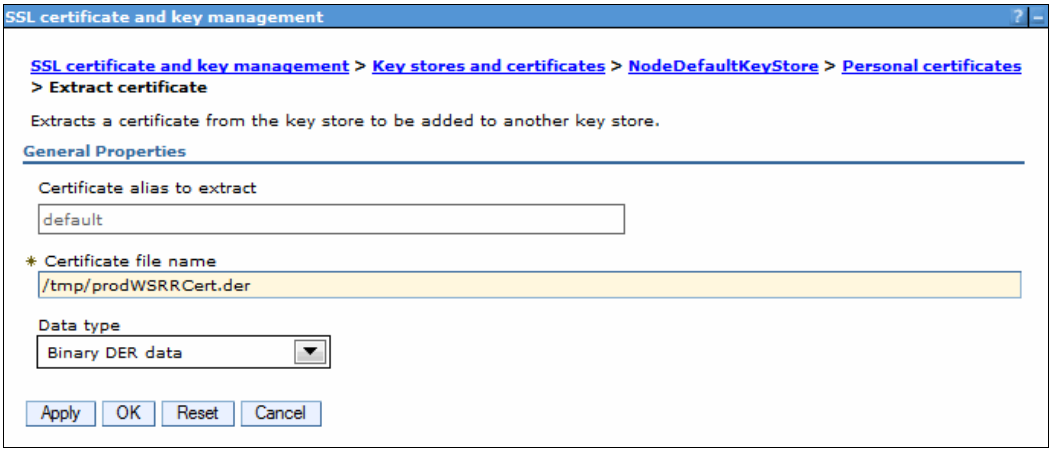


Figure 3-5 Extract SSL certificate

5. Click **OK**. The certificate is saved to the file system of the WSRR machine.
6. Copy the file from the WSRR machine to where you are running your browser.
7. Log in to your DataPower at a URL (such as `https://datapower:9090/`) and select the domain that you want to use for your production environment.

8. Navigate to **Objects** → **Crypto Configuration** → **SSL Proxy Profile**. On the next page, click **Add**. The Configure SSL Proxy Profile page opens (Figure 3-6).

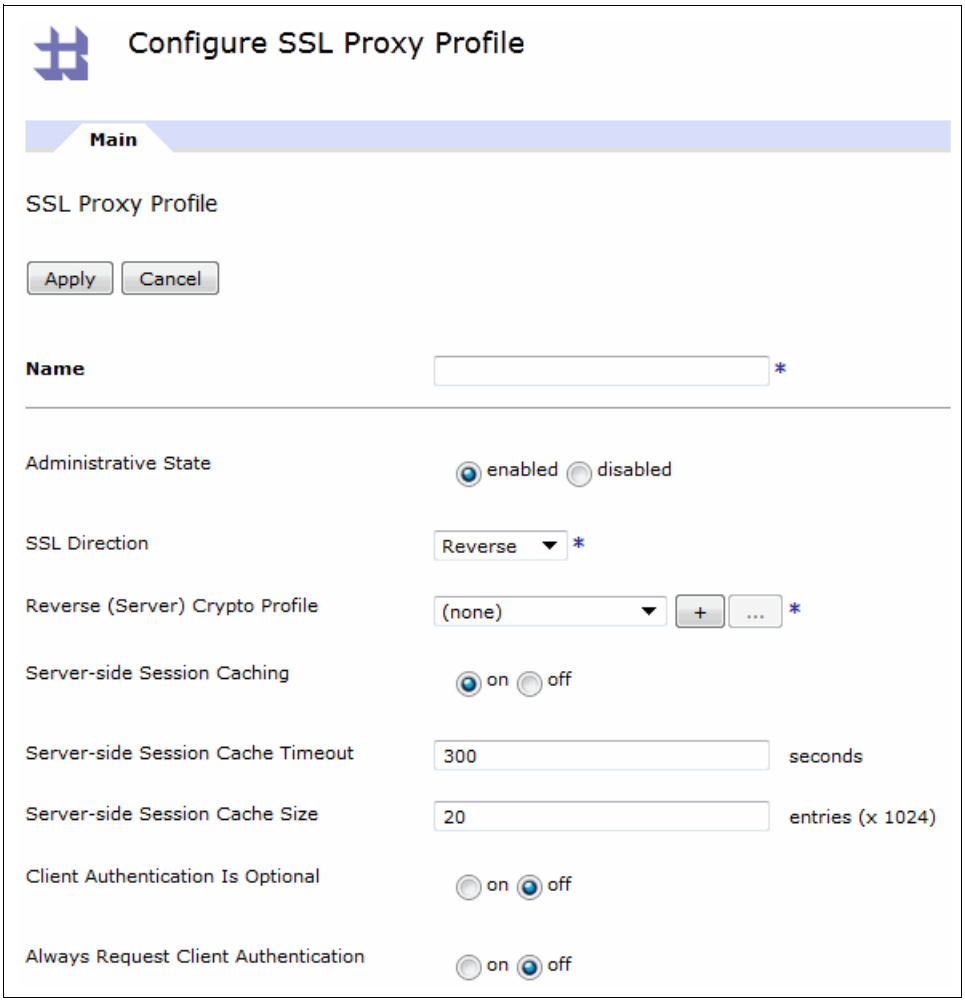


Figure 3-6 Configure SSL Proxy Profile page before adding details

9. Enter a suitable name for the profile, for example, enter the following name:

   `production-wsrr-ssl-proxy-profile`

10. Select **Forward** as the SSL Direction.

11. For Forward (Client) Crypto Profile, click the plus sign (+) to create a new profile. The Configure Crypto Profile page opens (Figure 3-7).

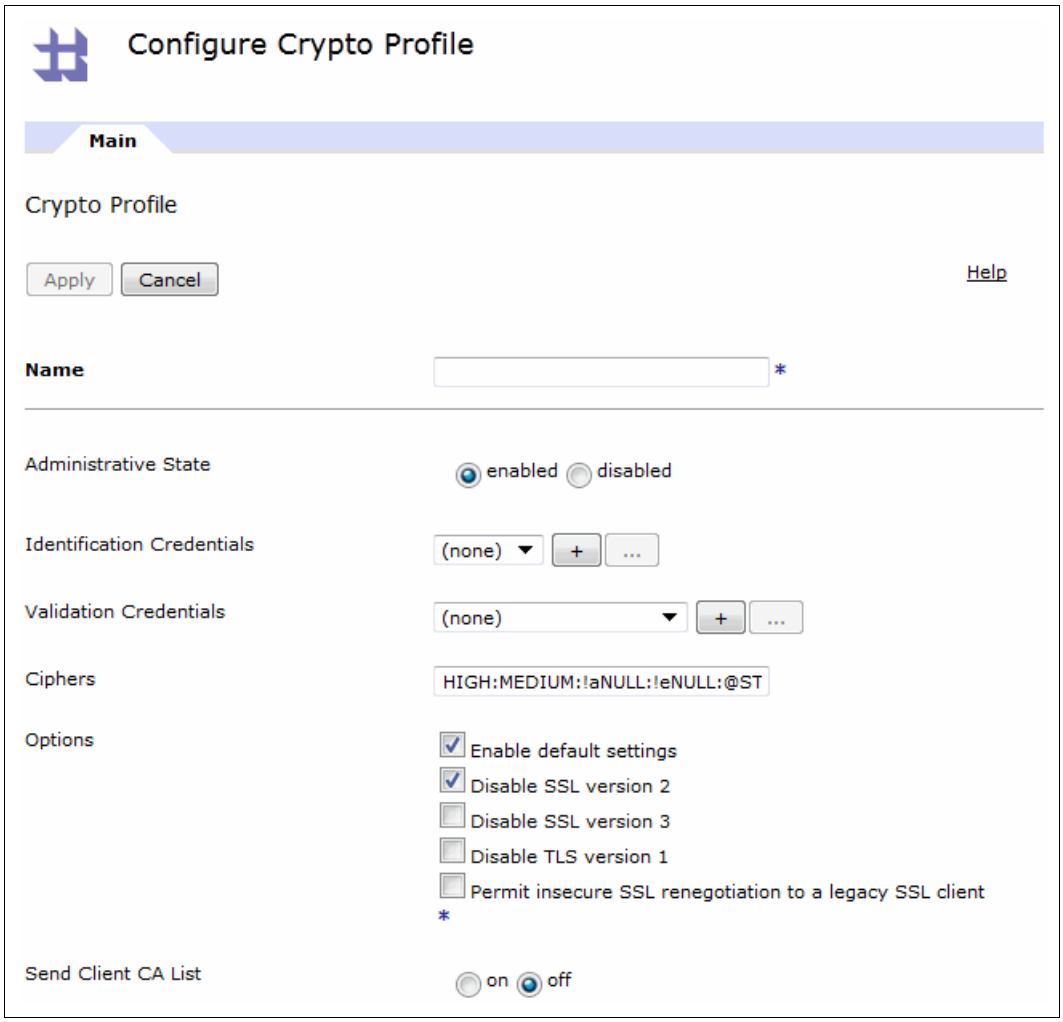


Figure 3-7 Configure Crypto Profile

12. Enter a suitable name for the profile, for example, use the following name:

```
wsrr-forward-crypto
```

13. For Validation Credentials, click the plus sign (+) to create new credentials. The Configure Crypto Validation Credentials page opens (Figure 3-8).

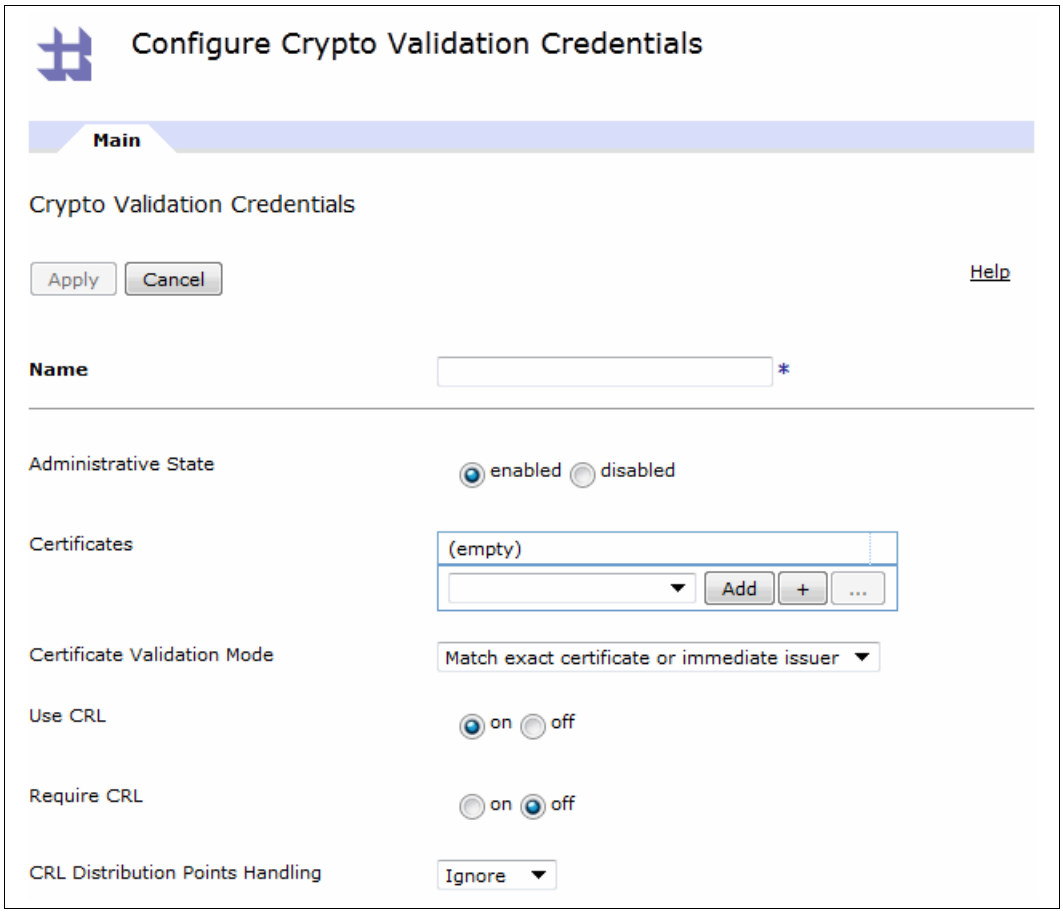


Figure 3-8 Configure Crypto Validation Credentials

14. Enter a suitable name for the credentials, for example, use the following name:

```
wsrr-crypto-valid-cred
```

15. Click the plus sign (+) in the Certificates section. The Configure Crypto Certificate page opens (Figure 3-9).

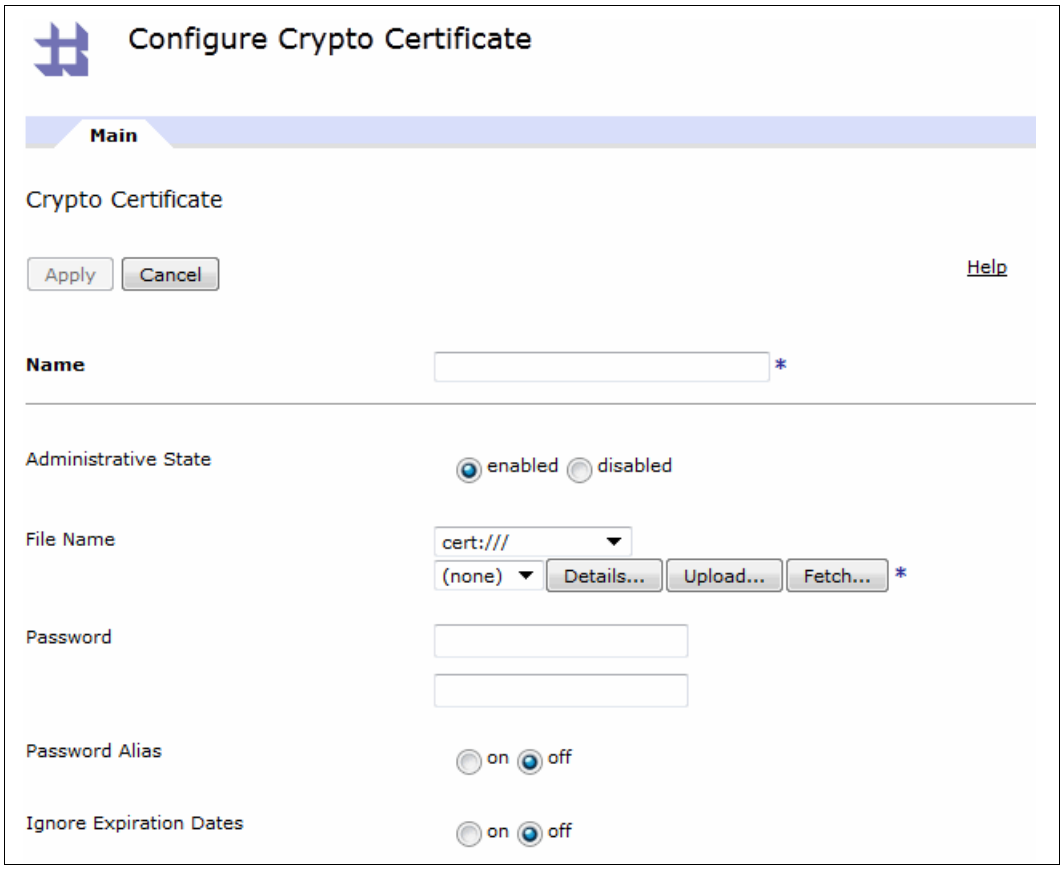


Figure 3-9 Configure Crypto Certificate

16. Enter a suitable name for the certificate, for example, `prod-wsrr-crypto-cert`.
17. In the File Name section, click **Upload**.
18. Click **Browse** to locate the certificate that you copied from the WSRR server earlier.
19. Select the file and click **Upload**. After the file is uploaded, click **Continue**.
20. Click **Apply** until you return to the SSL Proxy Profile page.

    The SSL Proxy Profile is now defined. Continue with the following steps to define the WSRR Server.
21. In the search box on the left, under Control Panel, type `WSRR Server` and click it when you see it in the list.

22. On the Configure WSRR Server page, click **Add**, which opens the panel that is shown in Figure 3-10.

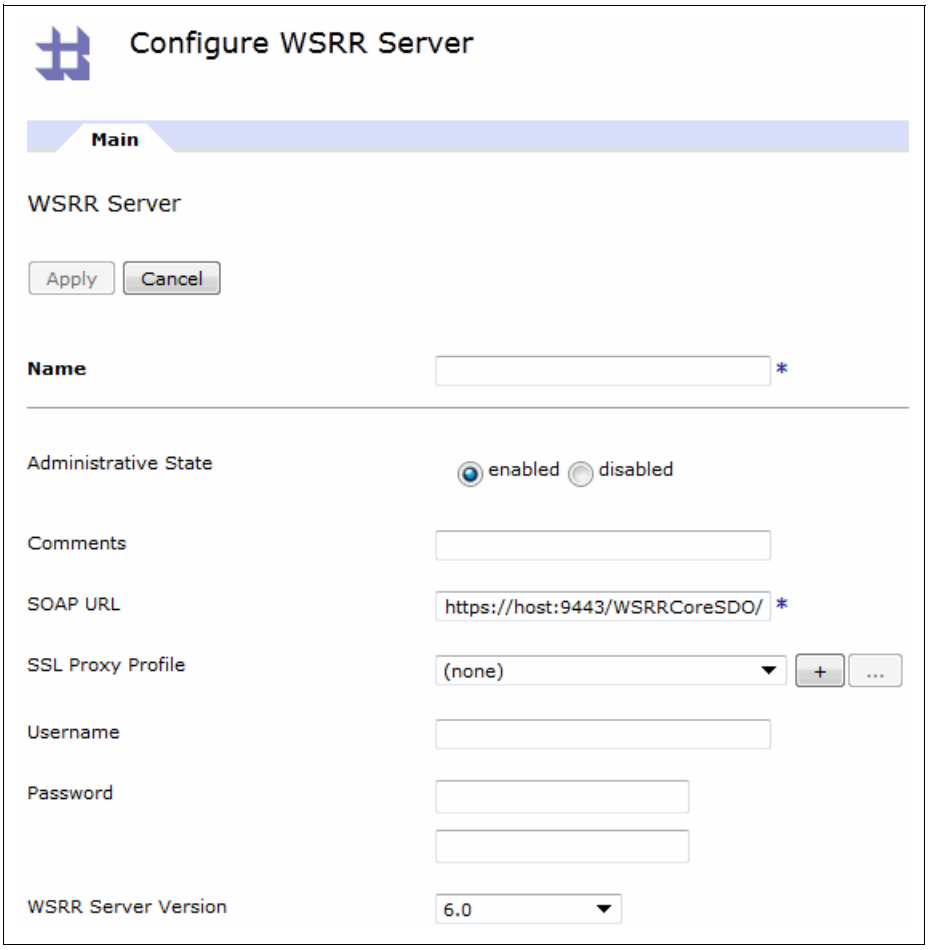


Figure 3-10 Adding a new WSRR Server configuration in DataPower

23. Enter a suitable name in the name field, for example, `production-wsrr`, and optionally any comments. The name has no fuctional effect, but is used to enable the user to more easily differentiate one instance of WSRR from another.
24. Set the SOAP URL appropriately for your environment. For example, in this scenario the URL is as follows:

    `https://productionwsrr:9443/WSRRCoreSDO/services/WSRRCoreSDOPort`

    The host must be that of the application server that is hosting the production WSRR, and port must the secure default host's secure port.
25. Select the SSL Proxy Profile that you created previously (for example, `production-wsrr-ssl-proxy-profile`) in the SSL Proxy Profile drop-down menu.
26. Enter the user name and password of the user who will access the production WSRR, and ensure the version is set to 7.5 or later.

Earlier versions: Policy enforcement does not work with versions of WSRR earlier than 7.5.

27. Click **Apply**, and then click **Save Config**.

    Ensure that the XML Management Interface has `WSRR Subscription` in its list of Enabled Services. This setting is made only in the default domain and you might need to have your DataPower administrator do this step for you.

    If you have the correct permission, do the remaining steps; otherwise, have your administrator do the steps for you.

28. Assuming you have permission, select the default domain in the Domain drop-down list.

29. Type `XML Management Interface` in the search box on the left and select the interface when it appears.

30. In the list of Enabled Services, ensure the WSRR Subscription check box is selected, as shown in Figure 3-11. Click **Apply**, and then click **Save Config**. If WSRR Subscription is already selected, click **Cancel** and go to the next section.

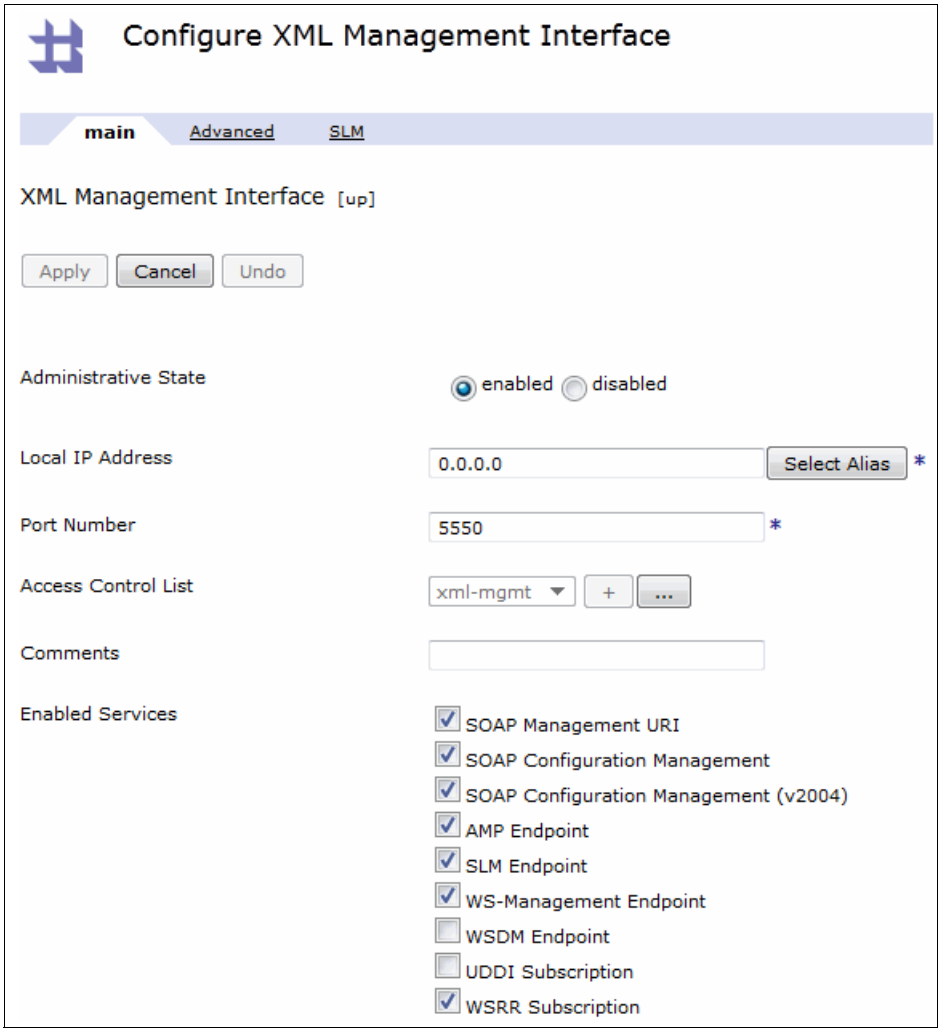


Figure 3-11 XML Management enabled services

WSRR configuration

To complete the WSRR side of the configuration, you must load the SSL certificate from DataPower into WSRR, enable the notifier schedule, and ensure the notifier is configured. Use the following steps.

1. Log in to the WebSphere Integrated Solution Console (or administrative console) with a user ID that has administrator privileges.
2. Expand Security on the left and click **SSL certificate and key management**.
3. Under Related Items, click **Key stores and certificates**.
4. Assuming you are using the default settings, click **NodeDefaultTrustStore**, and under Additional Properties, click **Signer certificates**.
5. On the new page, click **Retrieve from port** and then specify the host name of your DataPower appliance and the port, for example, `9090`. Enter an alias, a local name to easily identify the certificate by.
6. Click **Retrieve signer information**, and when the action finishes, click **OK** or **Apply**.
7. Click **Save** to save your changes. If you accept the default selection (`Dynamically update the run time when SSL configuration changes occur`), then you do not need to restart WebSphere Application Server. Otherwise, you must restart.
8. Log out from the administrative console and log in to the WSRR Console. Use a URL similar to the following example:

   `https://productionwsrr:9443/ServiceRegistry/`
9. If necessary, switch to the Configuration Perspective.
10. Navigate to **Active Profile** → **Scheduler**. Click **SubscriptionNotifierPluginScheduler** in the Scheduler Configurations list.

11. Figure 3-12 shows the WSRR Service Registry Subscription Scheduler Notification. Set the content of the <enabled> tag to `true` for automatic subscription, as shown.

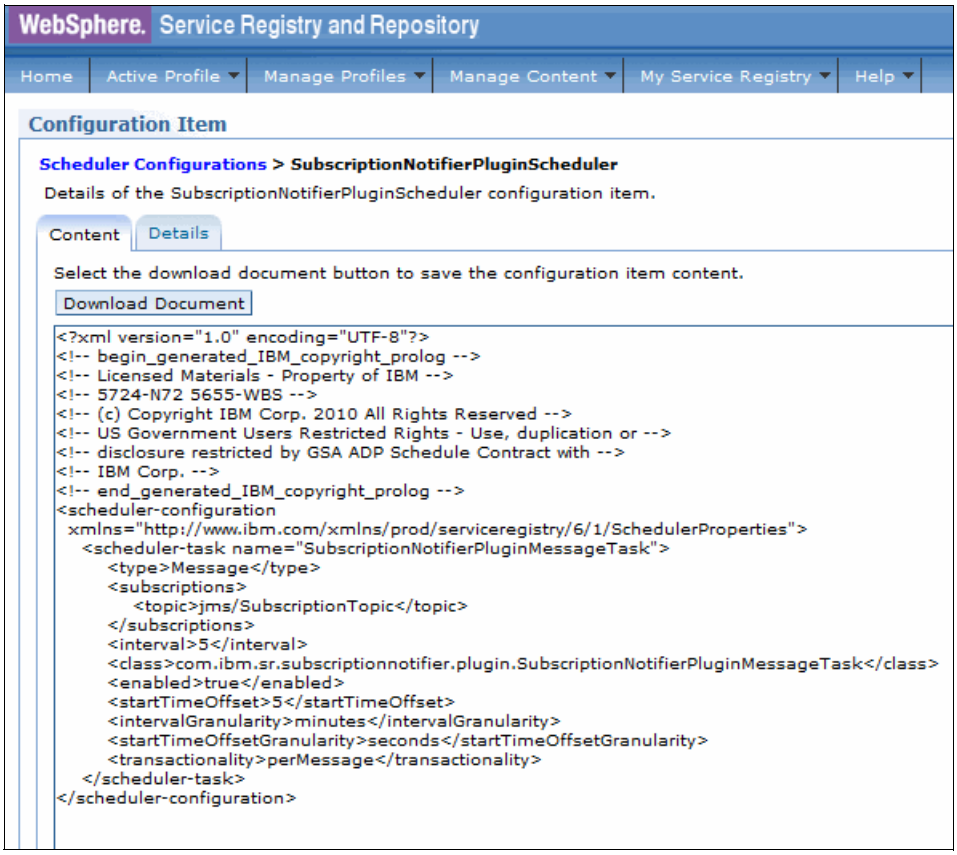


Figure 3-12 SubscriptionNotifierPluginScheduler Scheduler configuration

12. Optional: Change the content of the <interval> tags and <intervalGranularity> tags to make the scheduler run more frequently. The value of intervalGranularity can be one of the following values: ms, seconds, minutes, hours, days, months, or years.
13. Click **OK** or **Apply**. The settings take effect immediately.
14. Navigate to **Active Profile** → **Notifiers** → **Subscription Notifier** and click **SubscriptionNotifierPluginConfiguration**.
15. Find the XML tag shown in Example 3-2 and ensure that the tag is *not* surrounded by comment tags:
    - XML comments start with these characters: `<!--`
    - XML comments end with these characters: `-->`

    If it is commented, remove the comment tags and click **OK** or **Apply**.

Example 3-2 HTTP Post Notifier plug-in settings

```
<Plugin
identifier="com.ibm.sr.subscriptionnotifier.plugin.httppost.HttpPostNotifierPlu
gin
class="com.ibm.sr.subscriptionnotifier.plugin.httppost.HttpPostNotifierPlugin
type="httppost" />
```

Five-minute wait time: When DataPower is notified of a change in WSRR, it waits for five minutes before resynchronizing. If any more notifications are received during this time, the five minutes will restart. This is not configurable. See 3.3.7, “Synchronization with DataPower” on page 71 for more information.

DataPower and WSRR are now configured to communicate with each other in a secure environment.

Integrating WSRR and ITCAM

See Chapter 14, “ITCAM as policy monitoring point” on page 389 for the technical details about the integration of WSRR and ITCAM.

Integration between ITCAM and WSRR is loosely coupled. You might decide that the production runtime WSRR will create situations in the production ITCAM and the ITCAM will update metadata in the Governance Master or in the production run time or both. The use of classifications to filter the integration also means that a single WSRR can integrate with multiple ITCAMs. ITCAM can easily integrate with multiple WSRRs too. However, in that case, choose a naming convention so that identical SLD Policies are not created.

Integrating DataPower and ITCAM

ITCAM has a unidirectional integration with DataPower. ITCAM collects performance metrics from DataPower. These SOA metrics can be managed by a WSRR policy like any other SOA component. See more information about the integration of DataPower and ITCAM:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_install_guide/configdatapower.html

3.2 Service governance

Sample services are available for download with this book. After you deploy them to an application server, to use these services that are described in the scenario, you must register and govern the four services described in Table 3-1 on page 38 in your WSRR. Simply loading the WSDL for each service into WSRR is not sufficient. More information must be given to the registry and this data must then be governed so that DataPower and ITCAM for SOA know it is ready to be used in the run times.

Figure 3-13 on page 55 shows a subset of the objects and their relationships. It shows the pertinent objects that are concerned with the Itinerary Reservation’s consumption of the Itinerary Availability Service. All of these objects must be created by the user, they are not created as a consequence of loading a document such as a WSDL. Instead, the creation of these objects is required so that any supporting documents such as WSDL or XSD files can be loaded.

- A *business service* represents the abstract notion of a service. It can exist before any implementation of a service or design exists. When the need for a new service exists, you can create an object of this type to record the fact.
- A *service version* is a version of a business service. It represents an instantiation of a business service. It will link to the WSDL (for a web service) and the service model objects which are generated from the WSDL.
- A *service level definition* (SLD) provides details of the offered service, its operations, and endpoints where the service can be invoked. It is one of the two places polices can be

attached for enforcement by DataPower or monitoring by ITCAM for SOA. Policies attached to an SLD are applied to invocations of the service from all consumers.

- A *service level agreement* (SLA) represents the contract between a consumer and a provider. It provides the detail of how the service may be consumed by a given consumer. For example, messages per second or the times at which consumption is permitted. The SLA is another place where policies can be attached for enforcement by DataPower or monitoring by ITCAM for SOA. When policies are attached to the SLA, they apply only to invocations of the provided service by the given consumer. Policies can be used to enforce the restrictions of the SLA.
- A *SOAP service endpoint* object represents the actual endpoint where a provider service might be called, as in the following example. A service can have multiple endpoints and it is possible to control their availability by approving and revoking from use.

  `http://services:9081/redbooksTravel/ItineraryReservationService`

Figure 3-13 shows the relationships between the objects. These objects are typically created in a top-down fashion, as shown here. A service can have multiple SLDs, and multiple SLAs can exist between the same consumer-provider pair. When more than one SLA is present, this is typically so that each can offer differing levels of service.

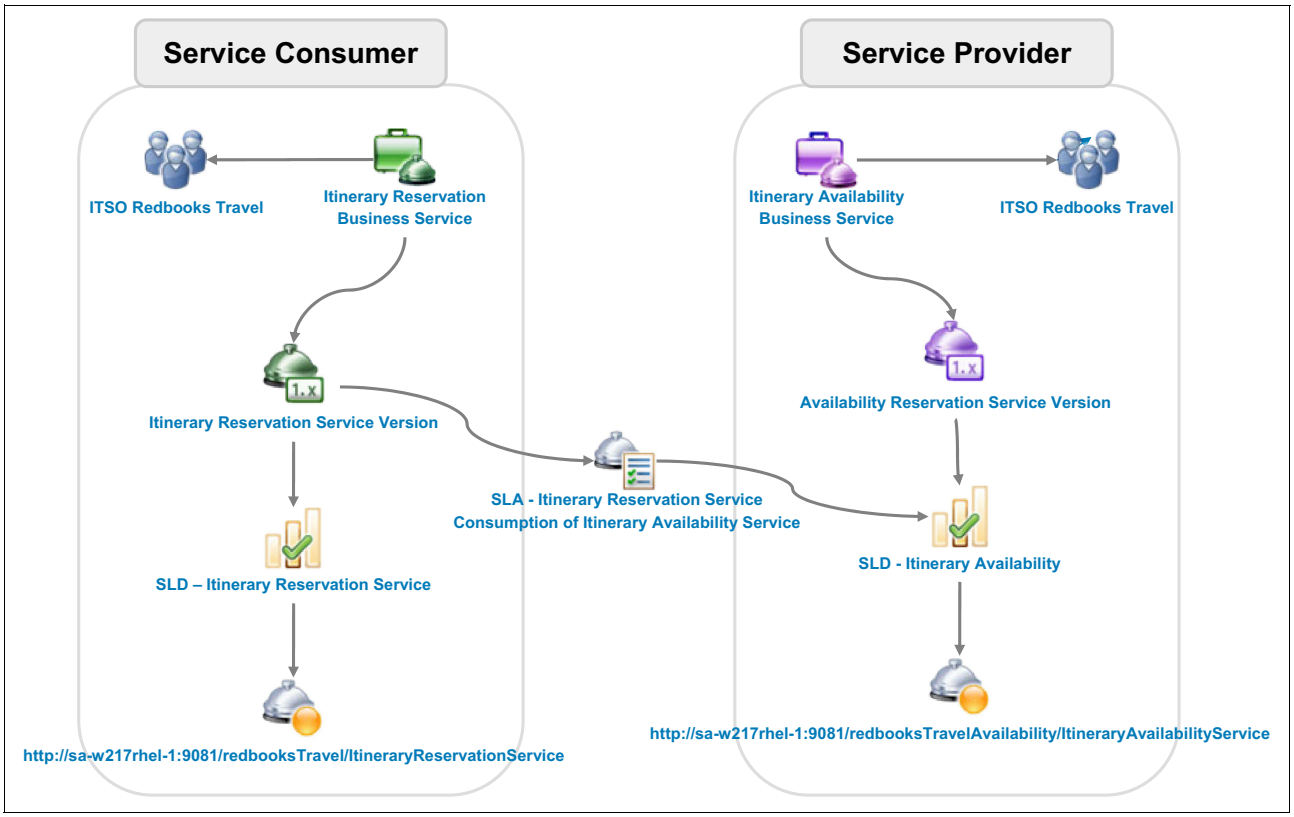


Figure 3-13 Relationship between objects

Each object in Figure 3-13 is subject to its own lifecycle, a series of states joined by transitions. For the integration to function, the objects must all be in the correct state. To transition each object through to the correct state, various criteria must be satisfied, for example, specifying the SLD for an SLA. Some objects, when moved through certain transitions also cause the object and all objects down the graph from it to be copied, or promoted, from the Governance Master WSRR to the Staging or Production registry.

3.2.1 Sample files

To save you the effort of governing the services and adding all of the information yourself, four compressed files that contain sample code are provided. These files are all exports from WSRR, one for each instance (Governance Master, Staging, and Production) and one that contains all of the policies that are ready for attachment.

Appendix C, “Additional material” on page 427 describes how to load these files into WSRRs. After loading the files, you must modify and apply the supplied routing policies. The reason is because the WSDL files that DataPower uses to proxy the services do not contain the correct endpoint information for where you deployed your copies of the services. By using routing policies, DataPower can be made to route to your deployment of the services.

3.2.2 Governing the services yourself

To govern the services yourself, see 3.3, “Govern existing services” on page 56, which describes the necessary steps to be ready to exercise the services and their applied policies.

3.2.3 Using your own services

If you are using your own services then you should see 3.3, “Govern existing services” on page 56, but using the WSDLs for your own services and using names and descriptions appropriate to those services instead of those used in this section.

3.2.4 Creating your business spaces

For the scenario in this book, four business spaces were created, one from each of the main templates, as shown in Figure 3-14. You must create these so that you can use the Business Space UI. For details about how to do this task, go to the following website:

http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/twsr_getting_started_with_wsrr_business_space.html

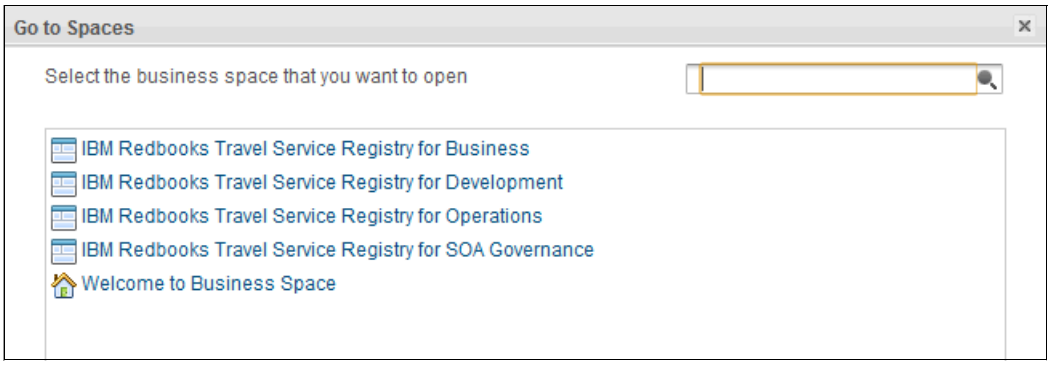


Figure 3-14 Business spaces

3.3 Govern existing services

Exported objects are provided with the sample services that you can import to save you from having to register and govern the services yourself. If you want to follow the process manually, continue with the following steps. If you are using your own services, modify the values submitted to match your own services.

3.3.1 Creating a business service and service version

A business service represents a business capability that is viewed as a service within the organization. Business services have realizations, called *service versions*.

These instructions: use these instructions only if you have already deployed the sample services, or have your own services deployed to WebSphere application server.

The steps in this section briefly describe the required objects. For comprehensive steps, see the information center:

http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/tutorial_gep_bspace_govern_new_service.html

Complete the following steps to create the business service.

1. Log in to the WSRR Business Space and go to your SOA Governance space.
2. In the Action Widget (on the Overview page), click **Create a Business Service** to display the window shown in Figure 3-15. Complete the form as shown.

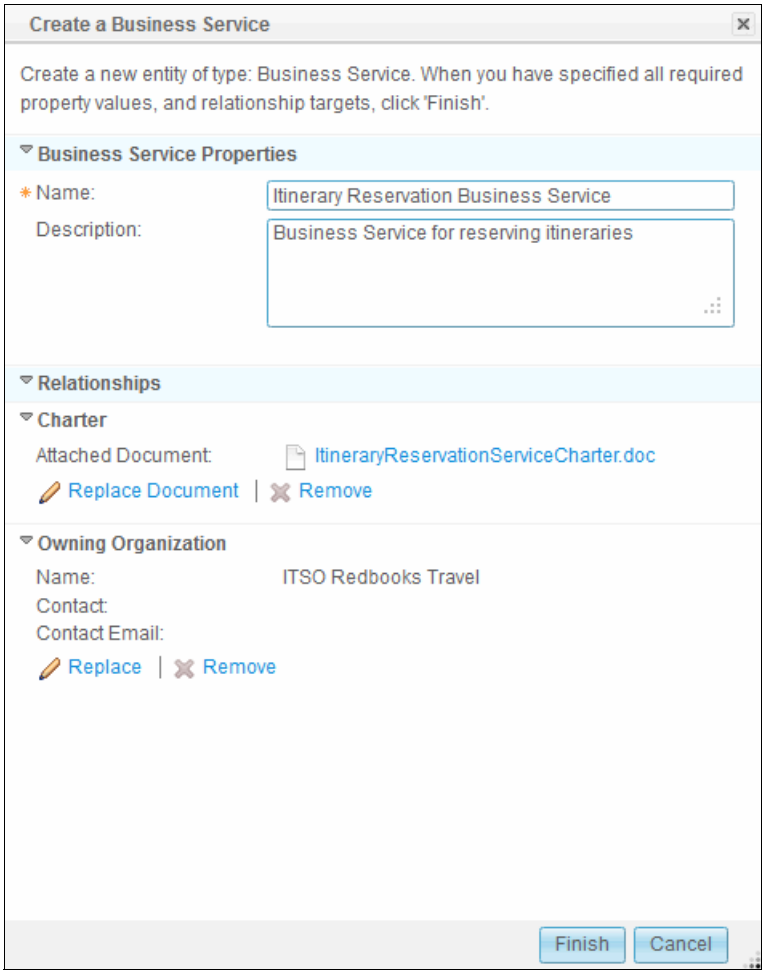


Figure 3-15 Create a Business Service

A business service requires a charter and an owning organization. The charter is a document that describes the service, its requirements, and business justification. It can be loaded from the widget, if not already loaded. The organization is an object that represents the department that owns the service. Again, if it does not already exist, it can be created.

3. Click **Finish**.
4. The business service must be approved before proceeding. From the Action menu, select **Propose Charter**. Click **OK** in the conformation dialog, return to the Action menu, and select **Approve Charter.**
5. After the business service is approved, add the service version to your business service and upload the WSDL. With the approved business service displayed in the Detail widget, click the **Edit** (pencil) icon and then click **Add Capability Version** under Versions. The Add Capability Version section is expanded.
6. Select **Service Version** in the Type drop-down list, and then click **Create**. At the next pop-up window, specify the details of the new service version.
7. Figure 3-16 on page 59 shows the information that is required for the Itinerary Reservation Service Version. Complete the form as shown; you must load the WSDL under Artifacts before you can add the Provided Service.

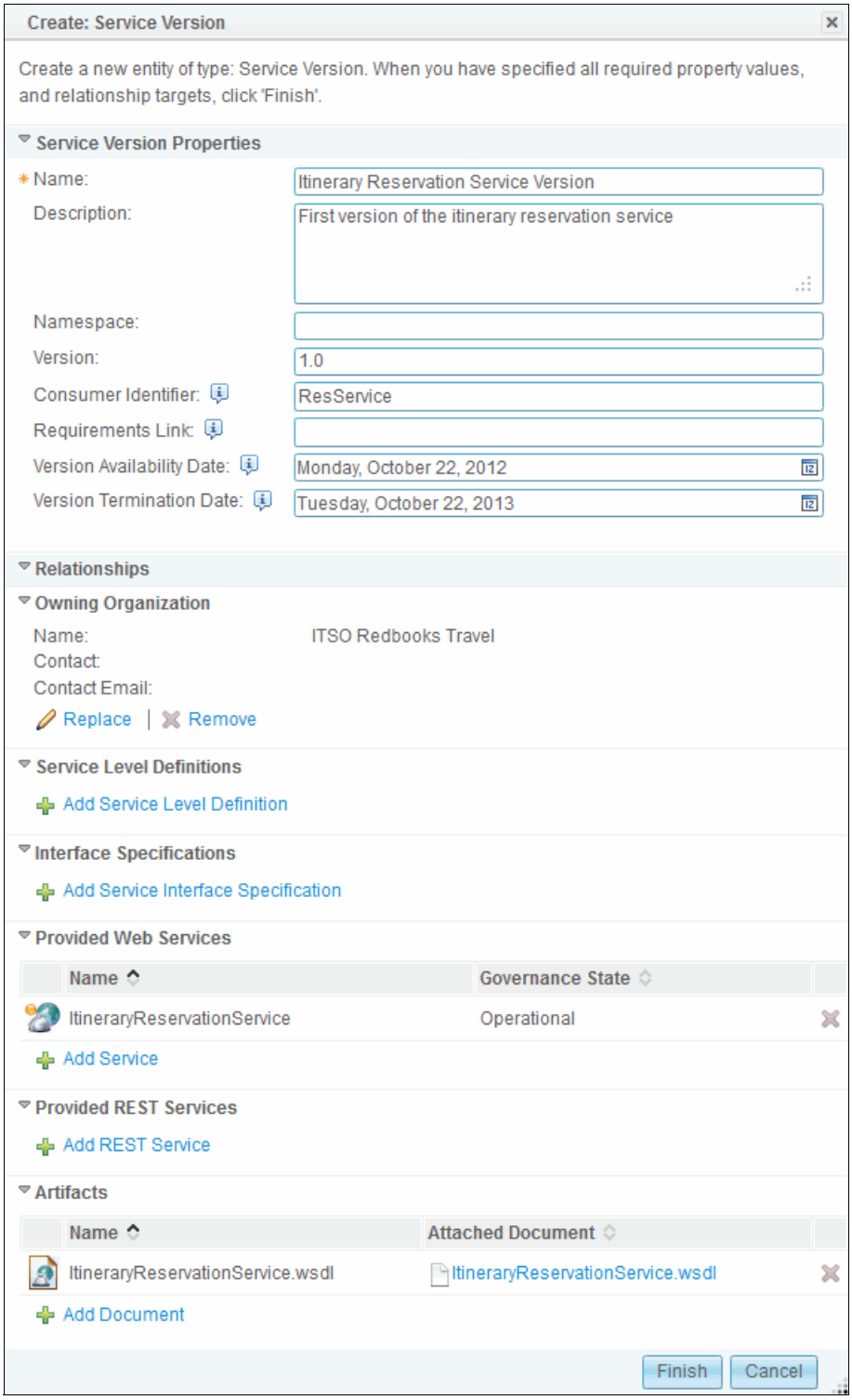


Figure 3-16 Create a service version

8. Wait for the service version to be transitioned through its lifecycle to the Planned state before proceeding with the next step.
9. With the Service Version displayed in the Detail widget, use the Action menu to transition the Service Version through the various lifecycle states.

3.3.2 Creating service level definitions

A service level definition (SLD) specifies the physical communication mechanism that is used by a service. An SLD is tied to a service version. Information such as endpoint, port, service interface, and operations are declared in an SLD. A service must have a minimum of one SLD to be accessible, but can have many SLDs, depending on the requirements.

SLDs are typically created by a member of the development team. Accordingly, you should create your SLD in a development space.

Use the following steps:

1. Navigate to the service version to which the SLD is intended and click the **Edit** icon.
2. Click **Add Service Level Definition** under Service Level Definitions and click **Create**.

   The Create Service Level Definition window opens (Figure 3-17 on page 61).

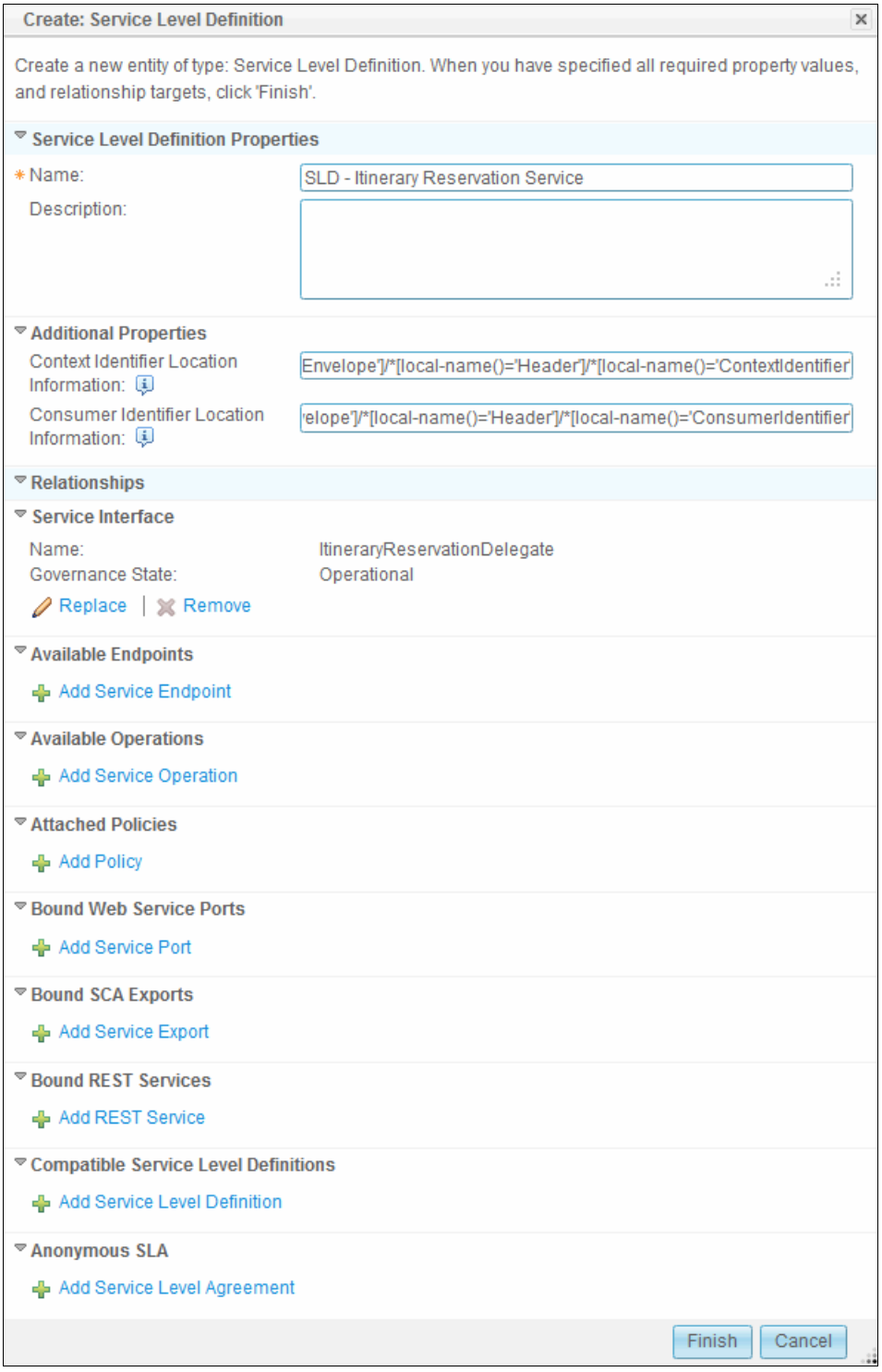


Figure 3-17 Create a service level definition (SLD)

3. Provide a name and define the Context Identifier Location Information and the Consumer Identifier Location Information, and click **Finish**.

   For more details about these values, see the following website:

   http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/rwsr_gep_service_level_definition.html

The SLD is transitioned to the Scoped state before the Operations team can add the details.

After transitioning to Scoped, switch to the Operations space and specify the following information:

- Bound Web Service Port as ItineraryReservationPort
- Available Endpoints as follows:

  `http://services:9081/redbooksTravel/ItineraryReservationService`

The SLD is transitioned to the SLD Subscribable state.

If you want to go straight to the next stage of completing the governance of your service, see 3.3.6, “Promoting the services to WSRR run times” on page 68.

3.3.3 Creating policies

A policy expression represents the declaration of a policy, and is equivalent to a `<wsp:Policy>` element in a WS-Policy document. See the following sections:

- 3.4, “Creating policies” on page 73 shows the detailed steps for creating mediation policies that are used in this book.
- 11.1, “Understanding the policy lifecycle” on page 334 explains the policy lifecycle.

3.3.4 Attaching policies to SLDs for services you want

The detailed steps for attaching policies to SLDs that match the service criteria are described in 3.5, “Attaching policies” on page 78.

Figure 3-18 on page 63 shows the policy attachments to various services, which are described in the following sections:

- 3.4, “Creating policies” on page 73
- 3.5, “Attaching policies” on page 78
- 3.6, “Reporting on services and policies applied to services” on page 84

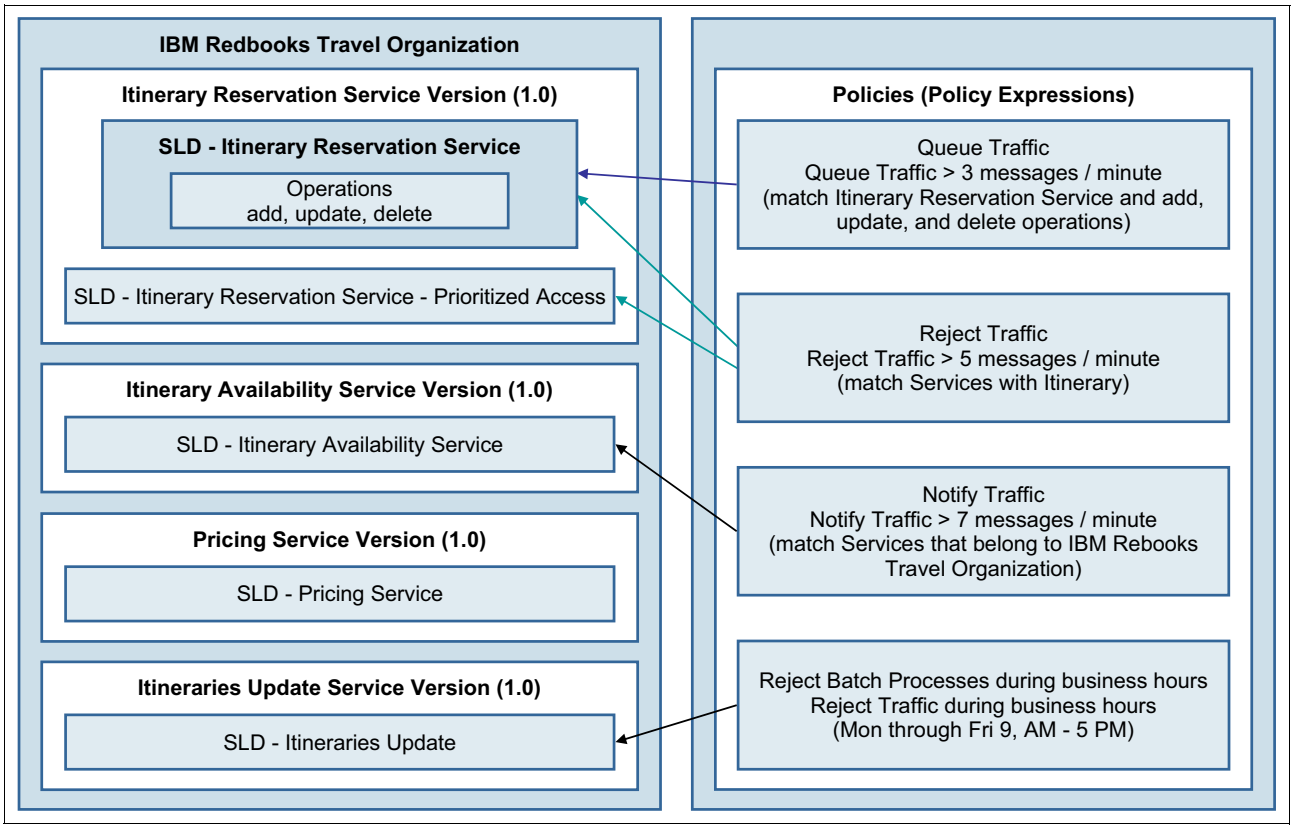


Figure 3-18 Policies and policy attachments

3.3.5 Setting up a service gateway in DataPower

The Web Service Proxy in DataPower can be configured to subscribe to a WSRR saved search. The following procedure shows how to create a WSRR saved search, which basically returns all of the WSDL files in the registry. In reality, a saved search can be more selective, for example, by using classifications.

Use the following steps to create the Saved Search in production WSRR.

1. Log in to the production WSRR. This scenario uses the following URL:

   `https://productionwsrr:9443/ServiceRegistry`

2. In the search field, type `wsdl`. Figure 3-19 on page 64 shows the results that match this search.

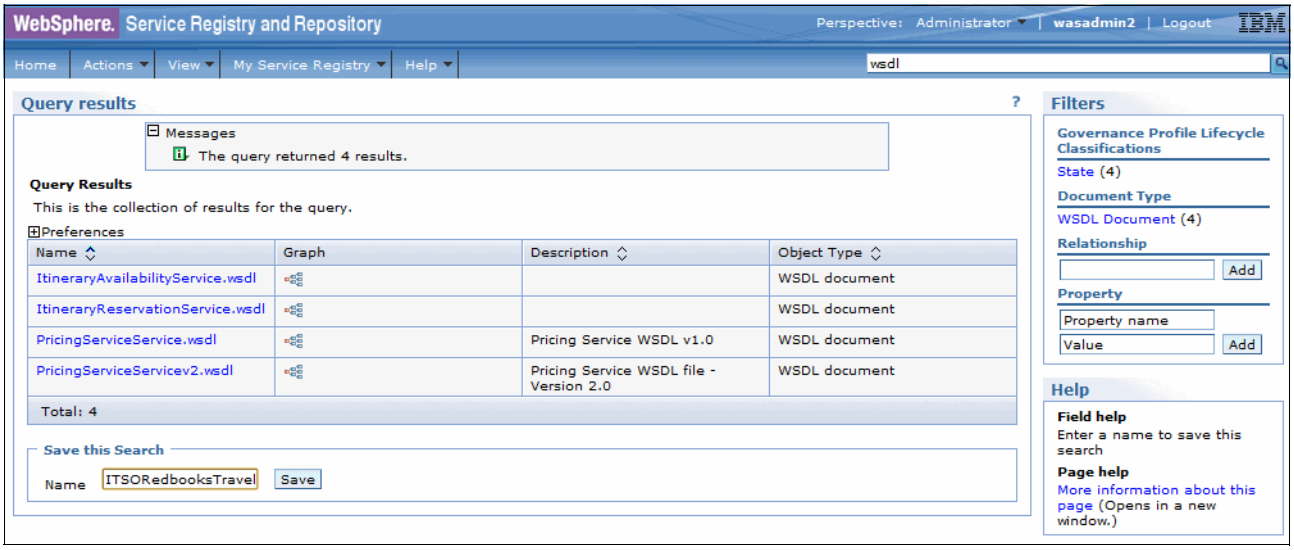


Figure 3-19 WSRR saved search

3. Save the search results with the following name: `ITSORedbooksTravel_WSDLs`

> **Tip:** You may also skip step 1, and create a new WSRR Saved Subscription in step 3 instead.

4. Use the WSRR saved search to set up a WSRR Saved Search Subscription in DataPower:
   a. Log in to your production domain on DataPower**.**
   b. Type `wsrr` in the search field.
   c. Go to **WSRR Saved Search Subscription** → **Add** → **WSRR Server**.
   d. Select **production-wsrr**.
   e. Click the **Find a Saved Search** button next to Saved Search Name. This retrieves the WSRR saved searches from the production-wsrr server.

Figure 3-20 shows the DataPower subscription to the `ITSORedbooksTravel_WSDLs` saved search in WSRR.

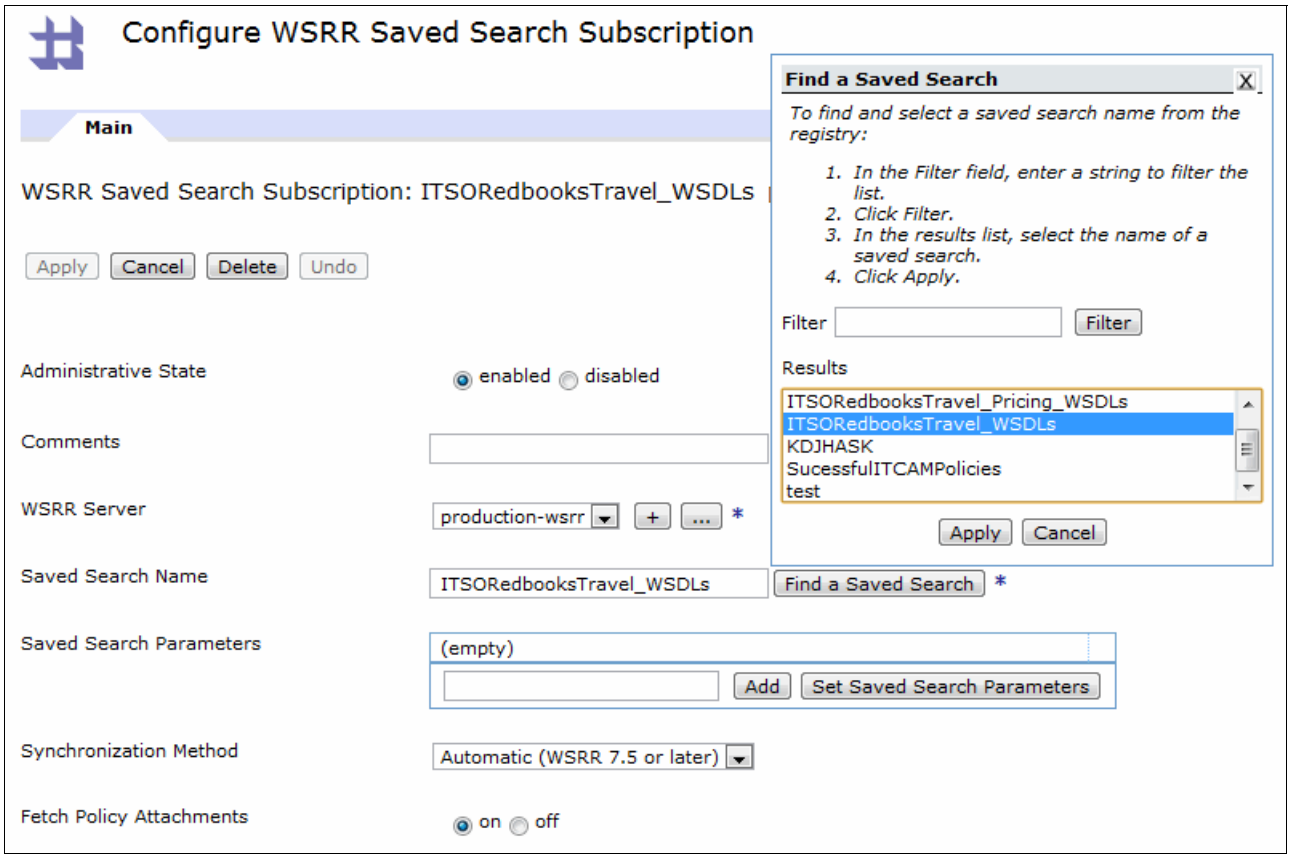


Figure 3-20 WSRR Saved Search Subscription

5. Set up a Web Service Proxy Gateway:
   a. Log in to your production domain on DataPower.
   b. Navigate to **Web Service Proxy** → **Add**.
   c. Enter the input name as `ITSORedbooksTravel_WSP`.
   d. Click **Create Web Service Proxy**.
   e. Click **Add WSRR Saved Search Subscription**.
   f. Under WSRR Saved Search Name, select **Existing** and then select **ITSORedbooksTravel_WSDLs** in the drop-down menu.

      The fields are automatically populated (Figure 3-21 on page 66).
   g. Click **Next**.

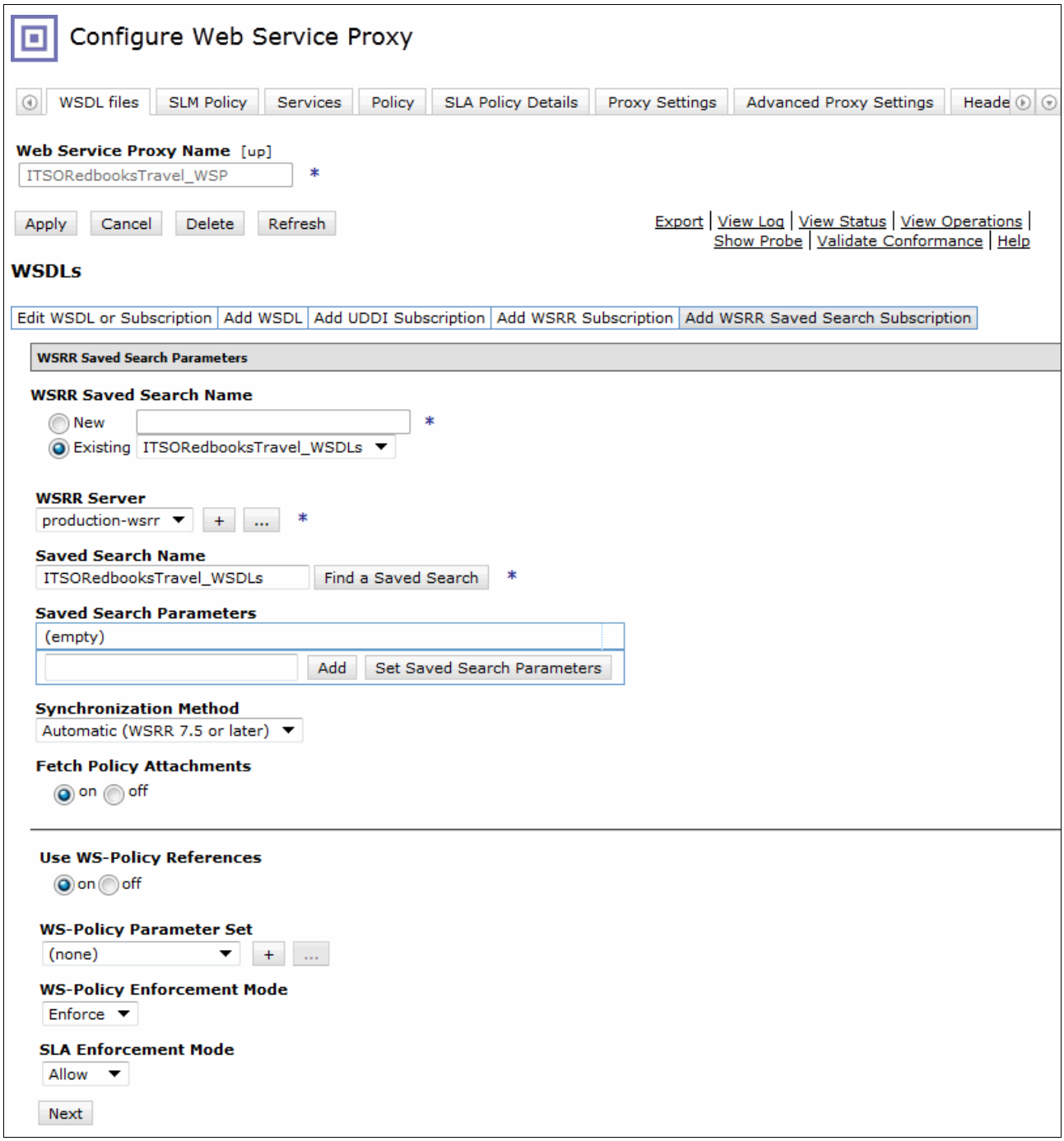


Figure 3-21 DataPower Web Service Proxy using WSRR Saved Search

6. Specify Front Side Handler and continue to complete the Web Service Proxy setup. See 12.4, “Front Side Handler” on page 353 for more details.

Figure 3-22 shows the automatic retrieval of WSDL configurations by DataPower by using a WSRR Saved Search Subscription.

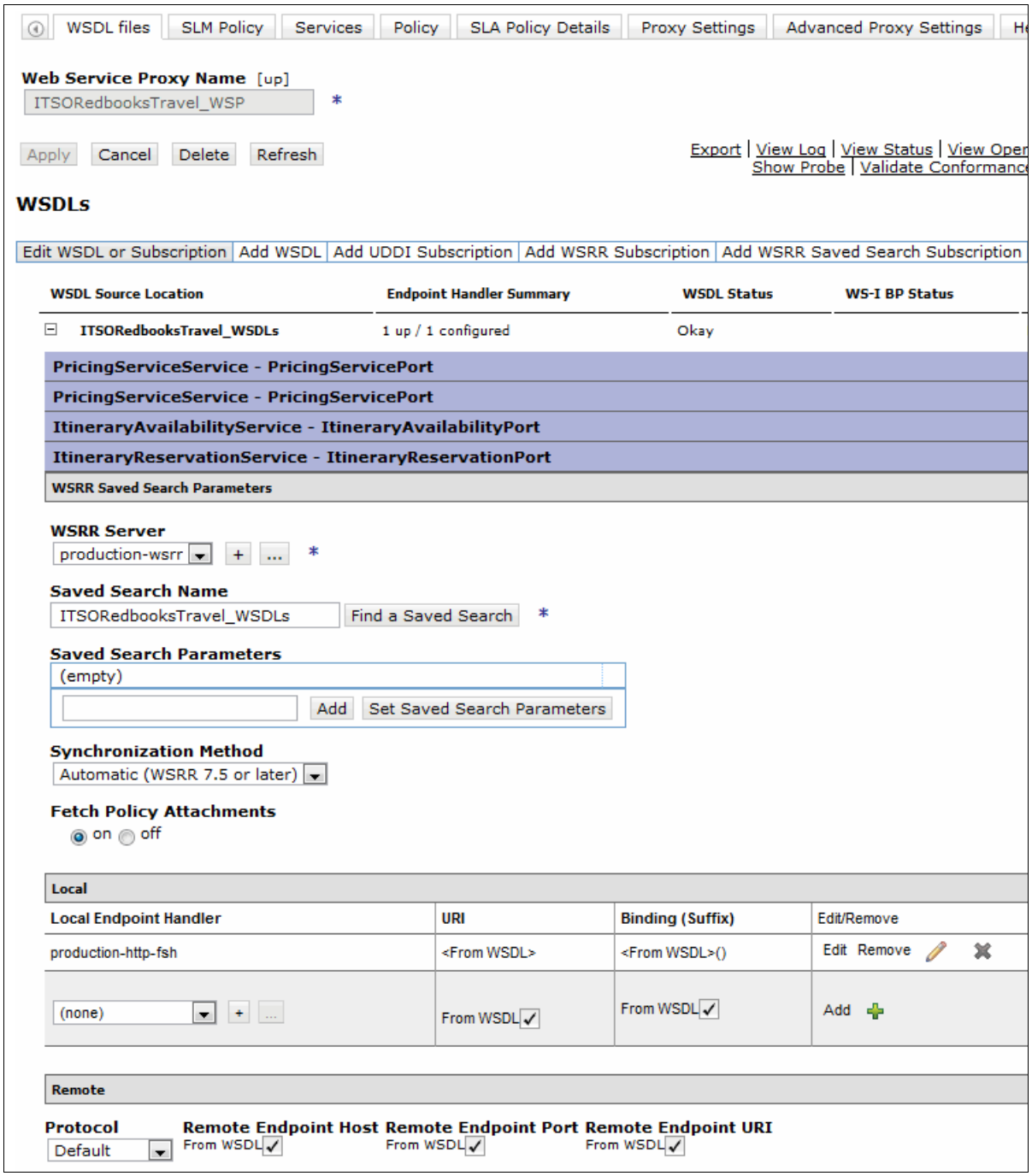


Figure 3-22 DataPower gateway showing synchronized WSDLs from WSRR

3.3.6 Promoting the services to WSRR run times

A full production deployment consists of multiple registries to support the various stages of the service development and testing lifecycle. Therefore, you can keep the service metadata for your staging and production environments in separate registries to more effectively control what will happen when a service goes into production. Users of the SOA service lifecycle can have a read-only WSRR for each environment.

For more details, see the following website:

http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/cwsr_overview_overview06.html

Figure 3-2 on page 39 shows the topology for the scenario in this book. These environments are described next.

Staging

A service can pass through a series of staging environments between its initial development and its release into production. In a full production deployment, there is one registry for each staging environment that you want to test in an isolated manner, for example, integration, acceptance testing, and preproduction.

The number and types of staging environments depend on the nature of your development process. The governance enablement profile (GEP), as used in this book, includes two predefined environments. You can add more to suit your needs.

For more information, see the following website:

http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/cwsr_runtime_env.html

Production

A limited set of content from the Governance Master is promoted to this registry as and when services are ready to go into production, by using the WSRR promotion feature.

Typically, users do not work directly with this registry, and the content is mostly read-only, although more metadata might be added to registry content, for example, to support ITCAM for SOA. Instead, users work in the Governance Master, and content is promoted automatically to the runtime production registry at the appropriate point in the governance lifecycle.

Promotion

Promotion is performed by transitioning a capability version (service version, or application version, in the case of this scenario). A service version should have at least one SLD for the service to be operational.

States: SLDs must be in *SLD Subscribable* state and policies must be in *Approved* state if they are to be promoted. The promotion will still occur, but these objects will be missing.

The current governance state of Itinerary Reservation Service Version (1.0) is Planned, but needs be Realized. Transition the service version to this state.

To promote the services to the Staging WSRR, complete the following steps:

1. Select **Action** → **Propose Staging Deployment**, and note that the new governance state is Staging Review.
2. Select **Action** → **Approve Staging Deployment** and note that the new governance state is Staged.

The *Staged* governance state indicates that the services have been promoted to Staging WSRR successfully. Any problems with the promotion will cause the transition to roll back. Figure 3-23 shows the objects involved in this transaction. The objects that are promoted when the Itinerary Reservation Service Version is Approved for Staging Deployment are all those from the service version to the endpoint. Also included are any (approved) policies that are attached the SLD. Other objects, such as WSDLs, XSDs, and so on, are also promoted.

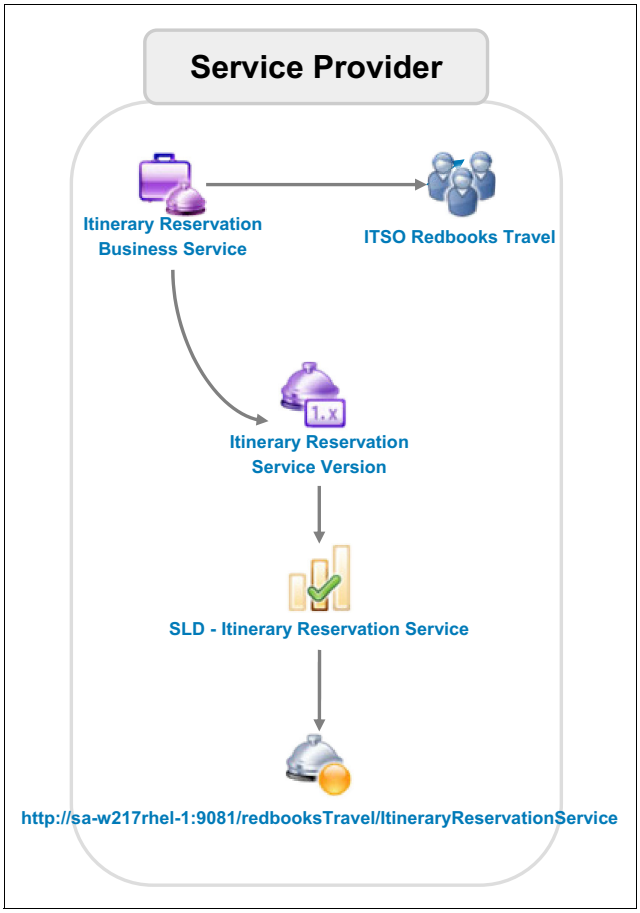


Figure 3-23 GEP objects that relate to the Itinerary Reservation Service

To promote the services to production WSRR run time, complete the following steps:

1. Select **Action** → **Propose Certification** to govern to `Certification Review` state.
2. Select **Action** → **Approve Certification** to govern to `Certified` state.
3. Select **Action** → **Propose Production Deployment** to govern to `Operational Review` state.
4. Select **Action** → **Approve Production Deployment** to govern to `Operational` state.

The Operational governance state indicates that the services have been promoted to Production WSRR successfully.

If any changes are made to objects after promotion, such as attaching policies or new SLAs created, then these changes are not automatically repromoted to the staging or promotion WSRRs. Follow the repromotion steps detailed in “Repromotion” on page 70.

Repromotion

To repromote objects to either the Staging or Promotion WSRRs, the Service Version upstream of the changed objects must, once again, be Approved for Staging (or Production) Deployment. Figure 3-24 shows the lifecycle the service version moves through. To repromote to the Staging WSRR, the service version must be redefined three times to be placed back in the `Realized` state. From here it can be moved through `Staged` and `Certified` states, on its way to becoming `Operational` once more when it is repromoted to `Production`. If you want to repromote only to `Production`, then you redefine once to `Certified` and then go forward to `Operational`.

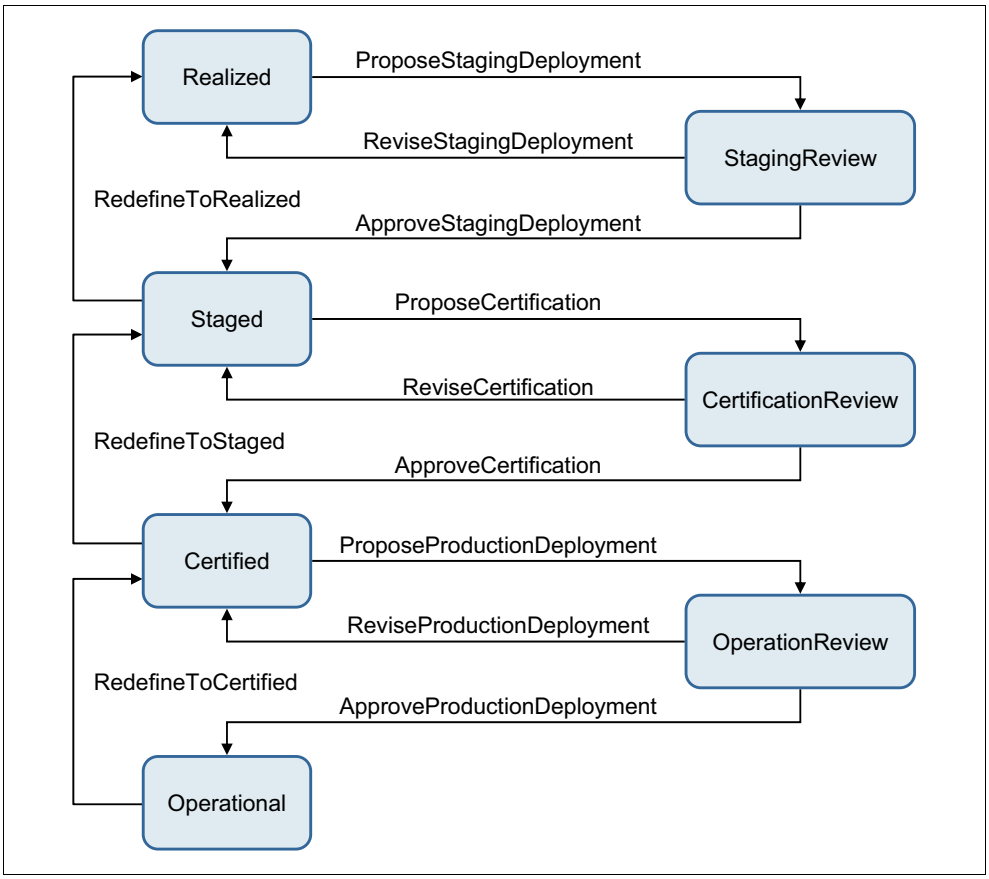


Figure 3-24 Deployment stage of the SOA Lifecycle from Realized to Operational

3.3.7 Synchronization with DataPower

Based on the synchronization mechanism (automatic, manual, or poll) that is defined in DataPower, the promoted services are propagated to DataPower and attached mediation polices are enforced.

Synchronization is set up in WSRR Saved Search Subscription in DataPower, as shown in Figure 3-25.

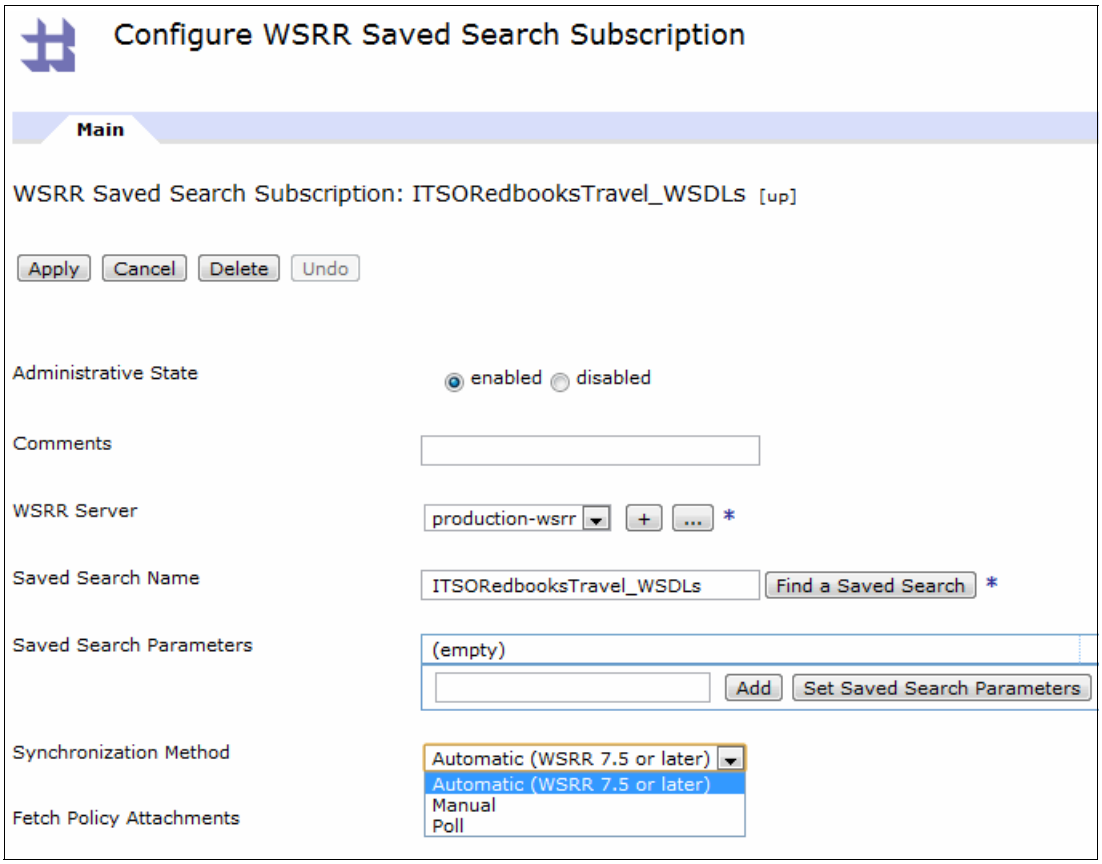


Figure 3-25 WSRR Synchronization methods in DataPower

For the scenario in this book, synchronization is set to automatic between WSRR and DataPower. To see the changes immediately in DataPower, a manual synchronization must be done.

Use the following steps to manually synchronize:

1. Log in to DataPower and select **Control Panel** → **Web Service Proxy** → **ITSORedbooksTravel_WSP**.
2. Click the **Okay** link under WSDL Status, as shown in Figure 3-26.

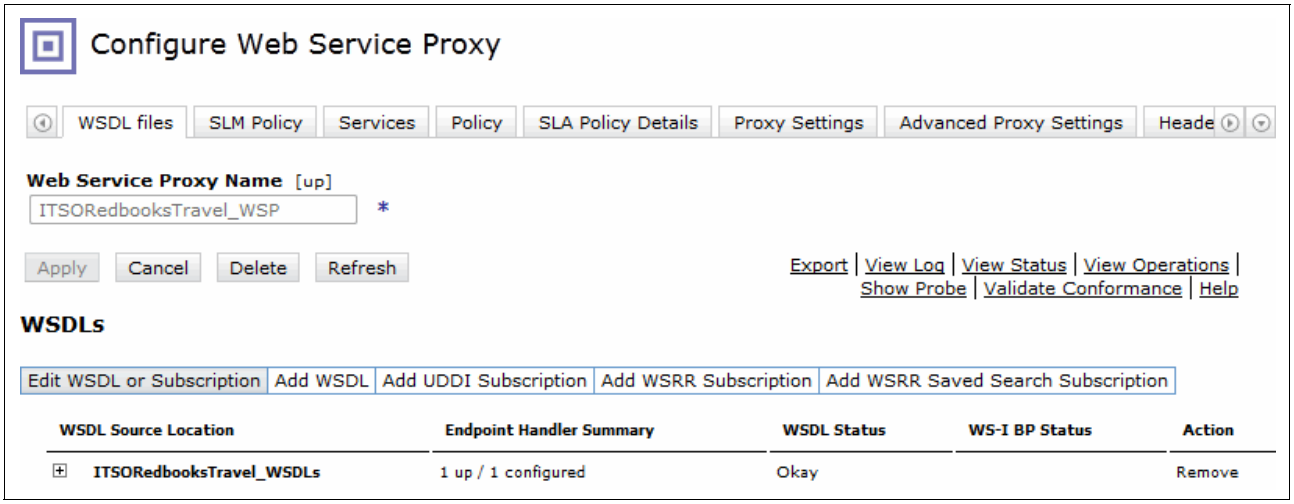


Figure 3-26 Subscription status link in DataPower Web Service Proxy

3. Figure 3-27 shows the WSRR Subscription Status page in DataPower with the Synchronize option. You can do either of the following actions:
   - Click **Synchronize** to manually synchronize the changes from WSRR run time to DataPower.
   - Click **Refresh status** to see the live synchronization status.

   During the synchronizing process, the status indicates `Synchronizing`. After successful synchronization, the status is `Okay`.

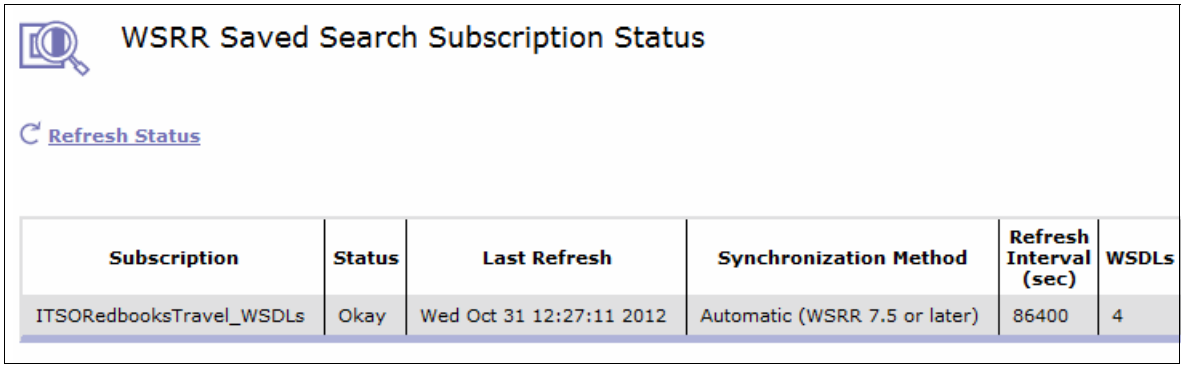


Figure 3-27 WSRR synchronize option in DataPower

4. To view the SLD and policy changes, navigate to **Web Service Proxy** → **ITSORedbooksTravel_WSP** → **SLA Policy Details tab** → **SLA table**, as shown in Figure 3-28. Currently, no policies are in place, so you see only the four WSDLs that match the Saved Search, with zero attachments.

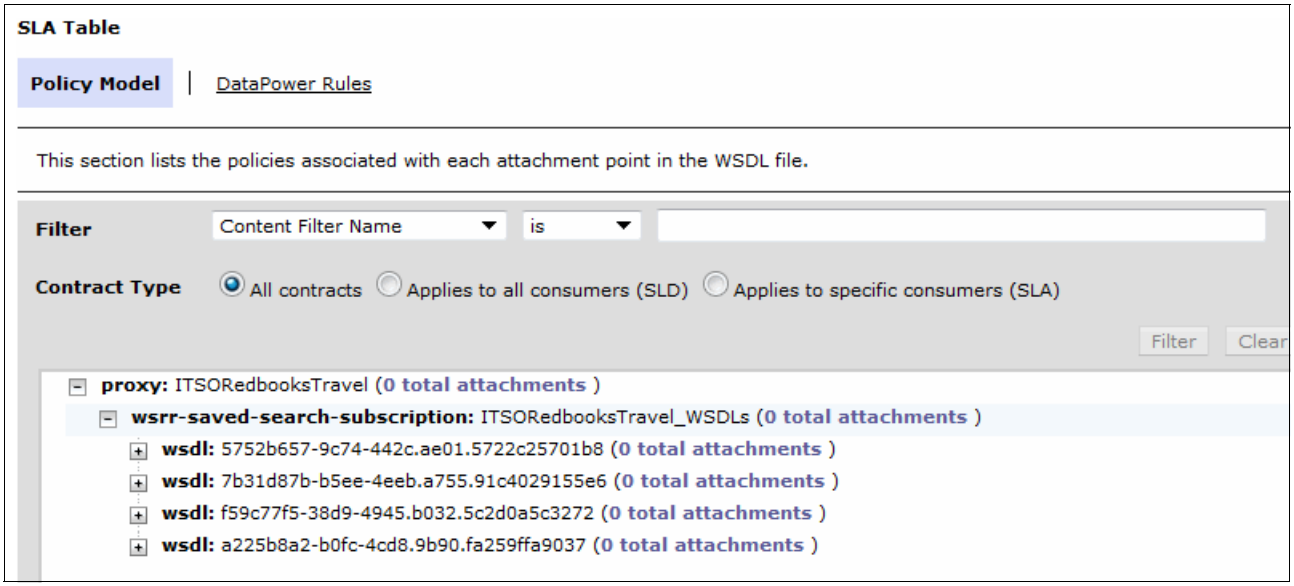


Figure 3-28 DataPower WS Proxy SLA table

3.4 Creating policies

IBM Redbooks Travel Company wants to author and enforce mediation policies to its existing services. This section shows creation of policies matching various actions like queuing, throttling, and notify for conditions exceeding specified number of messages. The policies are enforced at run time when conditions are met.

A selection of polices is available for import to your WSRR, or you can follow these instructions to create your own.

3.4.1 Creating mediation policy for queuing

Use the following steps to create a mediation policy named *Queue Traffic* that will queue traffic when receiving more than three messages per minute.

> **Numbers:** The numbers that are used here are unrealistically low so that the effects of the policy can be easily observed.

1. Log in to WSRR Business Space, and then select **Go To Spaces** → **IBM Redbooks Travel Service Registry Operations Space** → **Overview** → **Service Registry Actions** → **Create a Mediation Policy**.
2. Enter the values that are shown in Figure 3-29 on page 74, which shows the mediation policy for Queue Traffic.

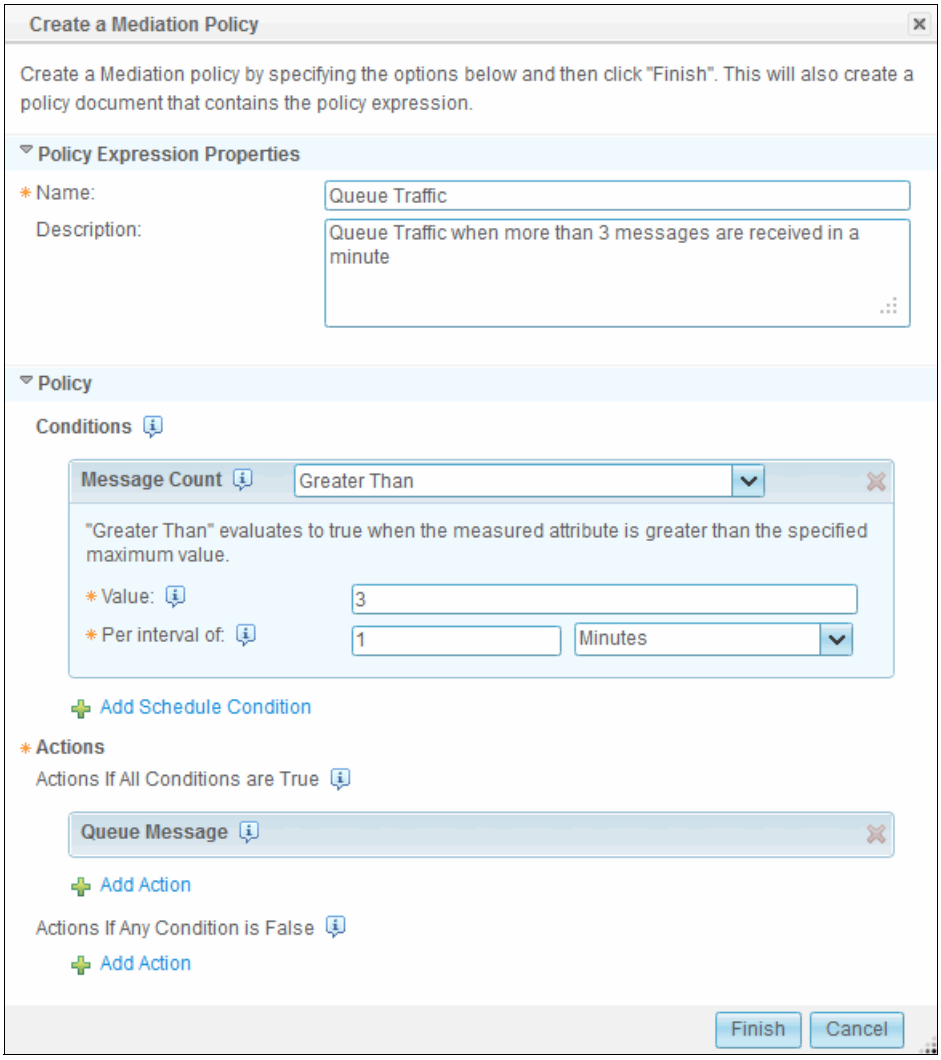


Figure 3-29 Create a mediation policy with queue action

3. After creating the mediation policy, propose and approve it:
   a. Navigate to **Policy Expression** → **Source Document** (`Queue Traffic.xml`).
   b. Take actions to propose and approve the specification:
      i. Select **Action** → **Propose Specification**.
      ii. Select **Action** → **Approve Specification**.

   The governance state now indicates the Approved state.

The Service Registry Navigator displays the artifacts, such as policy document and policy expression, as shown in Figure 3-30.

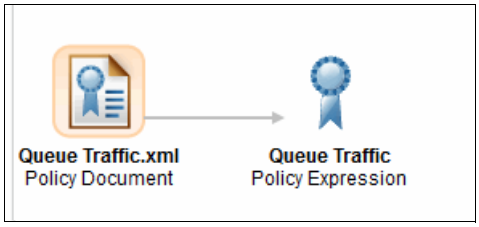


Figure 3-30 Service Navigator displaying queuing policy

3.4.2 Creating mediation policy for throttling

Use the following steps to create a mediation policy named *Reject Traffic* that will throttle (reject) traffic when receiving more than five messages per minute:

1. Log in to WSRR Business Space, and then select **Go To Spaces** → **IBM Redbooks Travel Service Registry Operations Space** → **Overview** → **Service Registry Actions** → **Create a Mediation Policy**.
2. Enter the values that are shown in Figure 3-31 on page 76, which shows a mediation policy for throttling.

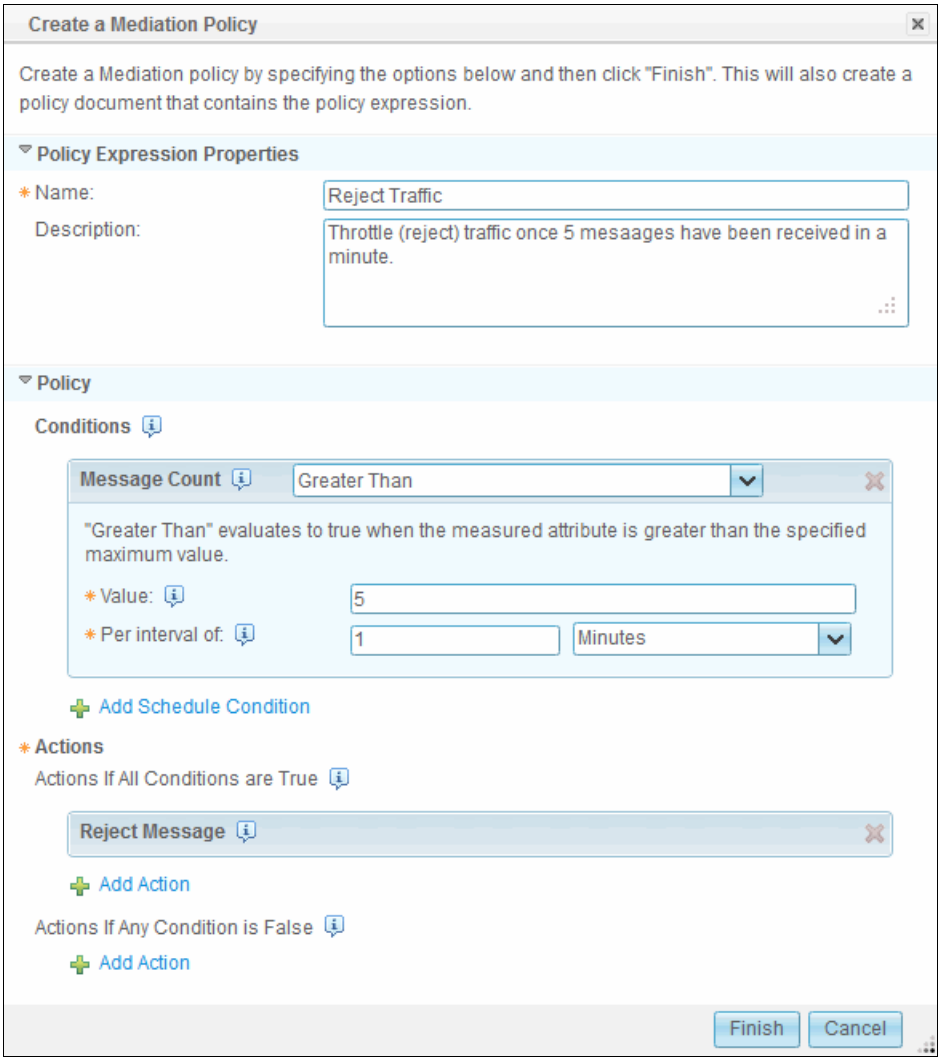


Figure 3-31 Mediation policy to throttle traffic

3. After creating the mediation policy, propose and approve it:
   a. Navigate to **Policy Expression** → **Source Document** (`Throttle Traffic.xml`).
   b. Take actions to propose and approve the specification:
      i. Select **Action** → **Propose Specification**.
      ii. Select **Action** → **Approve Specification**.

   The governance state now reflects the `Approved` state.

The Service Registry Navigator displays the artifacts, such as policy document and policy expression.

3.4.3 Creating Mediation Policy for Notify

Use the following steps to create a mediation policy named *Notify Traffic* that will notify of traffic when receiving more than seven messages per minute.

1. Navigate to **Go To Spaces** → **IBM Redbooks Travel Service Registry Operations Space** → **Overview** → **Service Registry Actions** → **Create a Mediation Policy**.
2. Enter the values shown in Figure 3-32, which shows a mediation policy to notify.

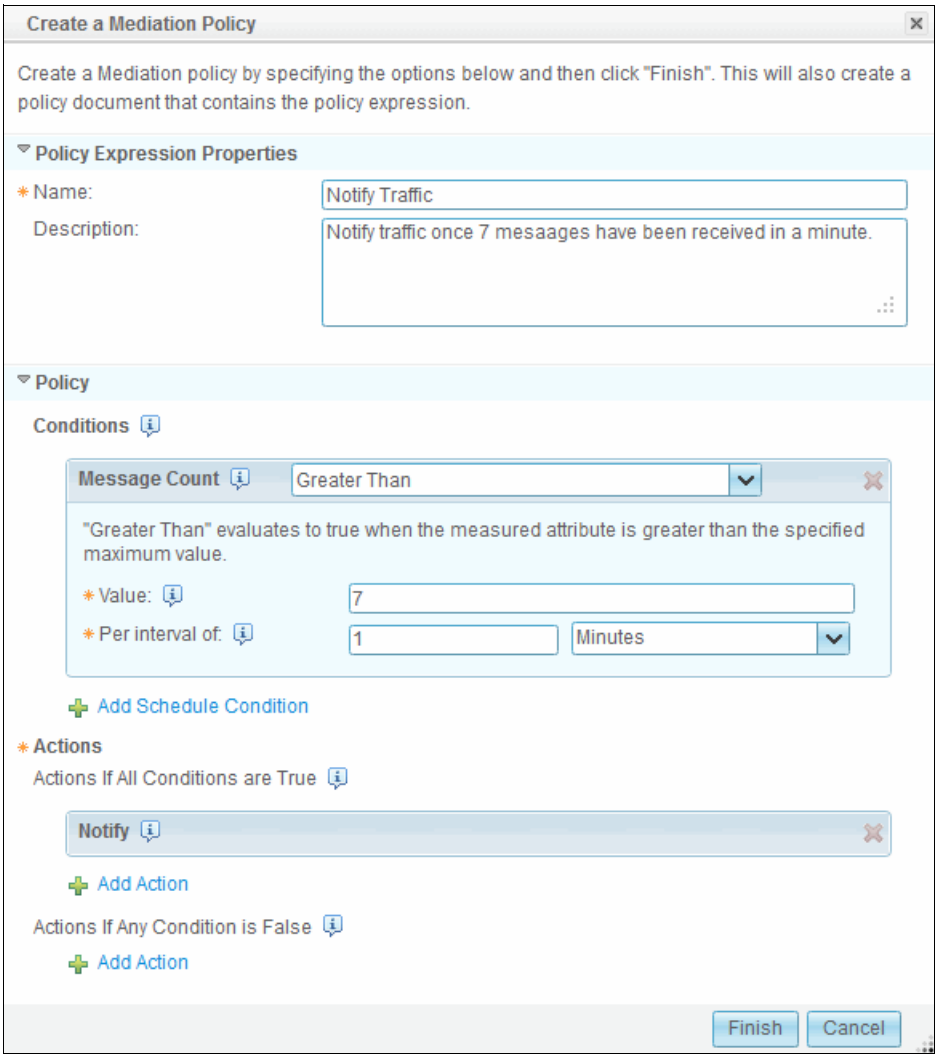


Figure 3-32 Mediation policy to notify

3. After creating the mediation policy, propose and approve the specification:
   a. Navigate to **Policy Expression** → **Source Document** (`NotifyTrafficGT7.xml`).
   b. Take actions to propose and approve the specification:
      i. Select **Action** → **Propose Specification**.
      ii. Select **Action** → **Approve Specification**.

   The governance state now reflects the `Approved` state.

The Service Registry Navigator displays the artifacts such as policy document and policy expression.

3.4.4 Running client transactions to validate provider services

After the promotion of services and the synchronization with ITCAM for SOA and DataPower occurs, you can use the web client to validate the services and that their policies are working correctly. See “Using the web material” on page 427 for further details. Ensure you update the endpoints to point to the Web Services Gateway endpoints in DataPower and use the correct consumer and context identifiers.

To validate the sample business services provided as part of this book, see Appendix A.2, “Validating installation of each individual product in the SOA Policy Solution” on page 417.

3.5 Attaching policies

Attach the policies to services that match a condition. Policies are attached to SLDs that match the services, based on the specified criteria.

3.5.1 Attaching a policy to a specific service and operations

If you want a policy to apply for only some of the operations that are available from a service, you must create an SLD that offers only these operations. Policies that are applied to SLDs with no specified operations are the same as polices that are applied to SLDs with all the operations specified. Ensure that all operations are covered by at least one SLD if they are to be available for consumption.

The procedures in the following sections describe how to attach the Queue Traffic policy to the Itinerary Reservation service for the add, update, and delete operations.

Adding operations to the SLD

Complete the following steps:

1. Choose **Service Level Definition** from the drop-down menu in the **Search** widget, and click **Search**.
2. Navigate to **SLD - Itinerary Reservation Service** → **Edit** → **Available operations** → **Find**.
3. Choose **add, update, and delete operations** → **Finish**.

Attaching a policy to the SLD

Complete the following steps:

1. Choose **Policy Expressions** from the drop-down menu in the **Search** widget.
2. Enter `Queue Traffic` in the text field and click **Search**.
3. Navigate to **Queue Traffic** → **Action** → **Manage Policy Attachments** → **Attach to Specific Items**.

The dialog in Figure 3-33 opens.

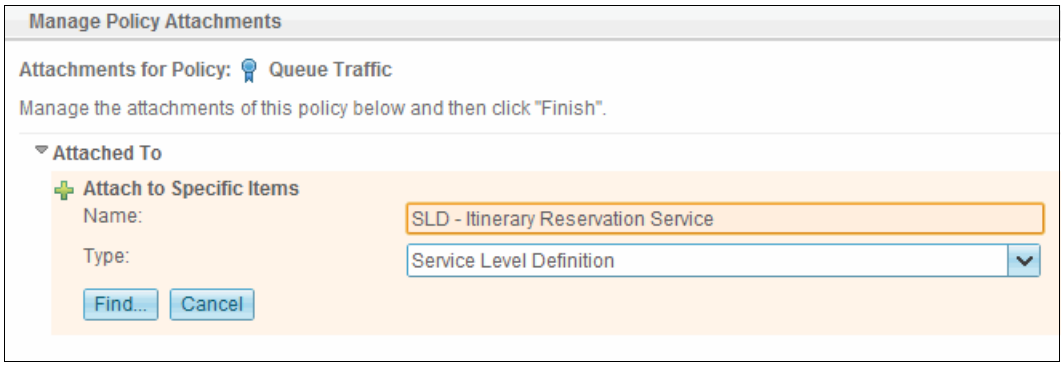


Figure 3-33 Manage Policy Attachments

4. Select **Service Level Definition** type and enter the SLD name, in this case, `SLD - Itinerary Reservation Service`. Click **Finish**.

Figure 3-34 shows the attached policies from the SLD perspective. Note that there are other policies attached.

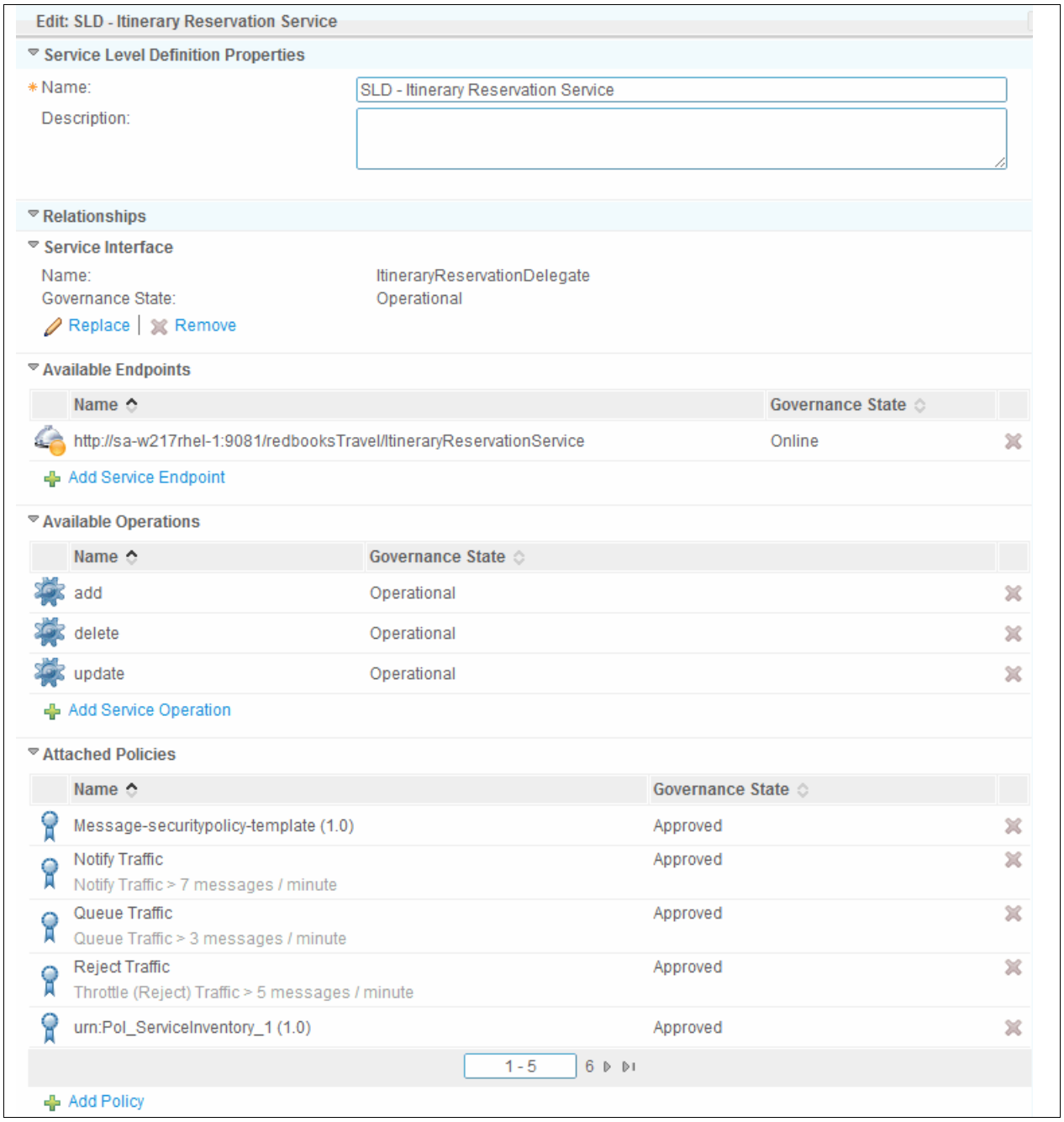


Figure 3-34 SLD with queuing policy attachment to ItineraryReservationService and operations

3.5.2 Attaching the policy to all services matching a name

The following steps show how to attach the Reject Traffic policy to all services that match the term *Itinerary*. In this case, two services, ItineraryReservation and ItineraryAvailability, match the condition.

1. Choose **Policy Expressions** from the drop-down menu in the **Search** widget.
2. Enter `Reject Traffic` in the text field and select **Search.**

3. Navigate to **Reject Traffic** → **Action** → **Manage Policy Attachments** → **Attach to Items Using a Query** → **Add Query -> Custom Query**. The dialog in Figure 3-35 opens.

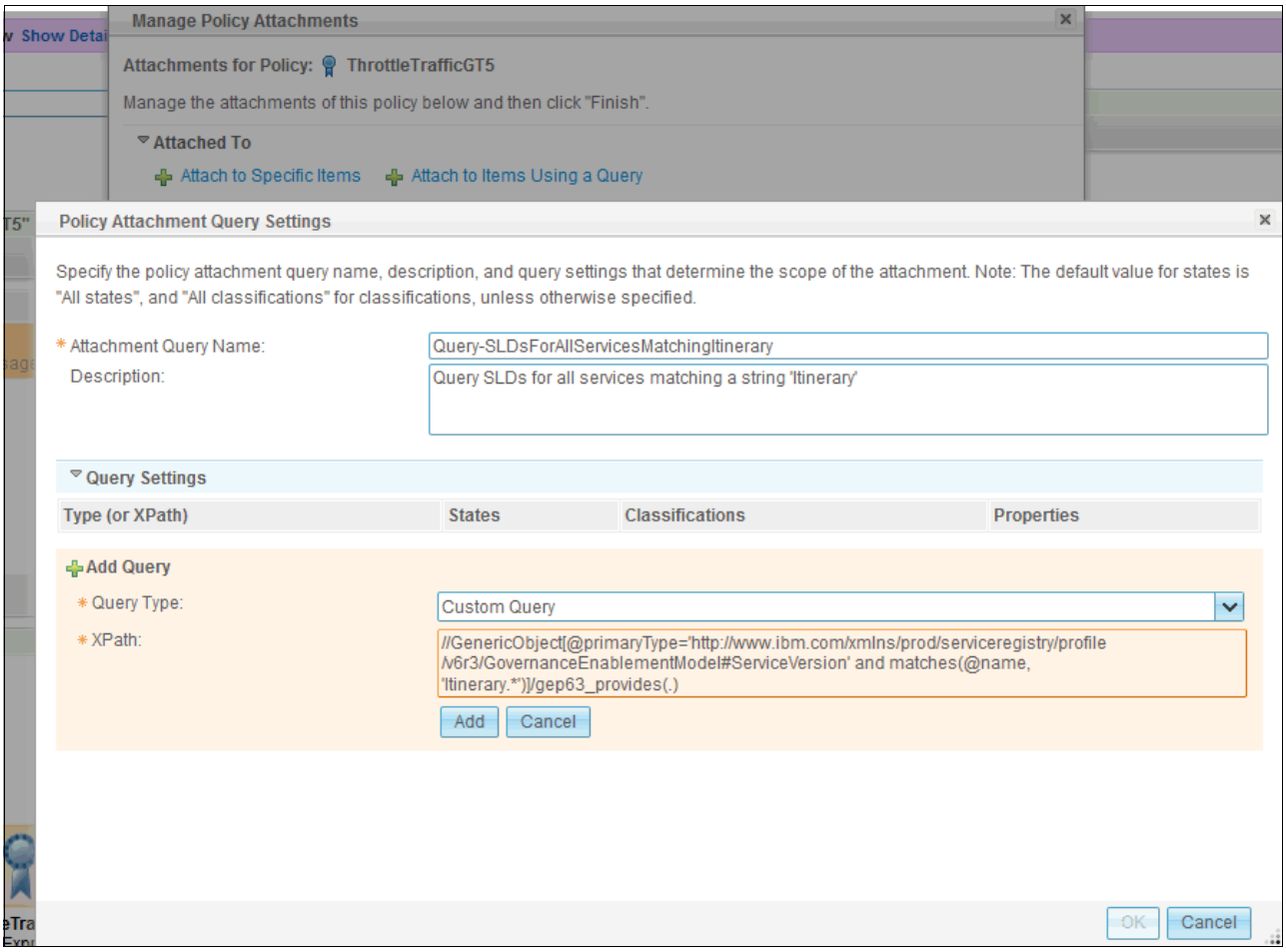


Figure 3-35 Policy attachment using XPATH custom query

In this case, you want to attach to SLDs where the service version name starts with the string `Itinerary`. This case is complex and requires the use of XPATH. Example 3-3 shows the XPATH to use in this case.

Example 3-3 XPATH to retrieve SLDs for Service Version matching a string

```
//GenericObject[@primaryType='http://www.ibm.com/xmlns/prod/serviceregistry/profile/v6r3/GovernanceEnablementModel#ServiceVersion' and matches(@name, 'Itinerary.*')]/gep63_provides(.)
```

4. Enter the values that are in Figure 3-35 and Example 3-3, and then click **Add**.
5. Click **OK**.
6. Click **Finish**.

Tip: The advantage of using the Attach by Query method is that it is dynamic. If a new service is added in the future, for example *Itinerary Confirmation*, the query will automatically apply policy to the new matching SLDs.

Figure 3-36 shows that the service navigator is displaying policy attachments on SLDs based on the XPATH query.

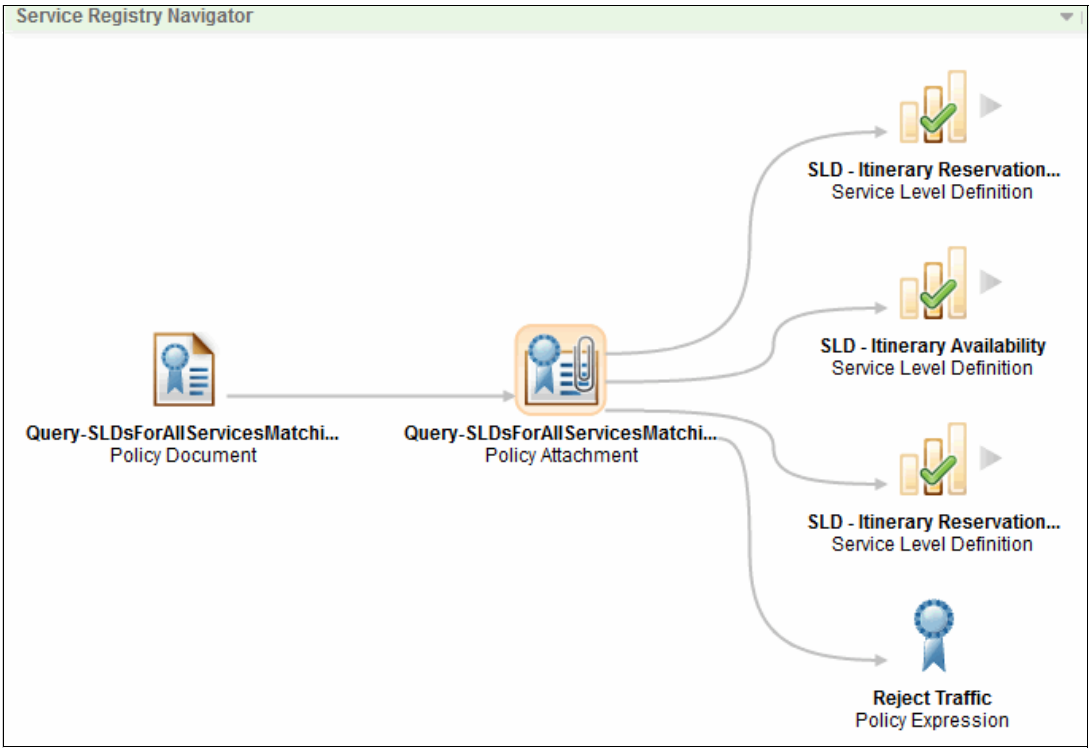


Figure 3-36 Service Registry Navigator view of query showing SLDs that match Itinerary

3.5.3 Attaching policy to all services for an organization

The following procedure shows how to attach the Notify Traffic policy to all services that belong to ITSO Redbooks Travel organization. In this case, three services will match the condition: ItineraryReservation, ItineraryAvailability, and Pricing.

1. Choose **Policy Expressions** from the drop-down menu in the **Search** widget.
2. Enter `Notify Traffic` in the text field, and select **Search**.
3. Navigate to **Notify Traffic** → **Action** → **Manage Policy Attachments** → **Attach to Items Using a Query** → **Add Query** → **Custom Query.**

   Again, this case is complex because you attach your policy to SLDs where the implementing Service Version has ITSO Redbooks Travel as the owning organization. Example 3-4 shows the XPATH query.

Example 3-4 XPATH to retrieve SLDs for Service Version matching organization

```
//GenericObject[@primaryType='http://www.ibm.com/xmlns/prod/serviceregistry/profile/v6r3/GovernanceEnablementModel#ServiceVersion' and ale63_owningOrganization(.)/@name='ITSO Redbooks Travel']/gep63_provides(.)
```

4. Enter a name for your query, for example, `All SLDs with Service Versions owned by IBM ITSO Redbooks Travel`.
5. Copy and paste in the XPATH shown in Example 3-4 on page 82.
6. Click **Add**.
7. Click **OK**.
8. Click **Finish**.

Figure 3-37 shows the service navigator displaying policy attachments on SLDs based on the XPATH query.

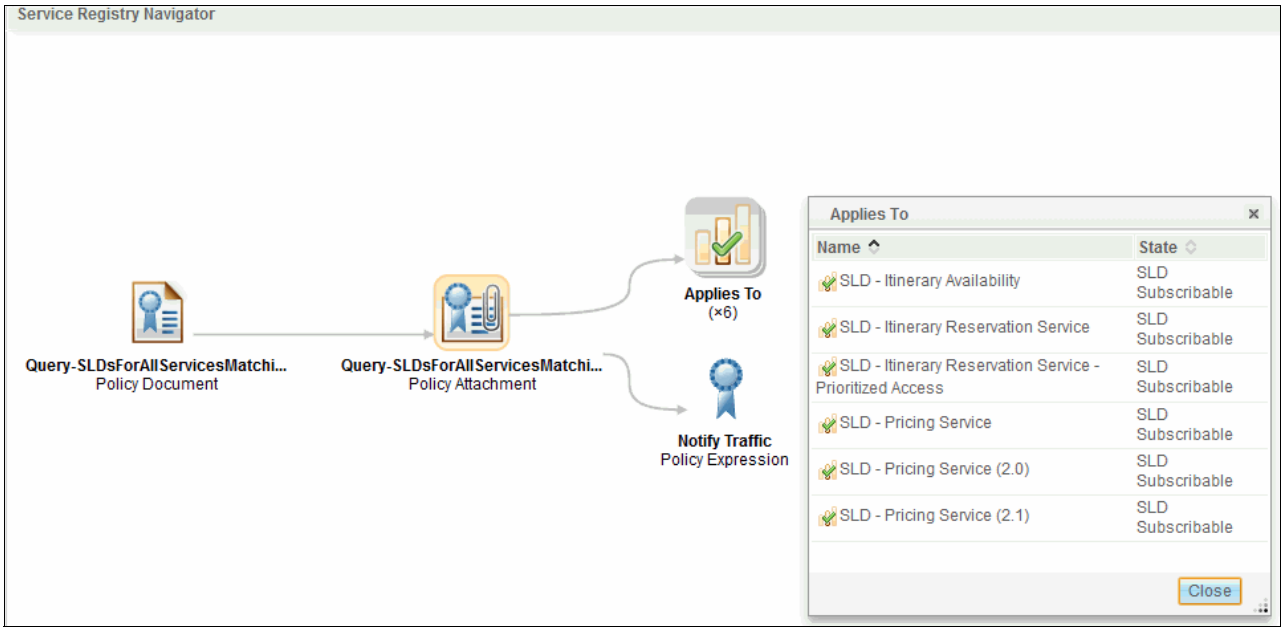


Figure 3-37 Service Registry Navigator view of query showing SLDs matching organization

3.6 Reporting on services and policies applied to services

To view all of the business services, use the following selection path from the Search menu:

Go To Spaces → IBM Redbooks Travel Service Registry for SOA Governance → Browse → Select Business Services

Figure 3-38 shows all available Business Services and details of the Itinerary Reservation Business Service.

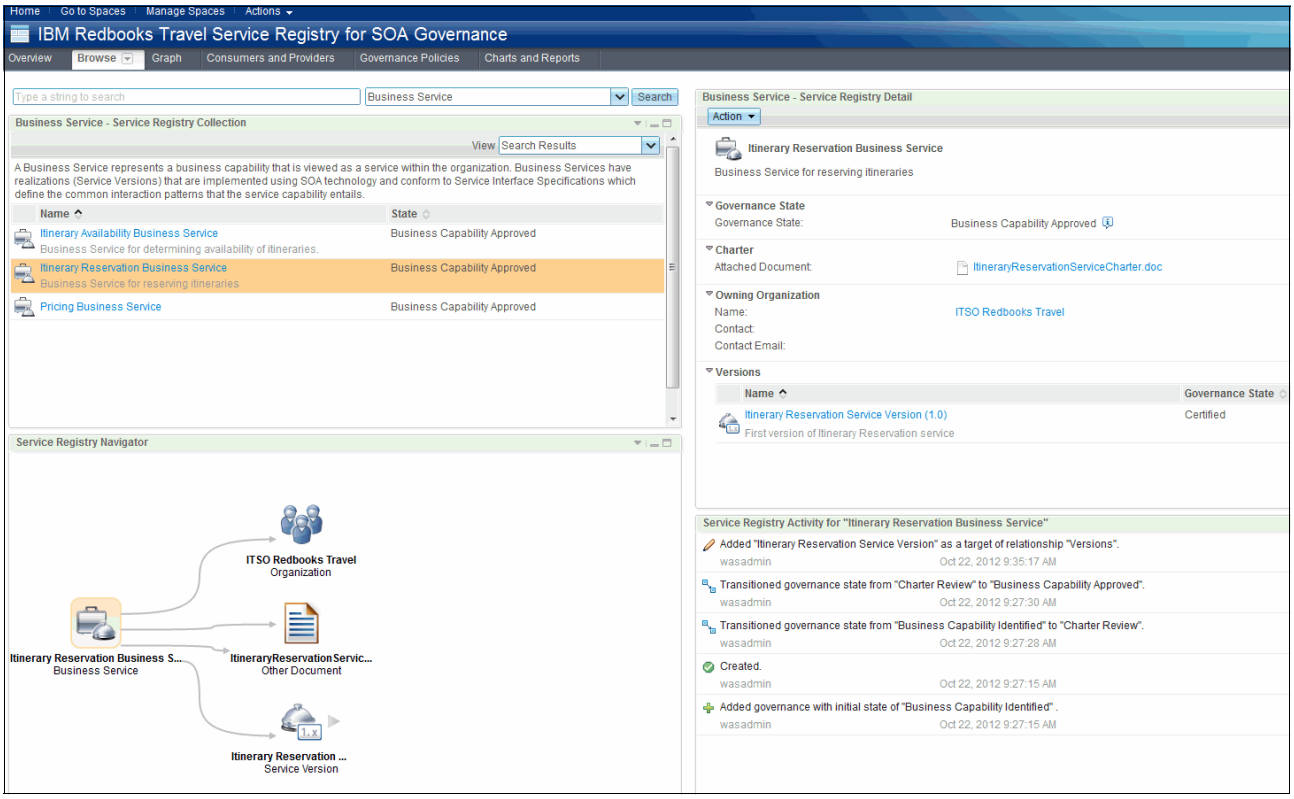


Figure 3-38 Report on Business Services

To view all Service Versions, use the following selection path from the Search menu:

Go To Spaces → IBM Redbooks Travel Service Registry for SOA Governance → Browse → Select Service Version

Figure 3-39 shows all available Service Versions and details of Itinerary Reservation Service Version (1.0). The Service Version also lists the SLDs.

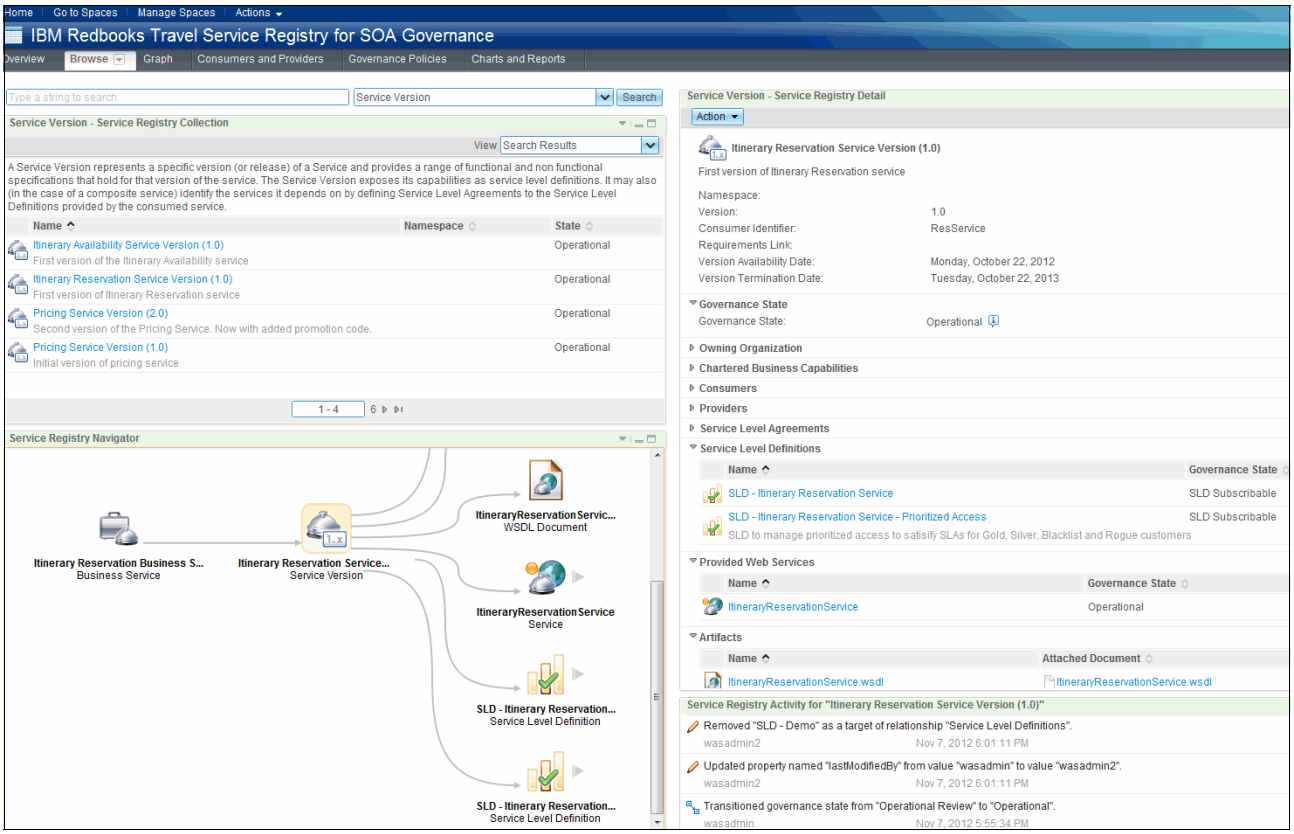


Figure 3-39 Report on Service Versions

To view the policies that are attached to an SLD, click the SLD name in the Detail widget.

Figure 3-40 shows policies like Queue Traffic, Notify Traffic applied to Itinerary Reservation Service Version (1.0) attached to SLD - Itinerary Reservation Service.

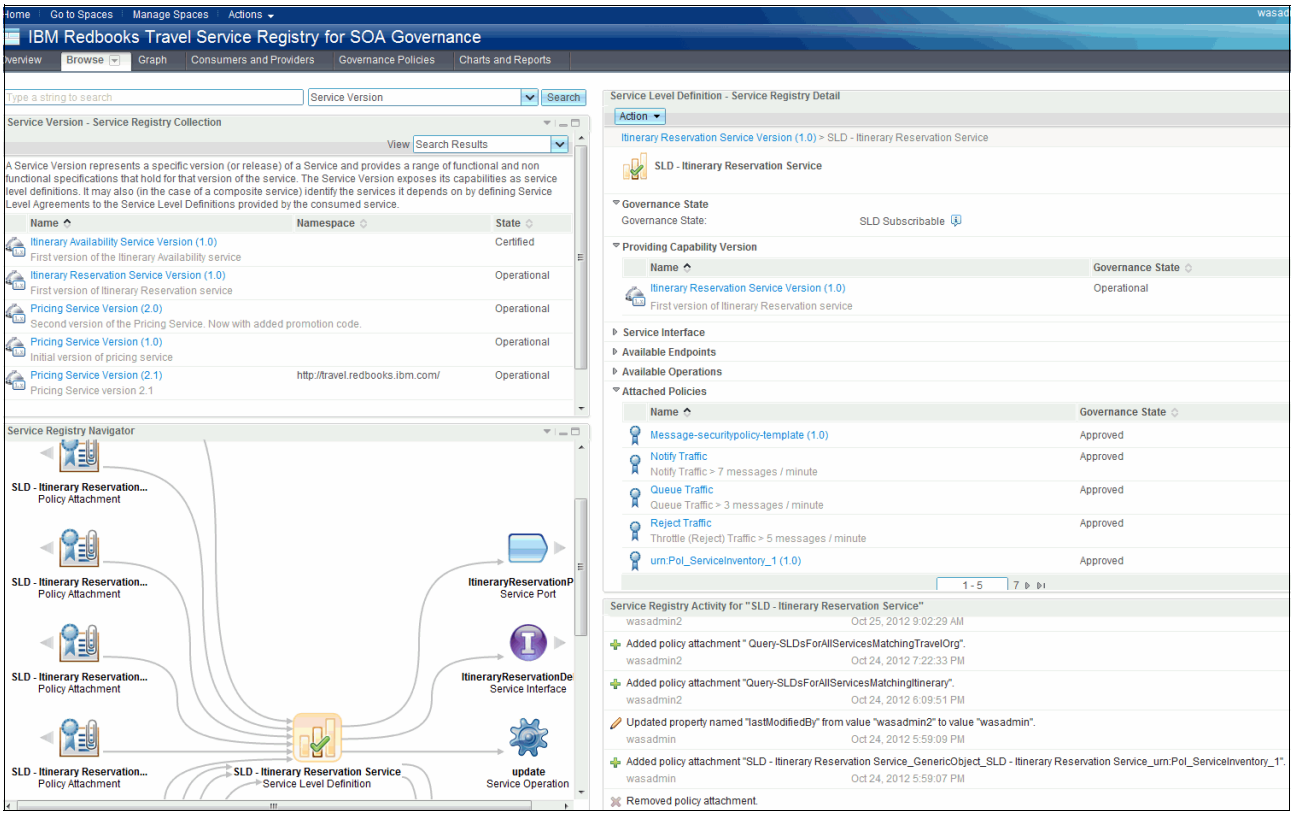


Figure 3-40 Reporting Policy Applied to a Service

4

Policy traffic management and consumer provider pairs

This chapter expands upon the previous chapter by extending the discussion about policy for provider services to providing policy for a consumer-provider service pair. Service level agreements between consumers and providers are illustrated with examples.

This chapter contains the following topics:

- 4.1, “Implement the consumer-to-provider services scenario” on page 88
- 4.2, “Determining customer priority” on page 99
- 4.3, “Customer priority policy example” on page 100

© Copyright IBM Corp. 2013. All rights reserved.

4.1 Implement the consumer-to-provider services scenario

ITSO Redbooks Travel Company set up provider services as described in Chapter 3, “Policy traffic management, provider only, with operations” on page 35. The services are ready to be consumed by business partners and clients through the service gateway. This section outlines the sequence of required steps to ensure that the consumer-to-provider services are completely functional with the service level agreements (SLAs).

1. Create a business application with capability application version.
2. Govern application services.
3. Create a service level agreement.
4. Create policies.
5. Add an SLD as an agreed endpoint to an SLA.
6. Attach policies to an SLA.
7. Promote the services to WSRR run times.
8. Synchronization with DataPower.
9. Run the client transactions to validate the SLA between consumer and provider.

4.1.1 Create a business application with capability application version

Create a business application (client application) to consume the services that are offered by the provider. A business application has a capability (application) version. An SLA is established between the application version and service version. The following steps create a business application and add a capability version:

1. Open the Business Space UI for the Governance Master WSRR at:

   `https://govmasterwsrr:9443/BusinessSpace/`

2. Navigate to **Go To Spaces** → **IBM Redbooks Travel Service Registry Operations Space** → **Overview** → **Service Registry Actions** → **Create a Business Application**.

   Figure 4-1 shows the Reservation Client Application that was created to invoke the Itinerary Reservation Service.

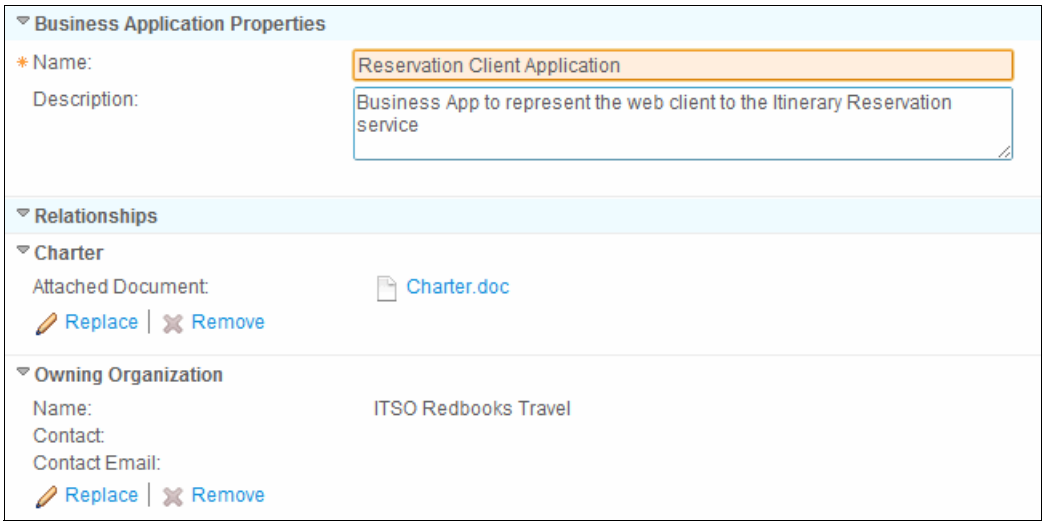


Figure 4-1 Create a business application for Itinerary Reservation

3. Add the Capability Version (Type: Application Version) to the business application. Specify the version and the consumer identifier. See Figure 4-2.

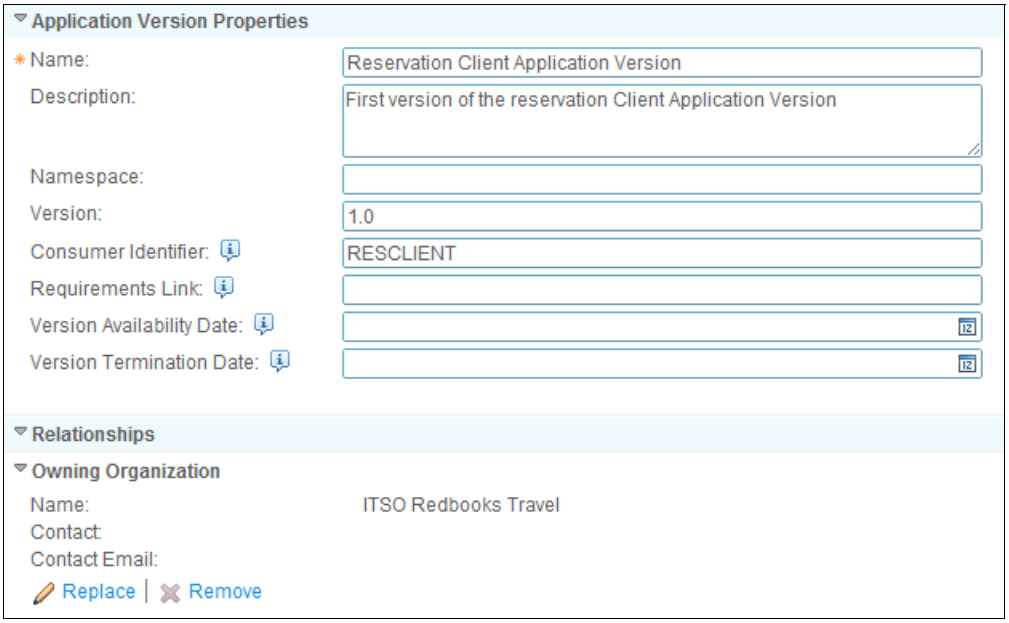


Figure 4-2 Create a capability (application) version for reservation client

4. Figure 4-3 shows the business application with added capability version. Click **Finish**.

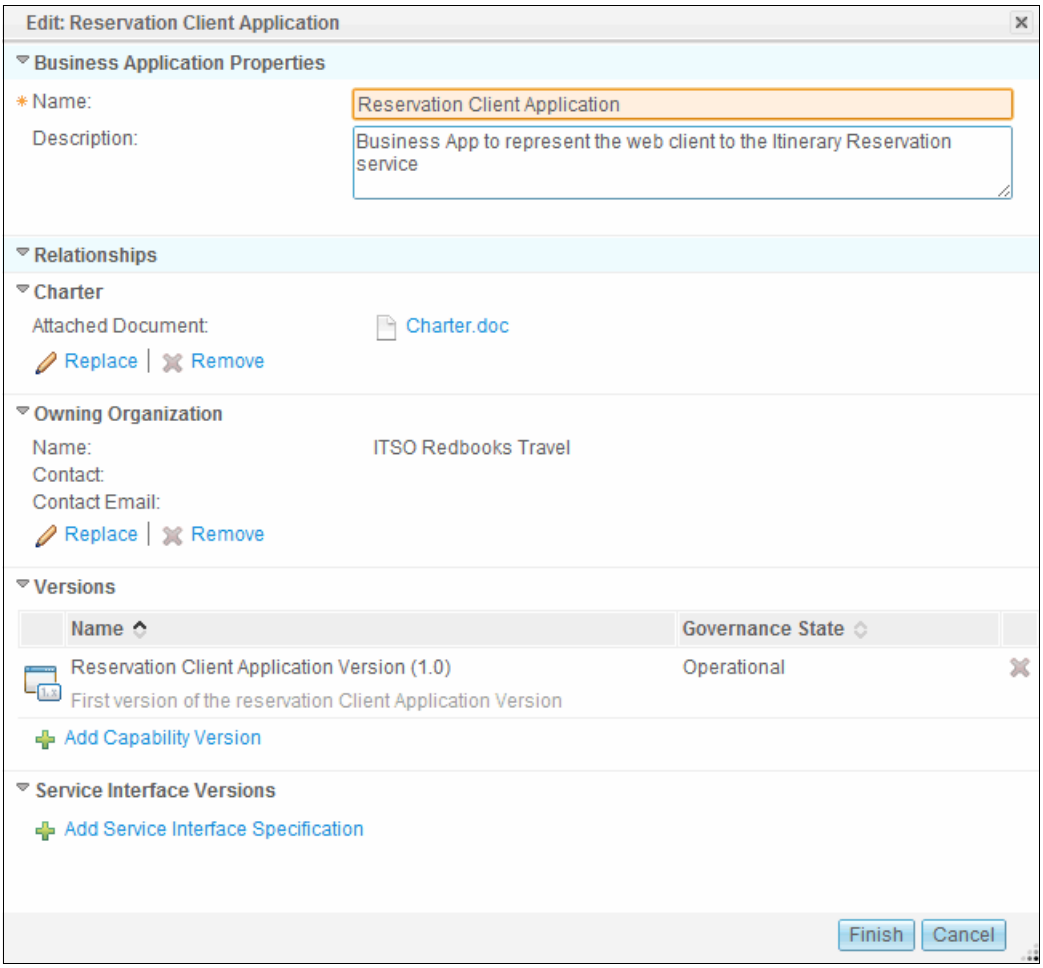


Figure 4-3 Reservation client with capability version

In this example, two business applications are created:

- Reservation Client Application (Consumer Identifier: RESCLIENT)
- Pricing Client Application (Consumer Identifier: PRICLIENT)

For the scenarios in this book, the Itinerary Reservation Service is also a client to the Itinerary Availability service, which, therefore, demonstrates how one provider service acts as a client to another service.

Figure 4-4 illustrates the consumer provider pairs for related services.

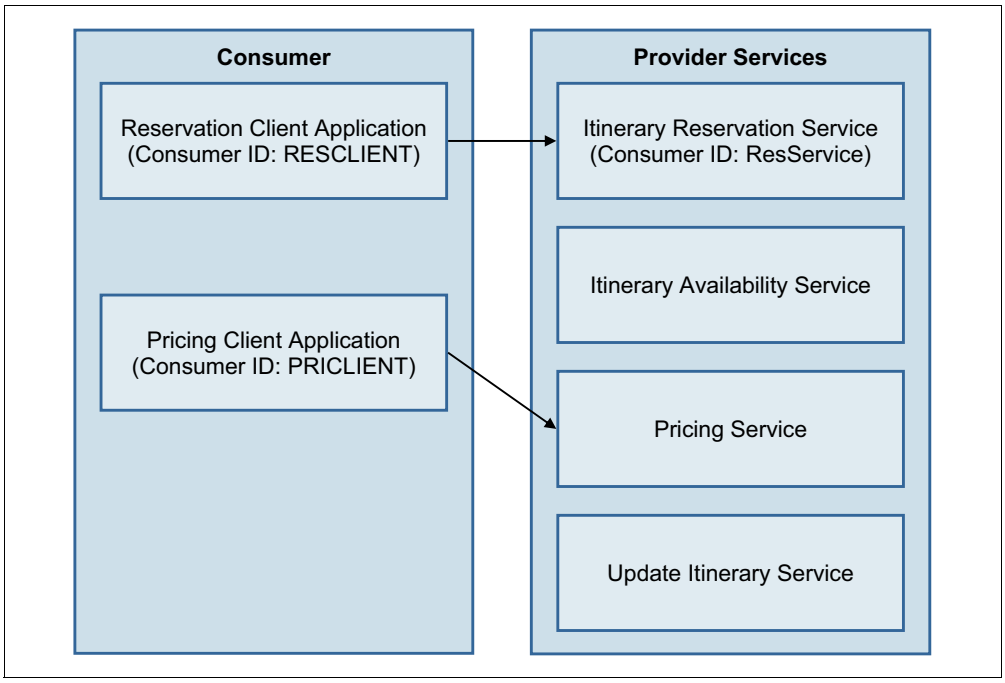


Figure 4-4 Consumer and provider pairs

4.1.2 Govern business application

A governing business application involves similar steps as governing business services. See the following sources:

- For service governance overview, see 3.2, “Service governance” on page 54.
- For further information about governing a new service, see the following WebSphere Service Registry and Repository V8.0 information center:

  `http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/tutorial_gep_bspace_govern_new_service.html`
- To understand the lifecycles in the governance enablement profile (GEP), see the following website:

  `http://pic.dhe.ibm.com/infocenter/sr/v8r0/index.jsp?topic=%2Fcom.ibm.sr.doc%2Frwsr_gep_life_cycles.html`

4.1.3 Create a service level agreement

The service level agreement (SLA) is a contract between the consumer and provider. To have an active SLA, a client application and a provider service version is required.

Use the following steps to create a sample SLA:

1. Navigate to **Go To Spaces → IBM Redbooks Travel Service Registry Operations Space → Overview → Search wizard → Application Version → Reservation Client Application Version (1.0) → Service Registry Detail → Edit → Service Level Agreements → Add Service Level Agreement → Create.**
2. Create the SLA by typing in the name `SLA - Gold Consumers - Itinerary Reservation Service` with the context identifier `Gold`, as shown in Figure 4-5 on page 92.

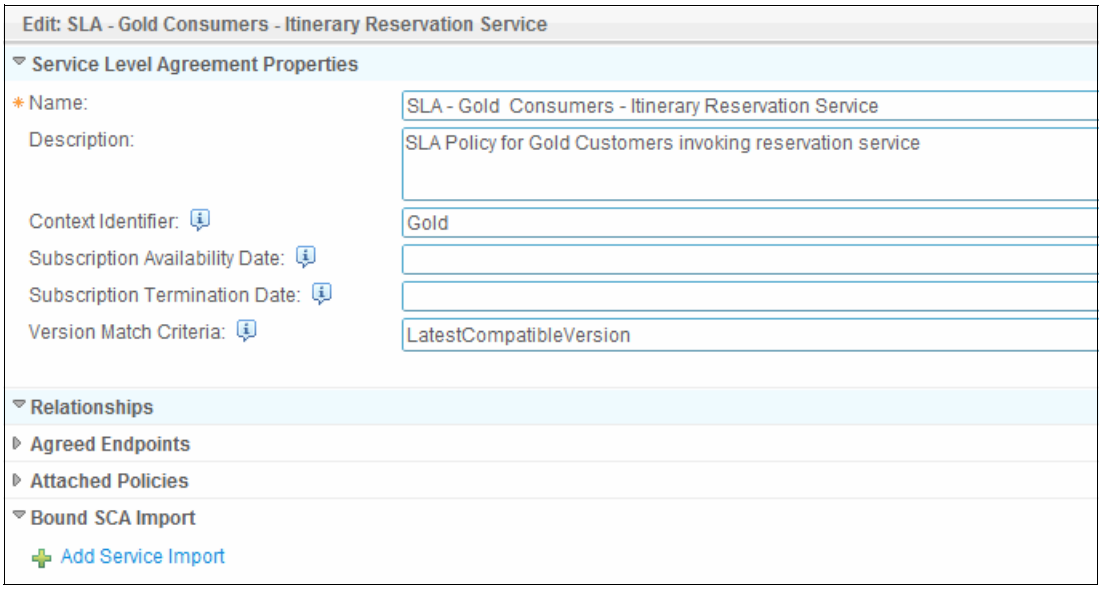


Figure 4-5 Create an SLA

More examples of creating SLAs are in 4.3, “Customer priority policy example” on page 100.

4.1.4 Create policies

A policy expression represents the declaration of a policy and is equivalent to a `<wsp:Policy>` element in the WS-Policy document.

Use the following steps to create a mediation policy named *Route Gold Customers*:

1. Define the attribute condition Average Total Latency with the value `1000` milliseconds and a per interval of `15 Seconds`.
2. Add the Action Route Message with secondary endpoint pointing to the following URL (note the higher port):

   `http://services:9082/redbooksTravel/ItineraryReservationService`

   When the condition of average total latency is greater than 1000 milliseconds per interval of 15 seconds is met, the Route Message action will redirect the requests to the secondary endpoint.

Figure 4-6 shows the results.

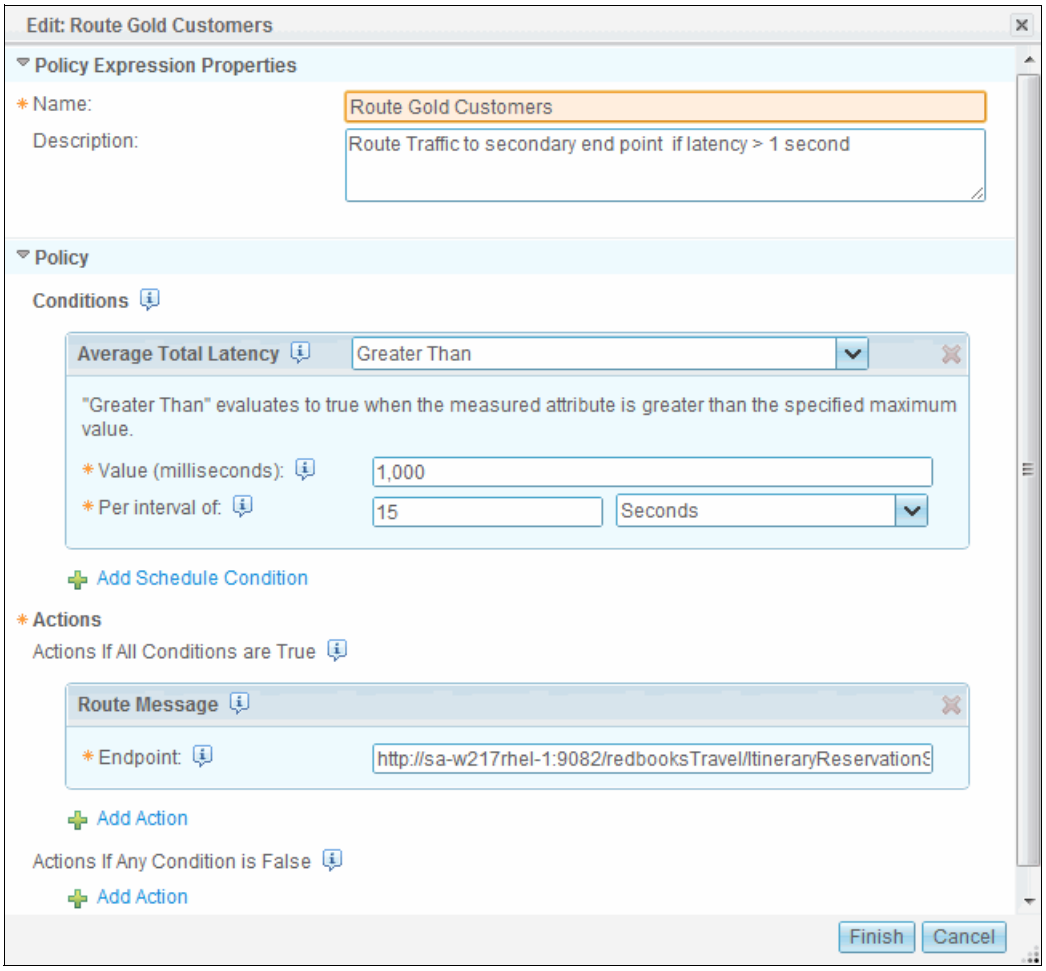


Figure 4-6 Create policy

3. Govern the policy to `Approved` governance state, click **Finish**.

Repromote: If you add or update a policy that is related to a capability version, repromote the capability version to see the changes.

More examples of creating different policies are in 4.3, “Customer priority policy example” on page 100. Also see 3.4, “Creating policies” on page 73.

4.1.5 Add the SLD as an agreed endpoint to the SLA

To have an active SLA, there must be at least one agreed endpoint, defined by an SLD. For example, add `SLD - Itinerary Reservation Service - Prioritized Access` as an agreed endpoint to `SLA - Gold Consumers - Itinerary Reservation Service`.

Navigate to **Search wizard** → **Service Level Agreements** → **SLA - Gold Consumers - Itinerary Reservation Service** → **Edit** → **Agreed Endpoints** → **Add Service Level Definition** → **Find** → **Select SLD - Itinerary Reservation Service** → **Finish**.

Figure 4-7 shows *S*LA - Gold Consumers - Itinerary Reservation Service tied to SLD - Itinerary Reservation Service - Prioritized Access.

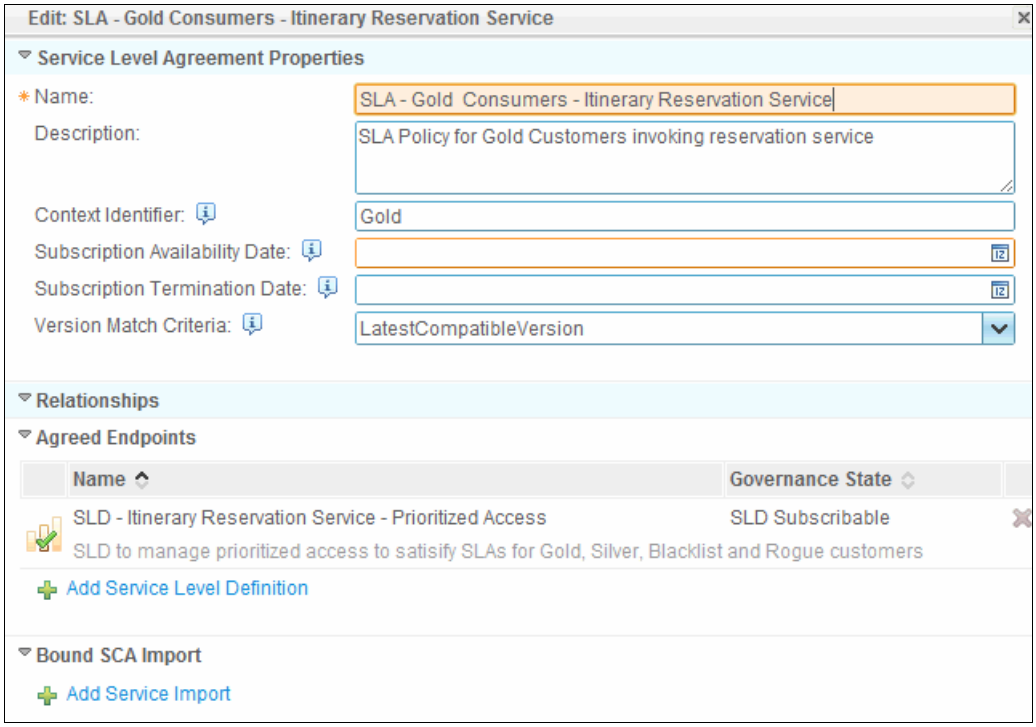


Figure 4-7 SLA tied to SLD for Gold customers

4.1.6 Attach policies to an SLA

To attach the Route Gold Customers mediation policy to the `SLA - Gold Consumers - Itinerary Reservation Service` SLA, navigate to **Search wizard → Service Level Agreements → SLA - Gold Consumers - Itinerary Reservation Service → Edit → Attached Policies → Add Policy → Find → Route Gold Customers → Finish.**

Figure 4-8 shows the `SLA - Gold Consumers - Itinerary Reservation Service` SLA with the `Route Gold Customers` policy attachment.

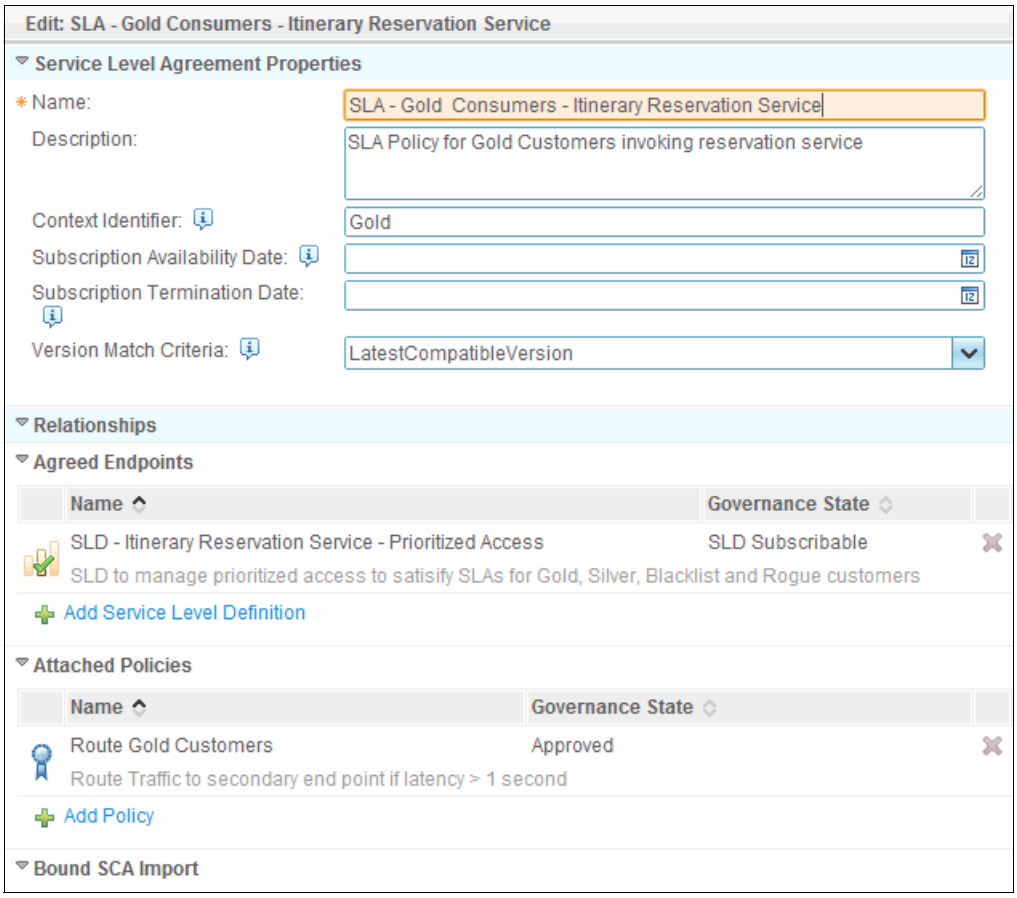


Figure 4-8 SLA with policy attached for Gold Customers

4.1.7 Promote the services to the WSRR run time

Promotion is done on a capability version. Govern the Capability (Application) Version through the various states until the Operational state. In this topology, there is one stage WSRR runtime environment and one production WSRR runtime environment. The SLA should be in the `SLA Active` state and policy should be in the `Approved` state before promoting the capability versions.

To promote the Application Version, the SLAs that are associated with it must be in the `SLA Active` state.

Use the following steps to govern the SLA to an `SLA Active` state. The SLA is initially in the `SLA Identified` state.

1. Select **Action** → **Request SLA** → **OK** to govern to `SLA Requested` state.
2. Select **Action** → **Approve SLA Request** → **OK** to govern to `SLA Inactive` state.
3. Select **Action** → **Activate SLA** → **OK** to govern to `SLA Active` state.

To be ready for promotion, key artifacts must be transitioned to the governance states listed Table 4-1. Any changes must again go through the lifecyle transitions.

Table 4-1 Governance states

Artifacts	Governance state ready to promote to Staging WSRR	Governance state ready to promote to Production WSRR
Business Application	Business Capability Approved	Business Capability Approved
Capability (Application) Version	Staging Review	Operational Review
Service Level Agreement	SLA Active	SLA Active
Policy	Approved	Approved

The current governance state of Application Version Reservation Client Application Version (1.0) is assumed to be in `Realized`.

1. Navigate to **Go To Spaces** → **IBM Redbooks Travel Service Registry Operations Space** → **Overview** → **Search wizard** → **Application Version** → **Reservation Client Application Version (1.0)** → **Service Registry Detail.**
2. To promote the services to Stage WSRR run time, perform the following steps:
   a. Select **Action** → **Propose Stage Deployment** → **OK to** govern to `Staging Review` state.
   b. Select **Action** → **Approve Staging Deployment** → **OK** to govern to `Staged` state.

   The `Staged` governance state indicates that the services were successfully promoted to the Stage WSRR run time.
3. To promote the services to production WSRR run time, use the following steps:
   a. Select **Action** → **Propose Certification** → **OK** to govern to `Certification Review` state.
   b. Select **Action** → **Approve Certification** → **OK** to govern to `Certified` state.
   c. Select **Action** → **Propose Production Deployment** → **OK** to govern to `Operational Review` state.
   d. Select **Action** → **Approve Production Deployment** → **OK** to govern to `Operational` state.

   The `Operational` governance state indicates that the services were successfully promoted to the production WSRR run time.
4. If there are any changes to artifacts after promotion, two steps more steps must be done for the changes to propagate to WSRR run times:
   a. Regovern the objects to required governance state.
   b. Redefine and repromote the capability version to staging and production WSRR.

4.1.8 Synchronization with DataPower

Based on the synchronization mechanism (automatic, manual, or poll) that is defined in DataPower, the promoted services are propagated to DataPower and the SLAs are enforced. For this scenario, automatic synchronization is set between WSRR and DataPower with a five-minute interval.

To see the changes immediately in DataPower, a manual synchronization can be done as follows:

1. Log in to DataPower and select **Control Panel** → **Web Service Proxy** → **ITSORedbooksTravel_WSP**. Then, click the **Okay** link under WSDL Status, shown in Figure 4-9.

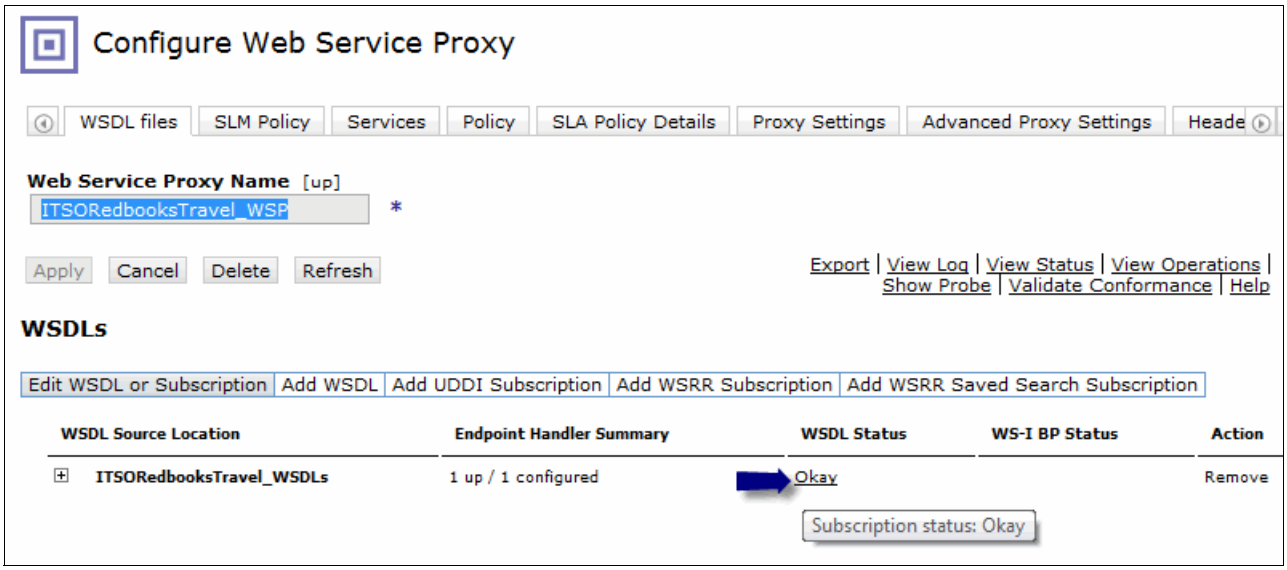


Figure 4-9 Subscription status link in DataPower WS Proxy

2. Figure 4-10 shows the WSRR Subscription Status page in DataPower with the **Synchronize** option. Do the following steps:

   a. Click **Synchronize** to manually synchronize the changes from WSRR run time to DataPower.

   b. Refresh the status to see the live synchronization status.

   During the synchronize process, the status indicates `Synchronizing`. Upon successful synchronization, status indicates `Okay`.

Figure 4-10 WSRR synchronize option in DataPower

To view the SLA and policy changes, navigate to **Web Service Proxy** → **ITSORedbooksTravel_WSP** → **SLA Policy Details tab** → **SLA table**.

In Figure 4-11, the arrows point to SLA and Policy in the SLA table that is associated with the Itinerary Reservation Service.

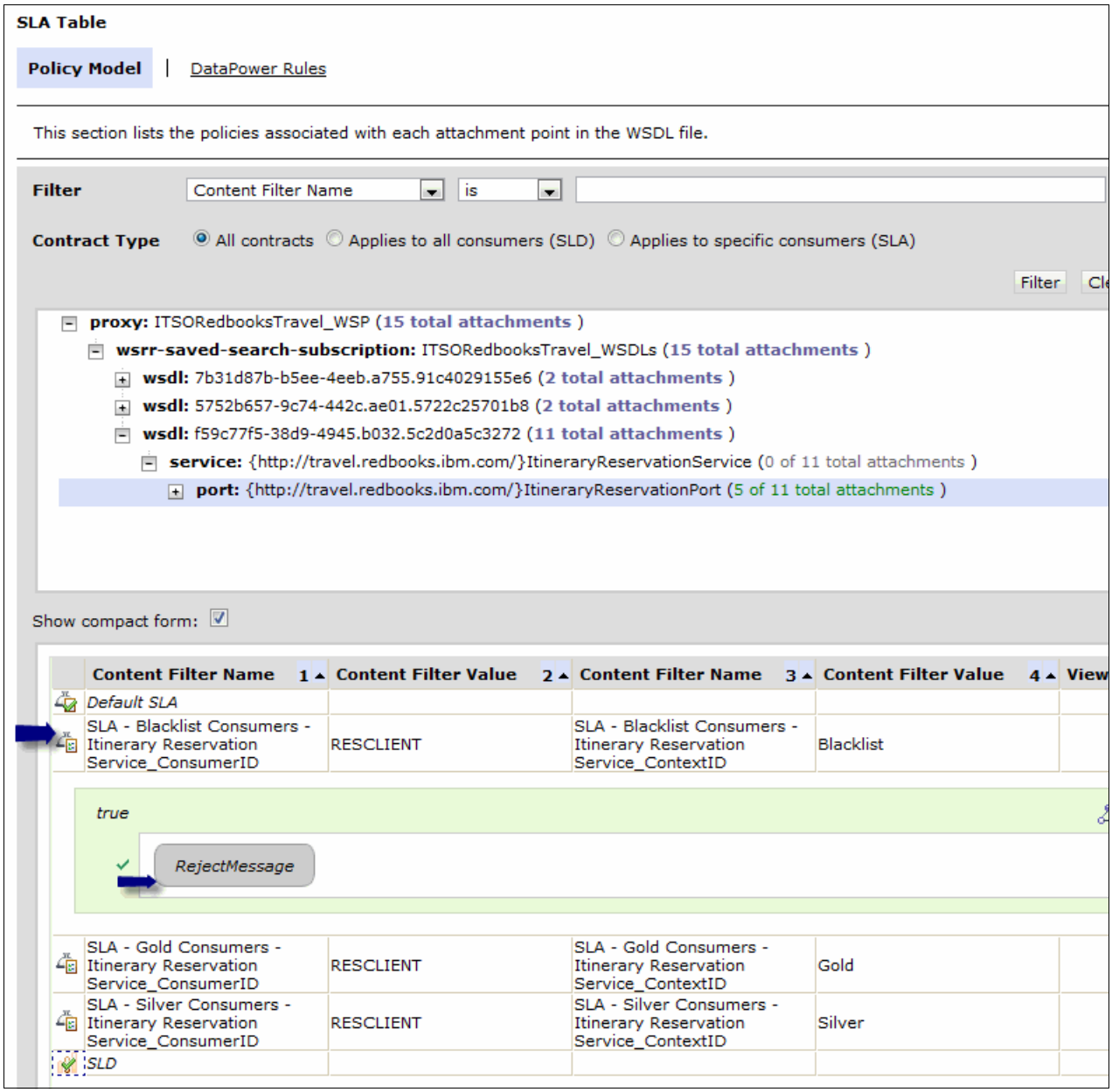


Figure 4-11 DataPower WS Proxy SLA table

4.1.9 Validate the services

After the promotion of services and synchronization occurs, you can run the client to validate the services. Client services can be run from SOAP UI or Java client. Update the endpoints to point to the service gateway endpoints in DataPower and validate the consumer-to-provider services with the SLAs.

4.2 Determining customer priority

It is possible to have multiple SLAs between the same consumer Capability Version (Application Version or Service Version in the example) and SLD. The justification is that separate customers of a service consumer might be required to receive separate levels of service.

For example, ITSO Redbooks Travel might want to reward regular customers by giving them a better level of service. This level might be realized by the underlying services being guaranteed to respond more quickly. Because this guarantee might be more expensive for ITSO Redbooks Travel, it would not want to use this for just anyone. The solution to this problem is that service requests coming from ITSO Redbooks Travel's most important customers can include a context identifier to match it to an SLA with a more strict set of requirements. Similarly, lesser customers might be given a lower level or service, and unknown customers are given the lowest quality of service.

Each SLA has a *context identifier* property. This property must match the context identifier in the incoming messages. In this example, the context and consumer identifiers are placed in the SOAP header. Example 4-1 shows a sample SOAP envelope, including the header with the consumer and context identifiers.

Example 4-1 Sample SOAP envelope calling the Iteration Availability Service

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Header>
        <ContextIdentifier xmlns="com.redbooks.travel">Gold</ContextIdentifier>
        <ConsumerIdentifier
xmlns="com.redbooks.travel">AVICLIENT</ConsumerIdentifier>
    </soapenv:Header>
    <soapenv:Body>
        <p755:available xmlns:p755="http://travel.redbooks.ibm.com/">
            <arg0>New York 123</arg0>
            <arg1>August 2012-December 2012</arg1>
        </p755:available>
    </soapenv:Body>
</soapenv:Envelope>
```

The SLD specifies where in the incoming messages the consumer and context identifiers are located, as Figure 4-12 shows.

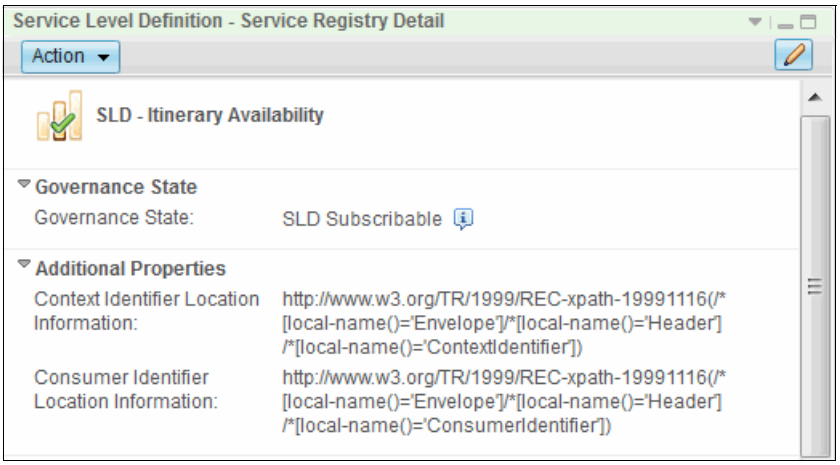


Figure 4-12 Service Level Definitions - Service Registry Detail

For further information about these settings, visit the information center:

http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/rwsr_gep_service_level_definition.html

4.3 Customer priority policy example

You can create and apply different policies to customers, based on priority levels. The high-level steps to implement the scenario are as follows:

1. Choose the business service to which the customer priority policy will be applied.
2. Create an SLD and attach the SLD to the service version of business service.
3. Create an SLA between Consumer and Provider Services and specify the SLD endpoints.
4. Create mediation policies.
5. Attach the policies to the SLA.
6. Govern and promote the provider services and business applications.

The following steps demonstrate the scenario (Itinerary Reservation Service):

1. The business service that the customer priority policy applied will be the Itinerary Reservation Service.
2. Create SLD - Itinerary Reservation Service - Prioritized Access and attach it to Itinerary Reservation Service.
3. Create the following SLAs and add them to the Reservation Client Application. SLD - Itinerary Reservation Service - Prioritized Access is specified as an endpoint for the following SLAs:
   - SLA - Gold Consumers - Itinerary Reservation Service
   - SLA - Silver Consumers - Itinerary Reservation Service
   - SLA - Blacklist Consumers - Itinerary Reservation Service
   - Anonymous SLA - Rogue Consumers - Itinerary Reservation Service

4. Create the following mediation policies:
   - Route Gold Customers (Reroute traffic to secondary endpoint for latency greater than 1 second)
   - Route Silver Customers (Reroute traffic to secondary endpoint for latency greater than 2 seconds)
   - Reject Blacklist Customers (Reject customers with context Blacklist)
   - Reject Rogue Customers (Reject anonymous customers)
5. Attach the following policies to the corresponding SLAs:
   - Route Gold Customers attached to `SLA - Gold Consumers - Itinerary Reservation Service`
   - Route Silver Customers attached to `SLA - Silver Consumers - Itinerary Reservation Service`
   - Reject Blacklist Customers attached to `SLA - Blacklist Consumers - Itinerary Reservation Service`
   - Reject Rogue attached to `Anonymous SLA - Rogue Consumers - Itinerary Reservation Service`
6. Govern and promote the provider services and business applications.

The arrows in Figure 4-13 on page 102 show the SLAs that are described.

Preprocess for setting up the type of customer: Describing the process for how customers are determined to be gold, silver, or blacklist is outside the scope of this book. A typical implementation involves a preprocess that accesses a customer database and determines what type of customer this is. Based on the results of this preprocess, a context identifier is created in the message header. Described next is how that context identifier is specified in policy and then used during policy execution from the message header. As you see later in this example, a *rogue* is defined as any consumer that does not match one of our defined consumer IDs.

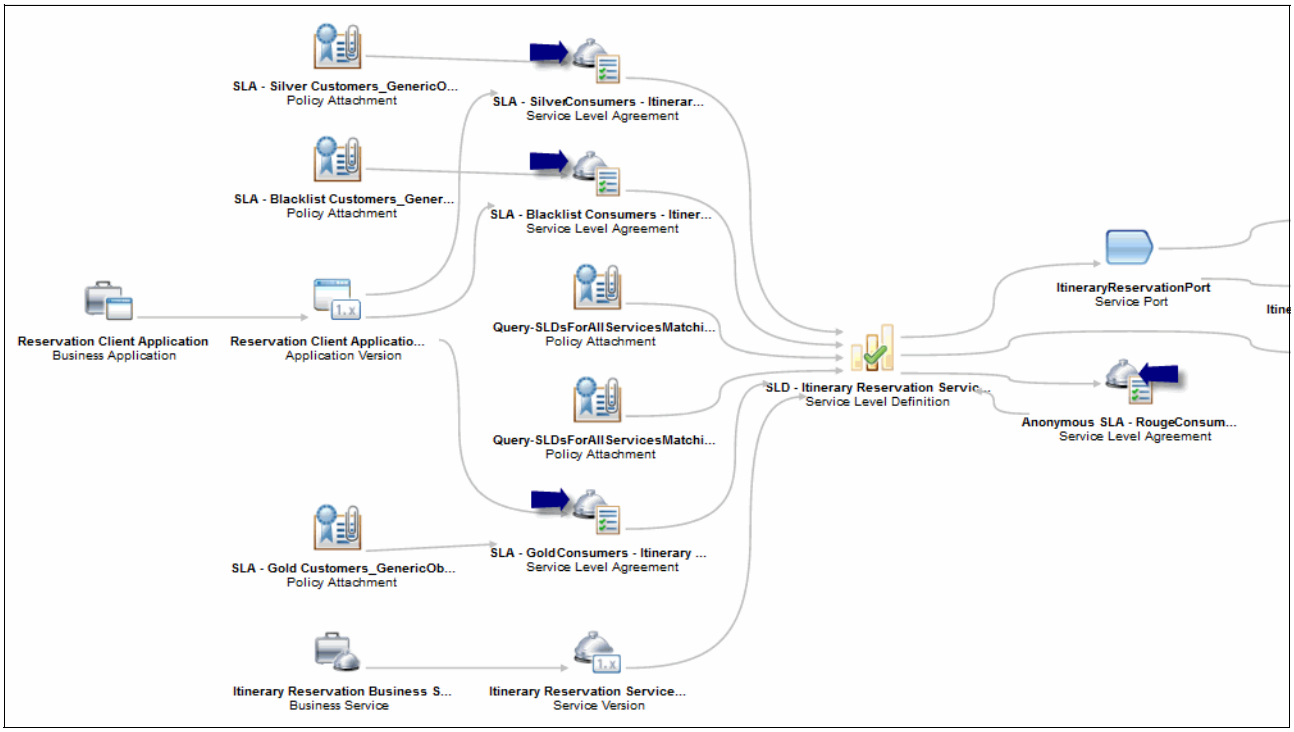


Figure 4-13 SLAs for Gold, Silver, Blacklist and Rogue customers

The assumption is that the business services and business applications and related capability versions are running before proceeding with the examples in the following sections.

4.3.1 Defining a Gold customer

ITSO Redbooks Travel wants to offer high performance and availability to the company's gold customers accessing the Itinerary Reservation Service. For this reason, the company wants to reroute its gold customers to a secondary endpoint when the primary endpoint exceeds the latency of one second. The company uses the context identifier *Gold* to identify the gold customers.

The following steps describe how to define a gold customer for this scenario:

1. Navigate to **Go To Spaces → IBM Redbooks Travel Service Registry Operations Space → Overview → Search wizard → Application Version → Reservation Client Application Version (1.0) → Service Registry Detail → Edit → Service Level Agreements → Add Service Level Agreement → Create**.
2. Create an SLA named d `SLA - Gold Consumers - Itinerary Reservation Service` with the `Gold` context identifier, as shown in Figure 4-14 on page 103.

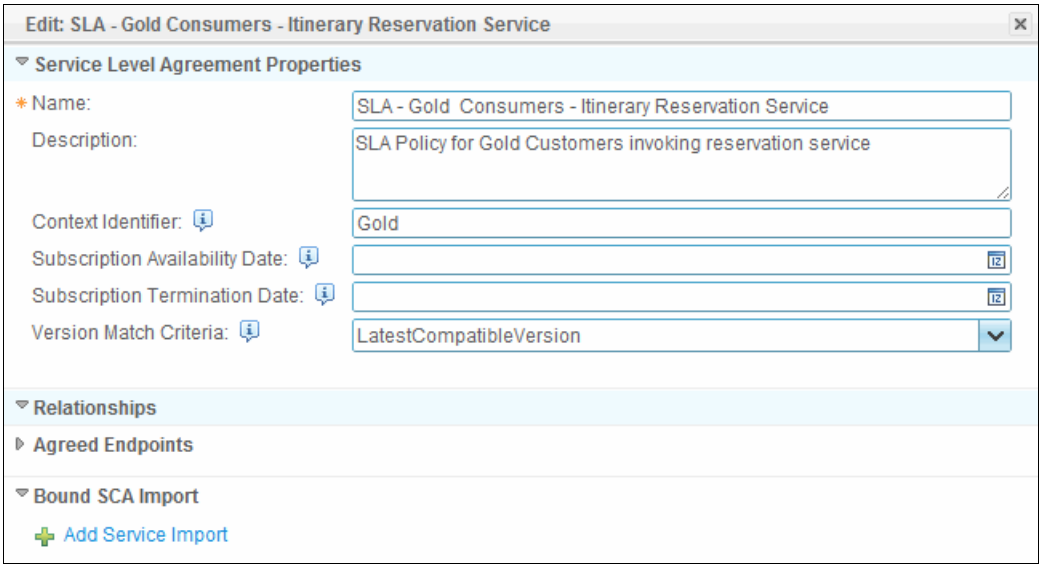


Figure 4-14 SLA for gold customers

Complete the following steps:

a. Navigate to **Go To Spaces** → **IBM Redbooks Travel Service Registry Operations Space** → **Overview** → **Service Registry Actions** → **Create a Mediation Policy**.

b. Create a mediation policy `Route Gold Customers` (Figure 4-15 on page 104).

c. Add the `Average Total Latency` attribute condition with the value of 1000 milliseconds and a per interval of 15 seconds.

d. Add the `Route Message` action with the secondary endpoint that points to the following URL:

```
http://services:9082/redbooksTravel/ItineraryReservationService
```

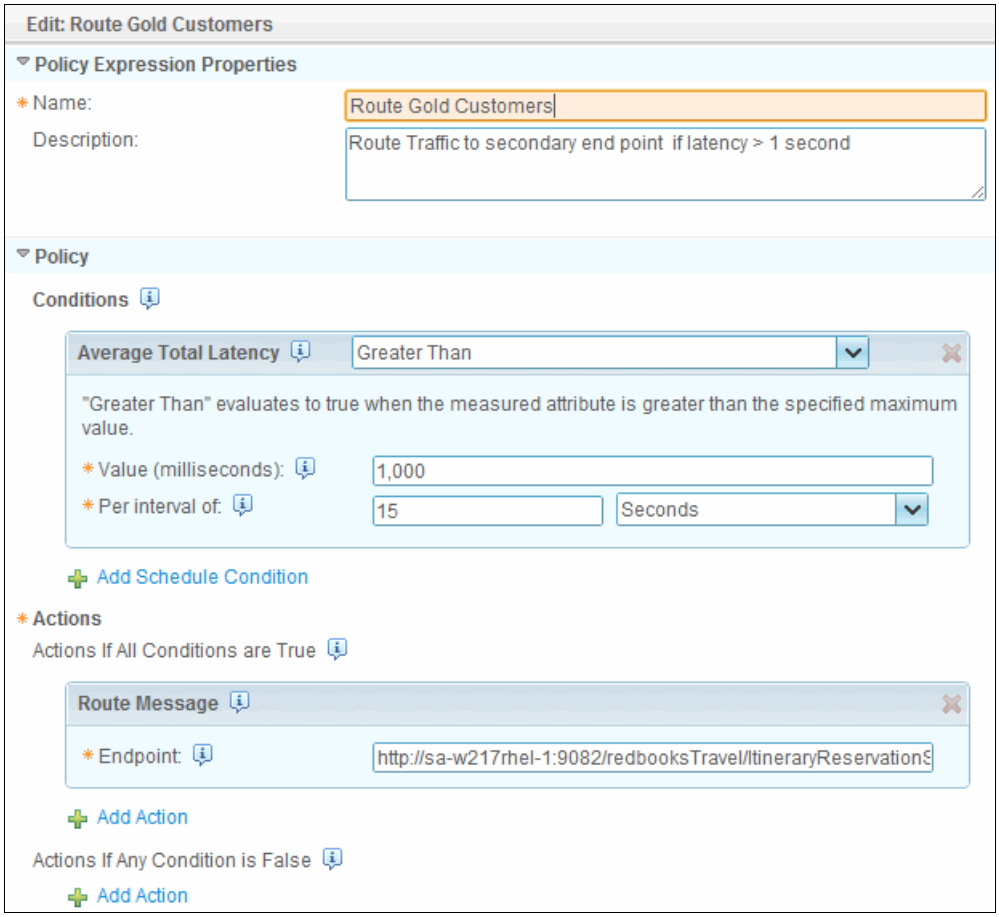


Figure 4-15 Mediation policy for gold customers

When the condition of *average total latency* is greater than 1000 milliseconds (1 second) during a 15-second interval, the action `Route Message` will redirect the requests to the secondary endpoint.

3. To attach the Route Gold Customers mediation policy to the SLA named `SLA - Gold Consumers - Itinerary Reservation Service`, select **Search wizard** → **Service Level Agreements** → **SLA - Gold Consumers - Itinerary Reservation Service** → **Edit** → **Attached Policies** → **Add Policy** → **Find** → **Route Gold Customers** → **Finish**.

Figure 4-16 shows the `SLA - Gold Consumers - Itinerary Reservation Service` SLA with the `Route Gold Customers` policy attachment.

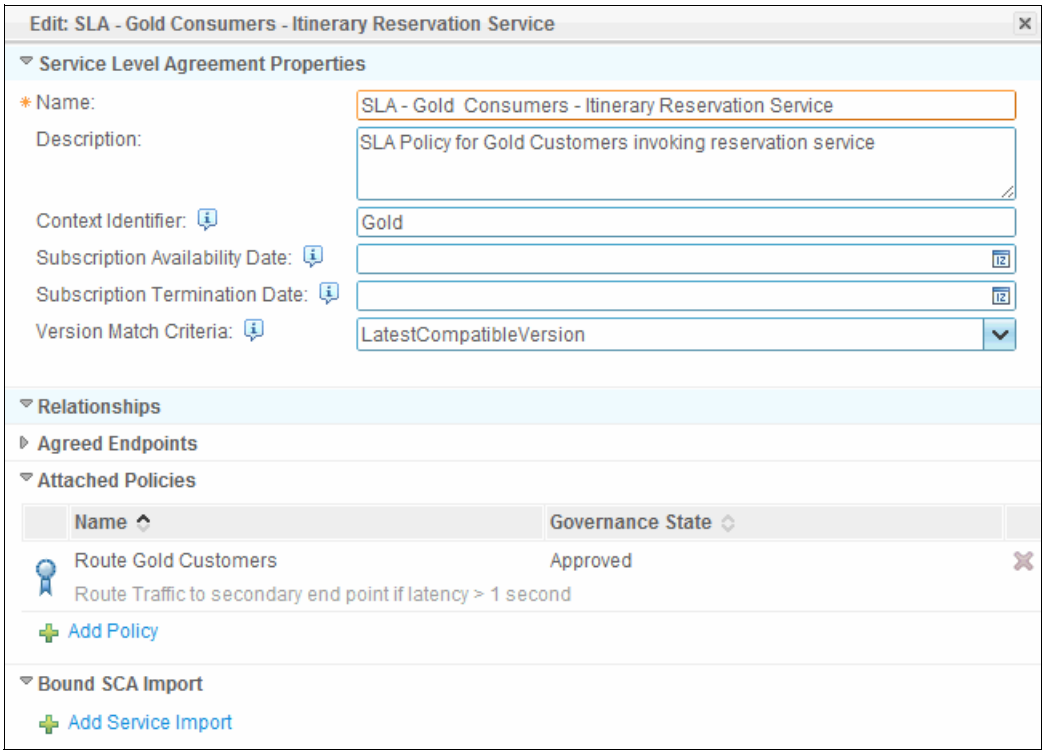


Figure 4-16 SLA with policy attached for Gold customers

4. To add `SLD - Itinerary Reservation Service - Prioritized Access` as an agreed endpoint to `SLA - Gold Consumers - Itinerary Reservation Service`, select **Search wizard → Service Level Agreements → SLA - Gold Consumers - Itinerary Reservation Service → Edit → Agreed Endpoints → Add Service Level Definition → Find → Select SLD - Itinerary Reservation Service → Finish**.

Figure 4-17 shows the `SLA - Gold Consumers - Itinerary Reservation Service` tied to `SLD - Itinerary Reservation Service - Prioritized Access`.

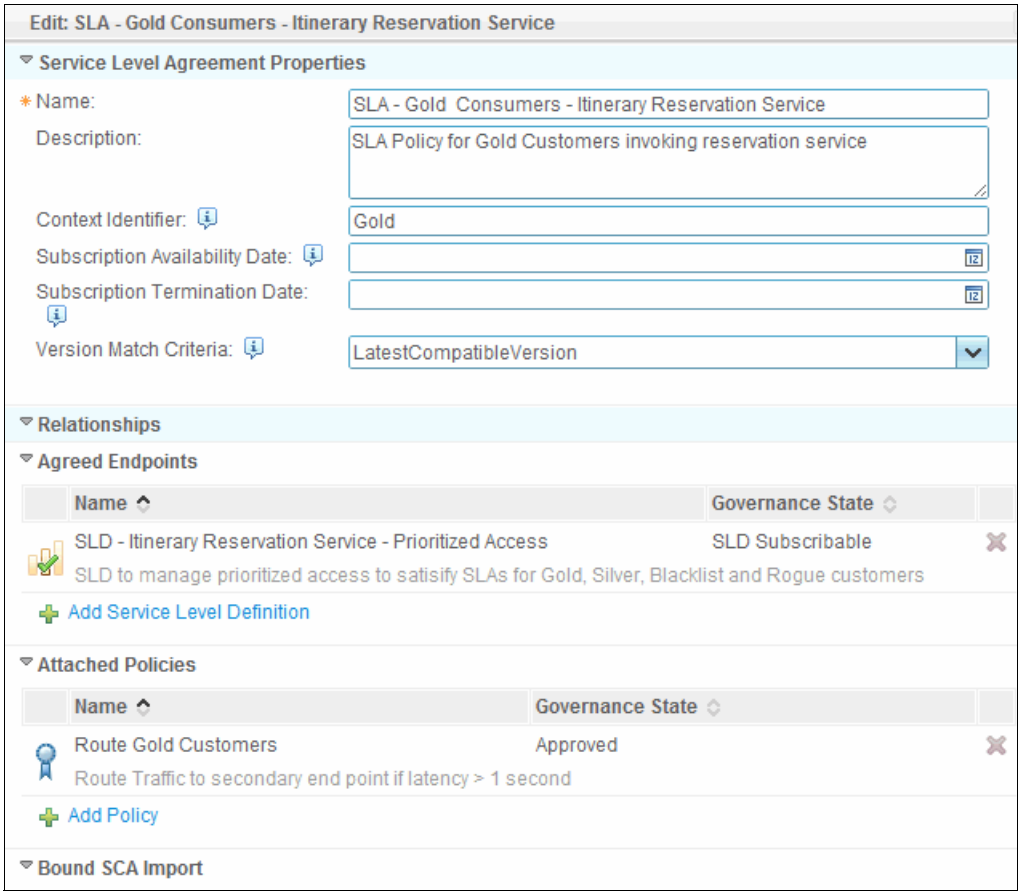


Figure 4-17 SLA tied to SLD for gold customers

5. Govern and promote the services to propagate the changes to WSRR and DataPower. See Appendix A, “Implementing a SOA Policy Solution flow of work” on page 415 for further details.

   Figure 4-18 shows the SLA for gold customers between Reservation Client and Itinerary Reservation Service.

Figure 4-18 SLA for Gold customers

By default, gold customers reaching Itinerary Reservation Service are routed to the following primary endpoint:

`http://services:9081/redbooksTravel/ItineraryReservationService`

When the SLA for Gold customers is enforced, the requests are routed to the secondary endpoint:

`http://services:9082/redbooksTravel/ItineraryReservationService`

4.3.2 Silver customer

ITSO Redbooks Travel offers high performance and availability to its gold customers that access Itinerary Reservation with an SLA of one-second latency. In addition, the company wants to establish an SLA with silver customers, offering them the transaction performance of no less than two seconds. For this reason, the company wants to reroute the silver customers to a secondary endpoint when the primary endpoint exceeds the latency of two seconds. ITSO Redbooks Travel uses *Silver* context identifier to identify the silver customers.

The following steps describe how to define a silver customer for this scenario.

1. Select **Go To Spaces** → **IBM Redbooks Travel Service Registry Operations Space** → **Overview** → **Search wizard** → **Application Version** → **Reservation Client Application Version (1.0)** → **Service Registry Detail** → **Edit** → **Service Level Agreements** → **Add Service Level Agreement** → **Create**.
2. Create the `SLA - Silver Consumers - Itinerary Reservation Service` SLA with a `Silver` context identifier, as shown in Figure 4-19.

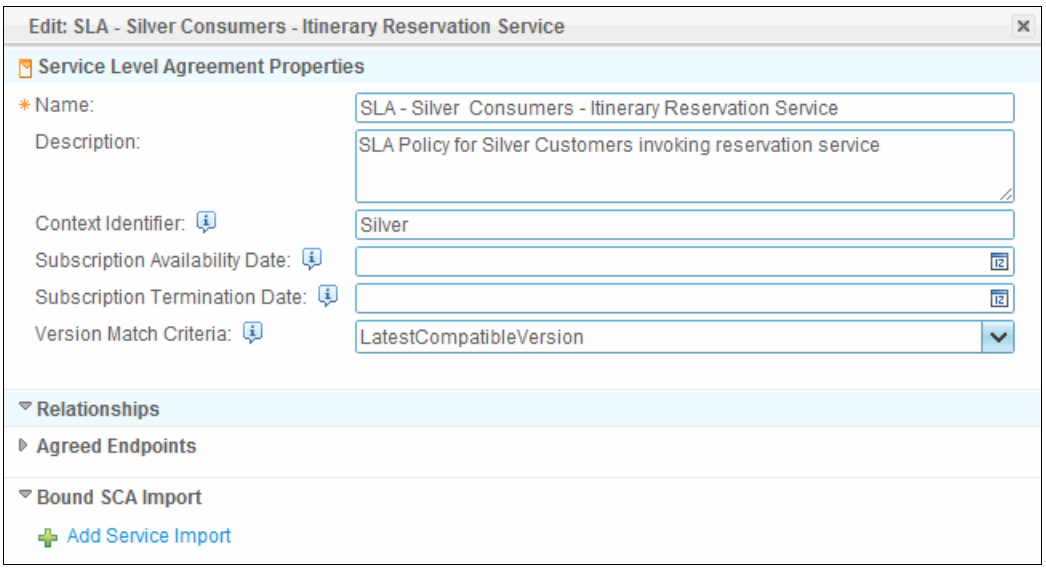


Figure 4-19 SLA for silver customers

   a. Select **Go To Spaces** → **IBM Redbooks Travel Service Registry Operations Space** → **Overview** → **Service Registry Actions** → **Create a Mediation Policy**.
   b. Create a `Route Silver Customers` mediation policy as shown in Figure 4-20 on page 108.
   c. Add the `Average Total Latency` attribute condition with a value of 2000 milliseconds (2 seconds) during a 15 second interval.

d. Add the `Route Message` action with the secondary endpoint that points to the following URL:

`http://services:9082/redbooksTravel/ItineraryReservationService`

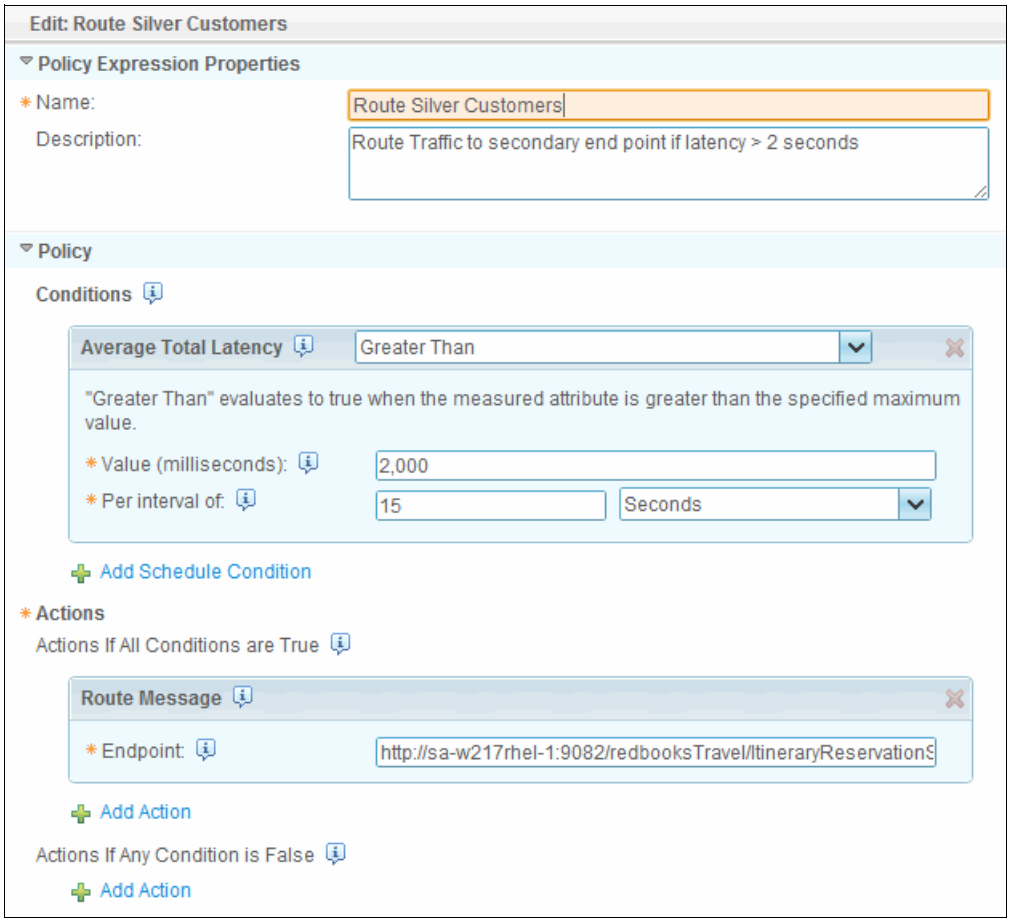


Figure 4-20 Mediation Policy for Silver customers

When the condition of *average total latency* is greater than 2000 milliseconds (2 seconds) during a 15-second interval, the `Route Message` action will redirect the requests to the secondary endpoint.

3. To attach the `Route Silver Customers` mediation policy to the `SLA - Silver Consumers - Itinerary Reservation Service` SLA, select **Search wizard** → **Service Level Agreements** → **SLA - Gold Consumers - Itinerary Reservation Service** → **Edit** → **Attached Policies** → **Add Policy** → **Find** → **Route Silver Customers** → **Finish**.

Figure 4-21 shows the `SLA - Silver Consumers - Itinerary Reservation Service` SLA with `Route Silver Customers` policy attachment.

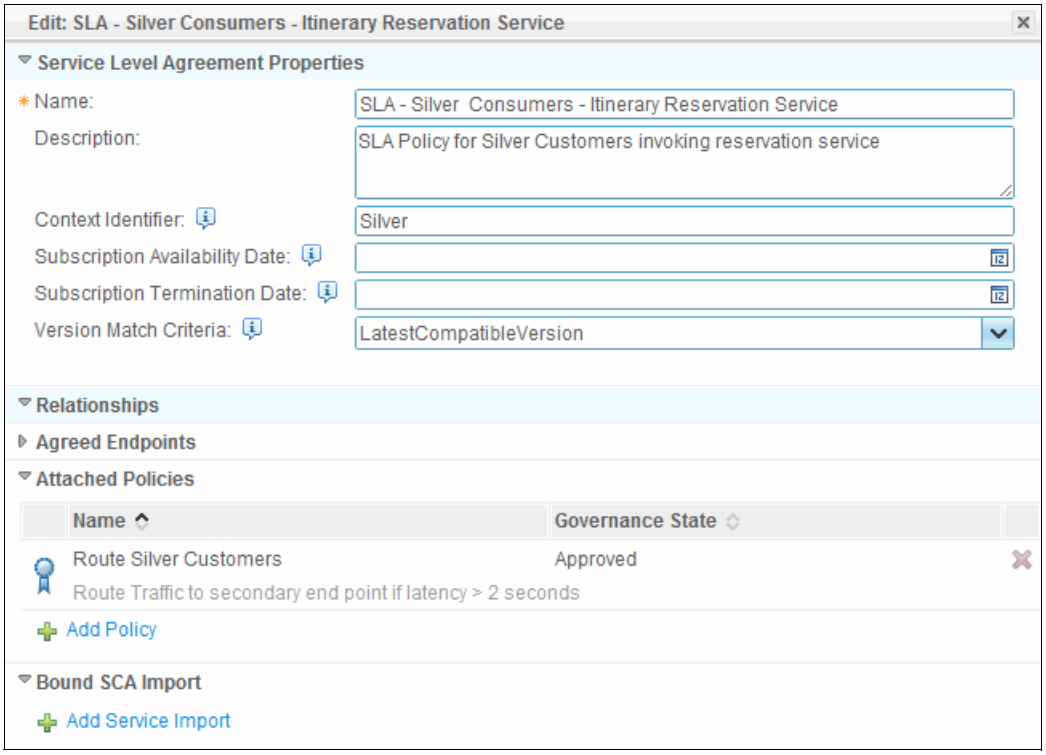


Figure 4-21 SLA with policy attached for Gold customers

4. To add `SLD - Itinerary Reservation Service - Prioritized Access` as an agreed endpoint to the `SLA - Silver Consumers - Itinerary Reservation Service`, select **Search wizard → Service Level Agreements → SLA - Silver Consumers - Itinerary Reservation Service → Edit → Agreed Endpoints → Add Service Level Definition → Find → Select SLD - Itinerary Reservation Service → Finish**.

Figure 4-22 shows `SLA - Silver Consumers - Itinerary Reservation Service` tied to `SLD - Itinerary Reservation Service - Prioritized Access`.

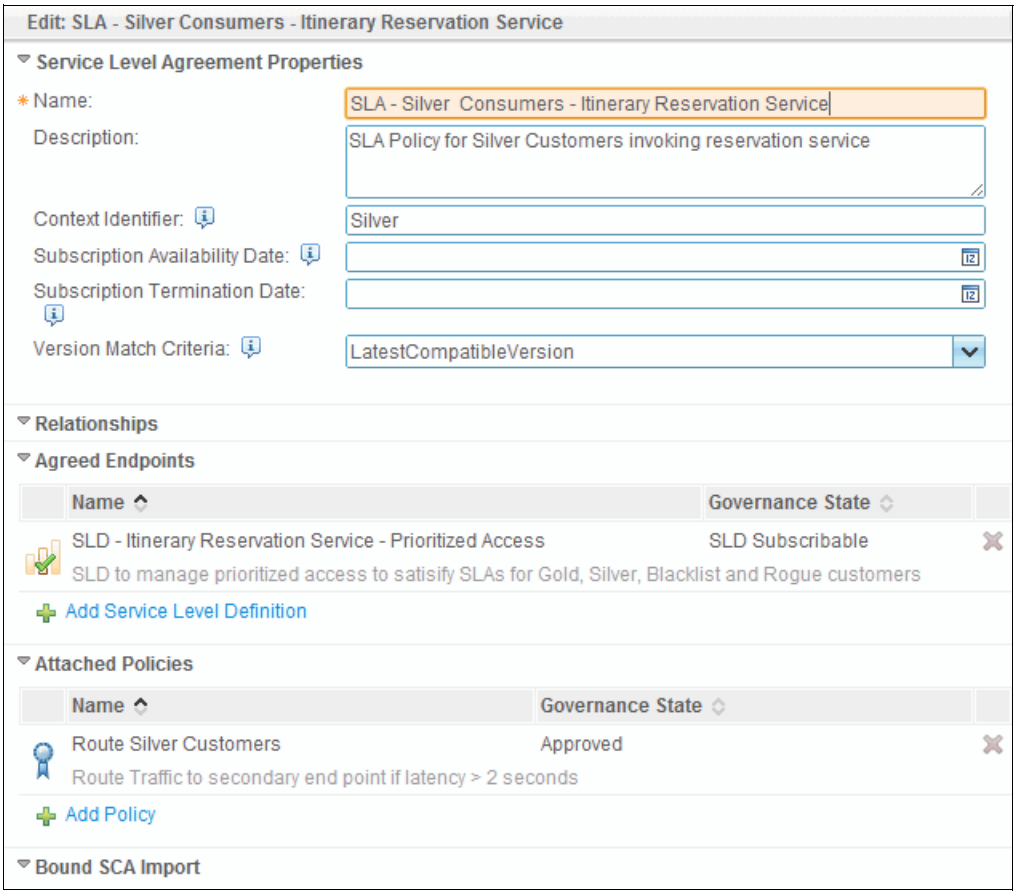


Figure 4-22 SLA tied to SLD for Silver customers

5. Govern and promote the services to propagate the changes to WSRR and DataPower. Figure 4-23 shows the SLA for silver customers between Reservation Client and Itinerary Reservation Service.

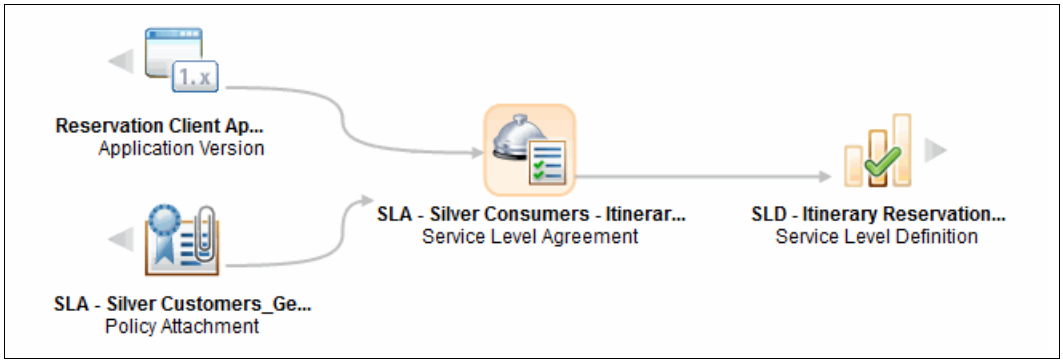


Figure 4-23 SLA for Silver customers

By default, silver customers reaching Itinerary Reservation Service are routed to the primary endpoint:

`http://services:9081/redbooksTravel/ItineraryReservationService`

When the SLA for silver customers is enforced, the requests are routed to the secondary endpoint:

`http://services:9082/redbooksTravel/ItineraryReservationService.`

4.3.3 Blacklist customer

ITSO Redbooks Travel company wanted to reject the blacklist customers that access Itinerary Reservation Service. The company used the *Blacklist* context identifier to identify the valid customers that are not valid.

The following steps describe how to define a blacklist customer for this scenario.

1. Select **Go To Spaces → IBM Redbooks Travel Service Registry Operations Space → Overview → Search wizard → Application Version → Reservation Client Application Version (1.0) → Service Registry Detail → Edit → Service Level Agreements → Add Service Level Agreement → Create.**
2. Create the `SLA - Blacklist Consumers - Itinerary Reservation Service` SLA with `Blacklist` context identifier, as shown in Figure 4-24.

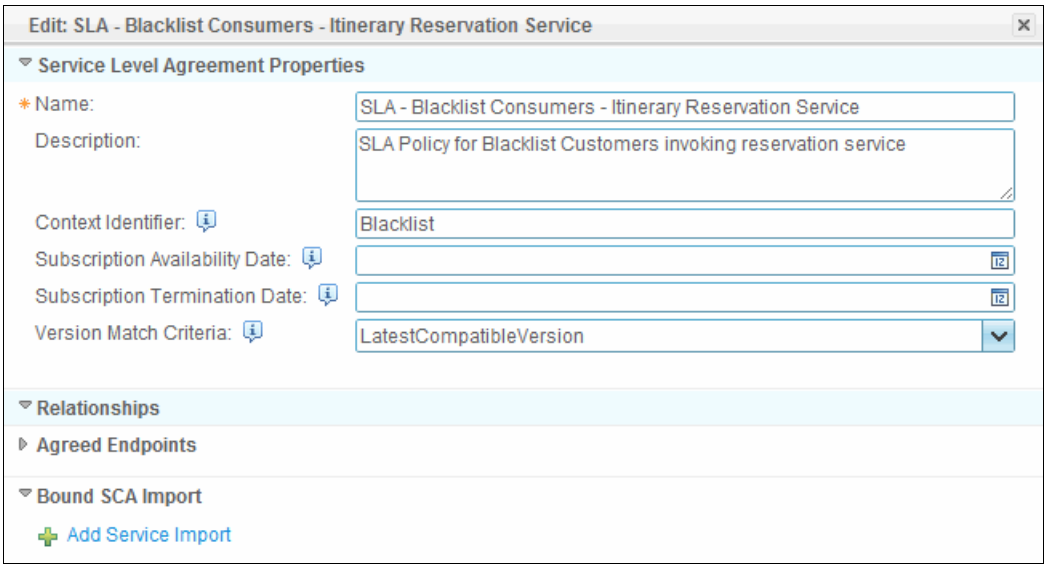


Figure 4-24 SLA for blacklist customers

   a. Select **Go To Spaces → IBM Redbooks Travel Service Registry Operations Space → Overview → Service Registry Actions → Create a Mediation Policy**.
   b. Create a `Reject Blacklist Customers` mediation policy, as shown in Figure 4-25 on page 112.
   c. Add the `Reject Message` action.

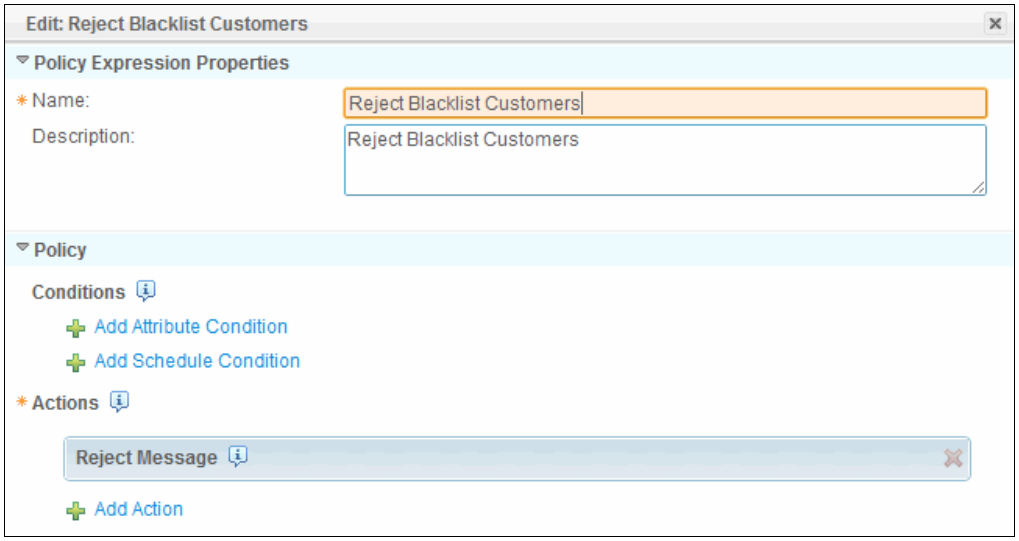


Figure 4-25 Mediation Policy for Blacklist customers

When the context of blacklist customers is met, the `Reject Message` action will reject the incoming messages. No specific condition is defined.

3. To attach the `Reject Blacklist Customers` to `SLA - Blacklist Consumers - Itinerary Reservation Service` mediation policy, select **Search wizard → Service Level Agreements → SLA - Blacklist Consumers - Itinerary Reservation Service → Edit → Attached Policies → Add Policy → Find → Reject Blacklist Customers → Finish**.

   Figure 4-26 shows `SLA - Blacklist Consumers - Itinerary Reservation Service` with the `Reject Blacklist Customers` policy attachment.

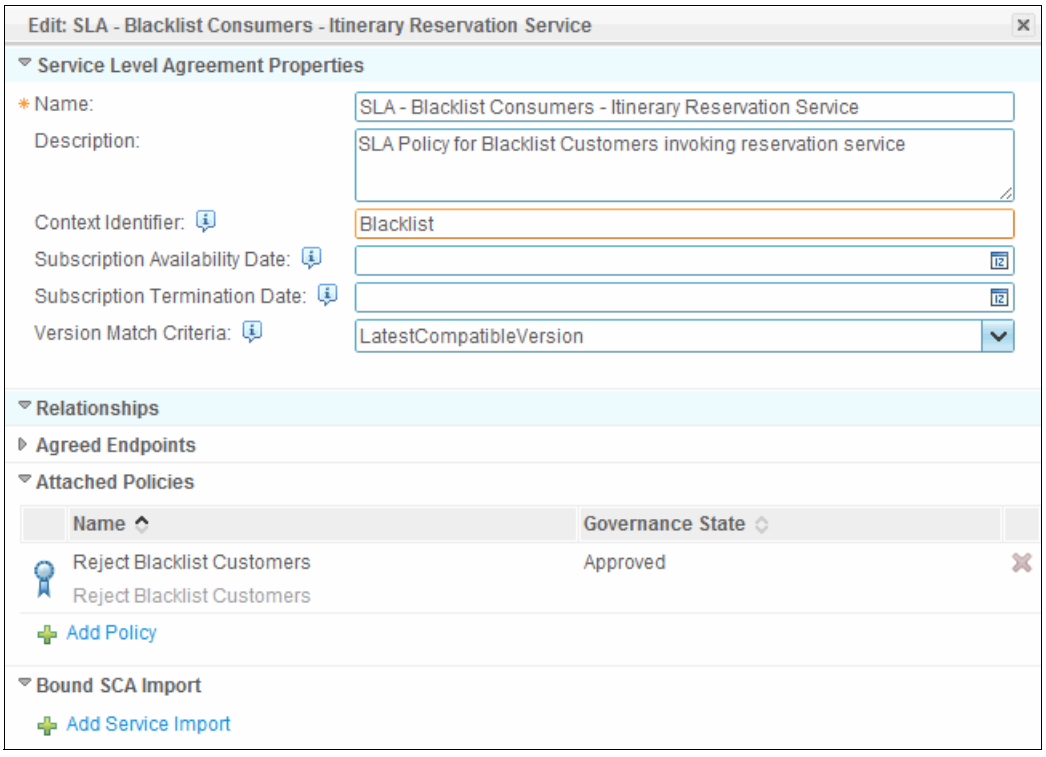


Figure 4-26 SLA with policy attached for blacklist customers

4. To add `SLD - Itinerary Reservation Service - Prioritized Access` as an agreed endpoint to `SLA - Blacklist Consumers - Itinerary Reservation Service`, select **Search wizard → Service Level Agreements → SLA - Blacklist Consumers - Itinerary Reservation Service → Edit → Agreed Endpoints → Add Service Level Definition → Find → Select SLD - Itinerary Reservation Service → Finish**.

   Figure 4-27 shows the `SLA - Blacklist Consumers - Itinerary Reservation Service` tied to `SLD - Itinerary Reservation Service - Prioritized Access`.

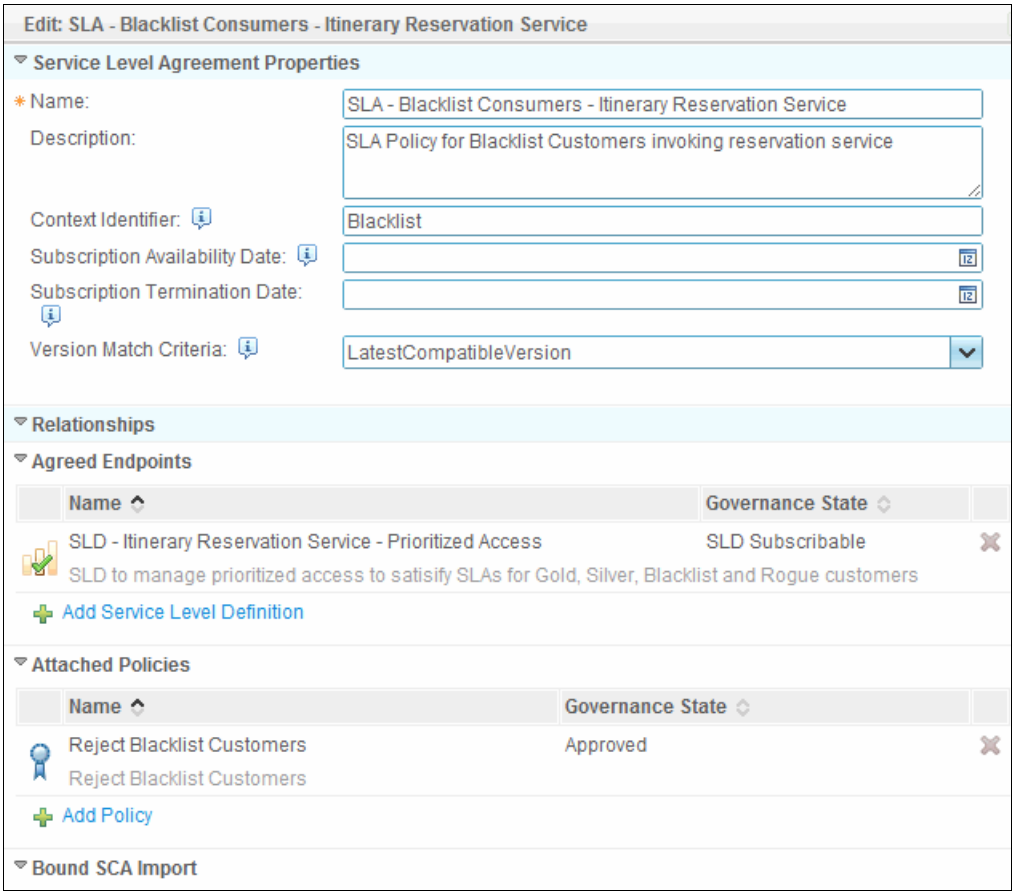


Figure 4-27 SLA tied to SLD for Blacklist customers

5. Govern and promote the services to propagate the changes to WSRR and DataPower.

   Figure 4-28 shows the SLA for blacklist customers between the Reservation Client and Itinerary Reservation Service.

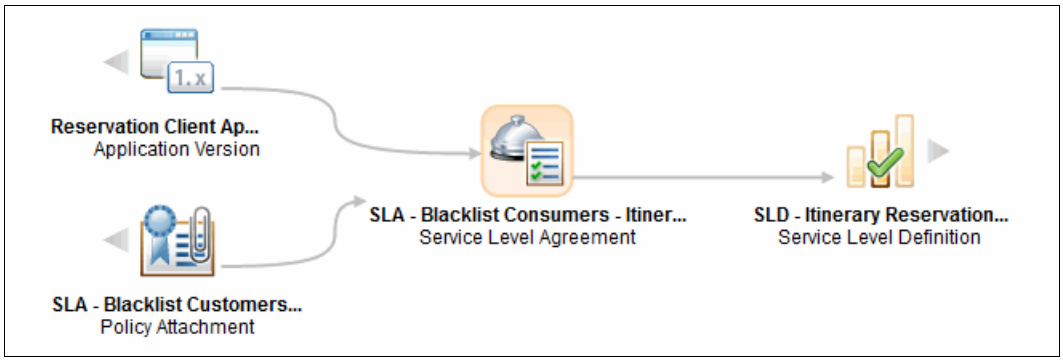


Figure 4-28 SLA for Blacklist customers

4.3.4 Rogue customer

ITSO Redbooks Travel wanted to reject rogue customers that access the Itinerary Reservation Service without an SLA. The company implemented an anonymous SLA to reject customers without an SLA.

An SLA is, by default, an agreement between a consumer and a provider. However, an *anonymous SLA* is not tied to any clients. An anonymous SLA is implemented on the Business Service Capability Version.

The following steps describe how to define a rogue customer for this scenario.

1. Navigate to **Go To Spaces → IBM Redbooks Travel Service Registry Operations Space → Overview → Search wizard → Service Level Definition → SLA - Itinerary Reservation Service - Prioritized Access → Service Registry Detail → Edit → Anonymous SLA → Add Service Level Agreement → Create.**
2. Create an `Anonymous SLA - Itinerary Reservation Service` SLA, as shown in Figure 4-29.

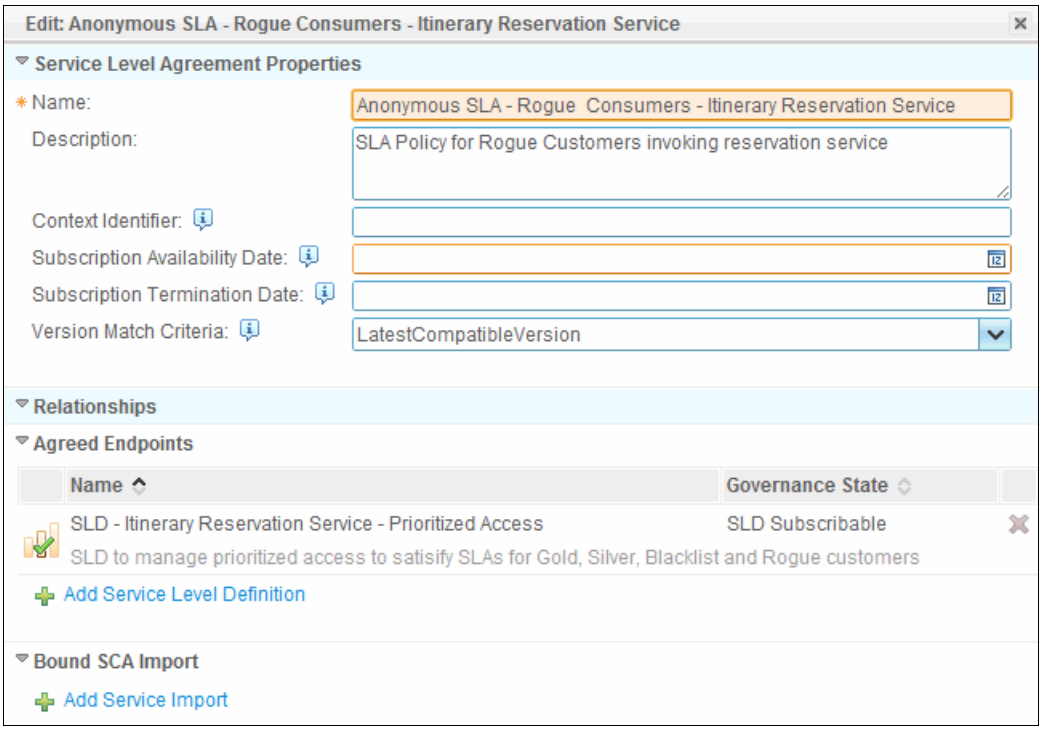


Figure 4-29 Anonymous SLA for Rogue customers

Complete the following steps:

a. Navigate to **Go To Spaces → IBM Redbooks Travel Service Registry Operations Space → Overview → Service Registry Actions → Create a Mediation Policy**.
b. Create a `Reject Rogue Customers` mediation policy, as shown in Figure 4-30 on page 115.
c. Add the `Reject Message` action.

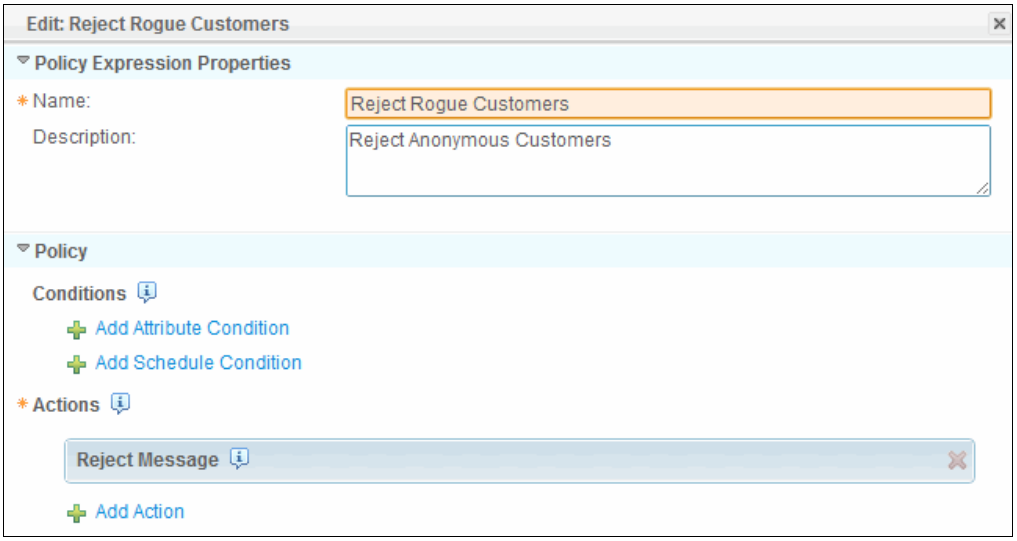


Figure 4-30 Mediation Policy for rogue customers

When incoming requests have no SLAs to Itinerary Reservation Service, the `Reject Message` action will reject the incoming messages. No specific condition is defined.

3. To attach the mediation policy `Reject Rogue Customers` to `SLA - Rogue Consumers - Itinerary Reservation Service`, select **Search wizard → Service Level Agreements → SLA - Rogue Consumers - Itinerary Reservation Service → Edit → Attached Policies → Add Policy → Find → Select ThrottleRogue → Finish**.

   Figure 4-31 shows the `SLA - Rogue Consumers - Itinerary Reservation Service` with the `Reject Rogue Customers` policy attachment.

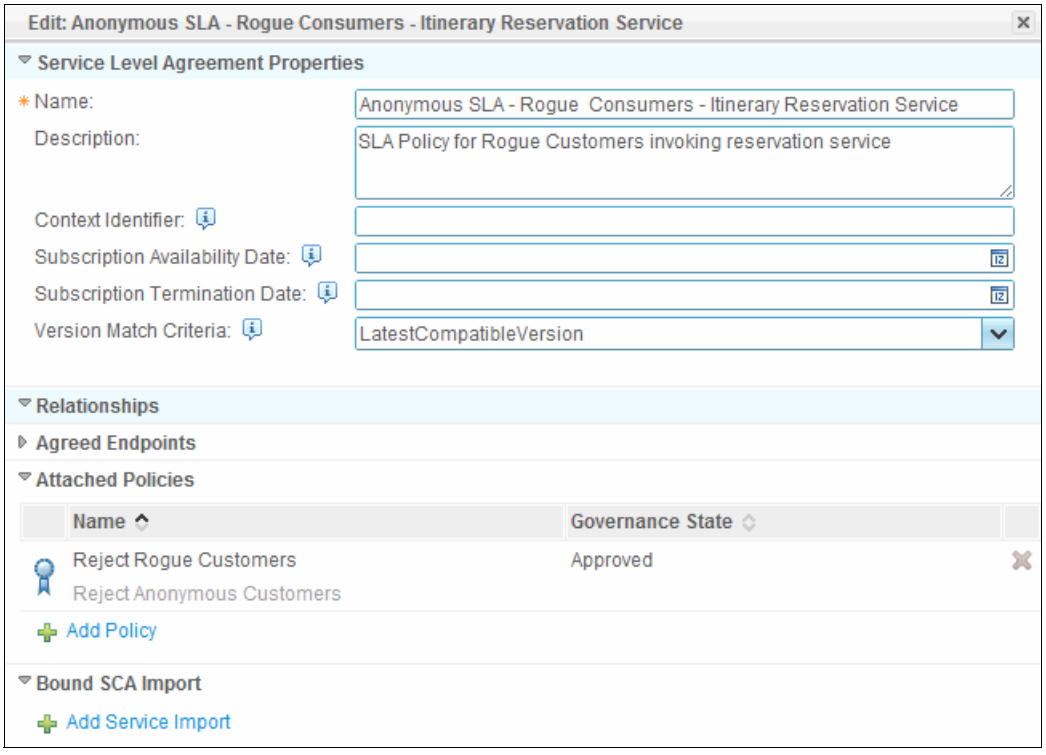


Figure 4-31 Anonymous SLA with policy attached for Rogue customers

4. To add `SLD - Itinerary Reservation Service - Prioritized Access` as an agreed endpoint to the `SLA - Rogue Consumers - Itinerary Reservation Service`, select **Search wizard → Service Level Agreements → SLA - Rogue Consumers - Itinerary Reservation Service → Edit → Agreed Endpoints → Add Service Level Definition → Find → Select SLD - Itinerary Reservation Service → Finish**.

   Figure 4-32 shows the `SLA - Rogue Consumers - Itinerary Reservation Service` tied to `SLD - Itinerary Reservation Service - Prioritized Access`.

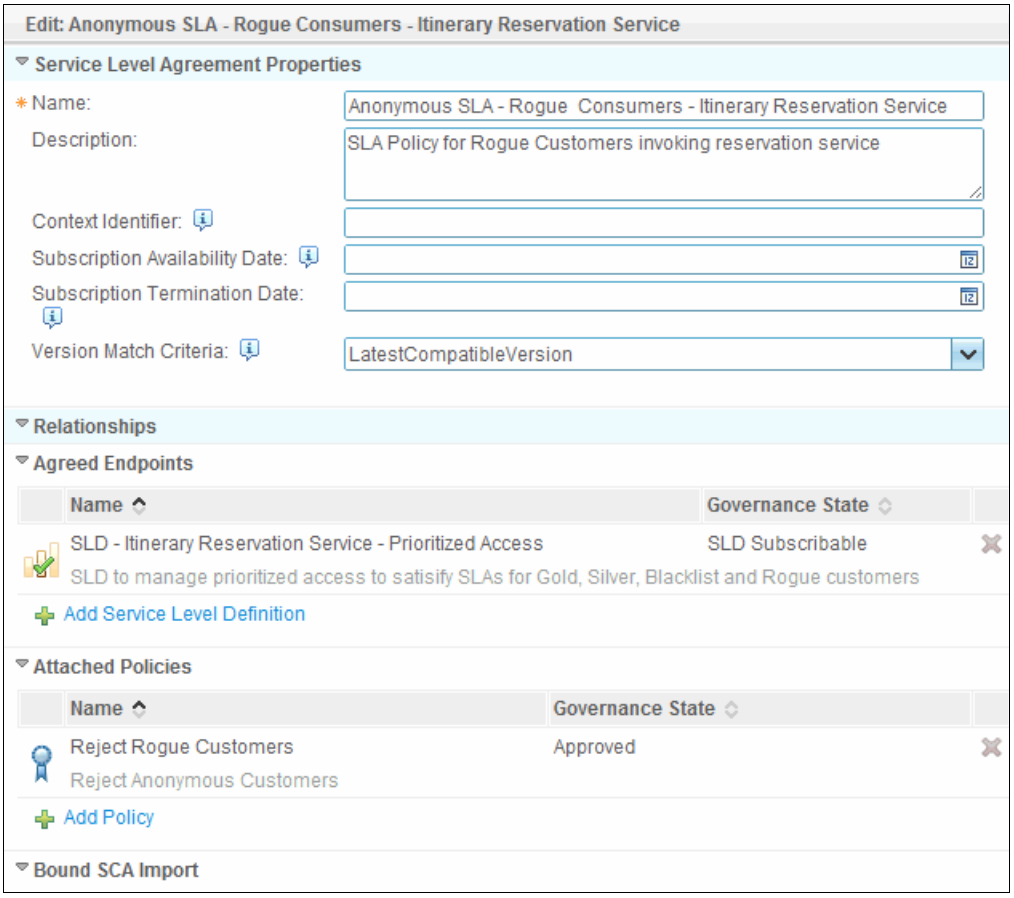


Figure 4-32 Anonymous SLA tied to SLD for Rogue customers

5. Govern and promote the service to propagate the changes to WSRR and DataPower.

Figure 4-33 shows the Anonymous SLA for rogue customers for Itinerary Reservation Service.

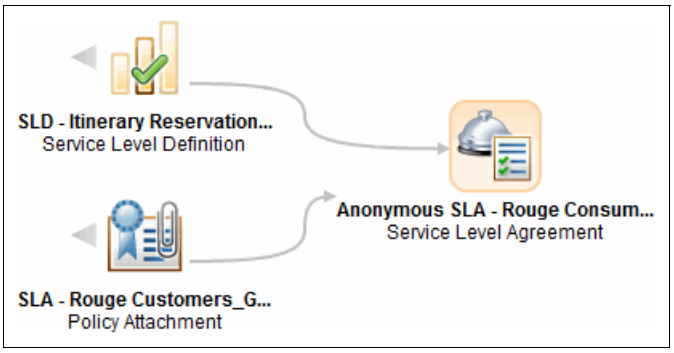


Figure 4-33 Anonymous SLA for rogue customers

5

Versioning with custom policy

The mediation policy capabilities that can be specified directly in the policy administration point (PAP) that is implemented in WSRR, is described in the previous two chapters. It is also possible to create policies, based on the full suite of policy capabilities that are available in the policy enforcement point (PEP) as implemented in WebSphere DataPower. This chapter and the next chapter have examples of how to create the policies. The example in this chapter focuses on applying policy to automatically cause consumers to use the latest version of a provider policy.

Versioning is an important topic for IT groups of many companies. In service-oriented architecture (SOA), managing the usage of various versions of a service when old versions must be retired can be complex to do. This chapter describes how the use of DataPower SOA appliances and WebSphere Service Registry and Repository (WSRR) provides a technique to manage old versions of a service.

This chapter presents a practical example of versioning based on the enforcement of a custom policy by using the Pricing service of the Fictional IBM ITSO Redbooks Travel Company.

This chapter contains the following topics:

- 5.1, “General description of the versioning use case” on page 120
- 5.2, “Implementing the versioning pattern with DataPower and WSRR” on page 122
- 5.3, “Custom versioning policy enforcement details” on page 125
- 5.4, “Pricing service versions” on page 128
- 5.5, “Creating custom policy domain and assertions for versioning” on page 134
- 5.6, “Creating custom policy XSL style sheet for the custom versioning policy” on page 139
- 5.7, “Attaching the custom versioning policy to a specific service” on page 159

© Copyright IBM Corp. 2013. All rights reserved.

5.1 General description of the versioning use case

The following list describes the assumptions that were made for the versioning use case:

- There are multiple consumers with two separate versions of a service.
- It is possible to transform the message format from one version to another.
- The versions of the service are governed in WSRR; services are exposed through a Web Service Proxy that is configured on a DataPower SOA device. Services that are exposed on the DataPower appliance are virtual services.

This initial state is shown in Figure 5-1.

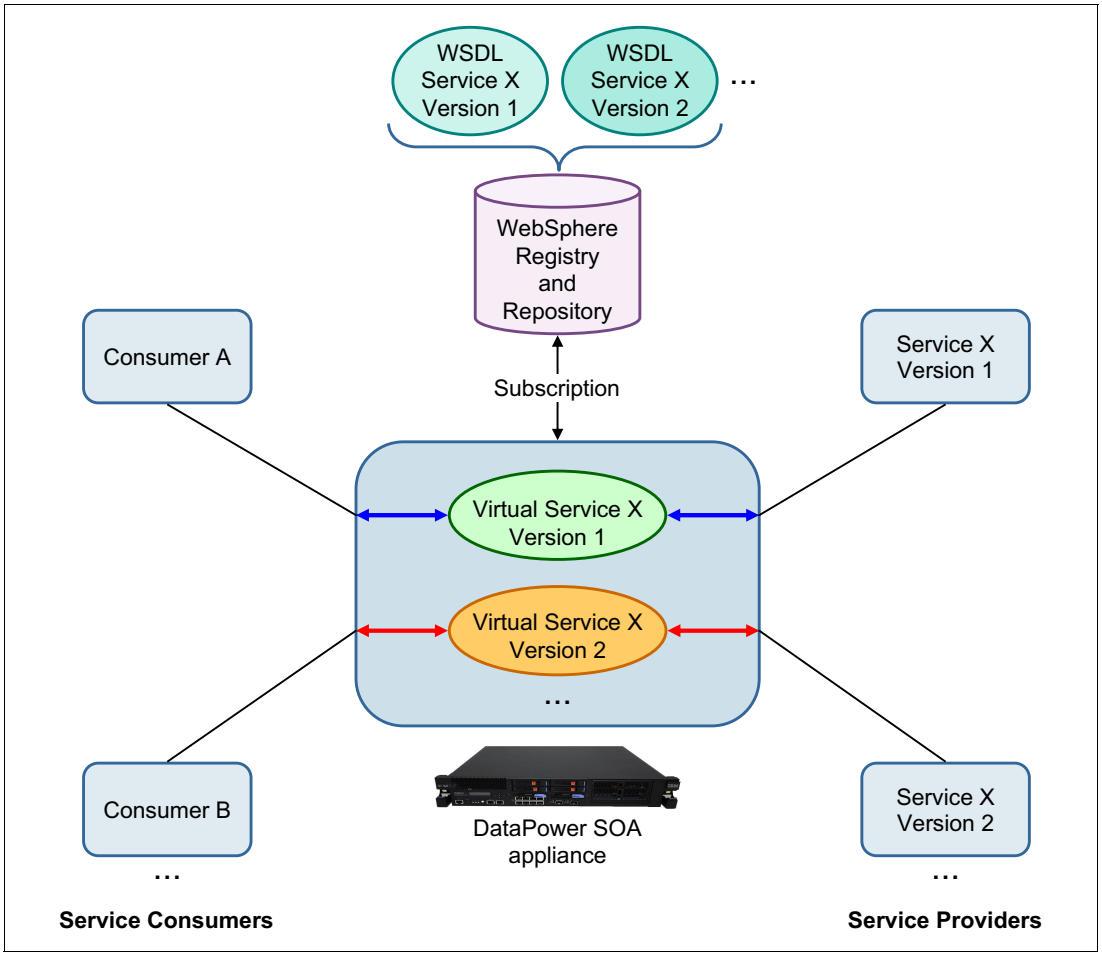


Figure 5-1 Presentation of the versioning use case

Services are exposed on the DataPower appliance as virtual services. Real implementations of services are located on back-end servers (service providers).

The versions of the back-end services are as follows:

- Version 1 for Service X
- Version 2 for Service X

Service X (version 1) was deprecated and is almost withdrawn from service. Unfortunately, not all consumers are able to migrate to version 2 of Service X. The versioning pattern that is described in this chapter can be used to resolve this specific need, as shown in Figure 5-2.

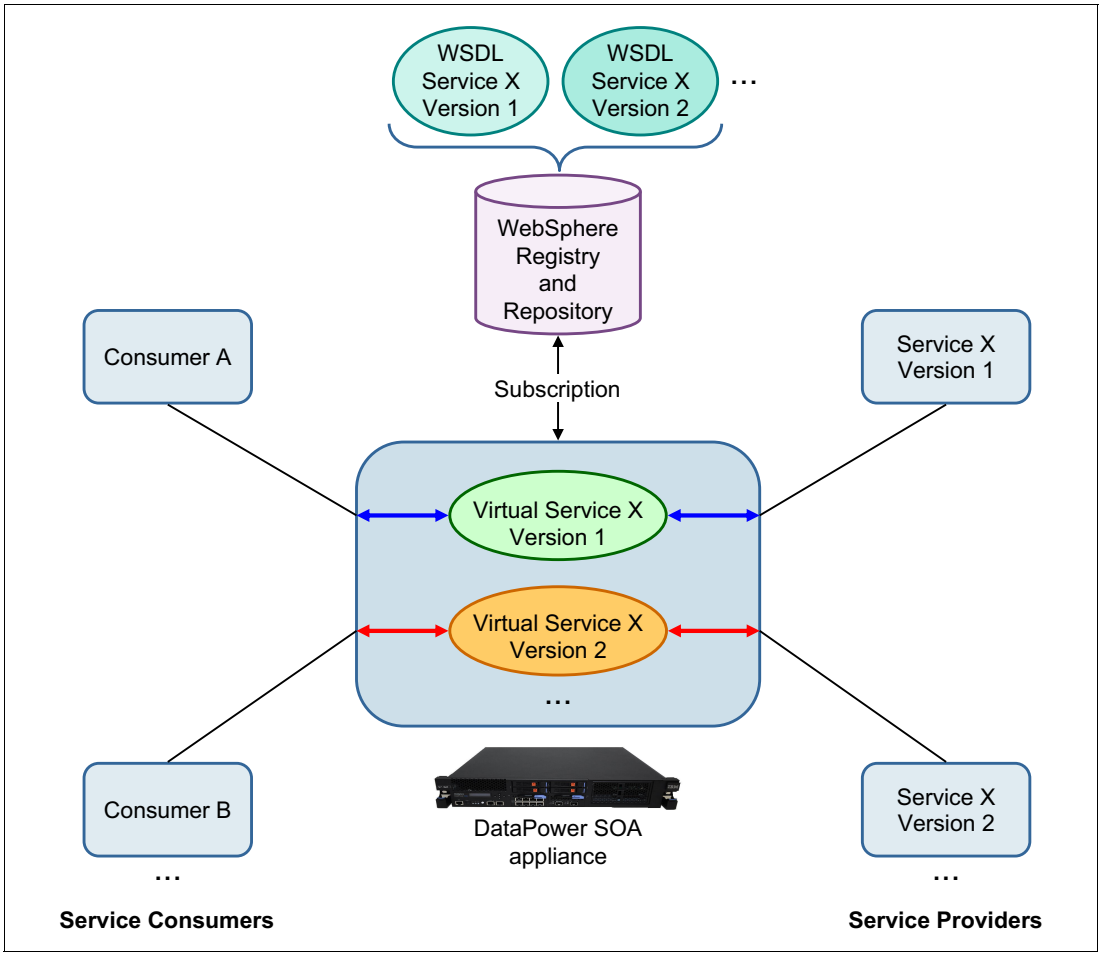


Figure 5-2 Versioning pattern

The back-end version 1 is no longer accessed, but it is still administered in WSRR and exposed as a virtual service in DataPower.

Regarding the versioning pattern, the DataPower appliance is responsible for enforcing the versioning pattern while executing the following tasks for messages for Service X (version 1):

- Exposing services retrieved from WSRR as virtual services (façade).
- Taking request messages for Service X (version 1) and transforming them into the format that is required for Service X (version 2).
- Routing the request on the back-end to Service X Version 2.
- Transforming responses messages from the back-end Service X Version 2 to the format expected by those consumers sending request messages to Service X (version 1).

5.2 Implementing the versioning pattern with DataPower and WSRR

To implement the versioning pattern, you must create a custom policy in DataPower that provides the required transformations, validations, and routing definitions. DataPower uses this policy to create configuration artifacts, that are required to enforce the versioning policy.

The example in this section is based on two services consumers, accessing two separate versions of the same service. The following assumptions are made to illustrate the detailed versioning example, from a WSRR perspective:

- Consumer A and Consumer C consume Service X Version 1.
- Consumer B consumes Service X Version 2.
- Each version of Service X has its own service level definition (SLD): SLD1 and SLD2.
- Each consumer is bound to SLD1 through service level agreement 1 (SLA1) or SLD2 through SLA2.

Figure 5-3 shows the components that are related to two versions of the same business service, from a WSRR perspective.

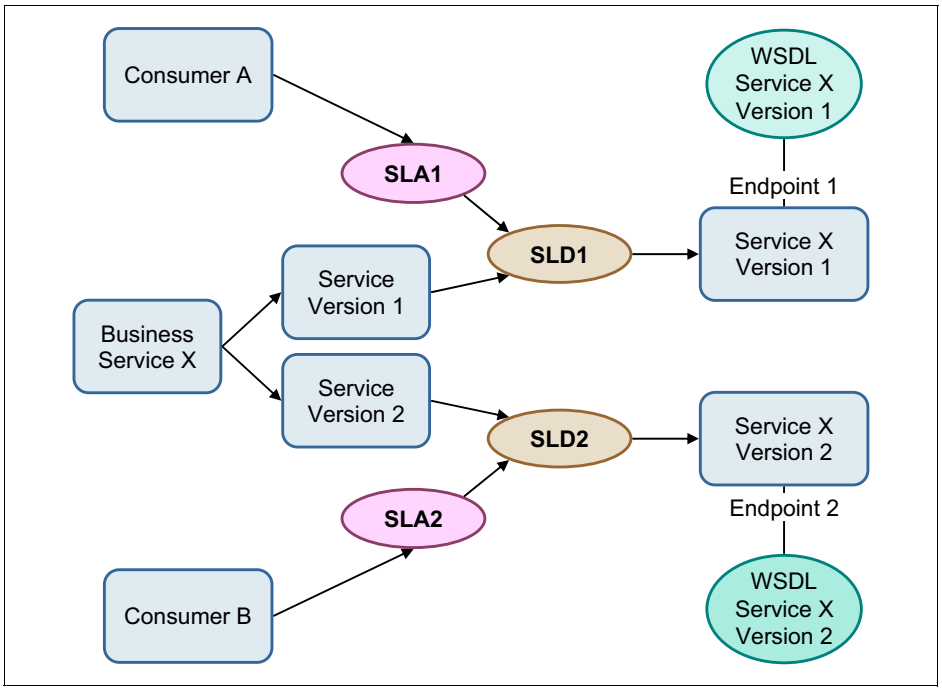


Figure 5-3 Business service components in WSRR

From a DataPower perspective, these components can be retrieved from WSRR by using a subscription mechanism. Virtual services are created on the DataPower device, as shown in Figure 5-4.

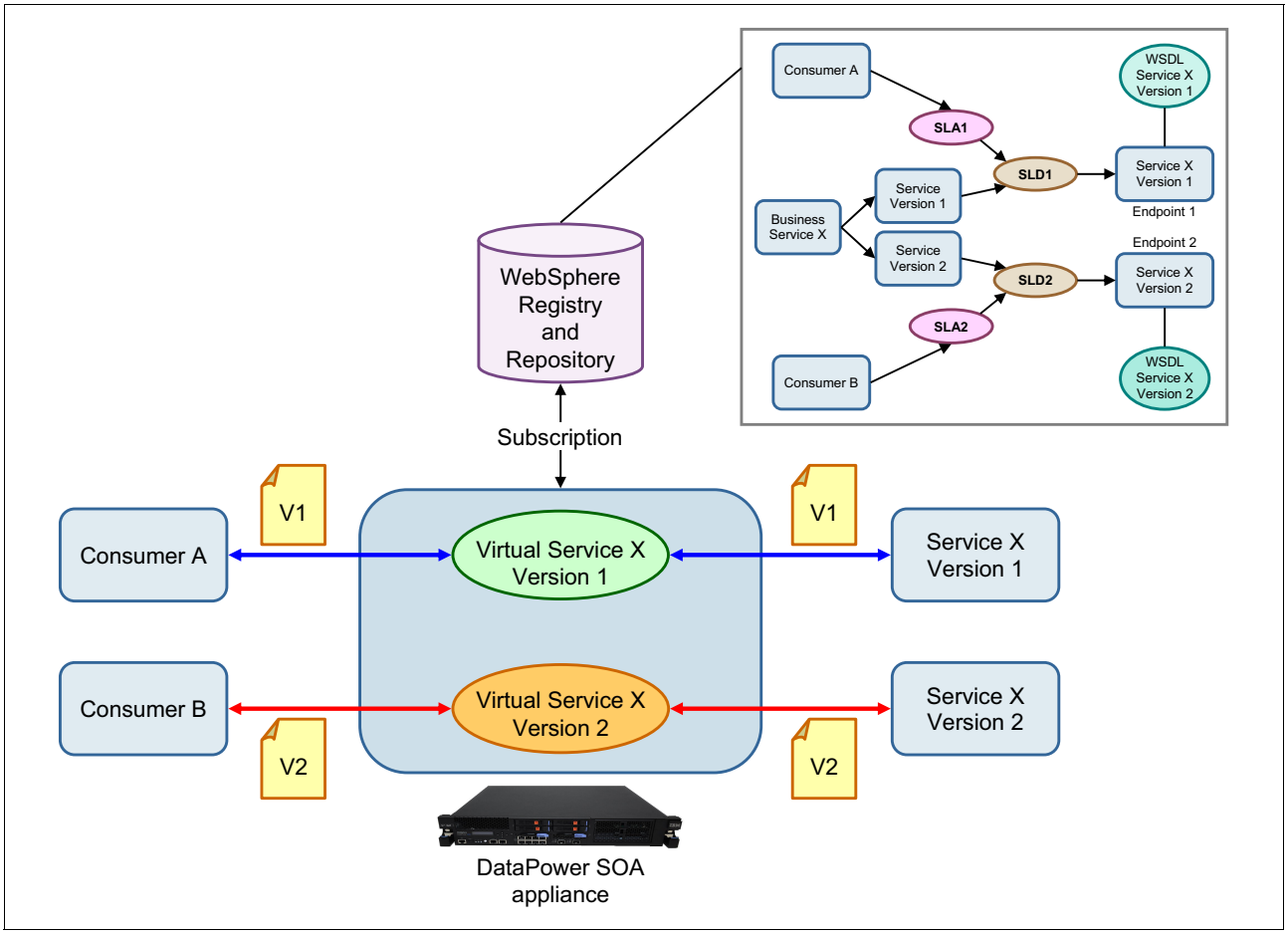


Figure 5-4 Virtual services in DataPower

As shown in Figure 5-4, Consumer A accesses Service 1 and Consumer B accesses Service 2. If a service is modified in WSRR, DataPower is notified of this modification. DataPower is able to resynchronize its subscription to benefit from the latest information provided in the registry.

To support this use case for versioning, it is necessary to create a custom policy to manage it. After this policy is created, administering the policy in WSRR, as a common WS-Policy, is possible.

Figure 5-5 shows the main WSRR components that are used for versioning.

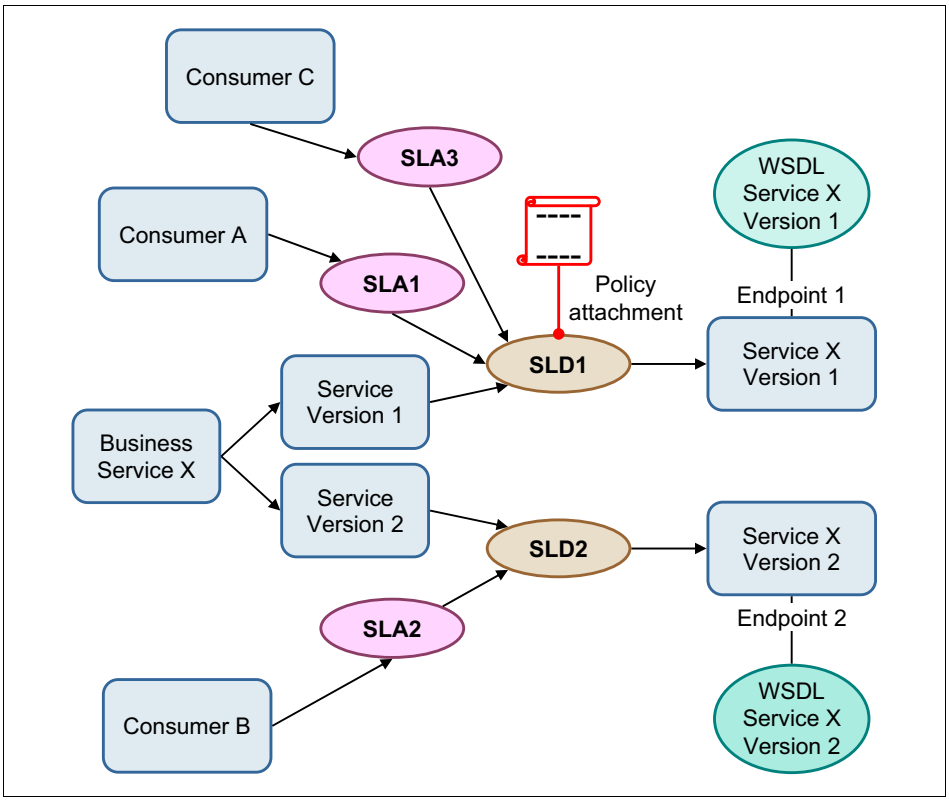


Figure 5-5 Custom versioning policy attachment in WSRR

A custom versioning policy that will be created in DataPower will be exported to WSRR and then be attached at the SLD level.

In this example, all Consumer A and Consumer C consumers that are bound to the SLD for Service X Version 1 benefit from the versioning policy enforcement, as shown in Figure 5-5.

The versioning policy can be enforced as shown in Figure 5-6.

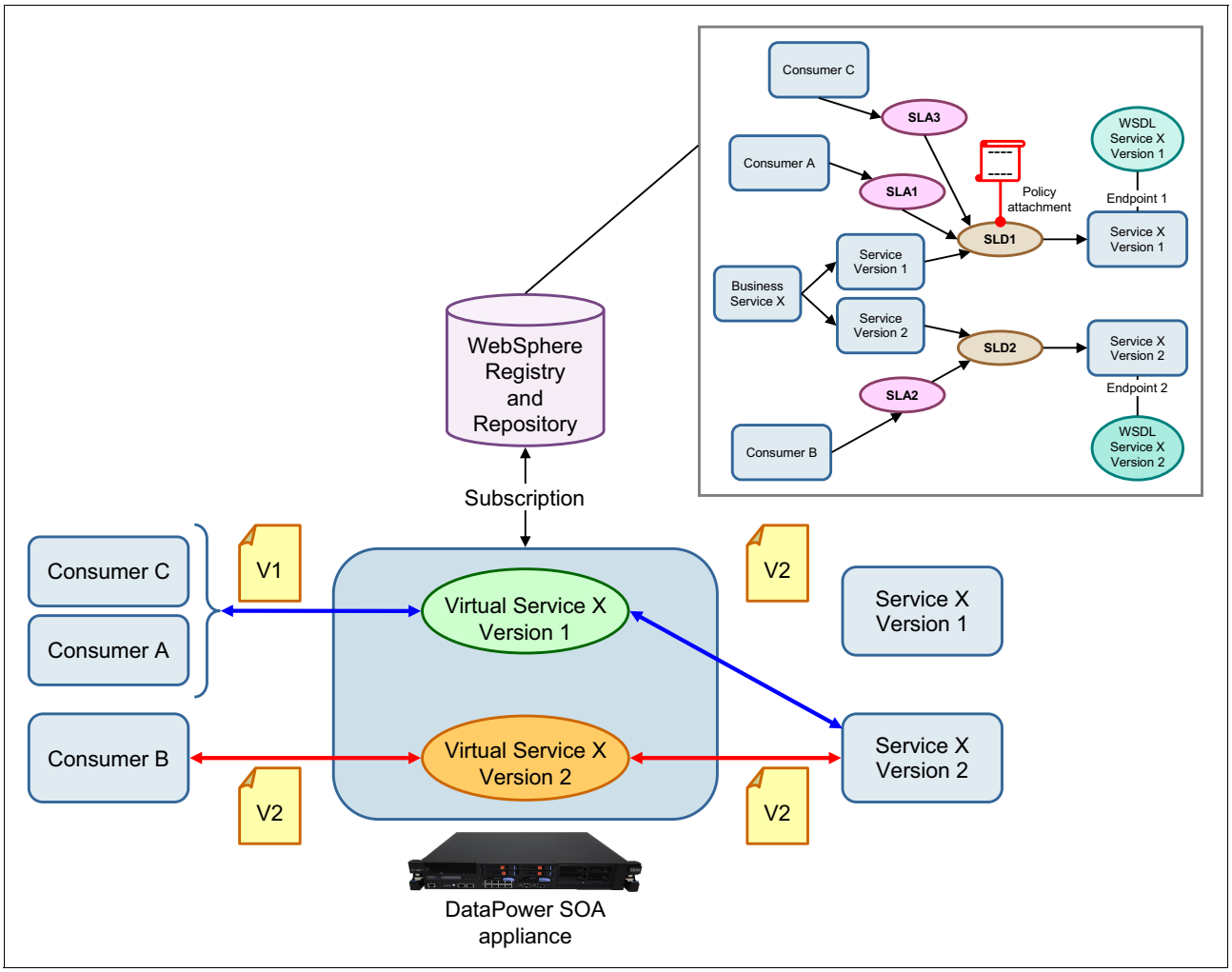


Figure 5-6 Versioning policy enforced on DataPower

Alternatively, if the versioning policy had been attached at the SLA level SLA1, only traffic of Consumer A for Service X Version 1 would be enforced on the DataPower device. Therefore, under this alternative, only Consumer A would be subject to the versioning policy.

5.3 Custom versioning policy enforcement details

The custom versioning policy contains the following information, as required by the versioning pattern:

- The transformation that is used to do data mediation during request processing to transform message version 1 to message version 2. In this example, the transformation is named `Transform_V1toV2`.
- The WSDL file is used to validate output and response:
  - The output of the transformation `Transform_V1toV2`, (the outgoing request)
  - The response of the back-end service

  In this example, the WSDL file is named `ServerWSDL`.

- The route information (route to service version 2).
- The transformation that is used to do data mediation during response processing to transform message version 2 back to message version 1. In this example, the name of the transformation is `Transform_V2toV1`.
- Optional: This step is used only for demonstrating how to validate the following items by using the WSDL file:
  - The result of the transformation `Transform_V2toV1`, (the outgoing response)
  - The request of the service consumer

  In this example, `ClientWSDL` is the name of the WSDL file.

Figure 5-7 presents the actions that are done on the request, and the processing rules of a DataPower appliance (policy enforcement point) during custom versioning policy enforcement.

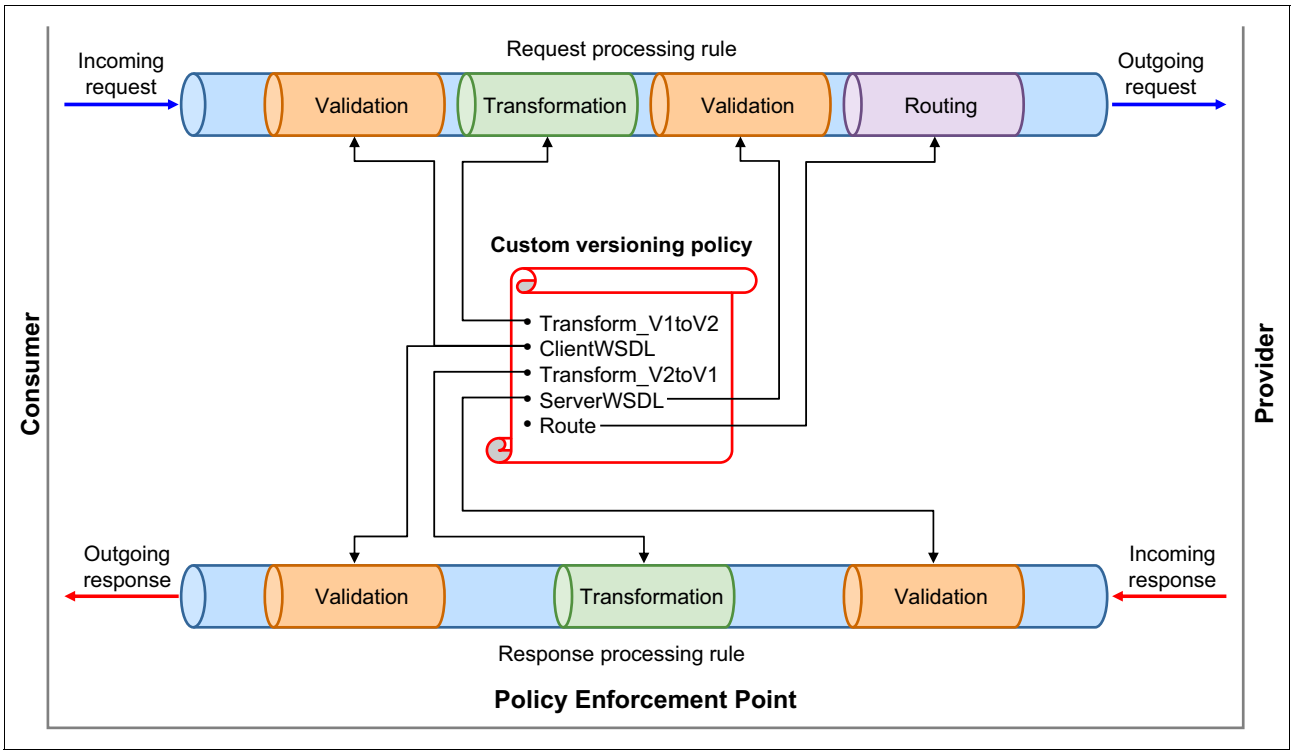


Figure 5-7 Custom versioning policy enforcement details

To minimize the amount of information that is provided in the custom versioning policy, it is possible to use only one transformation (in this example, the transformation is named `Transform`) rather than two. We set up `Transform` to handle the following transformations:

- Version 1 of a message into version 2
- Version 2 of a message into version 1

Important: The versioning use case can be supported in DataPower and WSRR as described in this chapter, only if the SOAP message can be transformed from one version to another. In other words, if a radical change occurred between Service X Version 1 and Version 2 so that the message formats are not backward compatible, this use case will not work for you.

Processing rules and actions (configuration artifacts) that are presented in Figure 5-7 on page 126 are created by the custom versioning policy style sheet and differ from the request and response processing rules that are automatically generated during the creation of a Web Service Proxy. These rules are listed in the **Policy** tab of a Web Service Proxy, as shown in Figure 5-8.

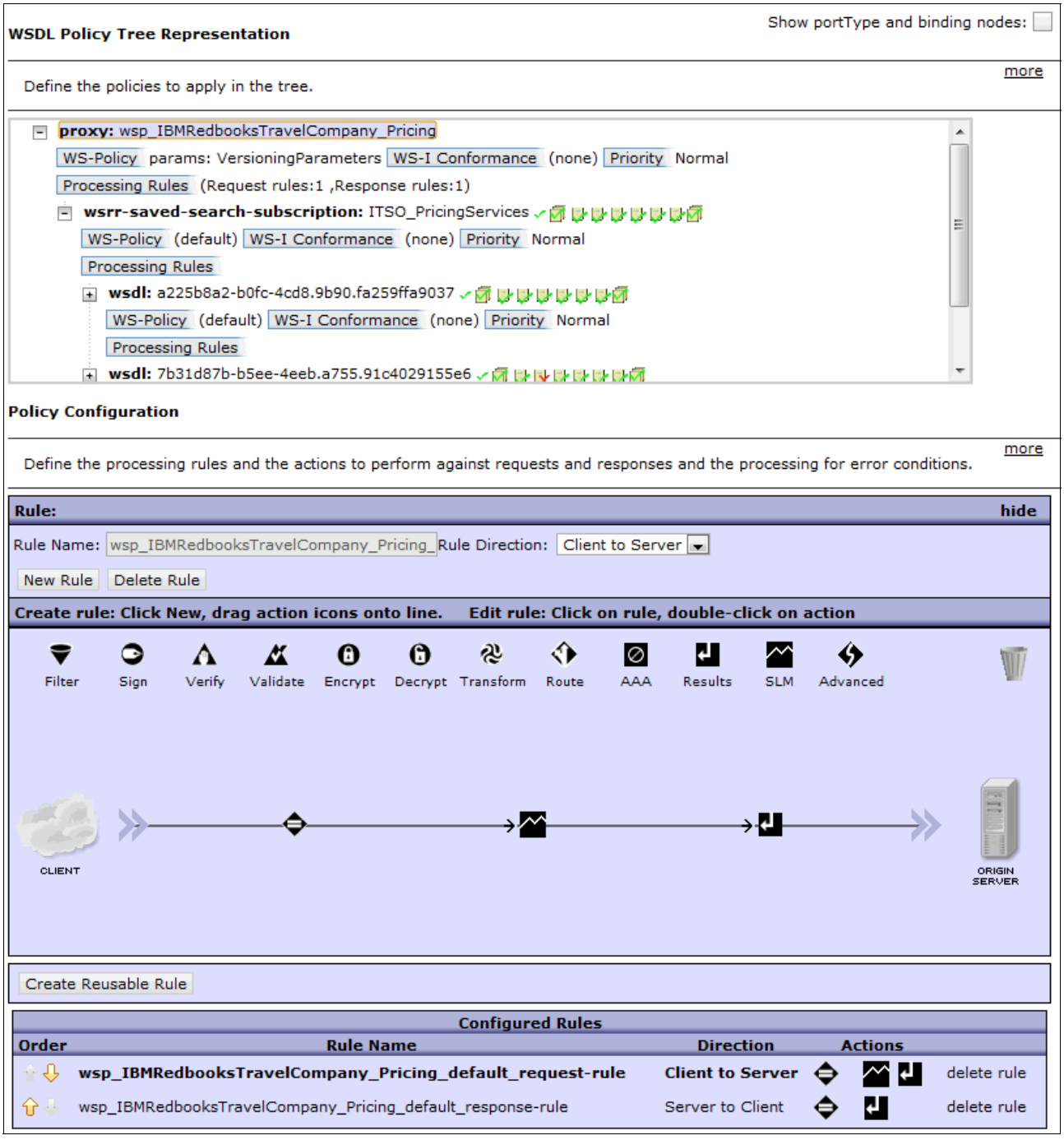


Figure 5-8 Default request and response processing rules in the Policy tab of a Web Service Proxy

Processing rules and actions that are created by a custom policy style sheet provide an enforcement on the client side of a proxy, as shown in Figure 5-9.

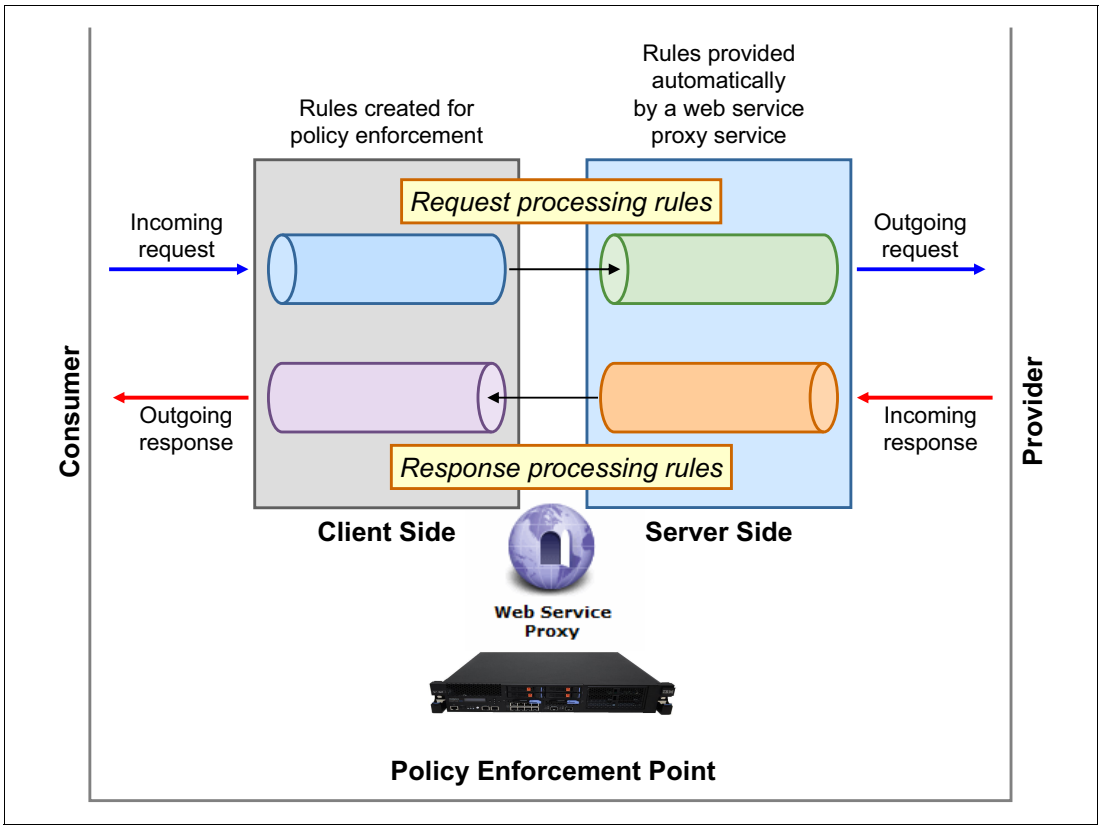


Figure 5-9 Client side and server side processing rules

Tips:

- During request processing: the request rule that enforces a custom policy precedes execution of the request processing rule of a Web Service Proxy.
- During response processing: the response rule of a Web Service Proxy precedes execution of the response processing rule that enforces a custom policy.

Because request and response processing rules of a Web Service Proxy provide automatic validations of request and response messages, deactivating these validations is important when a custom versioning policy, which performs data mediation must be implemented. If these deactivations are not configured, validation errors occur during request and response processing. Deactivation of automatic validations of request and response messages is described in step 10 on page 173 and step 11 on page 173.

5.4 Pricing service versions

The actual service from IBM Redbooks Travel Company that is used to demonstrate the versioning use case is the Pricing service. This section presents information about the various versions of the Pricing service.

5.4.1 Details of the Pricing service versions

The differences in the two Pricing service versions are as follows:

- Version 1.0 is the initial version of the Pricing service.
- Version 2.0 is the updated version of the Pricing service.

The Pricing service is used to get the price of a trip, based on various parameters.

Pricing service version 1.0

The initial version (1.0) of the service requires three parameters:

- The `trip` identifier
- The `date` interval to look for the price
- The `number` of people planned for the trip

The response consists of a price, expressed in US dollars.

The WSDL file, `PricingServiceService.wsdl`, defines the next version of the Pricing service. The file uses the `http://travel.redbooks.ibm.com/` target namespace.

Pricing service version 2.0

The updated version (2.0) of the service requires four parameters:

- The `trip` identifier
- The `date` interval to look for the price
- The `number` of people planned for the trip
- The `promotional code` to get the actual discounted price on the trip

The response consists of a price, expressed in US dollars.

The WSDL file, `PricingServiceServicev2.wsdl`, defines the version of the Pricing service. The file uses the `http://travel.redbooks.ibm.com/v2/` target namespace.

Tip: To manage versions of web services that are not backwards compatible, a good practice is to have separate namespaces for WSDL and XSD files. See the following article:

`http://www.ibm.com/developerworks/webservices/library/ws-version/`

5.4.2 Data mediation between versions of the Pricing service

Because version 1.0 was deprecated but is still used by many consumer applications that did not migrate yet to the latest version of Pricing service, the IBM Redbooks Travel Company wants to implement versioning for the consumer that accesses Pricing service version 1.0.

A data mediation and routing are necessary because the versions of the Pricing service differ in many areas, as follows:

- The two versions of the Pricing service use a different namespace.
- Version 2.0 of the Pricing service requires a supplemental parameter (promotional code).
- The endpoint of the two versions of the Pricing service differ, therefore a routing is necessary.

Because a promotional code is not defined in the version 1.0 of the Pricing service, a static value is added during the data mediation from version 1.0 to version 2.0 of the Pricing

service. The static value set as the default promotional code is `PROMO-CODE_00001234`, which will result in no discount being given.

To better understand the data mediation during request processing, view the following examples of request messages that present the transformation effort:

- Example 5-1 shows an extract (only the body) of a request message in version 1.0.

Example 5-1 Body example of a request message for Pricing service version 1.0

```
...
<soapenv:Body>
   <v1:getPrice xmlns:v1="http://travel.redbooks.ibm.com/">
      <!-- Trip Identifier -->
      <arg0>New York 0406</arg0>
      <!-- Date interval -->
      <arg1>June 2012-September 2012</arg1>
      <!-- Number of people planned for the trip -->
      <arg2>2</arg2>
   </v1:getPrice>
</soapenv:Body>
...
```

- Example 5-2 shows an extract (only the body) of a request message in version 2.0.

Example 5-2 Body example of a request message for Pricing service version 2.0

```
...
<soapenv:Body>
   <v2:getPrice xmlns:v2="http://travel.redbooks.ibm.com/v2/">
      <!-- Trip Identifier -->
      <arg0>Paris 0803</arg0>
      <!-- Date interval -->
      <arg1>October 2012-November 2012</arg1>
      <!-- Number of people planned for the trip -->
      <arg2>2</arg2>
      <!-- *** Promotional code *** -->
      <arg3>PROMO-CODE_0811</arg3>
   </v2:getPrice>
</soapenv:Body>
...
```

To better understand data mediation during response processing, view the following examples of response messages that present the transformation effort.

- Example 5-3 shows an extract (only the body) of a response message in version 1.0.

Example 5-3 Body example of a response message for Pricing service version 1.0

```
...
<soapenv:Body>
   <v1:getPriceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:v1="http://travel.redbooks.ibm.com/">
      <!-- Price -->
      <return>742.0</return>
   </v1:getPriceResponse>
</soapenv:Body>
...
```

- Example 5-4 shows an extract (only the body) of a response message in version 2.0.

Example 5-4 Body example of a response message for Pricing service version 2.0

```
...
<soapenv:Body>
   <v2:getPriceResponse xmlns:v2="http://travel.redbooks.ibm.com/v2/"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <!-- Price -->
      <return>391.0</return>
   </v2:getPriceResponse>
</soapenv:Body>
...
```

- Example 5-5 shows the XSL style sheet that transforms version 1.0 into version 2.0 of the Pricing service. This transformation can be applied on both request and response messages. The name of this XSL style sheet is `itso.pricingService.v1-v2.xsl` file.

Example 5-5 Details of the itso.pricingService.v1-v2.xsl XSL style sheet

```
<?xml version="1.0" encoding="utf-8"?>
<!--+

|*************************************************************************
*******************
   |*** Author: ITSO - gauci@fr.ibm.com
   |*** file: itso.pricingService.v1-v2.xsl
   |*** Description: XSLT used to transform pricing service messages from v1 to
v2 in both directions.
   |*** Revision : 1.0 : initial version

|*************************************************************************
*******************
   +-->
<xsl:stylesheet version="1.0"
         xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
         xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
         xmlns:v1="http://travel.redbooks.ibm.com/"
         xmlns:v2="http://travel.redbooks.ibm.com/v2/"
         extension-element-prefixes=""
         exclude-result-prefixes="soap">

   <!-- output definition -->
   <xsl:output method="xml" indent="yes" encoding="utf-8"/>

<!--+
   |*****************************************
   |*** Matching Template
   |*** Element: all
   |*****************************************
   +-->
   <!-- This template matches all elements of the SOAP input document. -->
   <xsl:template match="*">
      <xsl:copy>
         <xsl:apply-templates select="@*|*|text()|comment()"/>
      </xsl:copy>
   </xsl:template>
```

```
<!--+
    |************************************************
    |*** Matching Template
    |*** Element: attributes, text values and comments
    |************************************************
    +-->
    <!-- This template matches both attributes, text values and comments. -->
    <xsl:template match="@*|text()|comment()">
       <xsl:copy/>
    </xsl:template>

<!--+
    |************************
    |*** Matching Template
    |*** Element: v1:getPrice
    |************************
    +-->
    <!-- This template matches 'v1:getPrice' request  and transforms it into a
'v2:getPrice'. -->
    <xsl:template match="v1:getPrice">
       <v2:getPrice>
          <xsl:apply-templates select="@*|*|text()|comment()"/>
          <xsl:comment>arg3 (promo code) parameter added with default value:
PROMO-CODE_0001234</xsl:comment>
          <arg3>PROMO-CODE_0001234</arg3>
       </v2:getPrice>
    </xsl:template>

<!--+
    |*******************************
    |*** Matching Template
    |*** Element: v2:getPriceResponse
    |*******************************
    +-->
    <!-- This template matches 'v2:getPriceResponse' response element and
transforms it into a v1:getPriceResponse'. -->
    <xsl:template match="v2:getPriceResponse">
       <v1:getPriceResponse>
          <xsl:apply-templates select="@*|*|text()|comment()"/>
       </v1:getPriceResponse>
    </xsl:template>

</xsl:stylesheet>
```

The default value of the promotional code is set as a static value but it can be handled as a stylesheet parameter.

The XSL style sheet can be created by using any XML editor tool or IDE, such as IBM Integration Designer, for instance.

5.4.3 Pricing services in WSRR

Because the IBM Redbooks Travel Company wants to allow consumer applications in version 1.0 to access the pricing service in version 2.0, we load and manage the resultant custom versioning policy in WSRR.

WSRR contains the various information and objects that are related to the possible versions of the Pricing service. Figure 5-10 shows an overview of the Pricing service versions and consumer applications in WSRR.

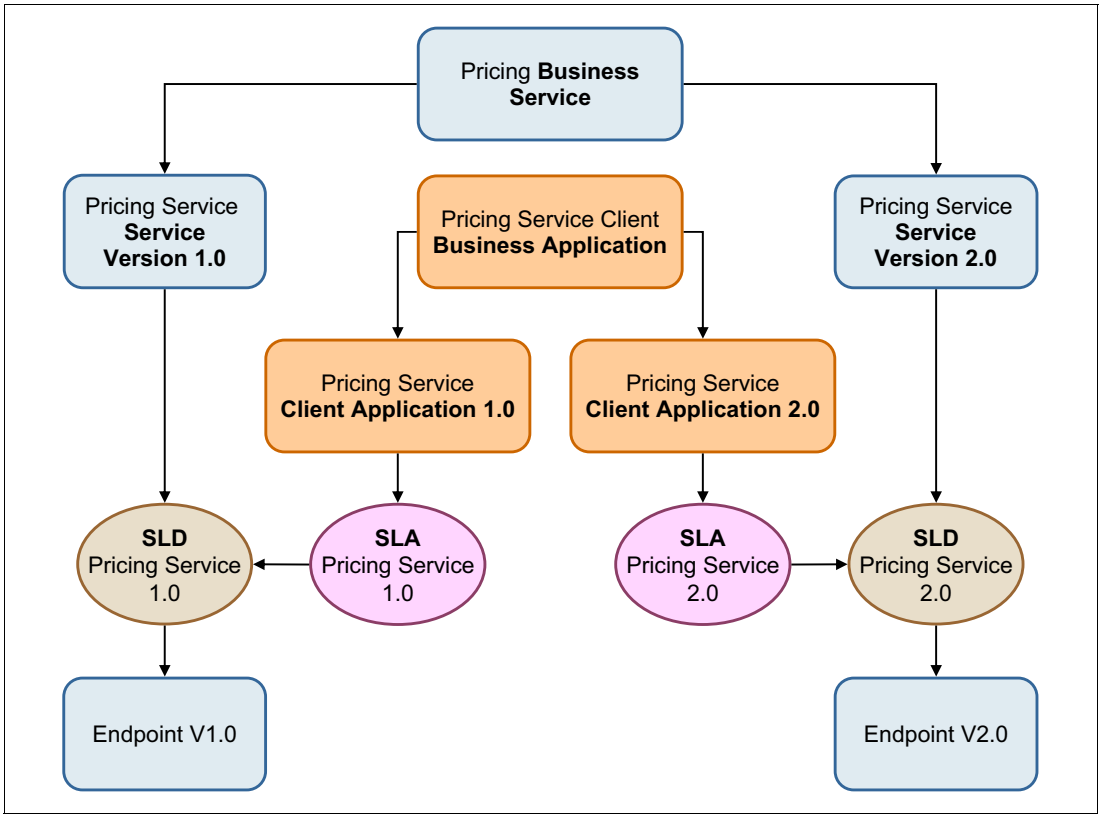


Figure 5-10 Pricing service versions and consumer applications in WSRR

Figure 5-10 shows the following components and relations:

- The Business Service of the Pricing service, and its child objects
- Business Application of the Pricing service, and its child objects
- Relations between the various objects that inherit from the Pricing Business Service and the Pricing Service Client Business Application

Consumers (in gold) and providers (in blue) are linked through a dedicated SLA, which represents a contract between a consumer and a provider. Moreover, the versioning is administered through WSRR only, using a custom policy attached to the correct SLD.

This policy must be attached at the SLD level, linked to the endpoint version 1.0, as shown in Figure 5-11.

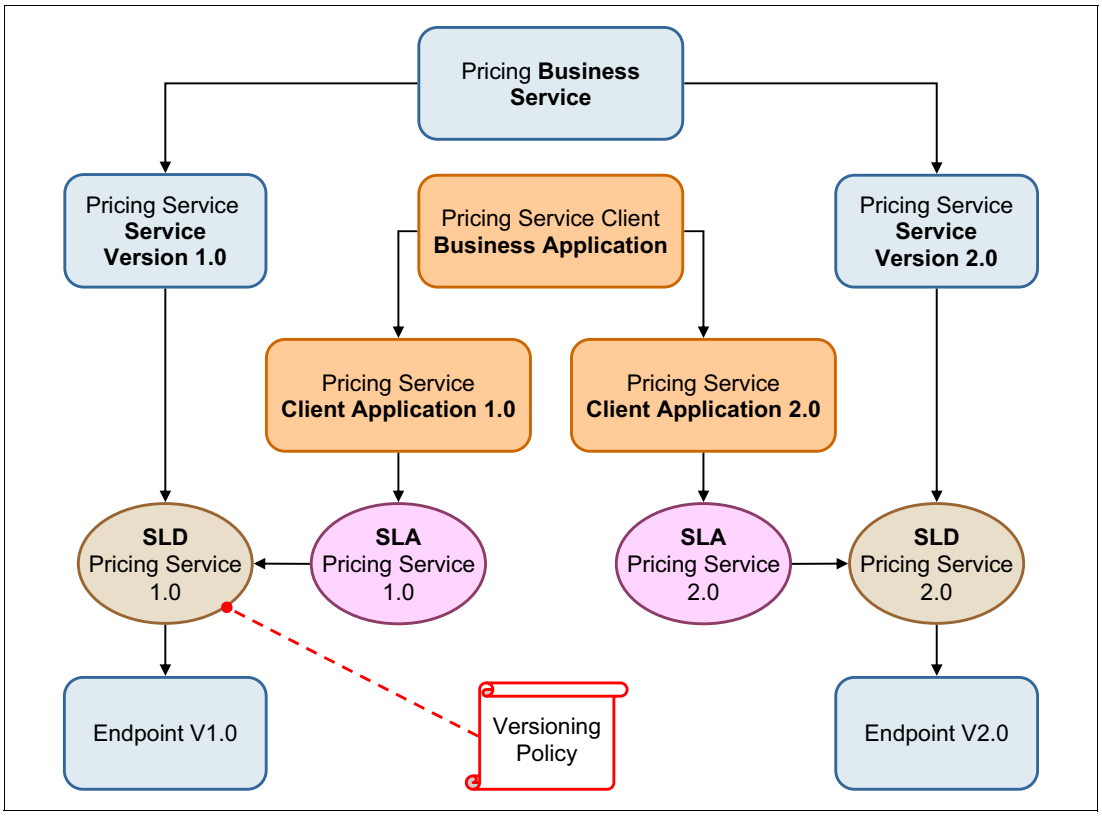


Figure 5-11 Attaching the custom policy at the SLD level for versioning

When the custom versioning policy is attached as described, the Client Application 1.0 will actually use the Pricing Service Version 2.0. At run time, DataPower is responsible for enforcing this versioning policy.

5.5 Creating custom policy domain and assertions for versioning

This section discusses the creation of a custom policy domain (namespace and vocabulary) that is related to the versioning policy that the IBM Redbooks Travel company requires.

The first step is to decide on a namespace for the custom versioning policy. This namespace is used when creating a policy, to qualify the elements of the custom versioning policy vocabulary. It is also used when the custom XSL stylesheet policy implements transform policy assertions into DataPower configuration artifacts.

The namespace is as follows; The prefix for this namespace is `itso`:

`http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11`

Tip: A namespace is used to qualify elements of a custom policy grammar. Namespace usage prevents collisions of various vocabulary elements (assertions) that are linked to separate custom policy domains.

This custom policy for versioning must reflect the capabilities of the versioning pattern that is implemented. These capabilities are described in 5.3, “Custom versioning policy enforcement details” on page 125.

At least, the vocabulary must be able to define routing information, and also the transformation that is used for data mediation and validation artifacts.

The following elements define the versioning grammar:

- `ClientWSDL`
- `ServerWSDL`
- `Route`
- `Transform`

To present this information clearly, the `ClientWSDL` and `ServerWSDL` are included in a `Validate` element. Moreover, all these elements are wrapped in a `Versioning` container, which specifies the type of enforcement. All of these elements are bound to the namespace:

`http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11`

Example 5-6 shows the custom policy for versioning, using the chosen namespace and vocabulary.

Example 5-6 Custom policy for versioning

```
<?xml version="1.0" encoding="utf-8"?>
<!--+
   **********************
   *** itso:Versioning ***
   ************************************************************************
   *** Author: ITSO IBM Redbooks Travel Company
   *** Revision: 1.0
   *** Description:Custom policy that implements the versioning pattern
   *** Namespace:http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11
   ************************************************************************
   +-->
<wsp:Policy wsp:Name="Message-versioning-template"
            wsu:Id="wsp-versioningpolicy"
            xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
            xmlns:wsmp="http://www.ibm.com/xmlns/stdwip/2011/02/ws-mediation"

xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.x
sd"
           xmlns:itso="http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11">


   <itso:Versioning>
      <!-- Validations- URL of WSDL files for validation | *** optional *** -->
      <itso:Validate>
         <itso:ClientWSDL>1234567890</itso:ClientWSDL>
         <itso:ServerWSDL>ABCDEF0123</itso:ServerWSDL>
      </itso:Validate>
      <!-- Data mediation - Same transformation for request and response | *** optional
*** -->
      <itso:Transform>local:///transformMessage.xsl</itso:Transform>
      <!-- Route information | *** required *** -->
      <itso:Route>http://ibm.redbooks.travelcompany/v2/pricing</itso:Route>
   </itso:Versioning>

</wsp:Policy>
```

The name of the custom security policy file is `versioning-customPolicy.xml`.

Table 5-1 lists the details of the elements of the custom versioning policy file.

Table 5-1 Elements and attributes of the custom policy for versioning

Element	Description
`/wsp:Policy`	Root element of a custom security policy.
`/wsp:Policy/@wsp:Name`	Attribute that defines the name for the custom security policy.
`/wsp:Policy/@wsu:Id`	Attribute that defines the identifier of the custom security policy.
`/wsp:Policy/itso:Versioning`	Container for the versioning information
`/wsp:Policy/itso:Versioning/itso:Validate`	Container for the declaration of the validation artifacts. This element is optional.
`/wsp:Policy/itso:Versioning/itso:Validate/ itso:ClientWSDL`	URL of the WSDL file that is used to validate the incoming request and outgoing response. The format is as follows: *<bsrURI_of_the_wsdl>*
`/wsp:Policy/itso:Versioning/itso:Validate/ itso:ServerWSDL`	URL of the WSDL file that is used to validate the outgoing request and incoming response. The format is as follows: *<bsrURI_of_the_wsdl>*
`/wsp:Policy/itso:Versioning/itso:Transform`	URL of the XSL transformation that is used to execute data mediation during request and response processing. The format is as follows: `local:///`*<name_of_the_stylesheet>* The style sheet is loaded on the DataPower appliance.
`/wsp:Policy/itso:Versioning/itso:Route`	URL of the target endpoint. The format is as follows: `http://`*<host>*`:`*<port>*`/`*<URI>*

In the example that is based on the Pricing service of the IBM Redbooks Travel Company, the aim is to implement versioning between version 1.0 and version 2.0 of the Pricing service.

The following information is related to the Pricing service versions that are required for creating the final version of the custom policy for versioning:

- The endpoint of the Pricing service version 2.0 is as follows:

  `http://sa-w217rhel-1.itso.ral.ibm.com:9081/redbooksTravelV2/PricingServiceService`

- The WSRR bsrURI of the WSDL file, which is used to validate a message that is related to the Pricing service in version 1.0, is `7b31d87b-b5ee-4eeb.a755.91c4029155e6`.

  Use the following steps to retrieve this information about DataPower from WSRR:

  a. Connect to the WSRR through the Service Registry interface with `Administrator` credentials (login and password)

  b. Click the **WSDL Documents** link, which is listed in the Service Documents section, shown in Figure 5-12 on page 137.

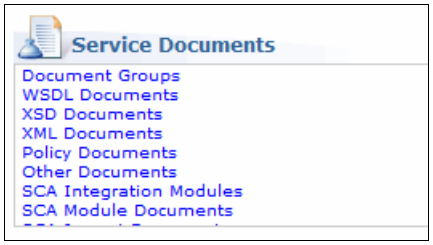


Figure 5-12 Accessing a WSDL document in WSRR

c. Select the WSDL document for which you are interested to get the bsrURI. The WSDL file of the Pricing service version 1.0 is `PricingServiceService.wsdl`.

d. In the Additional Properties list for the WSDL file, the bsrURI property contains the unique identifier of the current WSDL file, as shown in Figure 5-13.

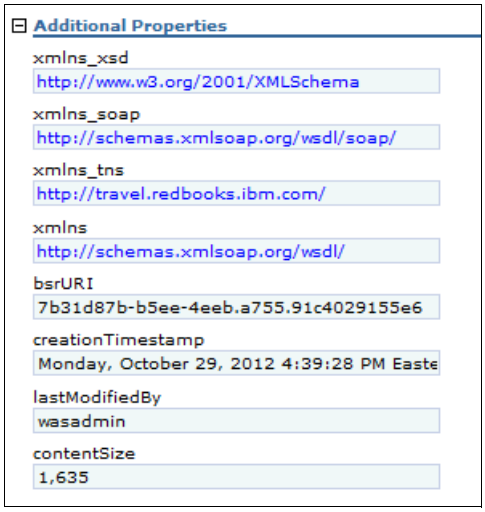


Figure 5-13 WSDL additional properties

e. Copy the bsrURI value so that you can use it in the custom policy for versioning.

- In a similar manner, determine that the WSRR bsrURI of the WSDL file, used to validate a message that is related to the Pricing service in version 2.0, is as follows:

  `a225b8a2-b0fc-4cd8.9b90.fa259ffa9037`

- The URL of the XSL style sheet, used to transform Pricing service versions (version 1.0 into version 2.0 and vice versa), is as follows:

  `local:///itso.pricingService.v1-v2.xsl`

  It is presented in 5.4.2, “Data mediation between versions of the Pricing service” on page 129. In this example, the data mapper is located on the DataPower appliance (see `local:///` directory). In particular, it is located on the domain on which the versioning enforcement of the Pricing service must be applied.

To retrieve the WSDL files that are related to WSRR more easily, create a saved search in WSRR and then create a subscription to that saved search in DataPower. A saved search is a smart mechanism that allows a user to specify a query to retrieve objects in WSRR. This query can be saved for future uses, as discussed in 5.7.2, “Creating a saved search in WSRR at design-time” on page 164. The saved search that was created in WSRR to retrieve the WSDL files, related to the Pricing service, is named `ITSORedbooksTravel_Pricing_WSDLs`.

The final version of the custom policy for versioning related to the Pricing service is shown in Example 5-7.

Example 5-7 Final version of the custom policy for versioning of the Pricing service

```
<?xml version="1.0" encoding="utf-8"?>
<!--+
   **********************
   *** itso:Versioning ***

********************************************************************************
**
   *** Author: ITSO IBM Redbooks Travel Company
   *** Revision: 1.0
   *** Description:Custom policy that implements the versioning pattern
   *** Namespace:http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11

********************************************************************************
**
   +-->
<wsp:Policy wsp:Name="Message-versioning-template"
            wsu:Id="wsp-versioningpolicy"
            xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
            xmlns:wsmp="http://www.ibm.com/xmlns/stdwip/2011/02/ws-mediation"

xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-util
ity-1.0.xsd"

xmlns:itso="http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11">

   <itso:Versioning>
      <itso:Validate>
         <itso:ClientWSDL>7b31d87b-b5ee-4eeb.a755.91c4029155e6</itso:ClientWSDL>
         <itso:ServerWSDL>a225b8a2-b0fc-4cd8.9b90.fa259ffa9037</itso:ServerWSDL>
      </itso:Validate>
      <itso:Transform>local:///itso.pricingService.v1-v2.xsl</itso:Transform>

<itso:Route>http://sa-w217rhel-1.itso.ral.ibm.com:9081/redbooksTravelV2/PricingSer
viceService</itso:Route>
   </itso:Versioning>

</wsp:Policy>
```

5.6 Creating custom policy XSL style sheet for the custom versioning policy

The custom policy style sheet is required for creating the DataPower configuration artifacts based on the custom security policy.

Figure 5-14 shows the generation process of the DataPower configuration artifacts for the custom policy.

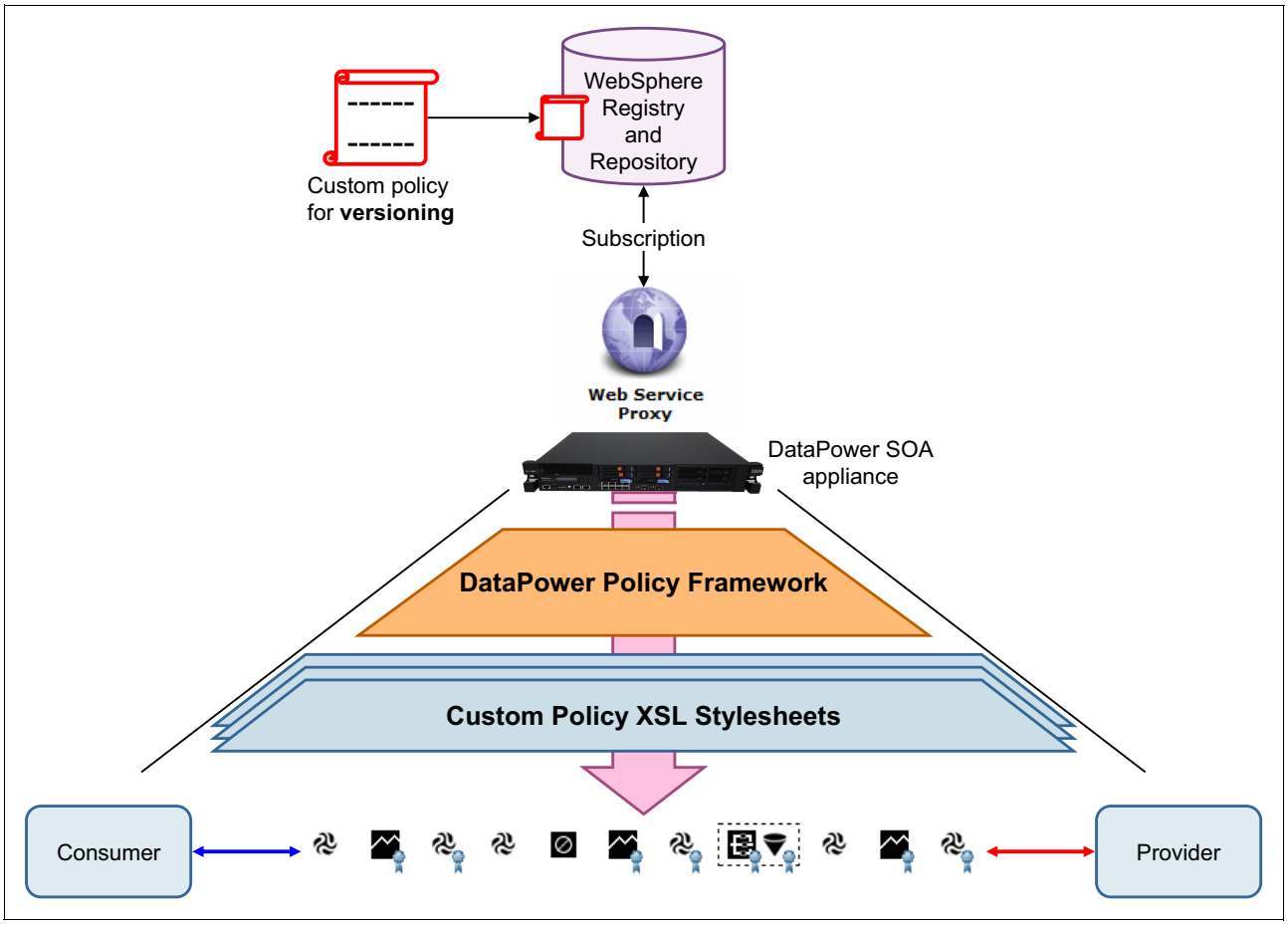


Figure 5-14 Generation of configuration artifacts based on a custom policy with DataPower

The custom policy for versioning can reside either on WSRR, where it can be governed and managed, or directly on the DataPower appliance. Because the purpose of this example is to show managing custom policy on WSRR, the focus here is on that use case.

The custom style sheet is loaded on the DataPower appliance in the default domain, in the `store:///polices/custom` directory.

5.6.1 Creating the DataPower configuration artifacts

Before creating the custom policy style sheet, a good approach is to create the configuration artifacts that must be generated by the style sheet itself. After these artifacts are created, their configurations can be used as an example (or pattern) for the custom versioning policy style sheet that must be coded.

The following actions of the versioning enforcement are performed:

- Validations based on a WSDL file
- Message transformation
- Routing

Use the following configuration steps to create these actions on a DataPower device. A simple XML firewall service is configured to provide a configuration example of the actions:

1. Connect to a DataPower appliance on a development domain by using your credentials (login and password).
2. Click the **XML Firewall** service icon, as shown in Figure 5-15.

Figure 5-15 XML Firewall icon

3. Click **Add Advanced**.
4. Enter a name for the XML Firewall service and the following properties:
   - Set the Firewall Type as `Loopback`.
   - Select a port number (Port Number property) that is currently available.

   An example of this configuration is shown in Figure 5-16 on page 141.

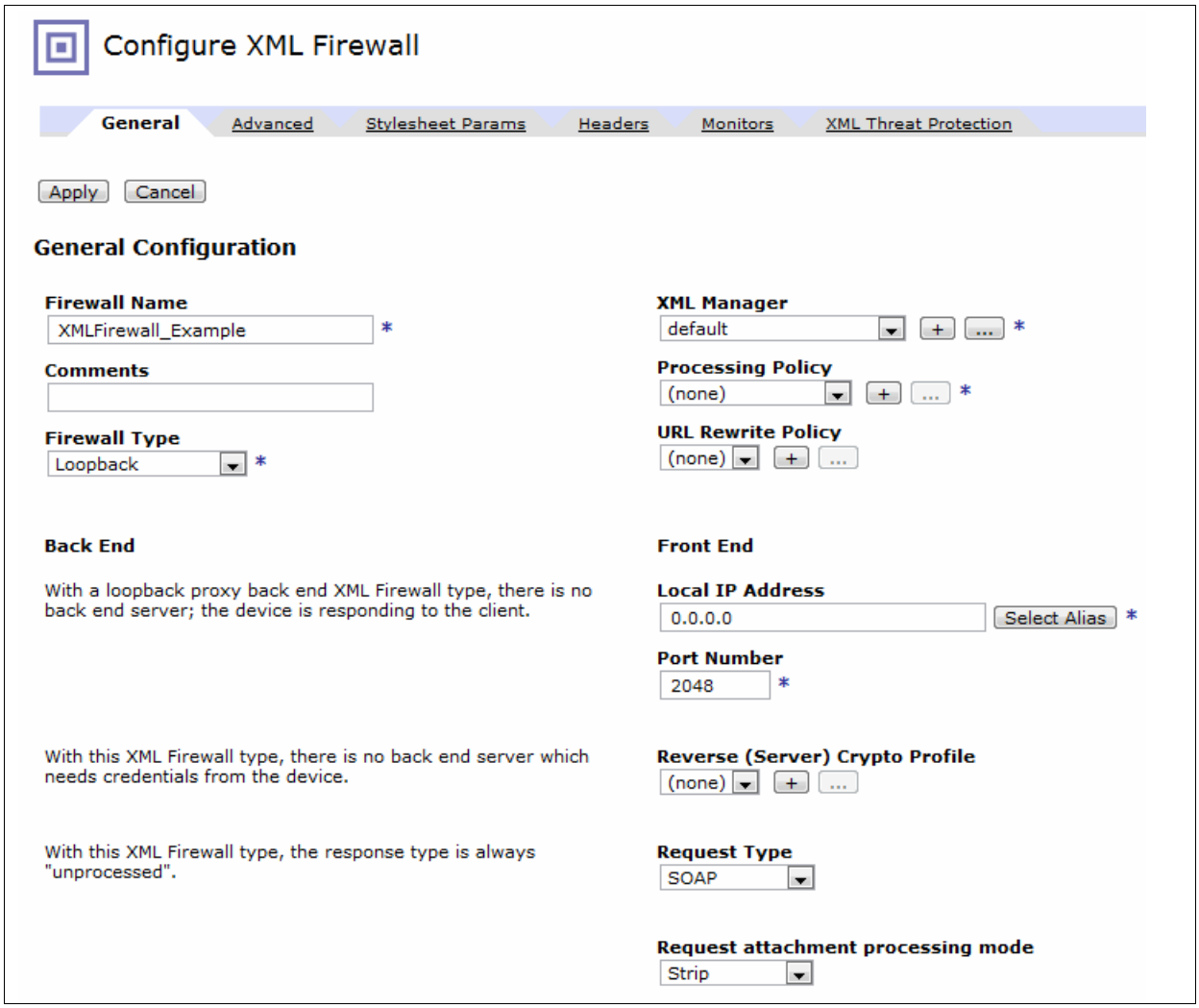


Figure 5-16 XML Firewall service main configuration panel

5. Click the plus sign (**+**) beside the Processing Policy property, as shown in Figure 5-17.

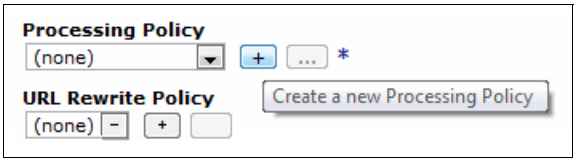


Figure 5-17 Create a new processing policy

6. In the window that opens, enter a name for the processing policy, for instance, enter the following name:

   `XMLFirewall_Example`

7. Click **New Rule** and select **Client to Server** as the direction of the rule. The resulting window is shown in Figure 5-18 on page 142.

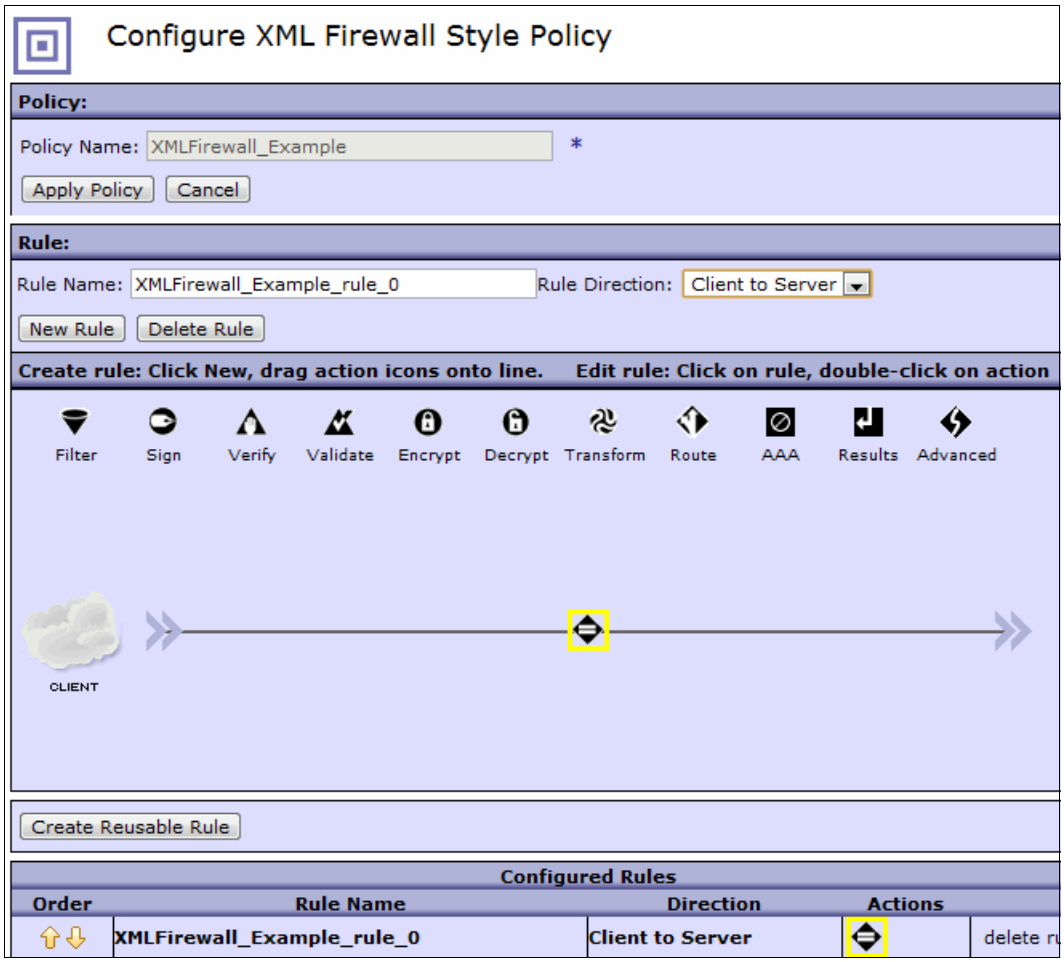


Figure 5-18 First step to create an example of configuration artifacts

8. Configure the match action (the first element of the request processing rule) at your convenience.
9. Drag **Validate**, **Transform**, and **Advanced** from the list of icons to the current request processing rule. The resulting processing rule is shown in Figure 5-19.

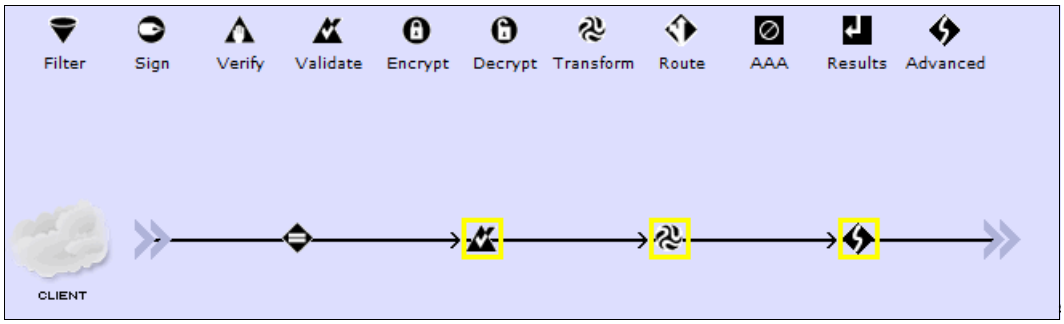


Figure 5-19 Validate, Transform, and Advanced icons

These actions will be used during versioning policy enforcement. The Advanced icon is configured as a Set Variable action, used for routing.

10. Double-click the **Validate** action. In the window that opens, provide the following properties:
   - Select **Validate Document via WSDL URL** as a Schema Validation Method.
   - Select **wsrr://** protocol as a protocol of the WSDL URL property.
   - Enter `123456` as the URI part of the WSDL URL.

   Figure 5-20 shows the configuration of the Validate action.

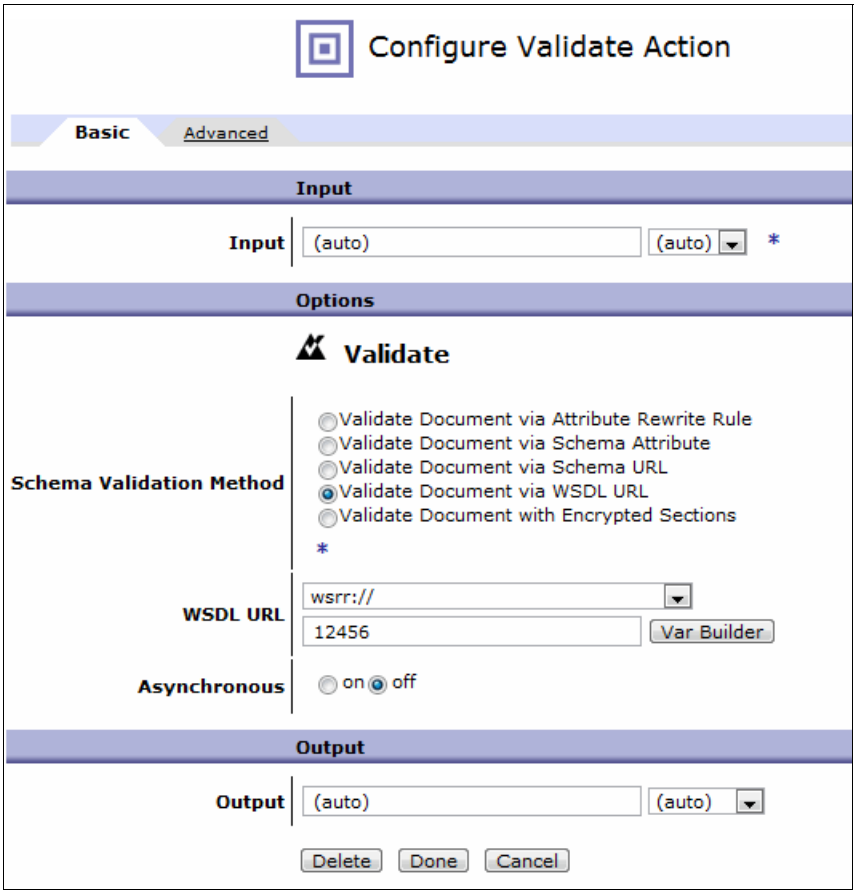


Figure 5-20 Validate action configuration

11. Click **Done** to complete the configuration of the Validate action.
12. Double-click the **Transform** action. On the window that opens, provide the following properties:
   - Select **store:///** as the protocol of the XSL stylesheet property.
   - To define the XSL stylesheet URL, select any XSL from the list of provided style sheets. For instance, select `identity.xsl`.

Figure 5-21 shows the configuration of the Transform action.

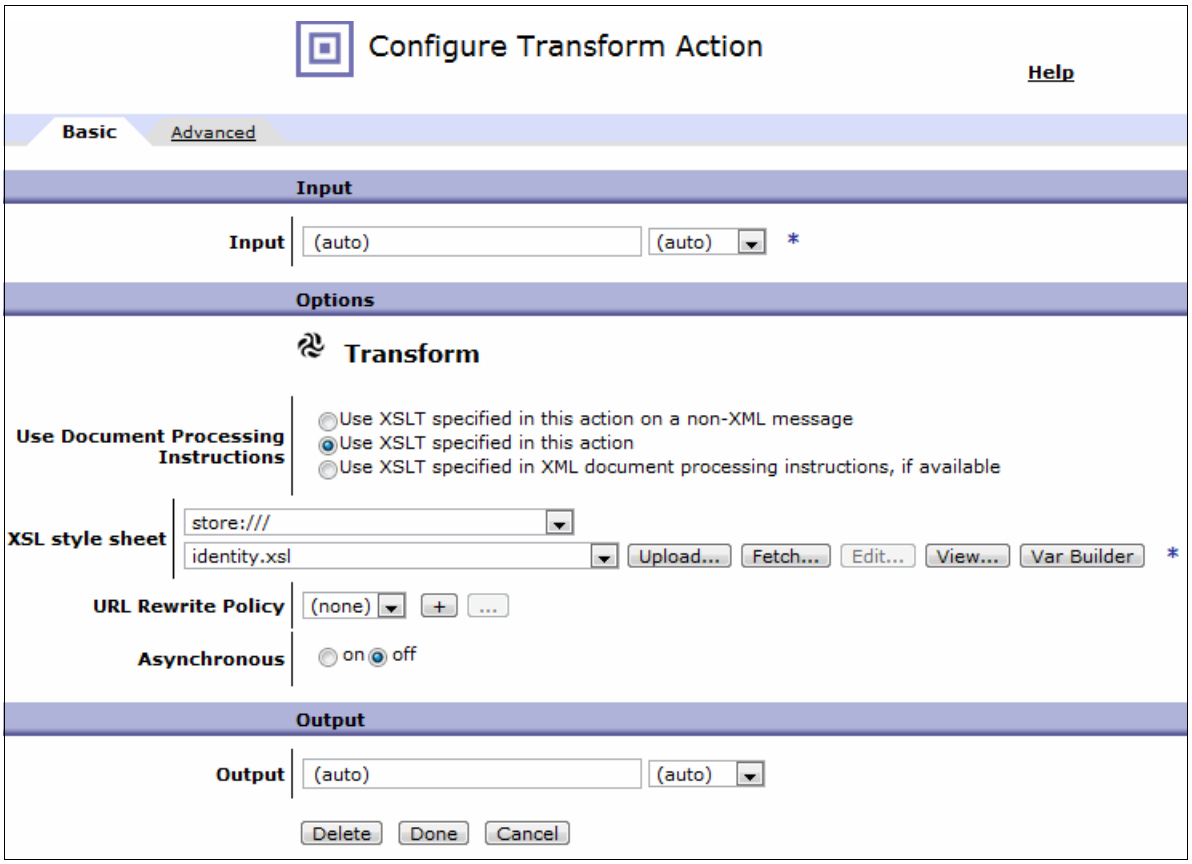


Figure 5-21 Transform action configuration

13. Click **Done** to complete the configuration of the Transform action.

14. Double-click the **Advanced** action. On the window that opens, select **Set Variable**, and click **Next**, as shown in Figure 5-22.

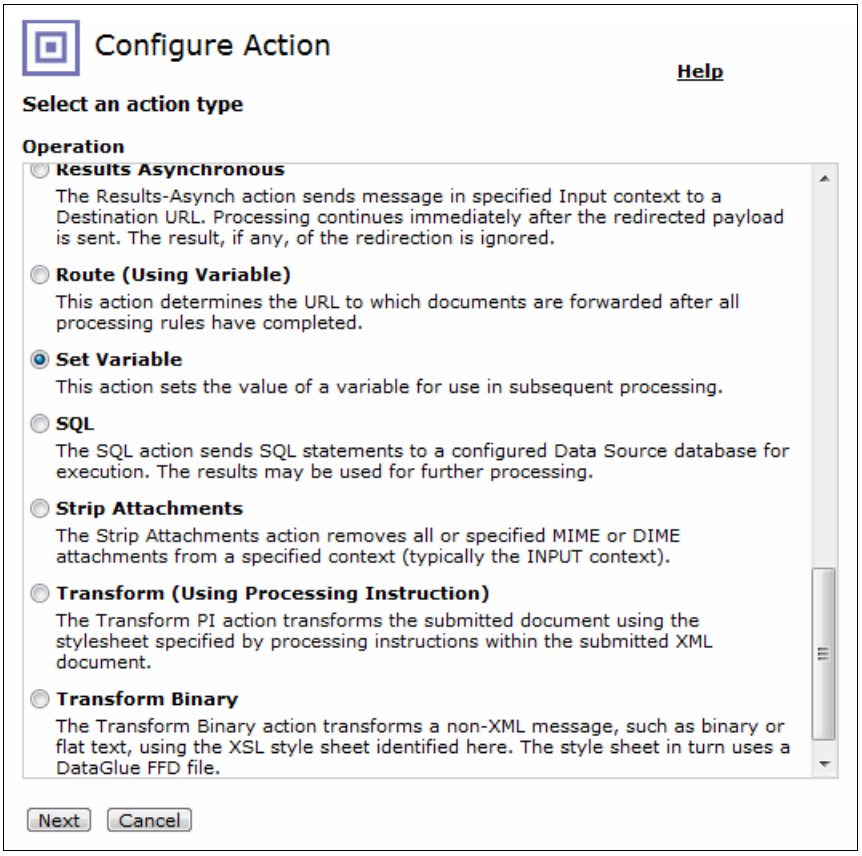


Figure 5-22 Select the Set Variable action from the list of advanced actions

15. From the Set Variable configuration panel, click **Var Builder**.
16. From the Service Variables section, select the **var://service/routing-url** service variable.

17. In the Variable Assignment text field, enter a URL, for example, `http://itso.com/redbooks/test`, as shown in Figure 5-23.

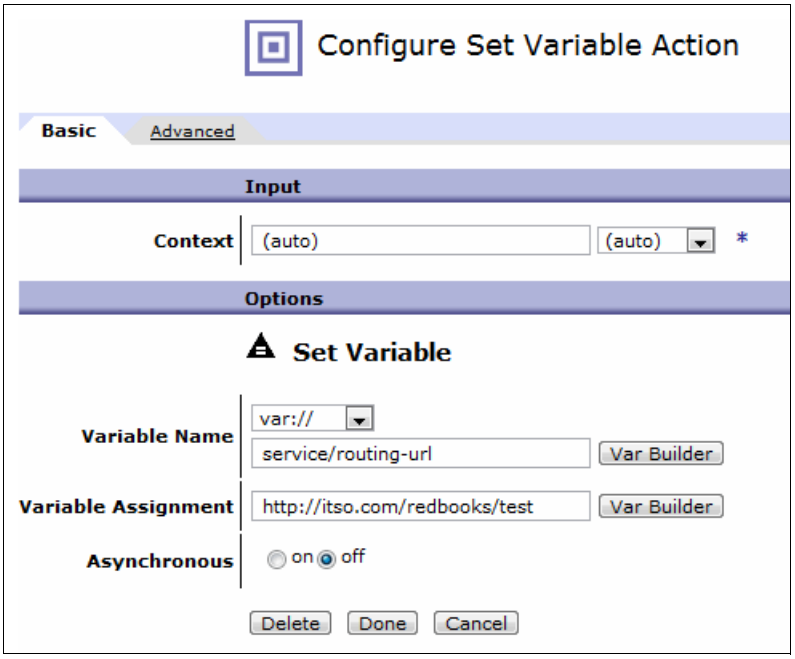


Figure 5-23 Define the variable assignment

18. Click **Done** to complete the configuration of the Set Variable action.

The various actions of the processing are now configured, as shown in Figure 5-24.

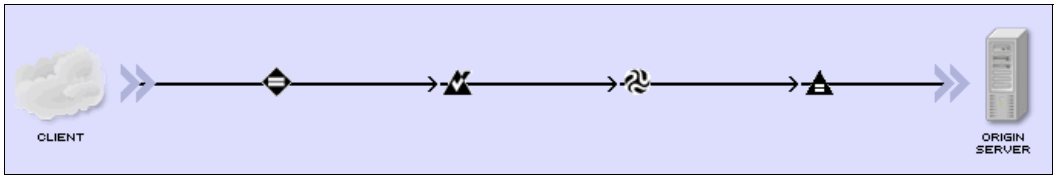


Figure 5-24 Final configuration of the processing rule example

5.6.2 Exporting the DataPower configuration artifacts

After an example of the artifacts of the versioning policy is configured, you can export the DataPower artifacts for the custom policy style sheet. Actually, the `export.xml` file of the archive (result of the export process) contains the configuration patterns that can be used to implement the custom style sheet for versioning.

Use the following steps to create a downloadable archive of the XML Firewall service (created in 5.6.1, “Creating the DataPower configuration artifacts” on page 140):

1. Connect to a DataPower appliance by using your credentials (login and password). Select the domain on which the XML Firewall, configured with the versioning artifacts, is defined.
2. Click the **Export Configuration** icon shown in Figure 5-25 on page 147.

Figure 5-25 Export configuration icon

3. Select **Export configuration and files of the current domain** from the list of current exportation capabilities, as shown in Figure 5-26.

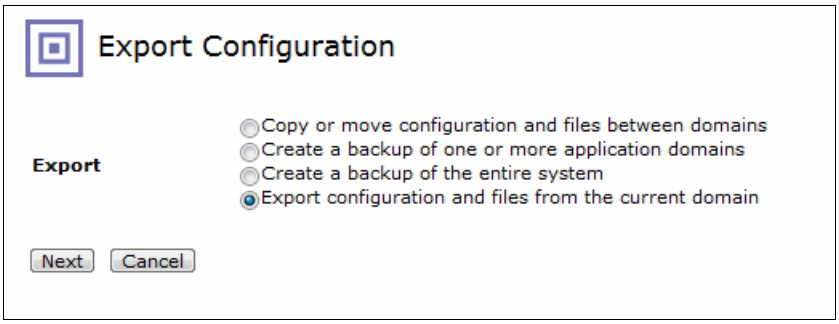


Figure 5-26 Export configuration and files of the current domain

4. Click **Next**.
5. Modify the export file name and set it to `export_VersioningPolicy`.
6. To finalize the exportation process, complete the following steps:
   a. Select **XML Firewall Service** from the list of configuration objects to export, and then select the **XMLFirewall_Example** service.
   b. Click the right arrow button (**>**) to move XMLFirewall_Example into the list of selected objects, as shown in Figure 5-27.

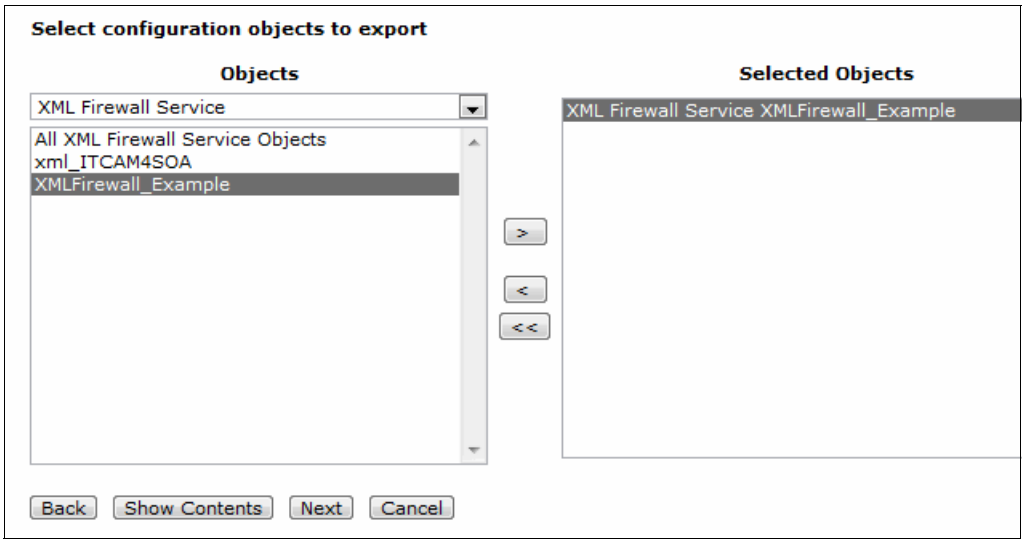


Figure 5-27 Select the XML Firewall service to export

   c. Click **Next**.

d. Click **Download** to download the export file, as shown in Figure 5-28.

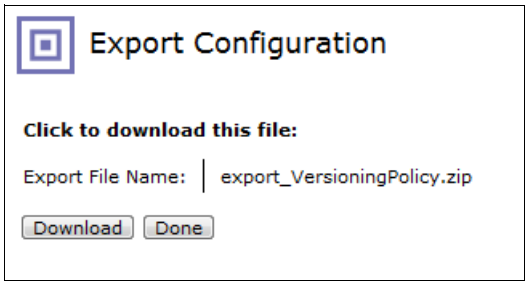


Figure 5-28 Download the versioning configuration artifacts examples

a. Click **Done** when the download task completes.

b. Extract the `export.xml` file from the `export_VersioningPolicy.zip` archive and open it with an XML editor or your integrated development environment.

The following elements in the `export.xml` file, contained in the archive, are of interest. These three configuration extracts are used to implement the custom versioning policy style sheet. These configuration examples are not copied and pasted in the style sheet directly but used as configuration examples to create required configuration artifacts that are generated by the style sheet itself.

- The `StylePolicyAction` element with the `validate` keyword in its name

  This action represents a Validate action. Example 5-8 shows the Validate action configuration, based on a WSDL file that is retrieved from WSRR.

Example 5-8 Configuration of a DataPower Validate processing action

```
<StylePolicyAction name="XMLFirewall_Example_rule_0_validate_0"
          xmlns:env="http://www.w3.org/2003/05/soap-envelope"
xmlns:dp="http://www.datapower.com/schemas/management">
          <mAdminState>enabled</mAdminState>
          <Type>validate</Type>
          <Input>INPUT</Input>
          <Output>dpvar_1</Output>
          <NamedInOutLocationType>default</NamedInOutLocationType>
          <WsdlURL>wsrr://wsrrserver/12456</WsdlURL>
          <Transactional>off</Transactional>
          <SOAPValidation>body</SOAPValidation>
          <SQLSourceType>static</SQLSourceType>
          <Asynchronous>off</Asynchronous>
          <ResultsMode>first-available</ResultsMode>
          <RetryCount>0</RetryCount>
          <RetryInterval>1000</RetryInterval>
          <MultipleOutputs>off</MultipleOutputs>
          <IteratorType>XPATH</IteratorType>
          <Timeout>0</Timeout>
          <MethodRewriteType>GET</MethodRewriteType>
          <MethodType>POST</MethodType>
          <MethodType2>POST</MethodType2>
      </StylePolicyAction>
```

- The `StylePolicyAction` element with the `xform` keyword in its name

  This action represents a Transform action. Example 5-9 shows the Transform action configuration, based on an XSL style sheet that is located on a DataPower appliance.

Example 5-9 Configuration of a DataPower Transform processing action

```
<StylePolicyAction name="XMLFirewall_Example_rule_0_xform_0"
          xmlns:env="http://www.w3.org/2003/05/soap-envelope"
xmlns:dp="http://www.datapower.com/schemas/management">
          <mAdminState>enabled</mAdminState>
          <Type>xform</Type>
          <Input>dpvar_1</Input>
          <Transform>store:///identity.xsl</Transform>
          <Output>dpvar_2</Output>
          <NamedInOutLocationType>default</NamedInOutLocationType>
          <OutputType>default</OutputType>
          <Transactional>off</Transactional>
          <SOAPValidation>body</SOAPValidation>
          <SQLSourceType>static</SQLSourceType>
          <Asynchronous>off</Asynchronous>
          <ResultsMode>first-available</ResultsMode>
          <RetryCount>0</RetryCount>
          <RetryInterval>1000</RetryInterval>
          <MultipleOutputs>off</MultipleOutputs>
          <IteratorType>XPATH</IteratorType>
          <Timeout>0</Timeout>
          <MethodRewriteType>GET</MethodRewriteType>
          <MethodType>POST</MethodType>
          <MethodType2>POST</MethodType2>
      </StylePolicyAction>
```

- The `StylePolicyAction` element with the `setvar` keyword in its name

  This action represents a Set Variable action, that can be used for routing purposes. Example 5-10 shows the Set Variable action configuration.

Example 5-10 Configuration of a Set Variable processing action

```
<StylePolicyAction name="XMLFirewall_Example_rule_0_setvar_0"
          xmlns:env="http://www.w3.org/2003/05/soap-envelope"
xmlns:dp="http://www.datapower.com/schemas/management">
          <mAdminState>enabled</mAdminState>
          <Type>setvar</Type>
          <Input>dpvar_2</Input>
          <NamedInOutLocationType>default</NamedInOutLocationType>
          <Variable>var://service/routing-url</Variable>
          <Value>http://itso.com/redbooks/test</Value>
          <Transactional>off</Transactional>
          <SOAPValidation>body</SOAPValidation>
          <SQLSourceType>static</SQLSourceType>
          <Asynchronous>off</Asynchronous>
          <ResultsMode>first-available</ResultsMode>
          <RetryCount>0</RetryCount>
          <RetryInterval>1000</RetryInterval>
          <MultipleOutputs>off</MultipleOutputs>
          <IteratorType>XPATH</IteratorType>
          <Timeout>0</Timeout>
          <MethodRewriteType>GET</MethodRewriteType>
          <MethodType>POST</MethodType>
          <MethodType2>POST</MethodType2>
      </StylePolicyAction>
```

5.6.3 Writing the custom policy style sheet for versioning

Remember, the custom policy style sheet is responsible for generating the versioning policy artifacts, based on the custom versioning policy in Example 5-6 on page 135.

To write this custom mapping, you can use the style sheet in Example 13-1 on page 374 as a starting point.

This style sheet is inspired by the `jk-example.xsl` custom policy style sheet, which is provided in the `store:///policies/custom` directory of any DataPower appliance that uses firmware version 5.0.0 or later. The name of the XSL style sheet is `itso.customPolicy.example.xsl`.

Important: The matching templates other than the following one, must not be modified:

```
<xsl:template match="myNamespace:MyAssertion" mode="assertion">
```

New matching and named templates, and also extension functions and DataPower Policy parameters may also be created.

The following steps describe how to complete coding of the style sheet, so it can be used to generate DataPower configuration artifacts based on the custom versioning policy (shown in Example 5-6 on page 135):

1. Declare the `itso` prefix bound to the namespace of the custom versioning policy domain.
2. Add the namespace declaration into the `</dppolicy:domain>` element.
3. Add a DataPower Policy parameter to declare the WSRR server, used to build the path to WSDL files in WSRR.

   Tip: In DataPower, a WSDL file that is located on a WSRR server can be accessed by using a URL with the following format; the end of the URL has the trailing slash (/) character:

   `wsrr://<WSRR_Server_Object_Name>/<WSDL_bsrURI>/`

   Using this URL format is the way (as a suggested practice) to access WSDL files in WSRR from a DataPower device. Importation of other WSDL files or schemas (XSD) in the main WSDL is automatically managed.

4. Create a matching template, which matches the following qualified element of the custom versioning policy:

   `{http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11}Versioning`

   The result of this matching template is the generation of two `<dppolicy:config>` elements, based on presence and value of the following elements:

   - `{http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11}Validate`
   - `{http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11}ClientWSDL`
   - `{http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11}ServerWSDL`
   - `{http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11}Transform`
   - `{http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11}Route`

The two elements are as follows:

– The first <dppolicy:config> element contains the following artifacts, defined on a request processing rule:
  - A validate action, used to validate the *incoming* request (value of ClientWSDL)
  - A transform action, used to transform the incoming request message (value of Transform)
  - A validate action, used to validate the *outgoing* request (value of ServerWSDL)
  - A route action (Set Variable), which is used to route the outgoing request (value of Route)
– The second <dppolicy:config> element contains the following artifacts, defined on a response processing rule:
  - A validate action, used to validate the *incoming* response (value of ServerWSDL)
  - A transform action, used to transform the incoming response message (value of Transform)
  - A validate action, used to validate the *outgoing* response (value of ClientWSDL)

Directions of the processing rule, on which the versioning artifacts must be created, are indicated through the direction attribute of the <dppolicy:config> elements.

In this case, the request processing rule uses the value request-rule and the response processing rule uses the value response-rule.

5. The authored custom policy XSL style sheet is now ready to be loaded on the DataPower device that is used to enforce the custom versioning policy.

The final version of the itso.customPolicy.versioning.xsl file is presented in Example 5-11.

Example 5-11 Custom policy XSL style sheet for versioning policy configuration artifacts generation

```
<?xml version="1.0"?>
<!-- +
     |*****************************************************
     |*** Author: ITSO - gauci@fr.ibm.com
     |*** file: itso.customPolicy.versioning.xsl
     |*** Description: Custom policy XSL styleheet for versioning policy enforcement
     |*** Revision: 1.0: Initial version
     |*****************************************************
    +-->

<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:func="http://exslt.org/functions"
        xmlns:dpe="http://www.datapower.com/extensions"
        xmlns:dppolicy="http://www.datapower.com/policy"
        xmlns:dpconfig="http://www.datapower.com/param/config"
        xmlns:dpfunc="http://www.datapower.com/extensions/functions"
        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:itso="http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11"
        extension-element-prefixes="dpe"
        exclude-result-prefixes="dpe dpconfig dppolicy wsp itso">

    <!-- ******************************************************* -->
    <!-- (1) Declare the policy domain this stylesheet implements -->
    <!-- ******************************************************* -->
```

```
    <dpe:summary xmlns="">
        <!-- Comment the following line to prevent the policy domain from being processed
-or- remove the comment if you want to process the policy domain. -->

<dppolicy:domain>http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11</dppolicy
:domain>
        <operation>xform</operation>
        <description>Implements versioning policy assertions for IBM Redbooks Travel
Company</description>
    </dpe:summary>

    <!-- ************************************************************** -->
    <!-- Declare any required DataPower Policy Parameters that are
         required for binding and associating policy assertions
         to configuration                                              -->
    <!-- ************************************************************** -->

    <!-- Specify the WSRR Server to use to retrieve WSDL files for 'validate' actions -->
    <dpe:param name="dpconfig:WSRR-Server" type="dmReference" reftype="WSRRServer"
xmlns="">

<dppolicy:assertion>{http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11}Versi
oning</dppolicy:assertion>
        <display>WSRR-Server</display>
        <description>Specify which WSRR server to use to retrieve WSDL files for
validations based on WSDL</description>
    </dpe:param>

    <!-- the policy domain namespace the stylesheet is executed for -->
    <xsl:variable name="seqno"
select="/dppolicy:request/dppolicy:header/dppolicy:SequenceNo"/>
    <xsl:variable name="nsuri"
select="/dppolicy:request/dppolicy:sequence/DomainNamespace[position()=$seqno]/@uri"/>

    <!-- The following global variables represent the input document -->
    <!-- header with aux information -->
    <xsl:variable name="header" select="/dppolicy:request/dppolicy:header"/>
    <!-- configured policy bindings defined as dpe:param above (Policy Parameters)-->
    <xsl:variable name="bindings" select="/dppolicy:request/dppolicy:bindings"/>
    <!-- ws-policy alternative -->
    <xsl:variable name="policy" select="/dppolicy:request/dppolicy:policy"/>
    <!-- previously generated configuration for this policy alternative -->
    <xsl:variable name="configuration" select="/dppolicy:request/dppolicy:configuration"/>
    <!-- general notepad to pass information between processing steps of one alternative
-->
    <xsl:variable name="notepad" select="/dppolicy:request/dppolicy:notepad"/>

    <!-- helper variables -->
    <xsl:variable name="rD" select="$header/dppolicy:RequestDomain"/>
    <xsl:variable name="lT" select="$header/dppolicy:LogType"/>
    <xsl:variable name="lC" select="$header/dppolicy:LogClass"/>
    <xsl:variable name="rO" select="$header/dppolicy:RequestObject"/>

    <!-- *** Global variables for our versioning implementation *** -->
    <!-- WSRR protocol and hostname -->
    <!-- Declaration of the WSRR Server prefix variable. This varibale is based on the
WSRRServer configuration parameter. -->
    <xsl:variable name="wsrr_prefix" select="concat('wsrr://',$bindings/WSRR-Server,'/')"/>

    <!--+
```

```
      |*****************************
      |*** Matching Template
      |*** Element: ROOT
      |*****************************
      +-->
    <xsl:template match="/">

    <!-- Log -->
      <xsl:message dpe:priority="error">Entering the root template</xsl:message>

        <!-- Get the Input context name -->
        <xsl:choose>
            <xsl:when test="string-length($notepad/shared-context/in-context) > 0">
                <dpe:set-local-variable name="'input-context'"
value="$notepad/shared-context/in-context"/>
                <xsl:message dpe:priority="debug" dpe:domain="{$rD}" dpe:type="{$lT}"
dpe:class="{$lC}" dpe:object="{$rO}">Input context=<xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:message>
            </xsl:when>
            <xsl:otherwise>
                <xsl:message terminate="yes" dpe:priority="error" dpe:domain="{$rD}"
dpe:type="{$lT}" dpe:class="{$lC}" dpe:object="{$rO}">Cannot find the input
context</xsl:message>
            </xsl:otherwise>
        </xsl:choose>

        <!-- Get the Output context name -->
        <xsl:choose>
            <xsl:when test="string-length($notepad/shared-context/out-context) > 0">
                <dpe:set-local-variable name="'output-context'"
value="$notepad/shared-context/out-context"/>
                <xsl:message dpe:priority="debug" dpe:domain="{$rD}" dpe:type="{$lT}"
dpe:class="{$lC}" dpe:object="{$rO}">Output context=<xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:message>
            </xsl:when>
            <xsl:otherwise>
                <xsl:message terminate="yes" dpe:priority="error" dpe:domain="{$rD}"
dpe:type="{$lT}" dpe:class="{$lC}" dpe:object="{$rO}">Cannot find the output
context</xsl:message>
            </xsl:otherwise>
        </xsl:choose>

       <!-- process single alternative -->
        <xsl:apply-templates select="$policy/*[local-name()='All']"/>

         <!-- Logs -->
      <xsl:message dpe:priority="error">Exiting the root template for
versioning</xsl:message>

    </xsl:template>

    <!--+
      |*****************************
      |*** Matching Template
      |*** Element: All
      |*****************************
      +-->
   <!-- process each assertion in the alternative -->
    <xsl:template match="*[local-name()='All']">
        <xsl:apply-templates mode="assertion"/>
```

```
    </xsl:template>

     <!-- my domain assertions -->

    <!--+
       |*****************************
       |*** Matching Template
       |*** Element: itso:Versioning
       |*** Mode: assertion
       |*****************************
      +-->
    <xsl:template mode="assertion" match="itso:Versioning">
        <xsl:message dpe:priority="error">Enter template Versioning: PolicyID=<xsl:value-of
select="$header/dppolicy:PolicyID"/></xsl:message>

        <!-- The framework passed in a policykey value, this must be added to the
             processing actions in order to properly instrument source policy origin
             information into monitoring subsystem and UI presentation -->
        <xsl:variable name="policyKey" select="./@policy-key"/>

        <!-- Log -->
      <xsl:message dpe:priority="error"><xsl:value-of select="concat('*** WSRR PREFIX:
wsrr://',$bindings/WSRR-Server,'/')"/></xsl:message>

        <!-- ************************************************************** -->
        <!-- (5) The configuration for assertion Versioning                 -->
        <!-- ************************************************************** -->
        <!-- generate processing action to execute (assertionNo = order of processing) -->
        <xsl:variable name="config">

        <!-- Request processing rule: Validate incoming request + Xform + Validate outgoing
request + Route -->
            <dppolicy:config uri="{$nsuri}" assertionNo="{position()}"
direction="{'request-rule'}">

             <!-- ClientWSDL conditional validation (incoming request) -->
            <xsl:if test="string-length(./itso:Validate/itso:ClientWSDL/text()) != 0">
                <xsl:message dpe:priority="error">Versioning: ClientWSDL
validation</xsl:message>
                    <xsl:variable name="actionNameValidate"
select="concat($header/dppolicy:PolicyID,'-validate-req-', position(),'-ClientWSDL')"/>
                    <xsl:element name="StylePolicyAction">
                        <xsl:attribute name="name"><xsl:value-of
select="$actionNameValidate"/></xsl:attribute>
                        <xsl:element
name="Type"><xsl:text>validate</xsl:text></xsl:element>
                        <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                        <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                        <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                        <!-- WSDL file that validates incoming request messages -->
                        <xsl:element name="WsdlURL"><xsl:value-of
select="concat($wsrr_prefix,./itso:Validate/itso:ClientWSDL/text(),'/')"/></xsl:element>
                        <xsl:element
name="SOAPValidation"><xsl:text>body</xsl:text></xsl:element>
                    </xsl:element>
                </xsl:if>
```

```
                <!-- Data mediation -->
                 <xsl:choose>
                 <xsl:when test="string-length(./itso:Transform/text()) != 0">
                 <!-- Message transformation (Vm to Vn)-->
                       <xsl:variable name="actionNameXform"
select="concat($header/dppolicy:PolicyID,'-xform-resp-', position(),'-Transform')"/>
                        <xsl:element name="StylePolicyAction">
                            <xsl:attribute name="name"><xsl:value-of
select="$actionNameXform"/></xsl:attribute>
                            <xsl:element
name="Type"><xsl:text>xform</xsl:text></xsl:element>
                            <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                            <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                            <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                            <xsl:element name="Transform"><xsl:value-of
select="./itso:Transform/text()"/></xsl:element>
                            <xsl:element
name="OutputType"><xsl:text>default</xsl:text></xsl:element>
                        </xsl:element>
                 </xsl:when>
                 <xsl:otherwise>
                 <!-- Identity transformation is used if no Transform is provided -->
                       <xsl:variable name="actionNameXform"
select="concat($header/dppolicy:PolicyID,'-xform-resp-', position(),'-Transform')"/>
                        <xsl:element name="StylePolicyAction">
                            <xsl:attribute name="name"><xsl:value-of
select="$actionNameXform"/></xsl:attribute>
                            <xsl:element
name="Type"><xsl:text>xform</xsl:text></xsl:element>
                            <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                            <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                            <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                            <!-- identity transformation -->
                            <xsl:element
name="Transform"><xsl:text>store:///identity.xsl</xsl:text></xsl:element>
                            <xsl:element
name="OutputType"><xsl:text>default</xsl:text></xsl:element>
                        </xsl:element>
                 </xsl:otherwise>
                 </xsl:choose>

          <!-- ServerWSDL conditional validation (outgoing request) -->
            <xsl:if test="string-length(./itso:Validate/itso:ServerWSDL/text()) != 0">
                <xsl:message dpe:priority="error">Versioning: ServerWSDL
validation</xsl:message>
                    <xsl:variable name="actionNameValidate"
select="concat($header/dppolicy:PolicyID,'-validate-req-', position(),'-ServerWSDL')"/>
                    <xsl:element name="StylePolicyAction">
                        <xsl:attribute name="name"><xsl:value-of
select="$actionNameValidate"/></xsl:attribute>
                        <xsl:element
name="Type"><xsl:text>validate</xsl:text></xsl:element>
                        <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
```

```
                        <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                        <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                        <!-- WSDL file that validates outgoing request messages -->
                        <xsl:element name="WsdlURL"><xsl:value-of
select="concat($wsrr_prefix,./itso:Validate/itso:ServerWSDL/text(),'/')"/></xsl:element>
                        <xsl:element
name="SOAPValidation"><xsl:text>body</xsl:text></xsl:element>
                    </xsl:element>
                </xsl:if>

                <!-- Setvar action for routing -->
                <xsl:variable name="actionNameSetvar"
select="concat($header/dppolicy:PolicyID,'-setvar-req-', position(),'-Route')"/>
                 <xsl:element name="StylePolicyAction">
                    <xsl:attribute name="name"><xsl:value-of
select="$actionNameSetvar"/></xsl:attribute>
                    <xsl:element name="Type"><xsl:text>setvar</xsl:text></xsl:element>
                    <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                    <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                    <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                    <xsl:element
name="NamedInOutLocationType"><xsl:text>default</xsl:text></xsl:element>
                    <xsl:element
name="Variable"><xsl:text>var://service/routing-url</xsl:text></xsl:element>
                    <!-- value of the target endpoint -->
                    <xsl:element name="Value"><xsl:value-of
select="./itso:Route/text()"/></xsl:element>
                 </xsl:element>

        </dppolicy:config>

        <!-- Response processing rule: Validate incoming response + Xform + Validate
outgoing response -->
        <dppolicy:config uri="{$nsuri}" assertionNo="{position()}"
direction="{'response-rule'}">

                <!-- ServerWSDL conditional validation (incoming response) -->
            <xsl:if test="string-length(./itso:Validate/itso:ServerWSDL/text()) != 0">
                <xsl:message dpe:priority="error">Versioning: ServerWSDL
validation</xsl:message>
                    <xsl:variable name="actionNameValidate"
select="concat($header/dppolicy:PolicyID,'-validate-resp-', position(),'-ServerWSDL')"/>
                    <xsl:element name="StylePolicyAction">
                        <xsl:attribute name="name"><xsl:value-of
select="$actionNameValidate"/></xsl:attribute>
                        <xsl:element
name="Type"><xsl:text>validate</xsl:text></xsl:element>
                        <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                        <xsl:element name="Input"><xsl:text>INPUT</xsl:text></xsl:element>
                        <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                        <!-- WSDL file that validates incoming response messages -->
                        <xsl:element name="WsdlURL"><xsl:value-of
select="concat($wsrr_prefix,./itso:Validate/itso:ServerWSDL/text(),'/')"/></xsl:element>
```

```
                        <xsl:element
name="SOAPValidation"><xsl:text>body</xsl:text></xsl:element>
                    </xsl:element>
                </xsl:if>

            <!-- Data mediation -->
                <xsl:choose>
                <xsl:when test="string-length(./itso:Transform/text()) != 0">
                <!-- Message transformation (Vm to Vn)-->
                     <xsl:variable name="actionNameXform"
select="concat($header/dppolicy:PolicyID,'-xform-resp-', position(),'-Transform')"/>
                    <xsl:element name="StylePolicyAction">
                        <xsl:attribute name="name"><xsl:value-of
select="$actionNameXform"/></xsl:attribute>
                        <xsl:element
name="Type"><xsl:text>xform</xsl:text></xsl:element>
                        <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                        <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                        <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                        <xsl:element name="Transform"><xsl:value-of
select="./itso:Transform/text()"/></xsl:element>
                        <xsl:element
name="OutputType"><xsl:text>default</xsl:text></xsl:element>
                    </xsl:element>
                </xsl:when>
                <xsl:otherwise>
                <!-- Identity transformation is used if no Transform is provided -->
                     <xsl:variable name="actionNameXform"
select="concat($header/dppolicy:PolicyID,'-xform-resp-', position(),'-Transform')"/>
                    <xsl:element name="StylePolicyAction">
                        <xsl:attribute name="name"><xsl:value-of
select="$actionNameXform"/></xsl:attribute>
                        <xsl:element
name="Type"><xsl:text>xform</xsl:text></xsl:element>
                        <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                        <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                        <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                        <!-- identity transformation -->
                        <xsl:element
name="Transform"><xsl:text>store:///identity.xsl</xsl:text></xsl:element>
                        <xsl:element
name="OutputType"><xsl:text>default</xsl:text></xsl:element>
                    </xsl:element>
                </xsl:otherwise>
                </xsl:choose>

          <!-- ClientWSDL conditional validation (outgoing response) -->
            <xsl:if test="string-length(./itso:Validate/itso:ClientWSDL/text()) != 0">
                <xsl:message dpe:priority="error">Versioning: ClientWSDL
validation</xsl:message>
                    <xsl:variable name="actionNameValidate"
select="concat($header/dppolicy:PolicyID,'-validate-resp-', position(),'-ClientWSDL')"/>
                    <xsl:element name="StylePolicyAction">
```

```
                        <xsl:attribute name="name"><xsl:value-of
select="$actionNameValidate"/></xsl:attribute>
                        <xsl:element
name="Type"><xsl:text>validate</xsl:text></xsl:element>
                        <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                        <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                        <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                        <!-- WSDL file that validates outgoing response messages -->
                        <xsl:element name="WsdlURL"><xsl:value-of
select="concat($wsrr_prefix,./itso:Validate/itso:ClientWSDL/text(),'/')"/></xsl:element>
                        <xsl:element
name="SOAPValidation"><xsl:text>body</xsl:text></xsl:element>
                    </xsl:element>
                </xsl:if>

        </dppolicy:config>
        <!-- Log -->
           <xsl:message dpe:priority="error">Exit template Versioning</xsl:message>
      </xsl:variable>

        <!-- Send the config to the console, whatever will fit (roughtly 3k) -->
        <xsl:message dpe:priority="error">Exit template Versioning - Config:
           <xsl:copy-of select="$config"/>
        </xsl:message>

        <!-- Send the config to the temporary://  -->
        <xsl:variable name="filename"
select="concat('Versioning-conf-',$header/dppolicy:PolicyID, '.xml')"/>
        <dpe:dump-nodes file="$filename" nodes="$config"/>

        <!-- Send the config to output -->
        <xsl:copy-of select="$config"/>

        <!-- Log -->
      <xsl:message dpe:priority="error">Exiting the Versioning template</xsl:message>

    </xsl:template>

    <!--+
       |*****************************
       |*** Matching Template
       |*** Element: text()
       |*****************************
       +-->
    <xsl:template match="text()"/>

</xsl:stylesheet>
```

The directory that contains this custom mapping is the `store:///polices/custom` directory, as shown in Figure 5-29.

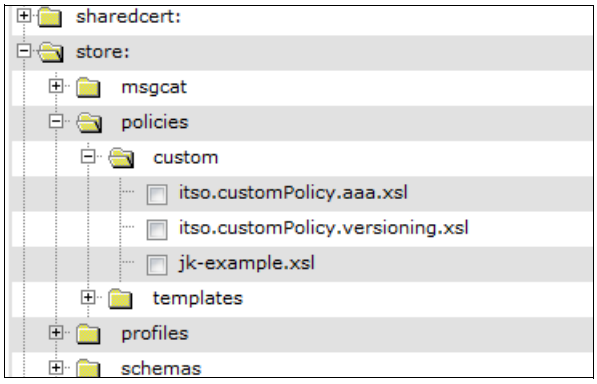


Figure 5-29 Directory for custom policy XSL style sheet

To verify that the custom versioning policy style sheet is parsed properly, review the `Policy Domains Supported` interface. You can access this interface through the Status menu by selecting **Status** → **Web-Service** → **Policy Domains Supported.** This status interface provides the information shown in Figure 5-30.

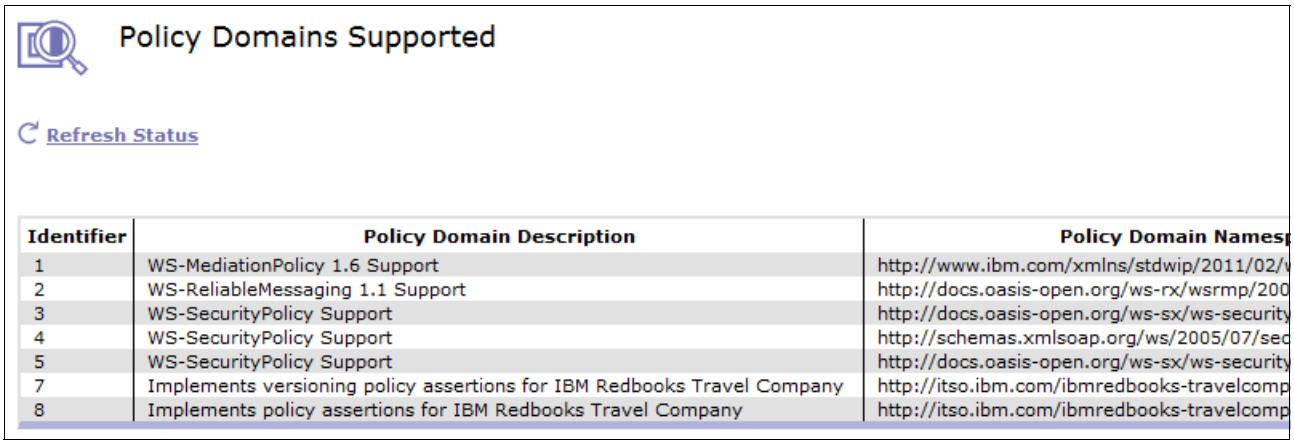


Figure 5-30 Policy Domains Supported user interface

5.7 Attaching the custom versioning policy to a specific service

You may attach the custom versioning policy to a specific service in WSRR, as described in this section. Also described are the design and runtime activities to enforce the custom versioning policy that is attached at design time and enforced at run time.

5.7.1 Attaching the custom versioning policy in WSRR at design-time

Use the following steps to attach a custom versioning policy to the Pricing service in WSRR.

1. Connect to the WSRR Governance Master Business Space by using your credentials.
2. Switch to the following space:

   `IBM Redbooks Travel Service Registry for Operations`

3. Click **Load Documents**, in the Service Registry Actions widget, as shown in Figure 5-31.

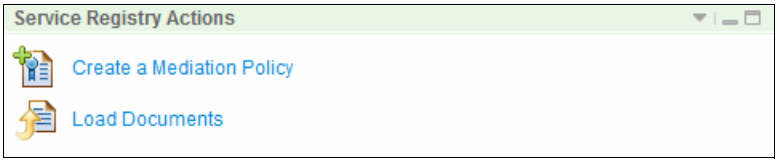


Figure 5-31 Load a custom policy document in WSRR

4. In the Load Documents window, click **Browse** to locate the custom policy to be loaded and select the **Policy** document type.

   The versioning policy has a version 2.0 in our example, as shown in Figure 5-32.

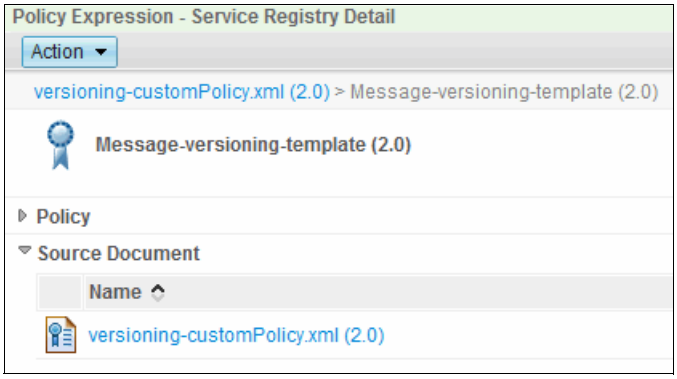


Figure 5-32 Custom versioning policy in WSRR

5. To attach the custom versioning policy, follow the steps in 13.4, “Custom policy attachment in WSRR” on page 383. Use the following information to help you with this policy attachment:
   - The name of the custom versioning policy is `versioning-CustomPolicy.xml`.
   - The SLD on which the custom versioning policy must be attached is the SLD of the `Service Version 1.0` of the `Pricing` service, as shown in Figure 5-11 on page 134.

   Figure 5-33 shows an overview of the versioning policy attachment.

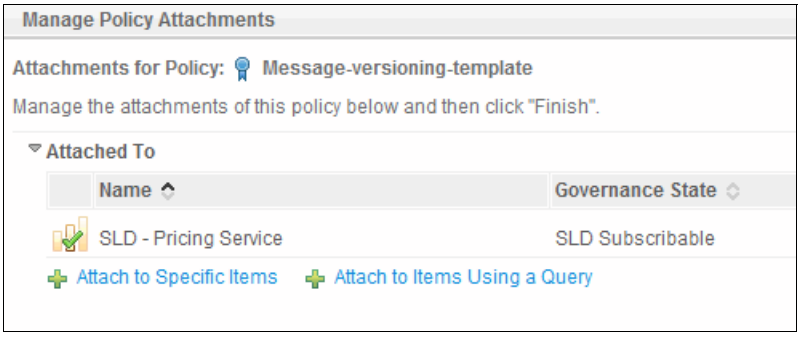


Figure 5-33 Versioning policy attachment at the SLD of the Service Version1.0

It is important to deploy this policy in production so the versioning policy attachment can affect the DataPower production appliance.

6. Select the Pricing Service Version 1.0, accessible through the Pricing Business Service, as shown in Figure 5-34.

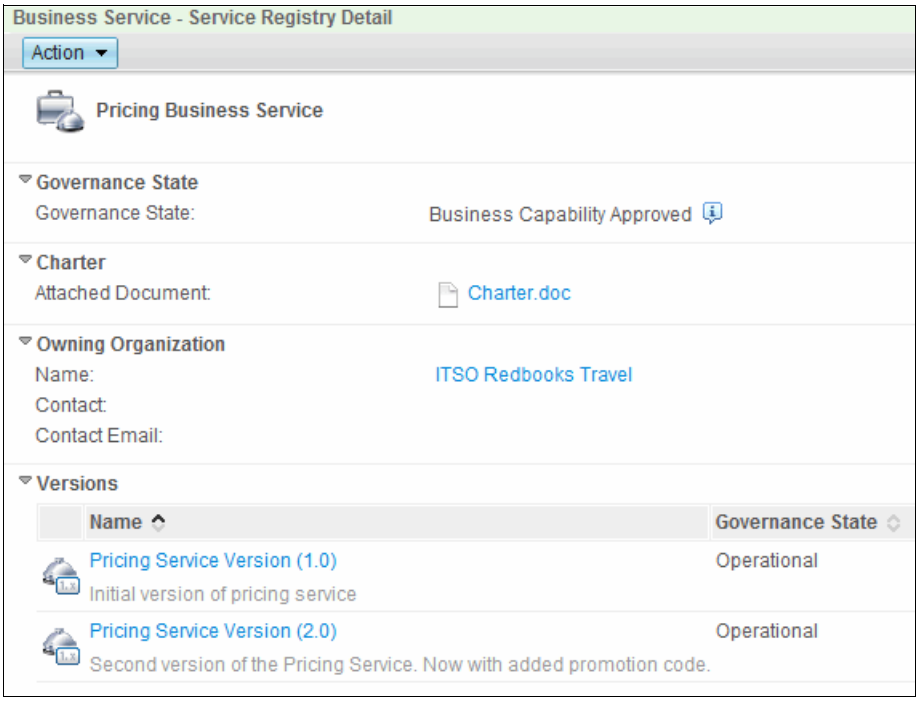


Figure 5-34 Pricing Business Service

7. Click the **Pricing Service Version (1.0)** link, in the Service Registry Detail widget.
8. Select **Action** → **Redefine**, as shown in Figure 5-35.

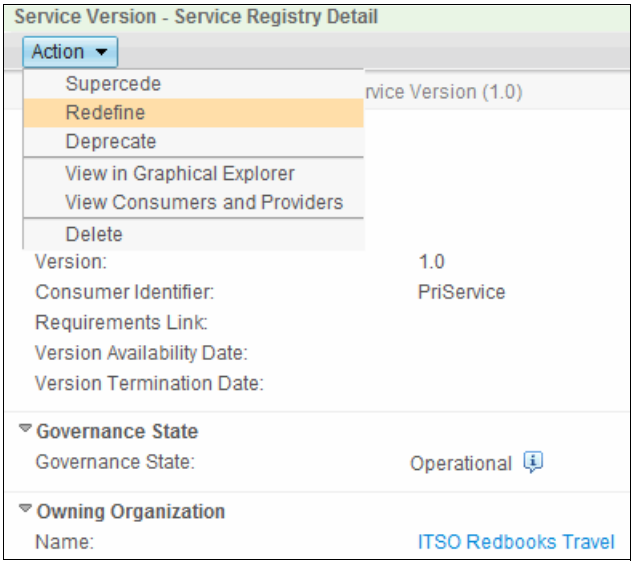


Figure 5-35 Redefining the Pricing Service Version 1.0

9. Click **OK** to go on the production deployment process, as shown in Figure 5-36.

Figure 5-36 Validated step

10. Select **Action** → **Propose Production Deployment**, as shown in Figure 5-37.

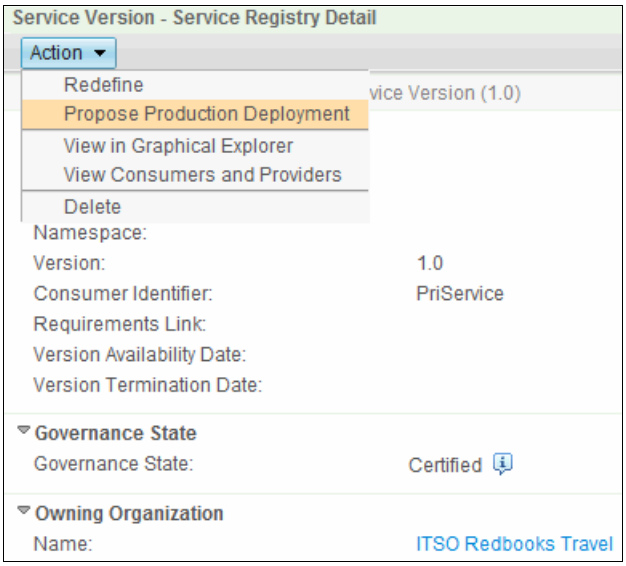


Figure 5-37 Proposing the Pricing Service Version 1.0 for deployment

11. Click **OK** to get to the production deployment process window.

12. Select **Action** → **Approve Production Deployment**, as shown in Figure 5-38.

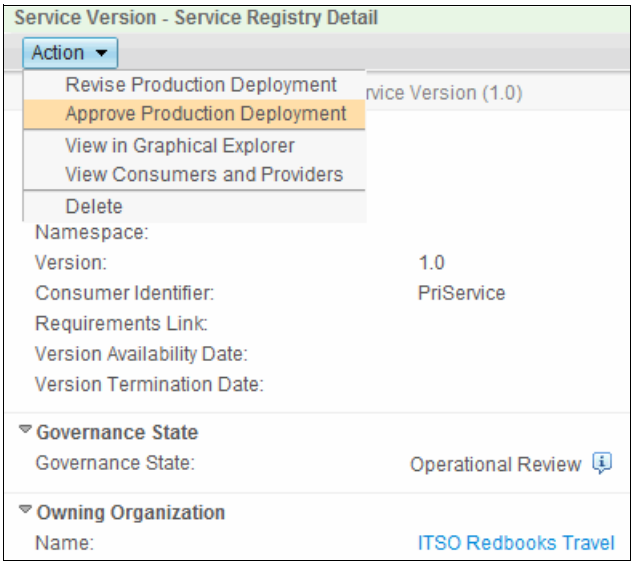


Figure 5-38 Approving the deployment of the Pricing Service Version 1.0

13. Click **OK** to validate the production deployment process.

Now, the custom versioning policy is deployed on the WSRR production instance.

Create a Web Service Proxy on the DataPower side in order to retrieve the following elements:

- The two WSDLs files of the Pricing service
- The custom versioning policy attached at the SLD level
- Information about the attachment of the custom versioning policy

5.7.2 Creating a saved search in WSRR at design-time

Before creating the minimum configuration on DataPower, a saved search can be created in WSRR to be able to retrieve the different versions of the `Pricing` service on DataPower. This WSRR saved search can be used on a DataPower Web Service Proxy to retrieve the WSDL files required to create a virtualization layer based on the Pricing service versions.

Use the following steps to create a saved search in WSRR to retrieve the versions of the Pricing service.

1. Connect to the production instance of WSRR, through the service registry user interface, by using your credentials.
2. Go to the **Administrator** perspective.
3. Select **Actions** → **Query Wizard**, as shown in Figure 5-39.

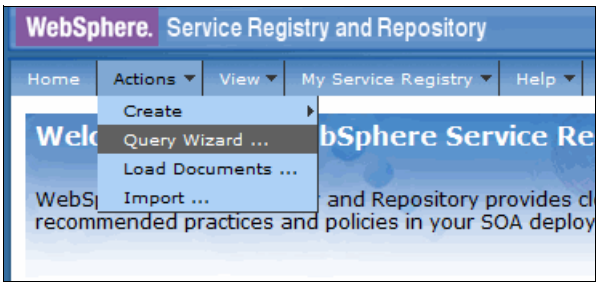


Figure 5-39 Select Query Wizard

4. In the Prepare to Run a Query panel (Figure 5-40), select **WSDL Document** as the type of entity to query and click **Next**.

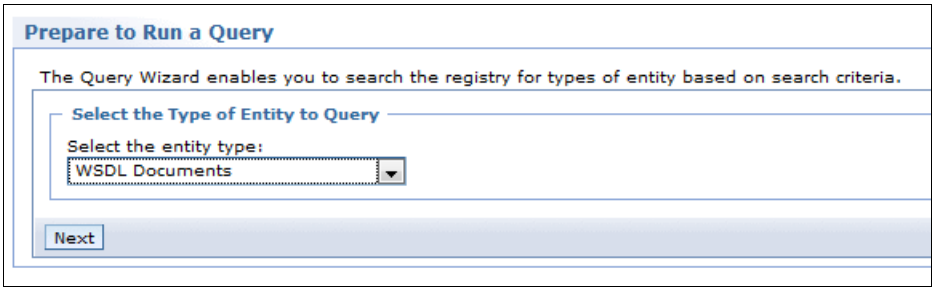


Figure 5-40 Prepare to run query

5. In the panel that is shown in Figure 5-41, enter details of the query. The `*Pricing*` name pattern is used for the query as the WSDL files of the Pricing service starts with the `Pricing` prefix (`PricingService.wsdl` and `PricingServicev2.wsdl`).

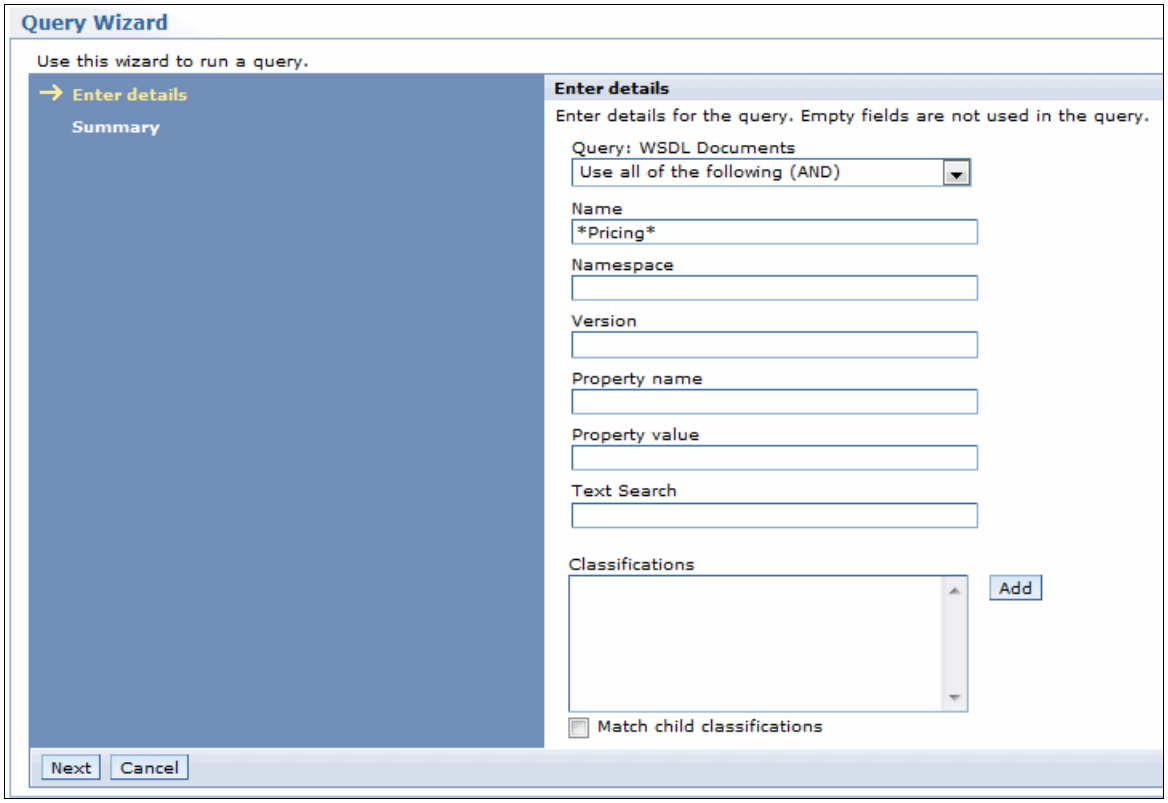


Figure 5-41 Details of the query to get Pricing service WSDLs

6. Click **Next** and then click **Finish** to display the current WSDL files of the various Pricing service versions, as shown in Figure 5-42.

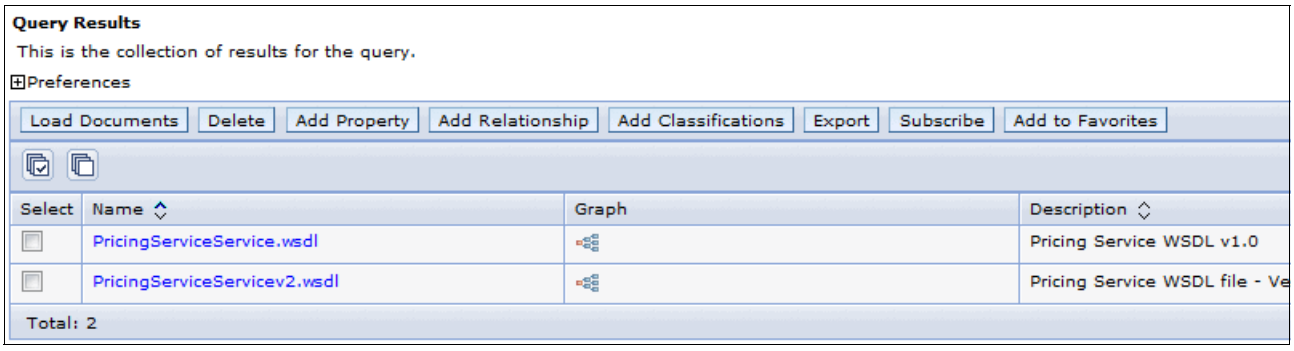


Figure 5-42 WSDL files of the Pricing service versions

7. WSRR allows you to save the search you executed. Enter `ITSO_PricingService_WSDLs` in the Name field, and then click **Save**, as shown in Figure 5-43.

Figure 5-43 Saving a search in WSRR

8. On the WSRR home page, the available saved searches are listed, as shown in Figure 5-44.

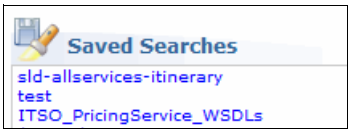


Figure 5-44 List of available saved searches

If a new `PricingServicev3.wsdl` file is added later in WSRR, running the saved search allows you to retrieve this new WSDL file too.

This saved search can be used while creating a subscription from a DataPower Web Service proxy to a WSRR server.

5.7.3 Creating a Web Service Proxy for versioning in DataPower

Use the following steps to design the time tasks that are necessary to enforce the custom versioning policy on the DataPower side at run time:

1. Connect to your DataPower SOA appliance by using your credentials, on a domain that is dedicated to service versioning of the Pricing service (`versioning-saw217` in the Pricing service example).
2. Create a Web Service Proxy object. Choose a name for this service (for example, `wsp_IBMRedbooksTravelCompany_Pricing`).

3. Create a WSRR subscription, based on the following configuration elements:
   – A WSRR Server object. The configuration for this object is shown in Figure 5-45.

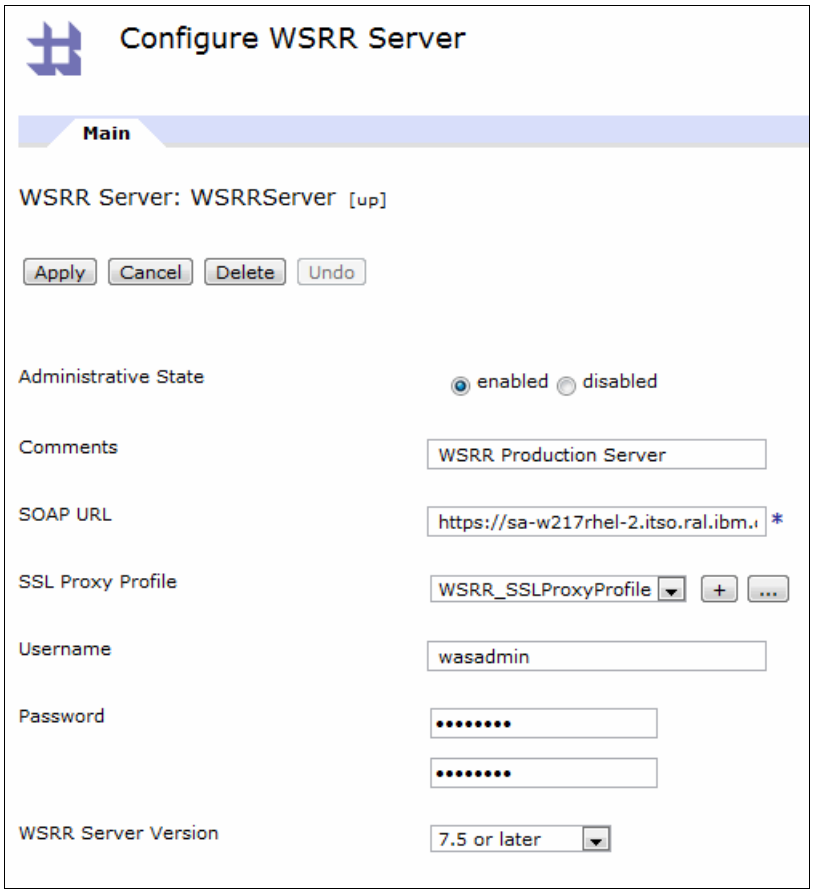


Figure 5-45 WSRR server object in DataPower

– A saved search retrieved from WSRR. This search is created in 5.7.2, “Creating a saved search in WSRR at design-time” on page 164.

Select the **ITSO_PricingService_WSDLs** saved search, as show in Figure 5-46.

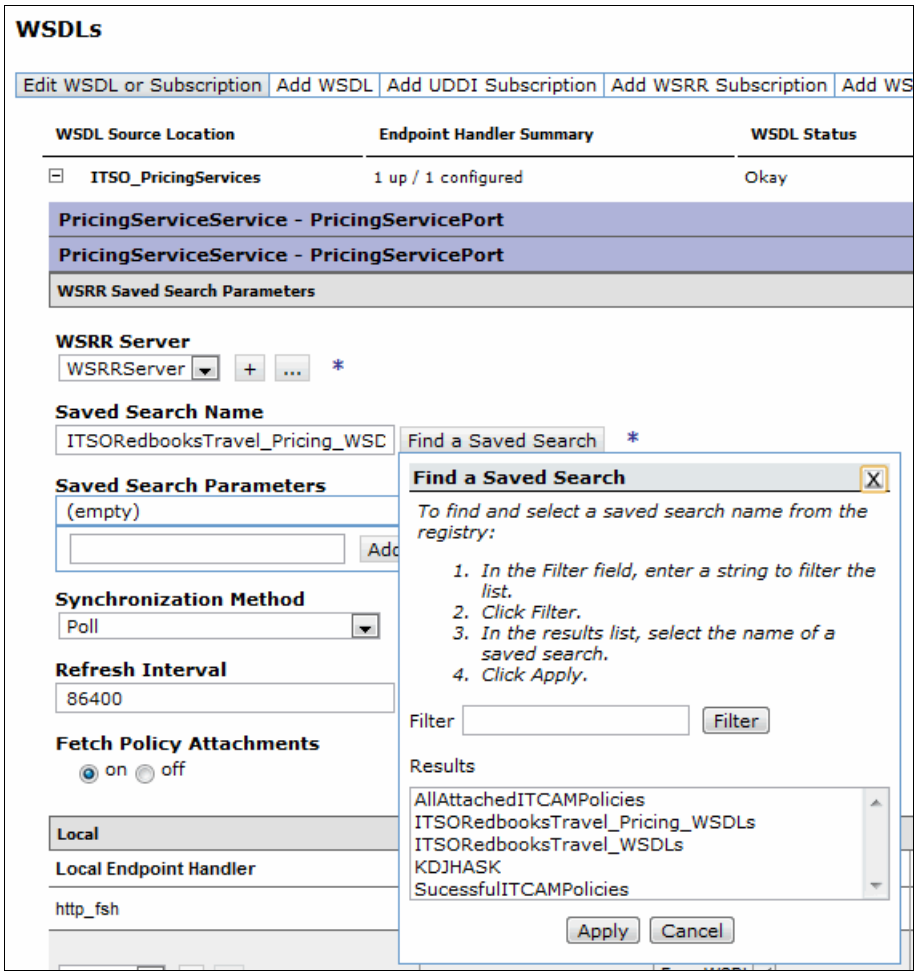


Figure 5-46 List of saved searches in DataPower

4. Create a `WS-Policy Parameter Set` object by clicking the plus sign (**+**) beside the **WS-Policy Parameter Set** property, as shown in Figure 5-47.

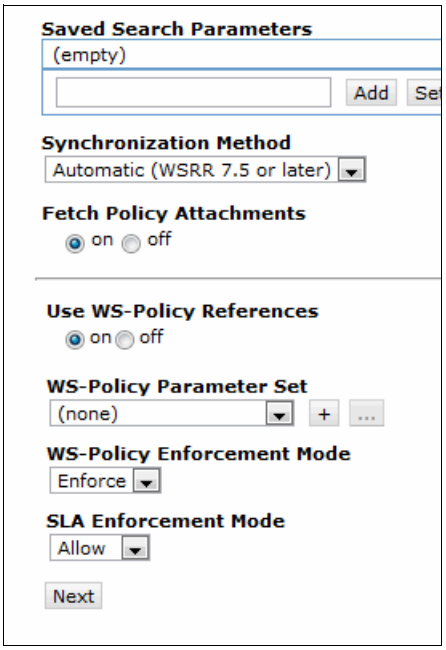


Figure 5-47 Create a WS-Policy parameter set

5. Choose a name for the WS-Policy Parameter Set object (for example, `VersioningPolicyParameterSet`), as shown in Figure 5-48, and provide the following information:
   a. Select the following value from the **Policy Domain** list:

      `http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11`

   b. Select the value **WSRRServer** from the Parameter Value list, as shown in Figure 5-48.

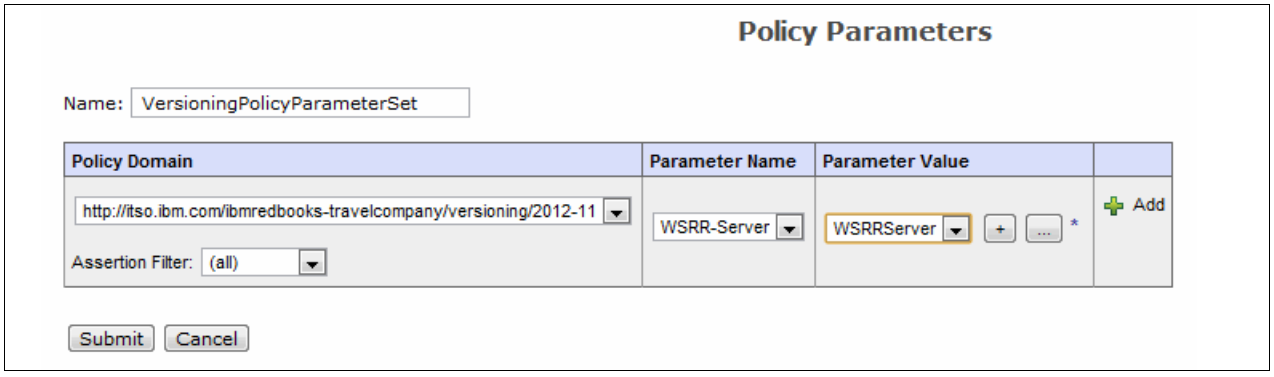


Figure 5-48 WS-Policy Parameter set for the custom versioning policy

6. Click **Add** to add the following parameter to the `VersioningPolicyParameterSet` policy parameter set, as shown in Figure 5-49

   `{http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012-11}WSRR-Server`

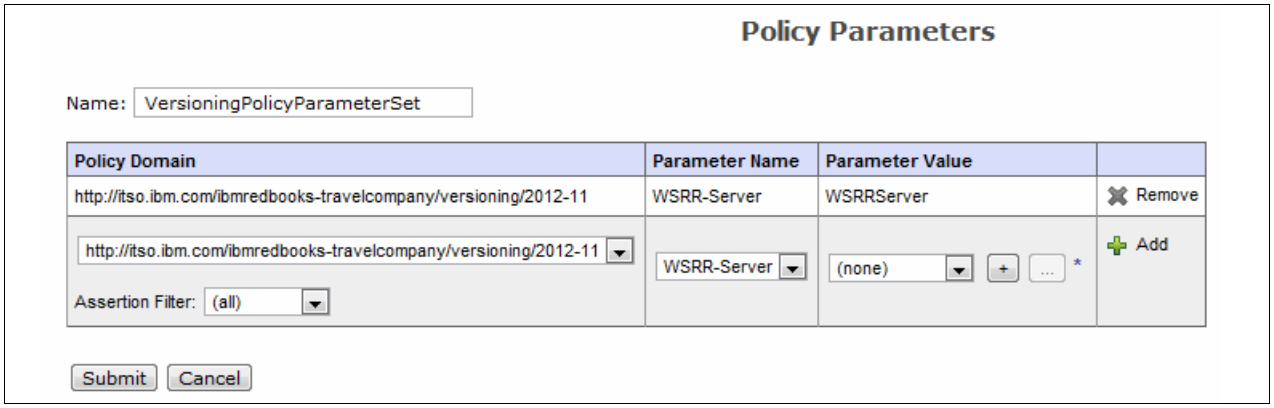


Figure 5-49 Policy parameter added to the VersioningPolicyParameterSet object

For now, the `WSRRServer` value can be retrieved on the custom policy XSL style sheet `itso.customPolicy.versioning.xsl`. This value is prominent to build the URLs to the WSDL files governed in WSRR.

7. Create an HTTP Front Side Handler to specify the interface, protocol, and listening port of the Web Service Proxy virtualization endpoints.

   The Pricing Web Service Proxy includes the following main information:

   – Protocol: `HTTP`
   – Listening port: `4081`
   – HTTP Methods: `GET - POST`
   – Listening interface: `DP.DATA` (defined as a host alias), which is a dedicated interface to data exchanges

The information regarding the HTTP Front Side Handler is shown in Figure 5-50.

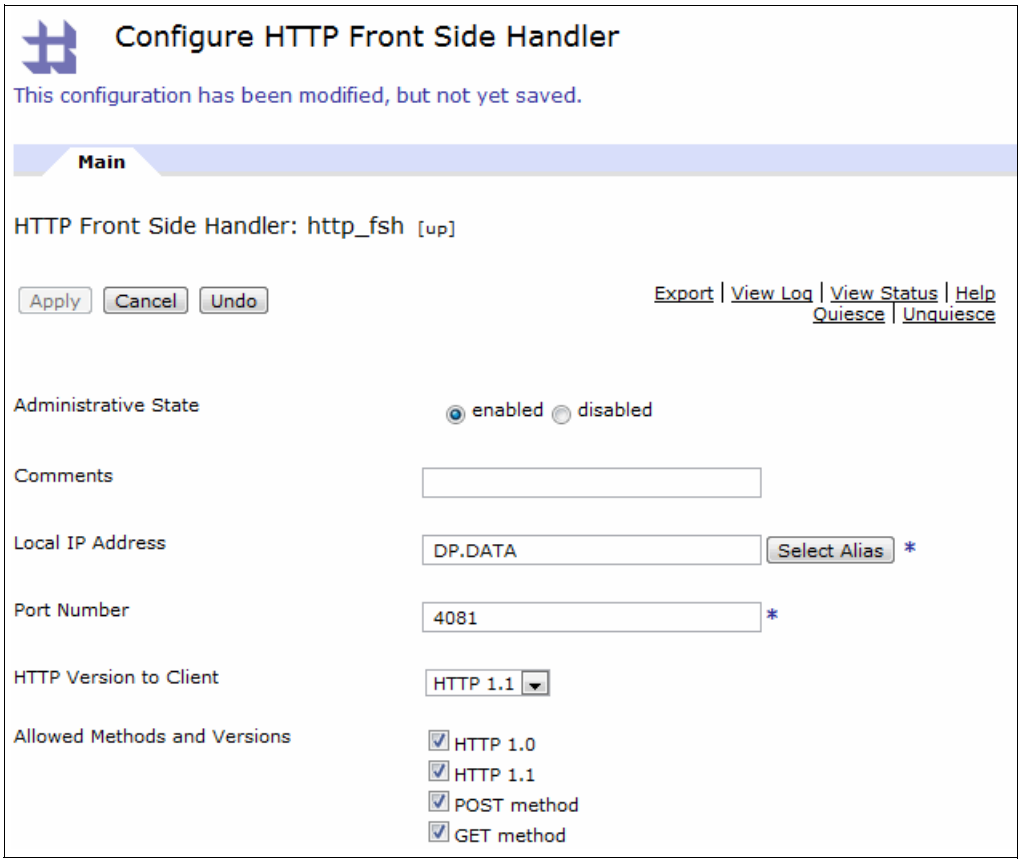


Figure 5-50 Main properties of the HTTP front side handler

Table 5-2 lists the Web-Service operations after the Web Service Proxy is created on DataPower and it retrieved the Pricing service WSDL file and policies.

Table 5-2 Web-Services operations

Operations	Port	Endpoint URI
{http://travel.redbooks.ibm.com/}getPrice	4081	/redbooksTravel/PricingServiceService
{http://travel.redbooks.ibm.com/v2/}getPrice	4081	/redbooksTravelV2/PricingServiceService

The subscription panel of the Web Service Proxy in charge of exposing Pricing service WSDLs is shown in Figure 5-51.

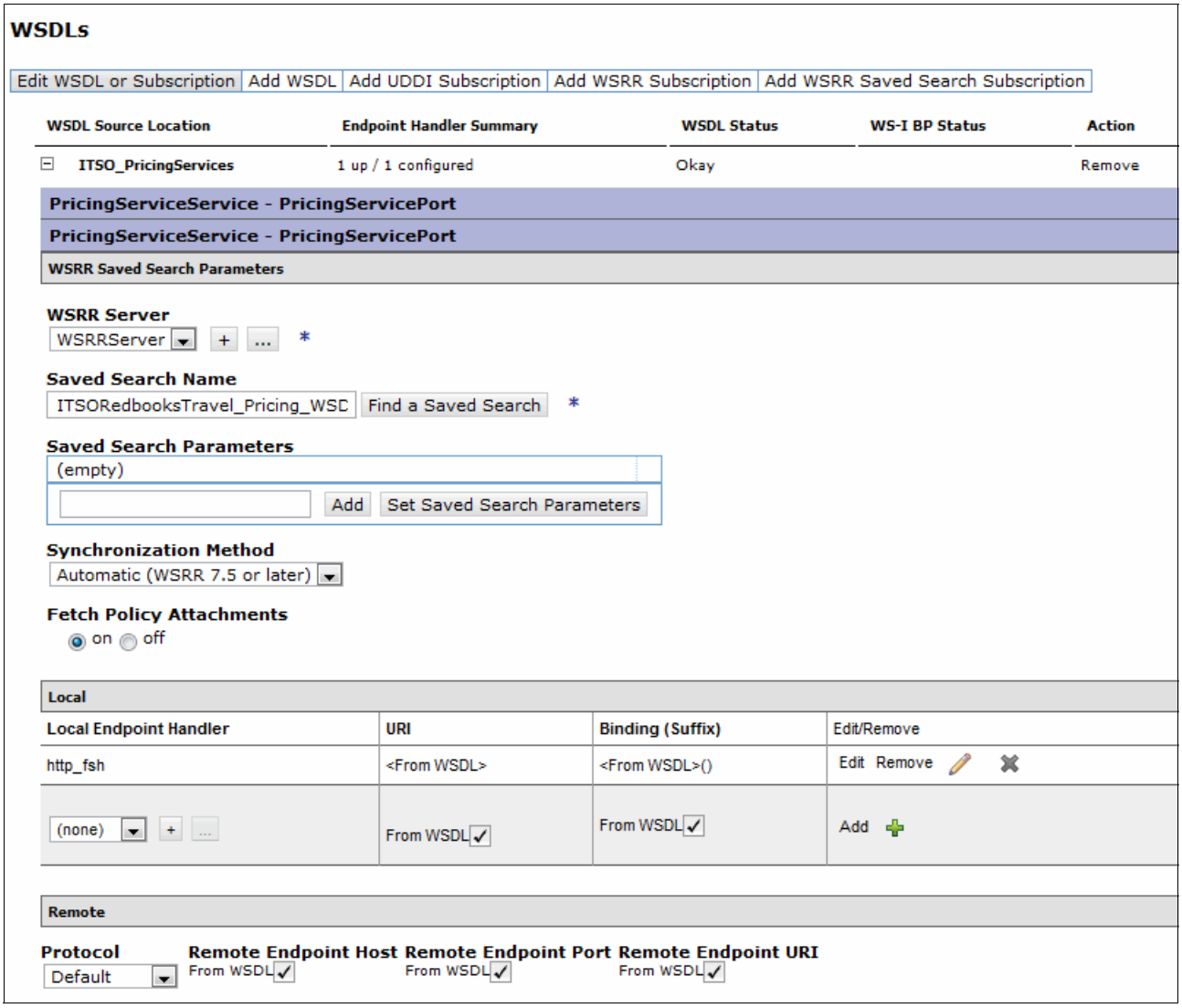


Figure 5-51 Subscription panel of the Web Service Proxy in charge of exposing Pricing service WSDLs

8. Click the **Policy** tab of the `wsp_IBMRedbooksTravelCompany` Web Service Proxy.
9. Look for the WSDLs of the Pricing service for which a versioning must be enforced. In this case, only Pricing service version 1.0 must be enforced.

10. Select the green check boxes (under Local Value) at the right side of the WSDL bsrURI, as shown in Figure 5-52.

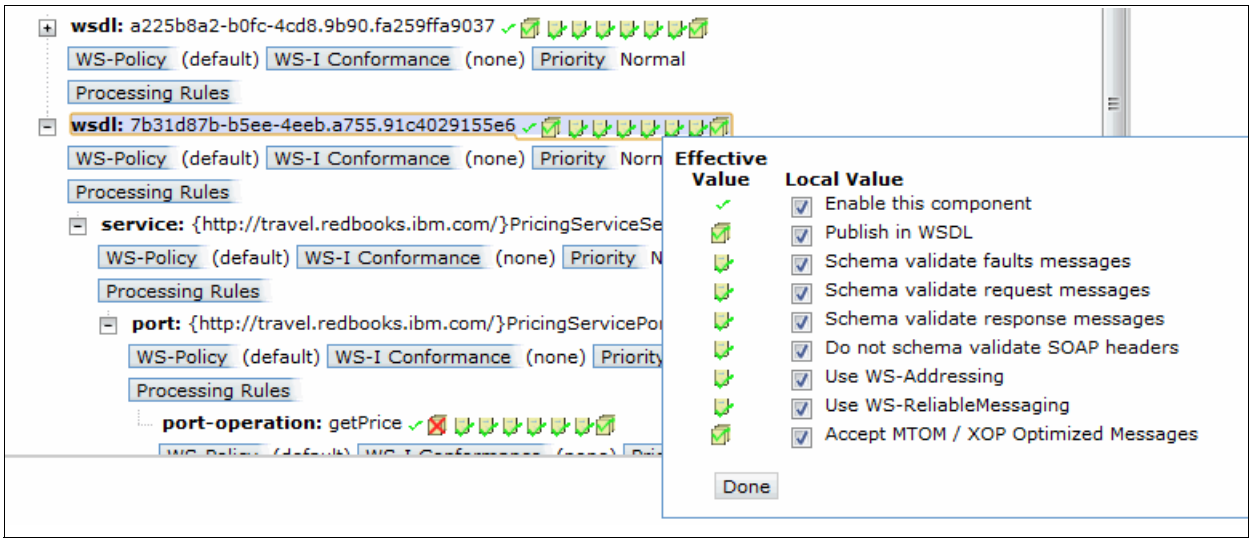


Figure 5-52 Automatic validations configuration at the WSDL level

11. Deactivate the request and response validations because these validations are done as defined in the custom policy for versioning. Figure 5-53 shows the deactivation of both request and response messages.

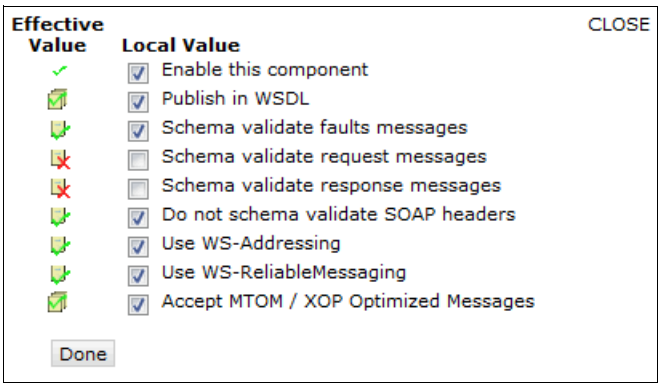


Figure 5-53 Deactivation of request and response validations

Tip: Deactivation of request and response messages can be done before a custom policy for versioning must be enforced for a service.

12. Click **Done**.
13. Click **Apply** at the Web Service Proxy level.
14. Click the **SLA Policy Details** tab of the `wsp_IBMRedbooksTravelCompany` Web Service Proxy.

15. In the SLA Table section of the panel, select the port level of the Pricing service version 1.0, as shown in Figure 5-54.

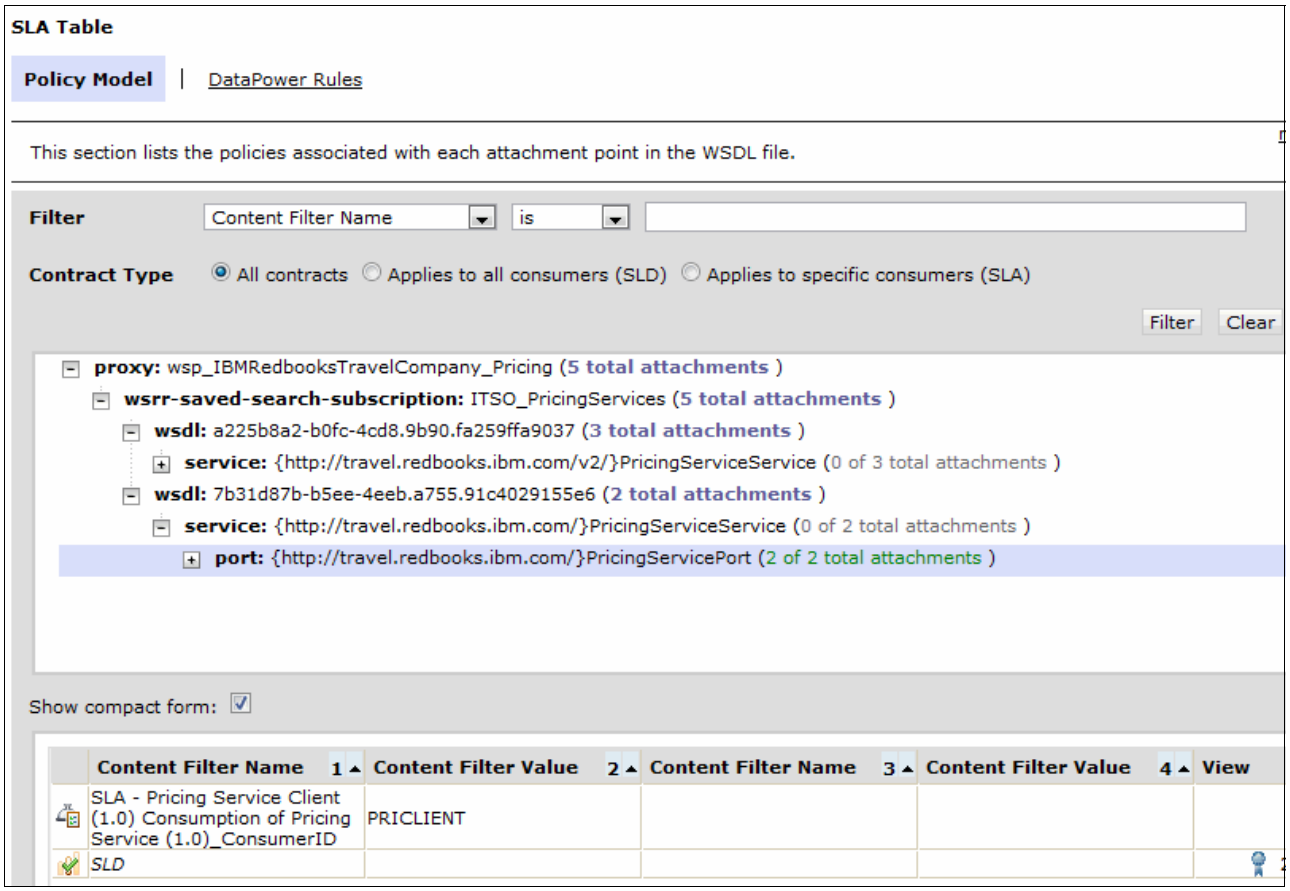


Figure 5-54 Policy model of the Pricing services

16. Click the **SLA** and **SLD** links to display the policy models. At the SLD level, you can see the versioning policy, as shown in Figure 5-55. The presence of a mediation policy is also attached at the SLD level.

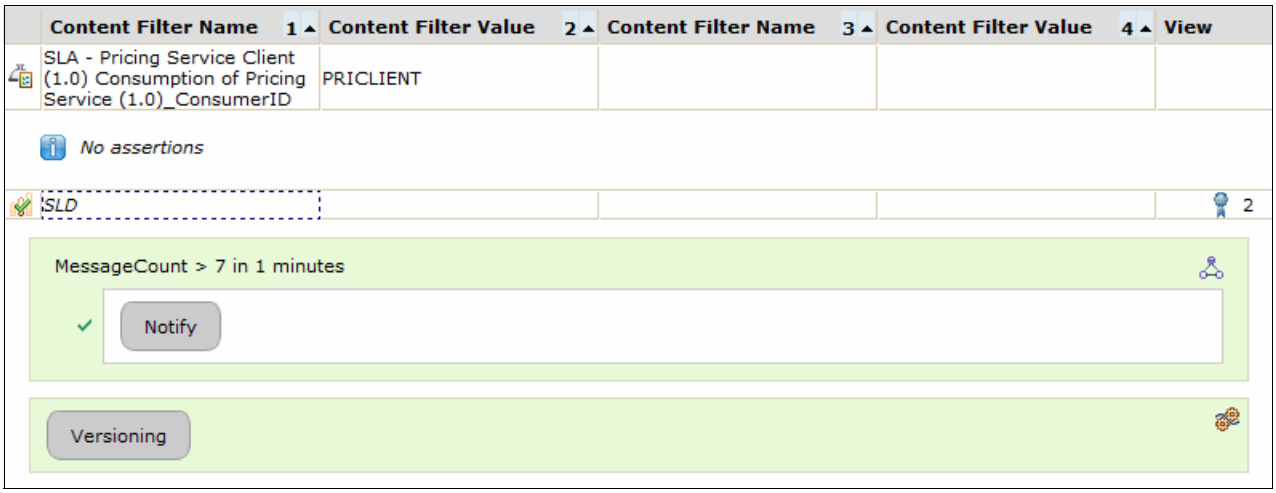


Figure 5-55 Policy model at the SLD level

17. Hover the mouse on the **Versioning** symbol. Details of the custom versioning display, as shown in Figure 5-56.

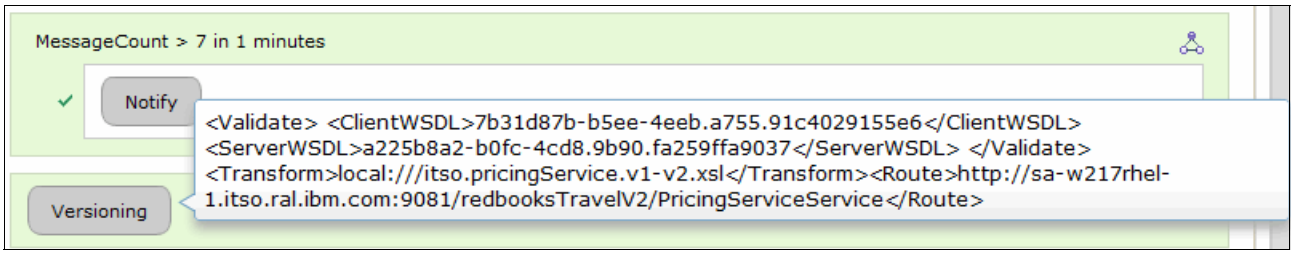


Figure 5-56 Details of the custom versioning policy

18. In the SLA Table section of the panel, click the **DataPower Rules** link and select the port level of the Pricing service version 1.0, as shown in Figure 5-57.

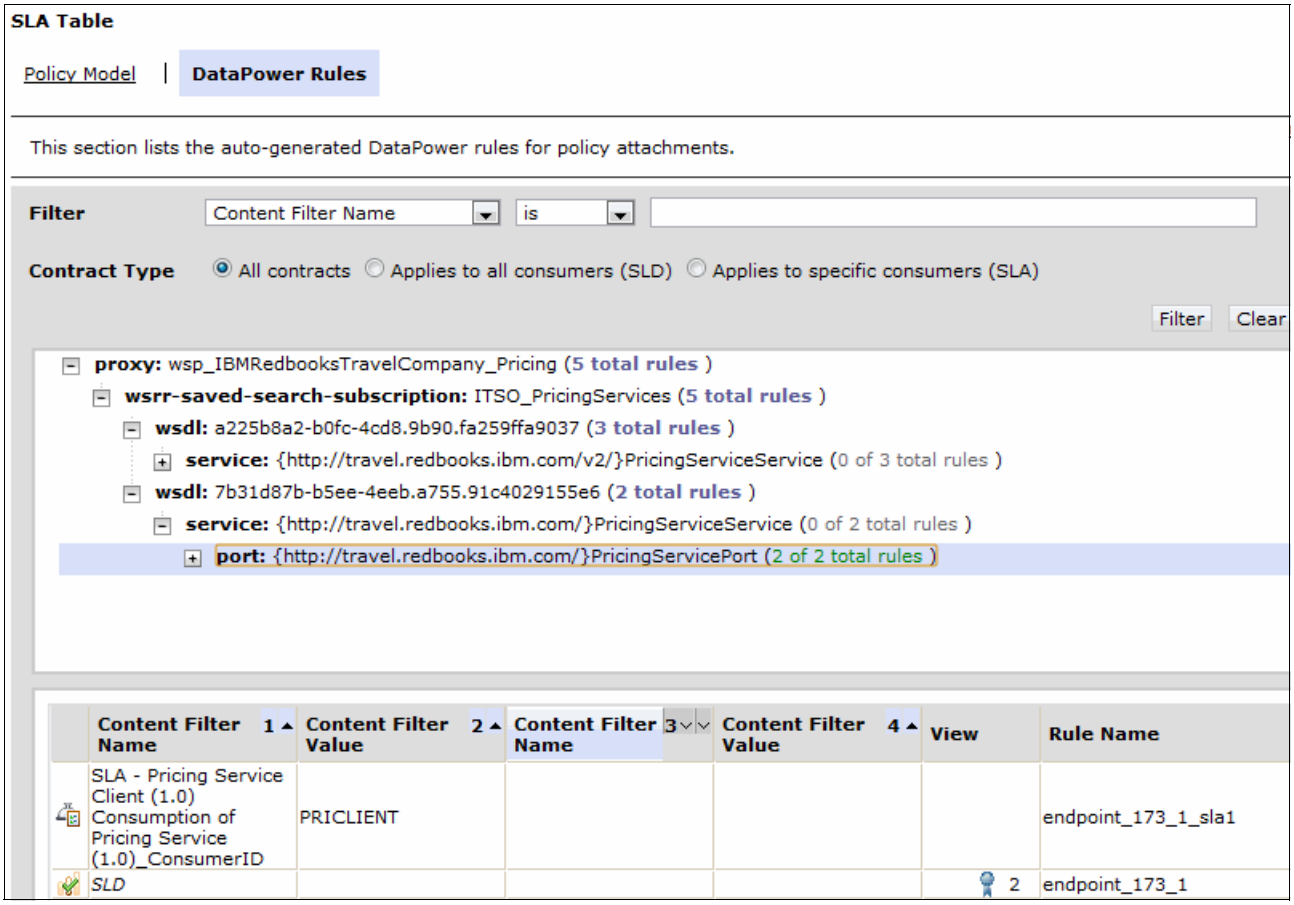


Figure 5-57 DataPower rules of the Pricing services

19. Click the **SLD** links to display the DataPower rules that were created by the custom versioning stylesheet policy.

You can see DataPower configuration artifacts that were created, as shown in Figure 5-58.

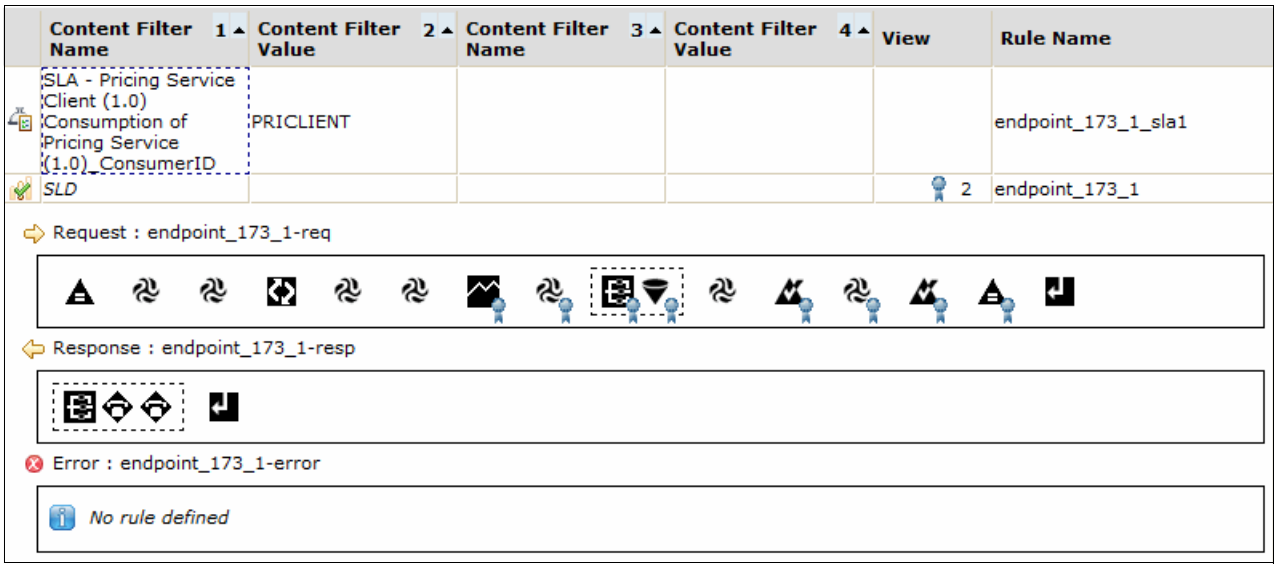


Figure 5-58 DataPower rules at the SLD level

Figure 5-59 shows the DataPower configuration artifacts that are created on the request processing rule.

Figure 5-59 DataPower configuration artifacts created on the request processing rule

From the left to the right, the DataPower actions are as follows:

a. A first `Validate` action. Figure 5-60 shows the details of this action.

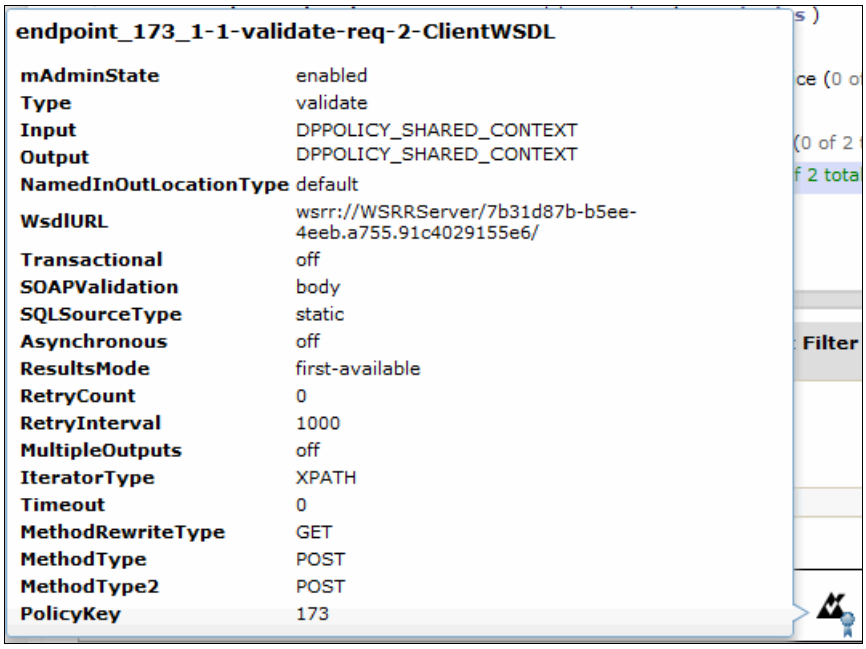


Figure 5-60 Details of the first Validate action on the request processing

b. A `Transform` action. Figure 5-61 shows the details of this action.

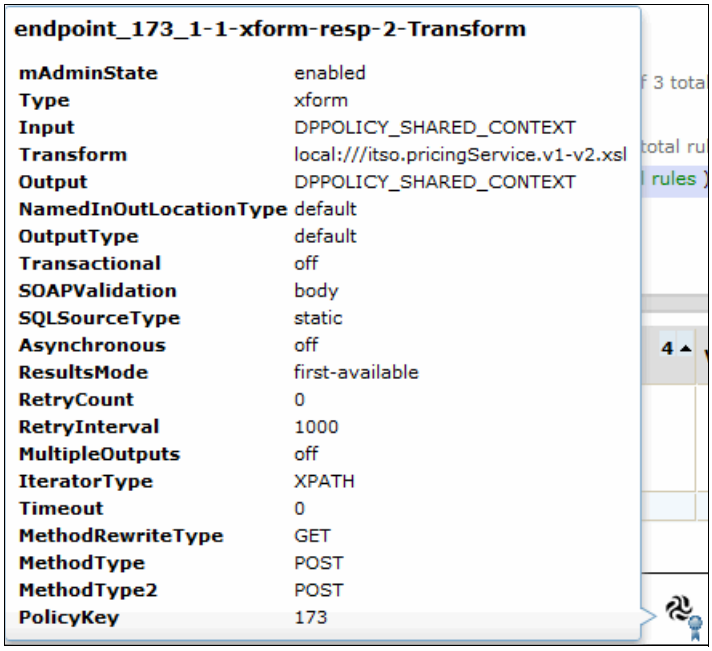


Figure 5-61 Details of the Transform action on the request processing

c. A second `Validate` action. Figure 5-62 shows the details of this action.

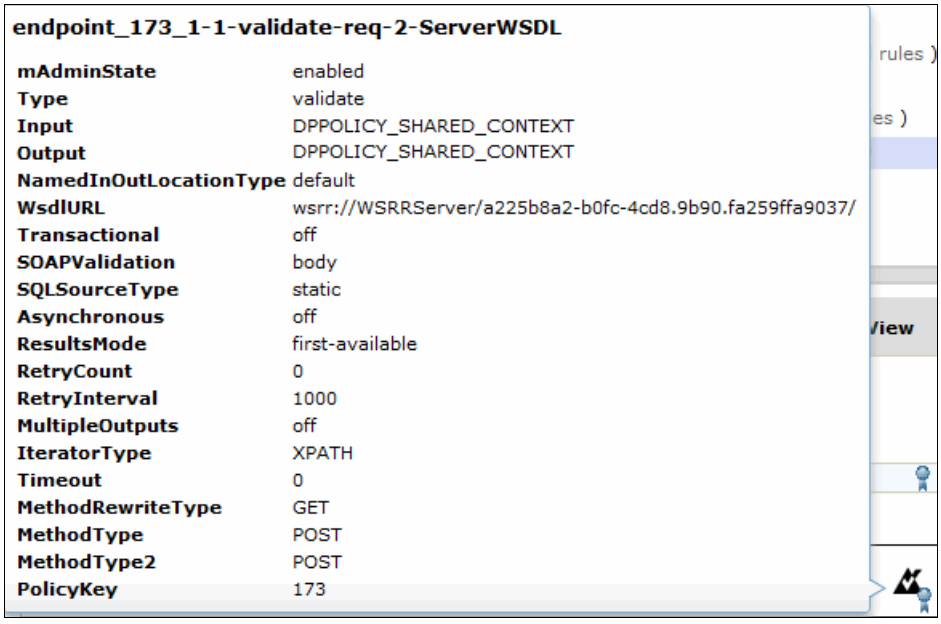


Figure 5-62 Details of the second Validate action on the request processing

d. A `Set-Variable` action for routing. Figure 5-63 shows the details of this action.

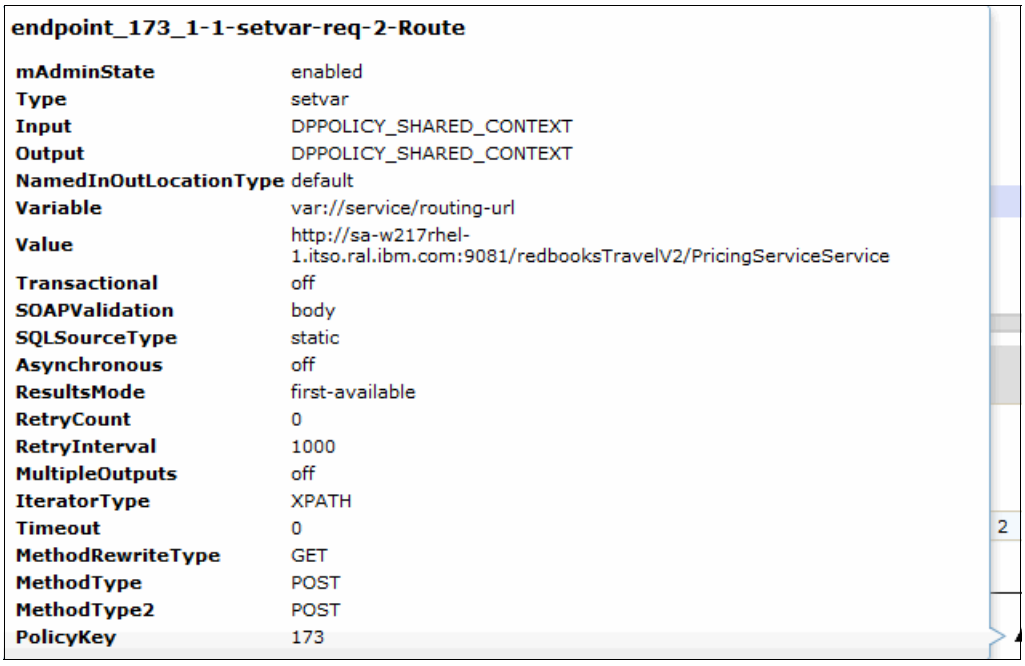


Figure 5-63 Details of the Set-Variable action on the request processing

The DataPower configuration artifacts that are created on the response processing rule are accessible through a search on the DataPower objects. The artifacts that are generated are accessible through a callable rule.

The callable rule references a processing rule, as shown in Figure 5-64.

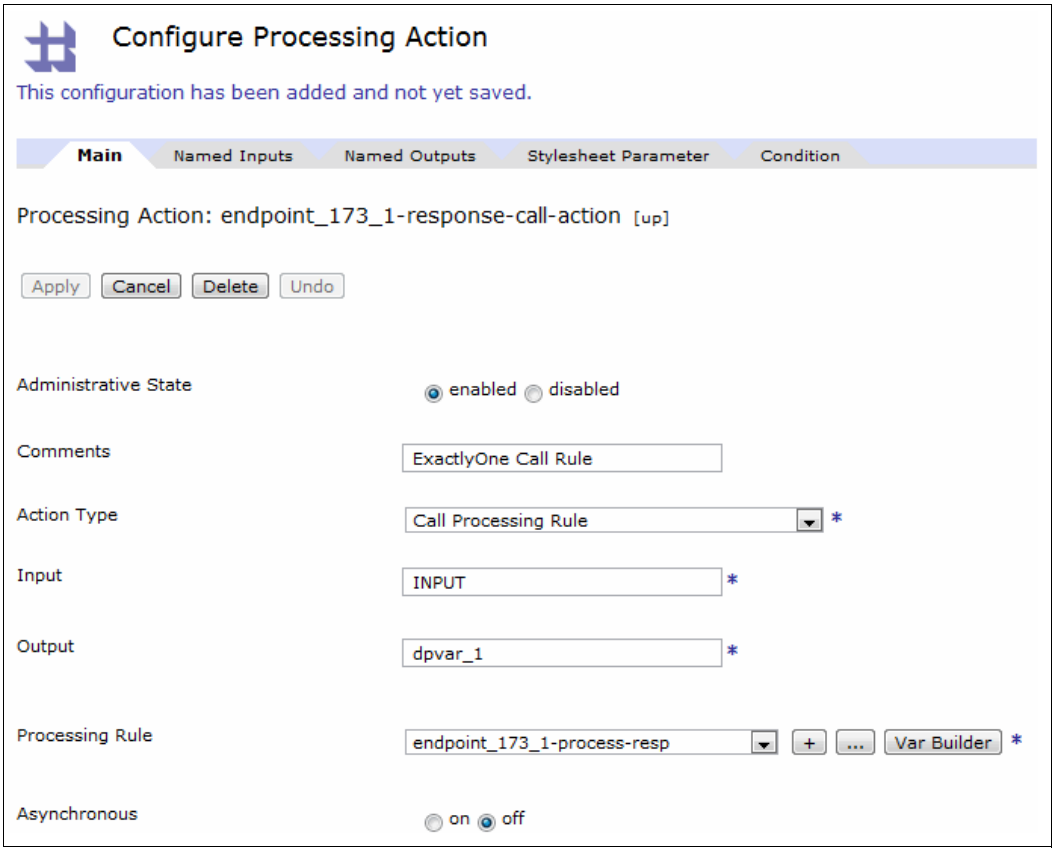


Figure 5-64 Details of the callable rule processed on the response

The processing rule executed is `endpoint_173_1-process-resp` in this example. Figure 5-65 shows the details of this processing rule.

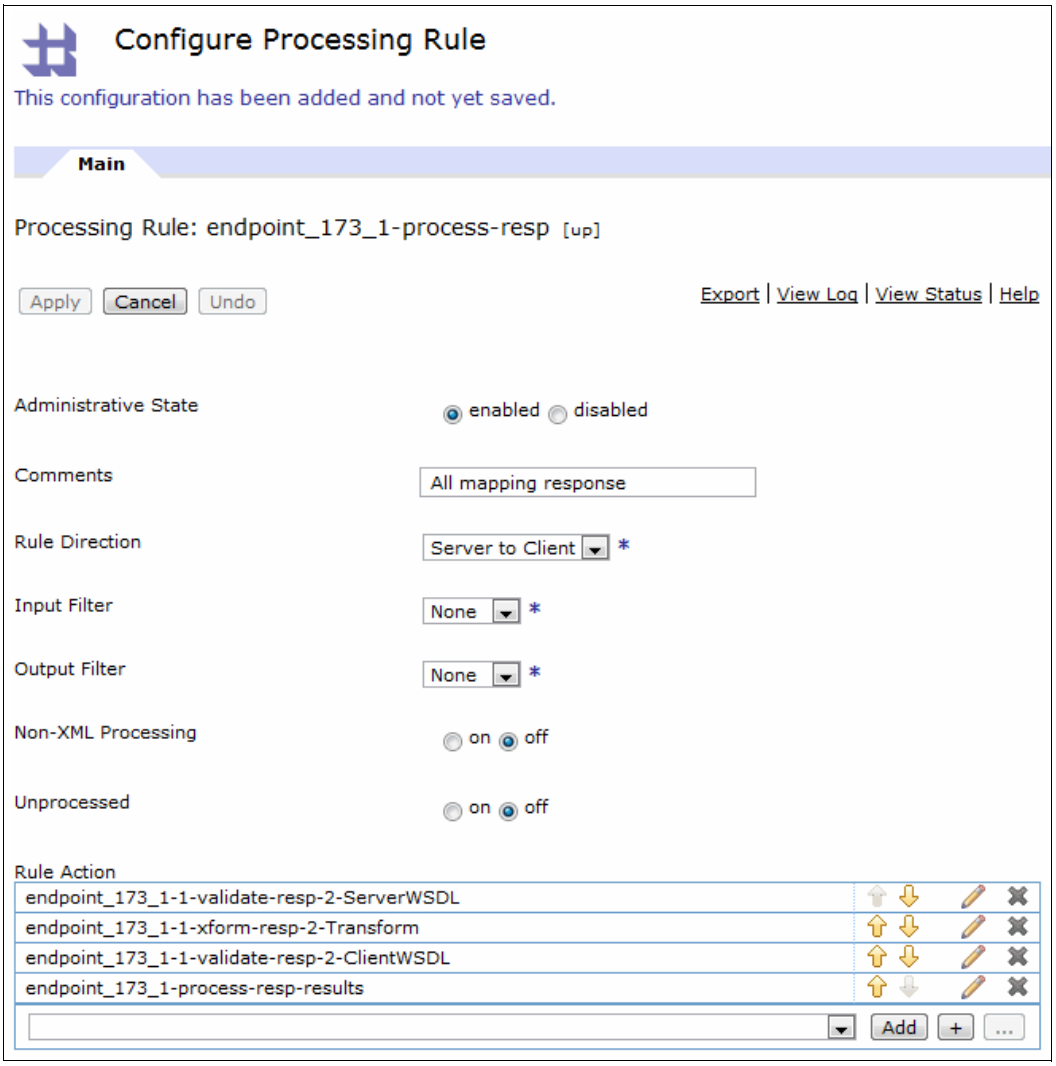


Figure 5-65 Processing actions configured on the response processing rule

The `Validate` and `Transform` actions are well-configured on the response processing.

5.7.4 Demonstration of the custom versioning policy enforcement

This section presents a demonstration of the enforcement of the custom policy for versioning.

Probes are activated on the Web Service Proxy for versioning, created in 5.7.3, “Creating a Web Service Proxy for versioning in DataPower” on page 166. The versioning policy is ready to be enforced, but only for consumers that are linked to the SLD of the Pricing Service Version 1.0.

The client that is used for the test is the one that was developed to access services of the IBM Redbooks Travel Company.

Use the following steps to run the demonstration.

1. Access the IBM Redbooks Travel Company test client user interface.
2. Click the **Pricing Service Web Services Test Client** link to access a consumer application linked to the Pricing Service Version 1.0.
3. Enter the endpoint URL of the virtualized Pricing service, exposed on the DataPower appliance.
4. Click **Invoke**, as shown in Figure 5-66.

Figure 5-66 Accessing Pricing Service Version 1.0

The result is displayed at the bottom of the page, as shown in Figure 5-67.

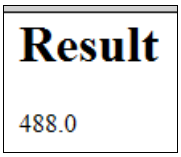


Figure 5-67 Result of the Pricing service

5. Refresh the DataPower probes to display the last transaction, as shown in Figure 5-68.

Figure 5-68 Inbound and outbound URLs in the DataPower probes

You can see that the outbound URL corresponds to the endpoint of the Pricing service in version 2.0. Therefore, the routing to version 2.0 occurred, as described in the 5.6, “Creating custom policy XSL style sheet for the custom versioning policy” on page 139.

6. Click the plus sign (+), beside the magnifying glass, of the transaction. The rules that are executed are listed, as shown in Figure 5-69.

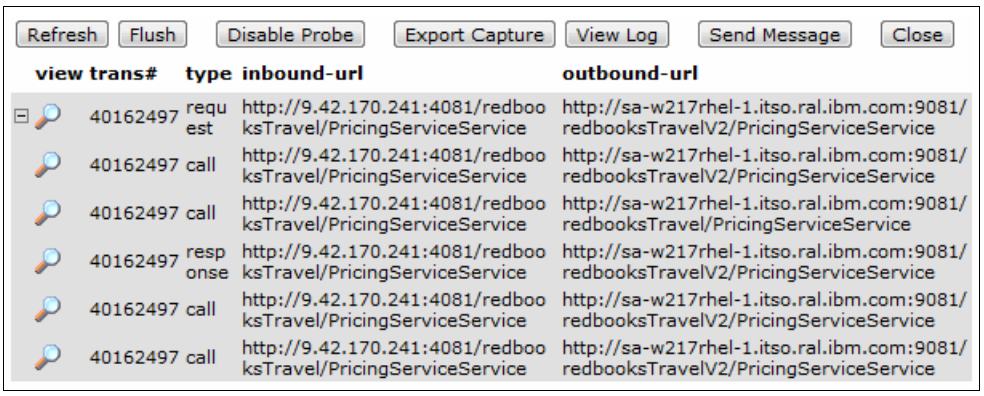


Figure 5-69 Processing rules of the Web Service Proxy for versioning

7. Click the first callable rule at the top of the transaction probes, as shown in Figure 5-70

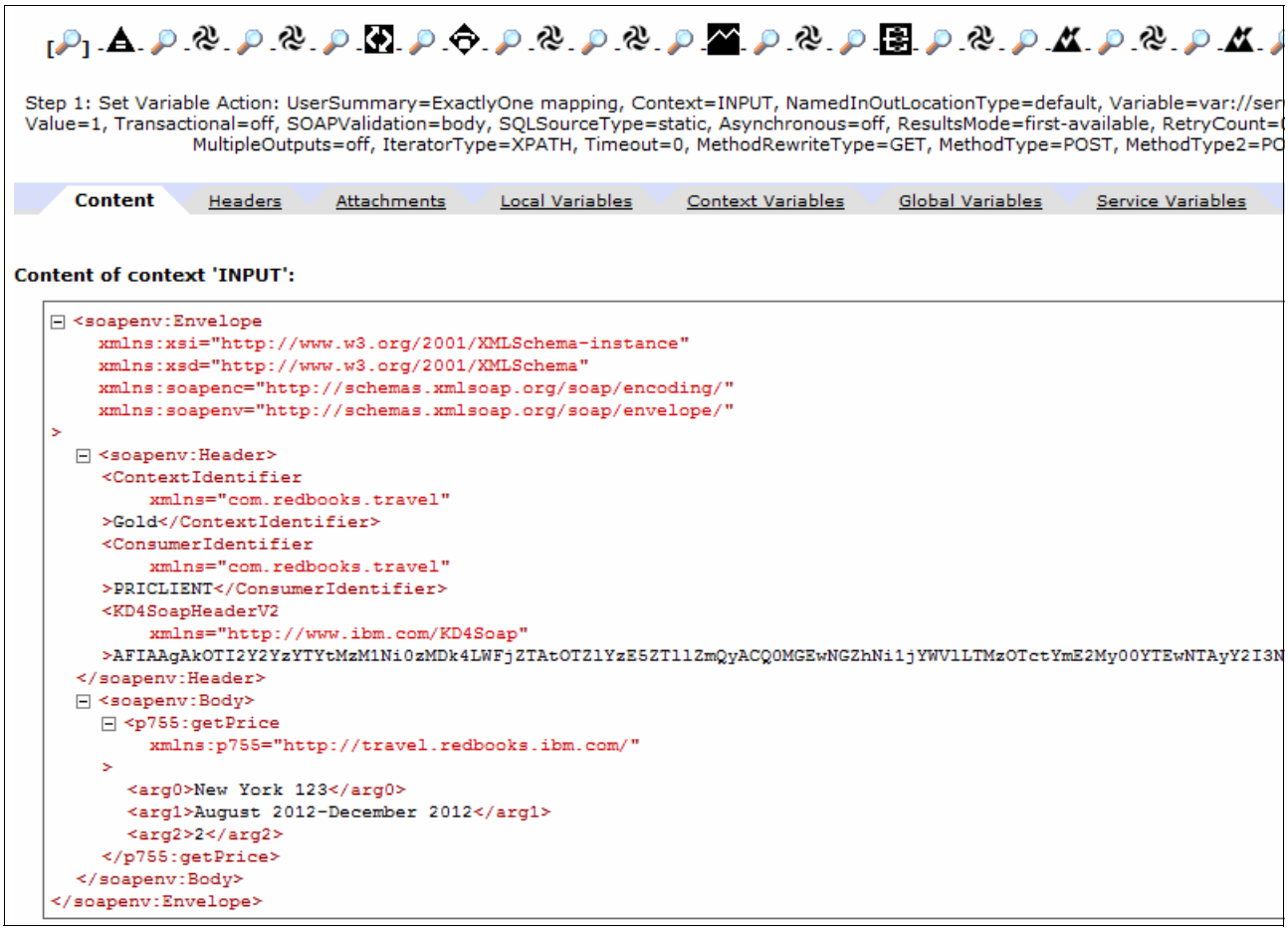


Figure 5-70 Processing actions executed on the request processing rule

The version 1.0 request message is displayed (sent by the consumer application).

The actions that are run during request processing display at the top of the probes panel, as shown in Figure 5-71.

Figure 5-71 Processing actions for versioning policy enforcement

8. Click the magnifying glass to the left side of the transform action. It displays the content before data mediation of the request, as shown in Figure 5-72.

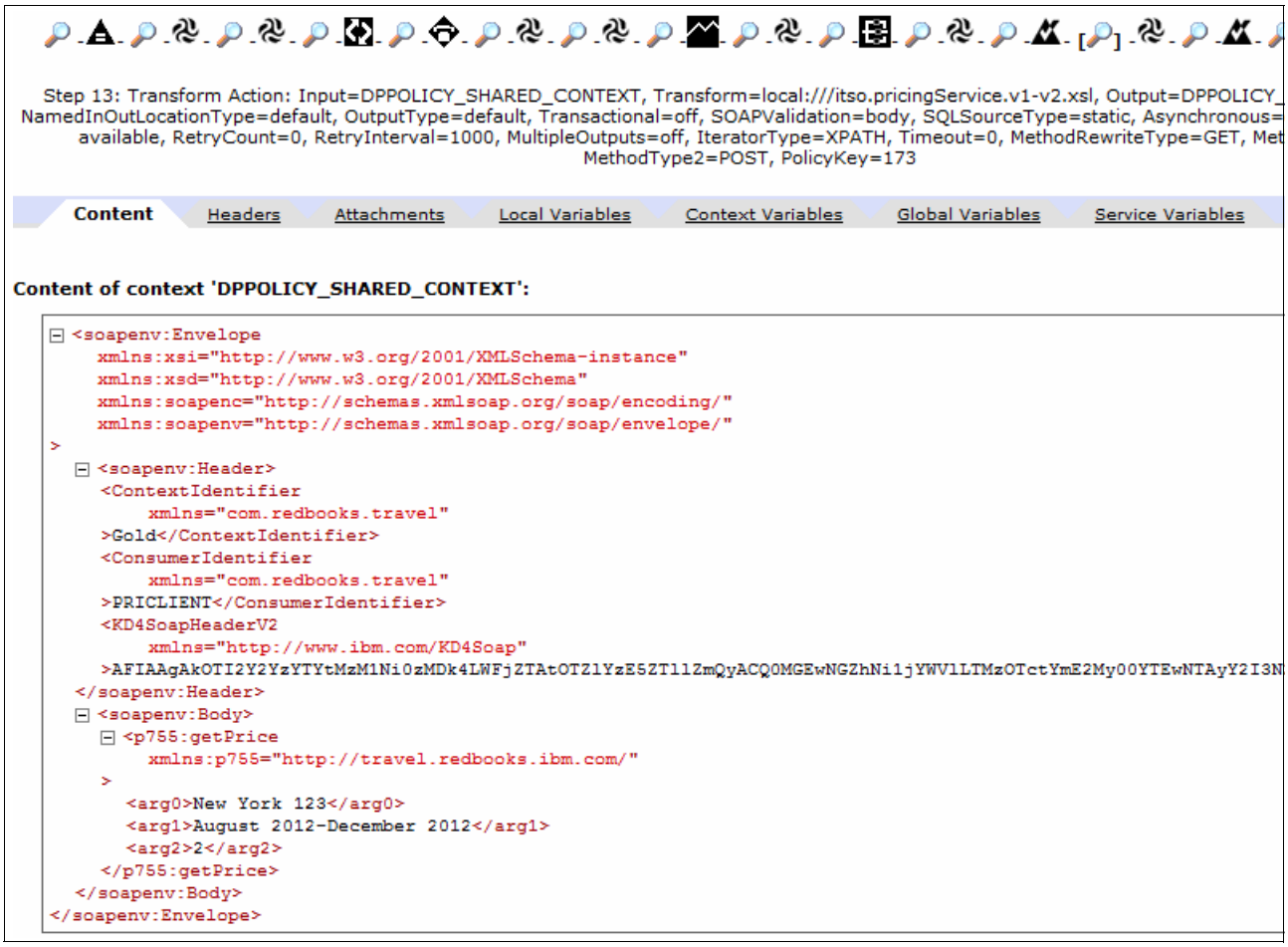


Figure 5-72 Request message before data mediation

9. Click the magnifying glass at the right side of the transform action. It displays the content after data mediation of the request, as shown in Figure 5-73.

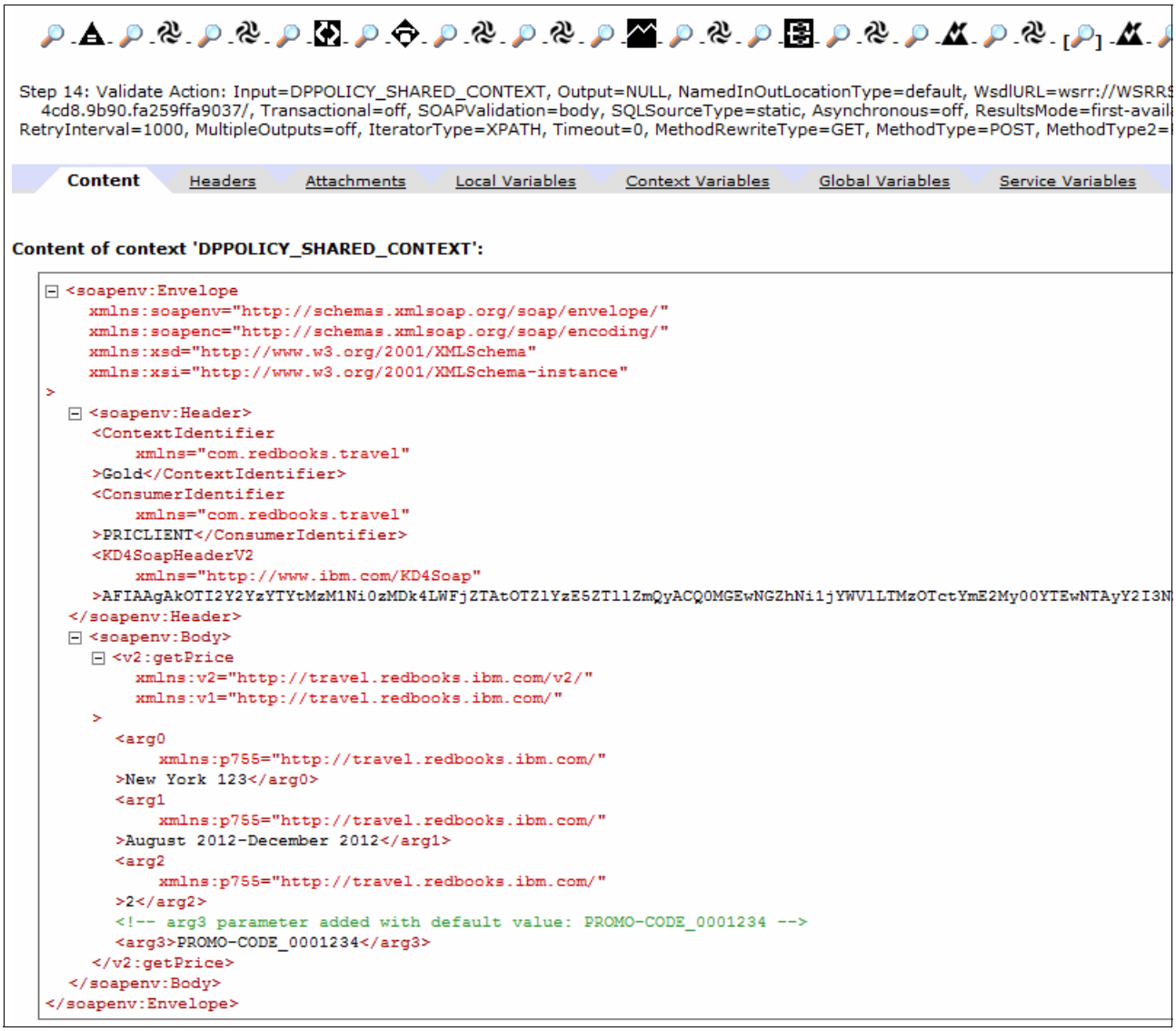


Figure 5-73 Request message after data mediation

During request processing, the message was adapted from version 1.0 into version 2.0 of the Pricing service, as required by the versioning policy that is enforced. The content after data mediation of the request consists of the following treatments:

- Namespace translation change from `http://travel.redbooks.ibm.com/` to `http://travel.redbooks.ibm.com/v2/`
- Added an `<arg3>` parameter (promotional code) in the request element `getPrice`. A default value is set for this element: `PROMO-CODE_0001234`.

The two Validate actions produce an output context. This means that validations, before and after the data mediation for versioning, are successful.

10. Go back to the transaction probes panel and select the last callable rule, at the bottom of the probes panel, as shown in Figure 5-74 on page 185. This callable rule corresponds to the response processing rule that is executed during the versioning policy enforcement.

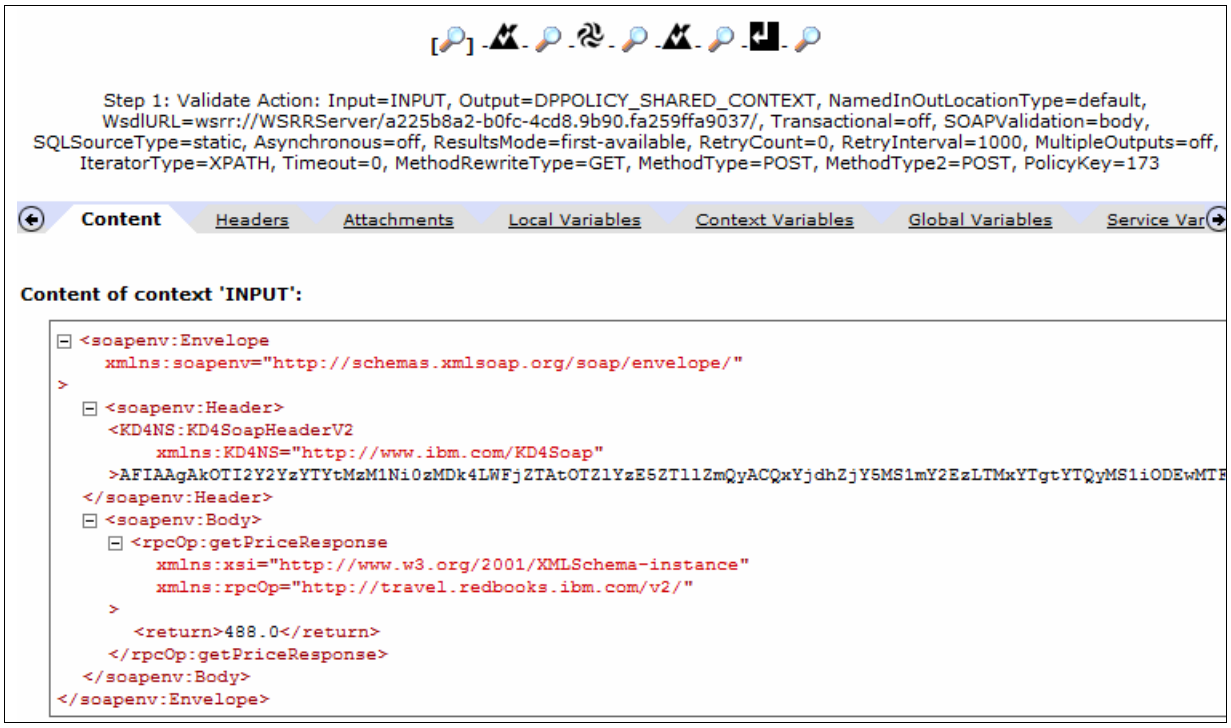


Figure 5-74 Response processing rule executed during versioning policy enforcement

11. Click the magnifying glass at the left side of the transform action. It displays the content before data mediation of the response, as shown in Figure 5-75.

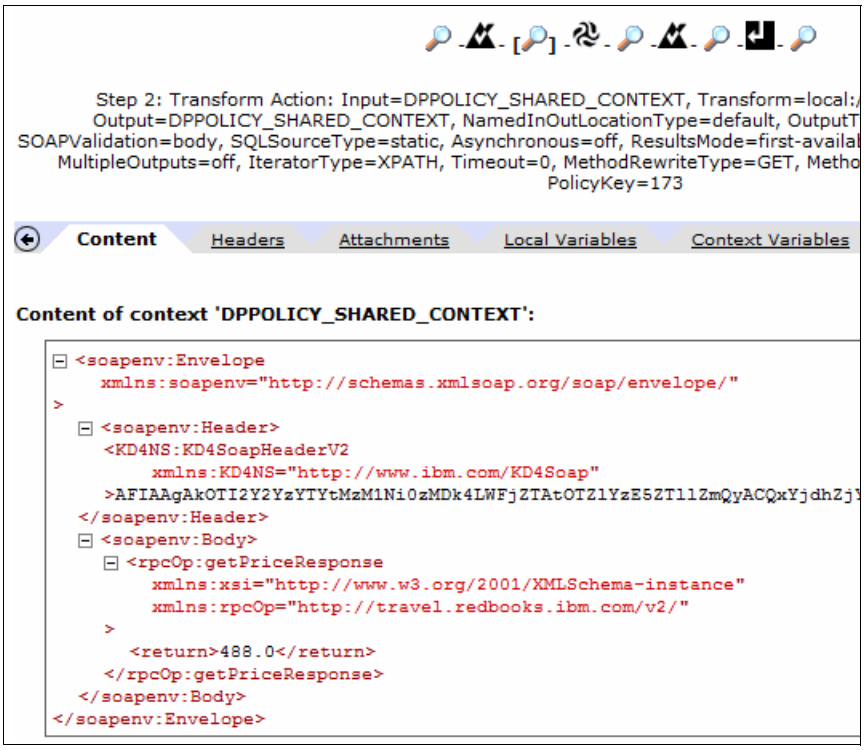


Figure 5-75 Response message before data mediation

12. Click the magnifying glass at the right side of the transform action. It displays the content after data mediation of the response, as shown in Figure 5-76.

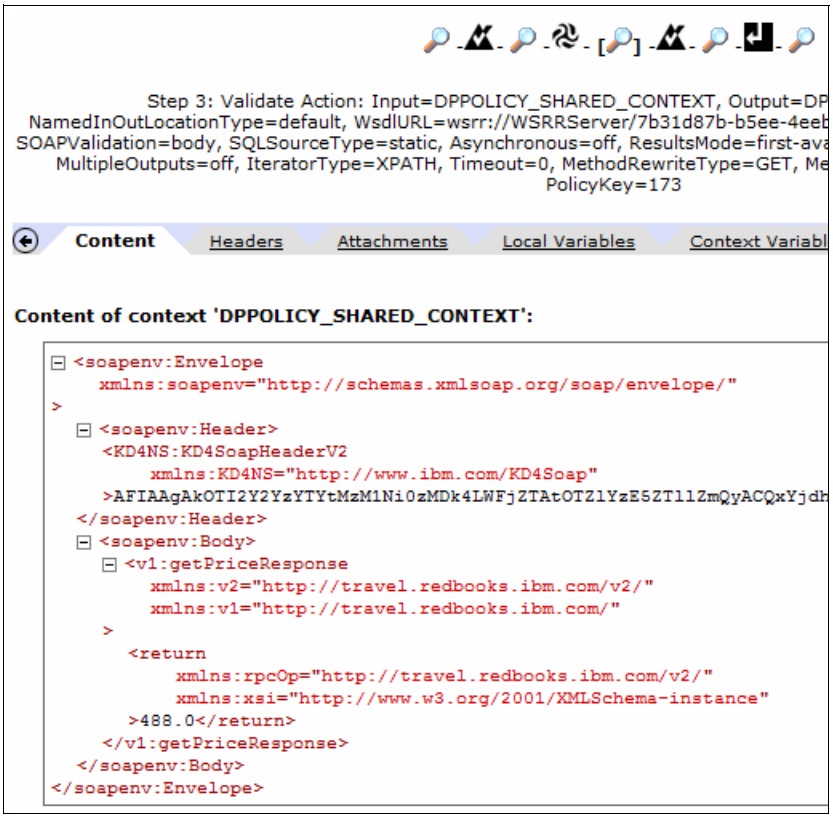


Figure 5-76 Response message after data mediation

During response processing, the message is adapted from version 2.0 into version 1.0 of the Pricing service, as required by the versioning policy that is enforced. The data mediation consists of the following treatment:

- Namespace translation from `http://travel.redbooks.ibm.com/v2/` to `http://travel.redbooks.ibm.com/`

13. Also notice that the two Validate actions produce an output context: this means that validations, before and after the data mediation for versioning, are successful.

Requesting the Pricing Service Version 2.0 shows that the custom versioning policy is not enforced, because the custom versioning policy is attached at the SLD of the Pricing Service Version 1.0.

6

Security using custom policy

Mediation policy capabilities can be specified directly in the policy administration point (PAP) that is implemented in WSRR. In addition, it is also possible to create policies that are based on the full suite of policy capabilities available in the policy enforcement point (PEP) as implemented in WebSphere DataPower. Chapter 5, “Versioning with custom policy” on page 119 describes how to use such policy (the *custom policy*) for the purpose of applying policy to automatically cause consumers to use the latest version of a provider policy. This chapter shows how customer policy can be used to create a security policy. It presents the steps to create the custom security policy in DataPower and then manage the policy in IBM WebSphere Service Registry and Repository (WSRR).

As part of the process of creating the custom security policy in DataPower, you see how to create the configuration artifacts. These artifacts are used in creating the custom security policy XSL style sheet.

This chapter contains the following topics:

- 6.1, “Service security” on page 188
- 6.2, “Creating a custom policy for security” on page 191
- 6.3, “Creating custom policy domain and assertion for security” on page 194
- 6.4, “Creating a custom policy XSL style sheet for the custom security policy” on page 197
- 6.5, “Attach custom policy to all services for an organization” on page 220

© Copyright IBM Corp. 2013. All rights reserved.

6.1 Service security

Creating a secure environment is a challenge in a distributed environment that uses web services. Many times, a services environment is the tipping point for identifying the limitations of existing security implementations. It Is important to recognize what those limitations are and work with the corporate security group to enhance the set of security capabilities that will be needed for a service-oriented architecture (SOA). Merely securing the perimeter with firewalls or routers is no longer sufficient. A business must have dynamic trust relationships over time with its partners, customers, and employees. To provide such flexibility, a business needs to use a security services infrastructure. The overall security principles that apply in any environment, whether SOA or not, are the same: identity, authentication, authorization, confidentiality, integrity, audit and compliance, policy management, and availability. What changes in a services environment is how those concepts are applied.

For aN SOA approach, separation of concerns is necessary, which means that application developers should not be deciding what security is needed. Instead, the security experts at the organization should be making those decisions and implementing that policy in the middleware. For the SOA Policy Solution, t the security team should be creating and implementing security policy that can be applied to services.

Security management permeates all aspects of the service-oriented lifecycle and is a key enabler for achieving the connectivity and flexibility goals of service orientation in a secure manner. The security function must consider the following aspects of service security to properly execute its duties of ensuring that services are architected in a secure manner:

- Management of identity and security across a range of systems and services that are implemented in a diverse mix of new and old technologies
- Protection of data in transit and at rest
- Demonstrable compliance with a growing set of corporate, industry, and regulatory standards

The first bullet mentions the need to manage identity and security across a diverse mix of technologies. A good SOA provides services that will be shared and reused, and connected in a variety of ways. Completely anticipating the permutations of service interconnection is impossible, but corporate security must ensure that the organizations, policies, and processes are in place so that services can be designed and run in a secure and auditable manner. Identity plays a key role in delivering on the promise of service orientation, because it must potentially be able to be used across various lines of business, customers, and partners. Identities exist for both users and services, and both must be subject to the same controls.

Consider Figure 6-1 with respect to the need for a security policy for services.

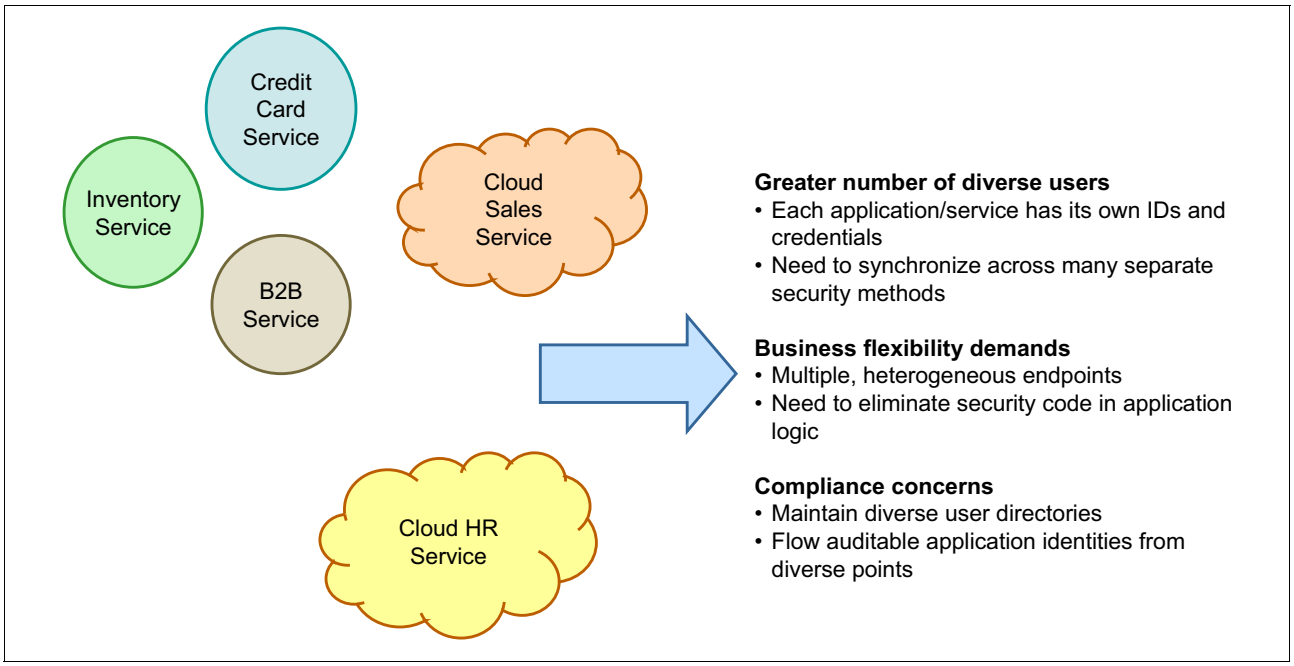


Figure 6-1 Security policy for services

The identities that are used for one service must sometimes be propagated throughout the SOA environment. In many cases, service implementations must mediate the formats and standards that are used for an identity to and from a service. Identity Services are needed in the infrastructure to handle identity mediation issues. This way allows services to be easily interconnected without worrying about how to map and propagate user identity from one service to the next. This approach should eliminate having to do this with application code and will also improve the speed and ease of service security by creating a standardized security pattern.

Another good reason for DataPower to manage your security is that DataPower provides a solution that supports a “defense in depth” security approach. It provides a new security layer for web services and XML with a single secure path through the firewall and threat protection from malicious XML. This concept is particularly useful if you have many unpredicted users of your web services. From the perspective of SOA Governance, the key concept is that the security appliance provides a single point of security for exposed services. Therefore, clients that invoke any number of services that are exposed through this appliance must be configured only to trust keys or tokens from this single proxy instead of trying to handle keys and tokens from each service.

Regardless of the form of the interaction, an imperative task is that security, identity, and access policies be defined and enforced for all transactions that cross a boundary where security credentials must be checked. One would not trust, for example, a transaction coming from outside the company without appropriate identity authentication and authorization.

Boundary security services, for example, must be able to provide coarsely grained verification that requests are coming from or going to trusted parties. Establishing a trust relationship between organizations is a key step in allowing inter-organization messaging. SOA Governance must ensure that rules are established around such interaction, such as defining identity information that must be propagated between organizations or usage of appropriate techniques to secure messages in a standardized manner across organizations.

Although you might consider a security policy for a particular service or application, a more complex activity is to consider the security policy for a combination of services. For example, a user or service might require specific privileges to allow them to access a service. However, when a number of services are combined, such as when they are choreographed into a higher level business process, the combination of these services might require another examination of the security policy. If a consumer service must be able to access a set of data across a variety of organizations, chances are good that before using a services approach, that access was not doable.

The services security strategy must account for the complexity of the SOA environment in relation to the security policy created and the potential for mixing and matching of services in various combinations. Predicting what those combinations might be is impossible. The governance function must allow for mixing and matching of services in various combinations with the proper mix of policy and process to examine any changes necessary to ensure it remains valid.

In regard to web services, security features specifically cover the following topics:

- Security based on secured protocols adoption, such as HTTPS for example
- Security based on the message itself, implementing cryptographic operations such as encryption/decryption and digital signatures
- XML threats, which must be detected and rejected.
- Access control to perform authentication, authorization, and auditing of service consumers

Security based on protocols or messages are sometimes requirements and constraints that are imposed by service providers. In this case, the WS-SecurityPolicy specification is the correct framework to adopt, because it defines dedicated protection and token assertions that can be used to enforce such security policies.

Example 6-1 is an example of a WS-SecurityPolicy policy, which specifies that a WS-Security Username token version 1.0 or version 1.1 must be present in a request message. Specifically required is that these security tokens must be declared in the SOAP header.

Example 6-1 WS-SecurityPolicy

```
<!--+
   *************************
   *** WS-SecurityPolicy ***
   ********************************************************************************
   *** Revision: 1.0
   *** Description:WS-SecurityPolicy example - WS-Security Username token 1.0 or 1.1
   ********************************************************************************
   +-->
<wsp:Policy xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
   xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512"

xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.x
sd"
   wsu:Id="usernametoken">

   <wsp:ExactlyOne>
      <!-- UsernameToken 1.0 -->
      <wsp:All>
         <sp:SupportingTokens>
            <sp:UsernameToken

sp:IncludeToken="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/Alw
aysToRecipient">
```

```
                <wsp:Policy>
                   <sp:WssUsernameToken10 />
                </wsp:Policy>
             </sp:UsernameToken>
          </sp:SupportingTokens>
       </wsp:All>
       <!-- UsernameToken 1.1 -->
       <wsp:All>
          <sp:SupportingTokens>
             <sp:UsernameToken

sp:IncludeToken="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/Alw
aysToRecipient">
                <wsp:Policy>
                   <sp:WssUsernameToken11 />
                </wsp:Policy>
             </sp:UsernameToken>
          </sp:SupportingTokens>
       </wsp:All>
    </wsp:ExactlyOne>

</wsp:Policy>
```

Custom policy within the SOA Policy Solution is a useful way to create standardized security patterns and exploit the rich security features of WebSphere DataPower as the policy enforcement point. As this chapter shows, the standardized security policy is created in DataPower and then managed and governed in the policy administration point in WSRR. Specifically, the steps are as follows:

1. Create the standardized security policy in DataPower (the policy enforcement point).
2. Export the policy to WSRR (the policy administration point).
3. Create a Policy Attachment query in WSRR to identify the set of services that will be attached to the standardized security policy.

6.2 Creating a custom policy for security

This section describes the best practices for creating a custom policy for security. It presents the custom security policy requirements of the Fictional IBM ITSO Redbooks Travel Company, before creating the custom policy security assertions.

A custom policy for security should be developed to create a security pattern that will be used repeatedly in your organization. The policy will be applied to a set of services that use the policy attachment capability in WSRR as the policy administration point (PAP).

The scenario for security in this chapter is the requirement to standardize access control at the enterprise level. Assume that the ITSO Redbooks Travel Company has a set of DataPower appliances, which perform security enforcement with authentication, authorization and audit (AAA) policies configured on the DataPower devices. These AAA policies must be applied to control access to our company services.

6.2.1 DataPower AAA policy

A DataPower authentication, authorization and audit (AAA) policy identifies a set of resources and procedures which a company can use to determine whether a requesting client or

consumer should be granted access to a specific service, file, or document. The AAA policies can be considered types of filters, because they accept or deny a specific client or consumer request.

For more information about DataPower AAA Policy, see *IBM WebSphere DataPower SOA Appliances Part II: Authentication and Authorization*, REDP-4364.

6.2.2 Prerequisites

Before creating the custom policy grammar with a DataPower AAA configuration, you should have an understanding of the current security enforcement in terms of access control at the company level. Therefore, before creating a custom security policy, work with the security group to find or create an inventory of the types of AAA policies that are currently used. If AAA policies do not exist, the inventory consists in gathering the security needs that are related to the access control that must be enforced.

The policy vocabulary, which follows from the AAA policy inventory and understanding, must not be ambiguous and must remain simple. Preferably, use generic terms and specify policy assertions by using policy assertion parameters.

To describe this custom policy security example, the following assumptions exist with regard to the ITSO Redbooks Travel Company:

- The ITSO Redbooks Travel Company needs to control access of its services.
- Every consumer must be authenticated to access backend services.
- Access control is based on consumer identifiers. These identifiers are sent by the consumer application and are used to identify the security access that is allowed for a specific application user.
- Access control consists in checking that the extracted identity (consumer identifier) is registered into a *white* list. If not present in the list, the consumer traffic is rejected.

  Identity checking database: Normally identity checking uses an LDAP or other type of identity checking database. We use a list to simplify this example.

- There is no authorization need in the access control policy of the ITSO Redbooks Travel Company. Any authenticated consumer is allowed to access the requested service.

  Not used in example: Normally there are levels of access allowed per the LDAP or identity management mechanism, which this example does not use.

- The white list of authenticated consumers is based on an XML file loaded in the DataPower appliance.

6.2.3 Policy decision point (PDP) versus policy administration point (PAP)

Before going further, an important step is to analyze the differences between an SLA and access control. As you create a custom policy for security, based on a DataPower AAA configuration, the authentication process will identify consumers that can access back-end services.

In WSRR, SLAs provide the mechanism to prevent a consumer from accessing a back-end service if the SLA prohibits such access. For example, if the SLA between a consumer and a

provider is set to *inactive* in WSRR, then the SLA check, enforced at the DataPower level, rejects consumers for which the SLA is inactive. The reason for the rejection during the SLA check is simple: there is no active contract (because SLA is inactive) between the consumer and the provider.

Nevertheless, there is an important difference between making an SLA inactive in WSRR and deciding that a consumer is not authenticated to access a service, even though the result is the same: the consumer is rejected (during security policy enforcement or SLA check).

Making the SLA inactive is an example of *course grained* security, where all customers for a consumer-provider pair are rejected. Checking for authentication of a particular customer on a consumer-provider pair transaction is an example of *fine grained* security.

The decision about whether an SLA must be deactivated or activated, or if a consumer is authenticated or not, are not the same. A modification is executed in WSRR (SLA state is active or inactive); however, the modification consists in modifying a repository of authenticated consumers (an XML file in this example, an LDAP server for many companies).

WSRR is a policy administration point (PAP); DataPower (based on an LDAP server or in XML file) is a policy enforcement point (PEP) and a policy decision point (PDP). The repository for authentication and authorization is the policy information point (PIP).

Figure 6-2 is a reminder schema of the administration, decision, enforcement, and monitoring points in the IT of a company. Based on our example, DataPower is the PEP and the PDP.

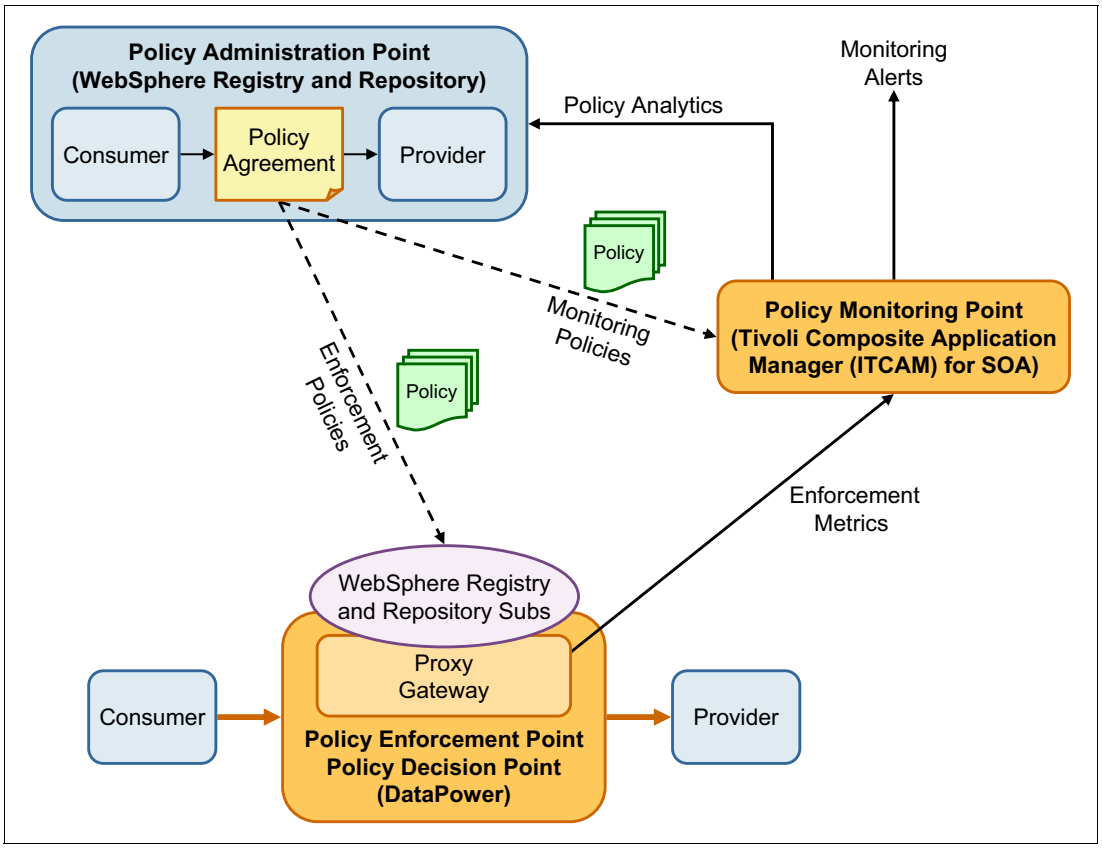


Figure 6-2 Policy administration, enforcement, decision, information and monitoring points

In the following situations for the example, a consumer is rejected during policy enforcement in DataPower:

- A consumer is defined as authenticated in the PEP, but no contract (SLA) is defined in the PAP.
- A consumer is not defined as authenticated in the PEP, but a contract (SLA) is defined in the PAP.
- A consumer is not defined as authenticated in the PEP, and a contract does not exist in the PAP.

When a consumer is defined as authenticated in the PEP and its SLA to a service provider is active in the PAP, it is allowed to access this service.

6.3 Creating custom policy domain and assertion for security

Create a custom policy domain and assertion to standardize a DataPower AAA action.

The first step is to choose a namespace for the custom security policy. This namespace is used when creating a policy, to qualify the elements of the custom security policy vocabulary. It is also used when creating the custom XSL stylesheet policy that is implemented to transform policy assertions into DataPower configuration artifacts.

The following namespace is used in this example:

`http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11`

Tip: A best practice is to add a date or version to the custom policy namespace. This step can ease the support of future versions of the policy domain.

The prefix for this namespace is `itso`.

The custom policy for security must reflect the capabilities of the access control you want to provide. Because the business need is to be able to authenticate consumer identifiers, you can define a simple element as the vocabulary of your custom security policy.

The unique element (assertion) for the custom security policy is as follows:

`{http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11}ControlConsumerAccess`

This element does not provide any information about the PEP configuration that is responsible for enforcing the access control policy. Moreover, the name of the tag that describes the action (`ControlConsumerAccess`) is clear, because the aim of the security policy is to control access of consumers.

Example 6-2 shows a custom policy for security, using the chosen namespace and vocabulary.

Example 6-2 Custom security policy for AAA enforcement

```
<?xml version="1.0" encoding="utf-8"?>
<!--+
   *********************************
   *** itso:ControlConsumerAccess ***
   *********************************
   *** Author: ITSO IBM Redbooks Travel Company
   *** Revision: 1.0
   *** Description:Custom policy for security (access control of consumers.
   *** Namespace:http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11
   +-->
```

```
<wsp:Policy wsp:Name="Message-securitypolicy-template"
            wsu:Id="wsp-custom-securitypolicy"
            xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
            xmlns:wsmp="http://www.ibm.com/xmlns/stdwip/2011/02/ws-mediation"

xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.x
sd"
            xmlns:itso="http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11">

   <itso:ControlConsumerAccess/>

</wsp:Policy>
```

The name of the custom security policy file is `aaa-customPolicy.xml`.

Table 6-1 lists details of the elements of the custom security policy file.

Table 6-1 Elements and attributes of the custom policy for security

Element	Description
/wsp:Policy	Root element of a custom security policy
/wsp:Policy/@wsp:Name	Attribute that defines the name for the custom security policy
/wsp:Policy/@wsu:Id	Attribute that defines the identifier of the custom security policy
/wsp:Policy/itso:ControlConsumerAccess	Custom element mentioning consumer access must be controlled in regard to an access control policy

Policy assertions can use parameters, even though it is not the case in this example. Assuming that the ITSO Redbooks Travel company has a security need to distinguish a standard security policy (such as the one described in Example 6-2) from a restricted security policy, the following parameters can be introduced in the custom policy vocabulary:

- `{http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11}Standard`

  Only authenticate the consumer identifier.

- `{http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11}Restricted`

  Authenticate and verify authorization to access a service, based on the consumer identifier identity information.

The `Restricted` parameter can specify that a consumer must be authenticated and authorized through a PDP to consume a service.

Example 6-3 shows a custom security policy that uses assertion parameters.

Example 6-3 Custom security policy with an assertion parameter

```
<?xml version="1.0" encoding="utf-8"?>
<!--+
   *********************************
   *** itso:ControlConsumerAccess ***
   *********************************
   *** Author: ITSO IBM Redbooks Travel Company
   *** Revision: 1.1
   *** Description:Custom policy for security (access control of consumers.
Assertion parameter usage example.
```

```
   *** Namespace:http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11
   +-->
<wsp:Policy wsp:Name="Message-securitypolicy-param-template"
            wsu:Id="wsp-custom-securitypolicy-param"
            xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
            xmlns:wsmp="http://www.ibm.com/xmlns/stdwip/2011/02/ws-mediation"

xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-util
ity-1.0.xsd"

xmlns:itso="http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11">

   <itso:ControlConsumerAccess>
      <itso:Restricted/>
   </itso:ControlConsumerAccess>

</wsp:Policy>
```

6.4 Creating a custom policy XSL style sheet for the custom security policy

This section describes details of the custom policy style sheet to create the DataPower configuration artifacts based on the custom security policy.

Figure 6-3 shows the generation process of the DataPower configuration artifacts in regard to a custom policy.

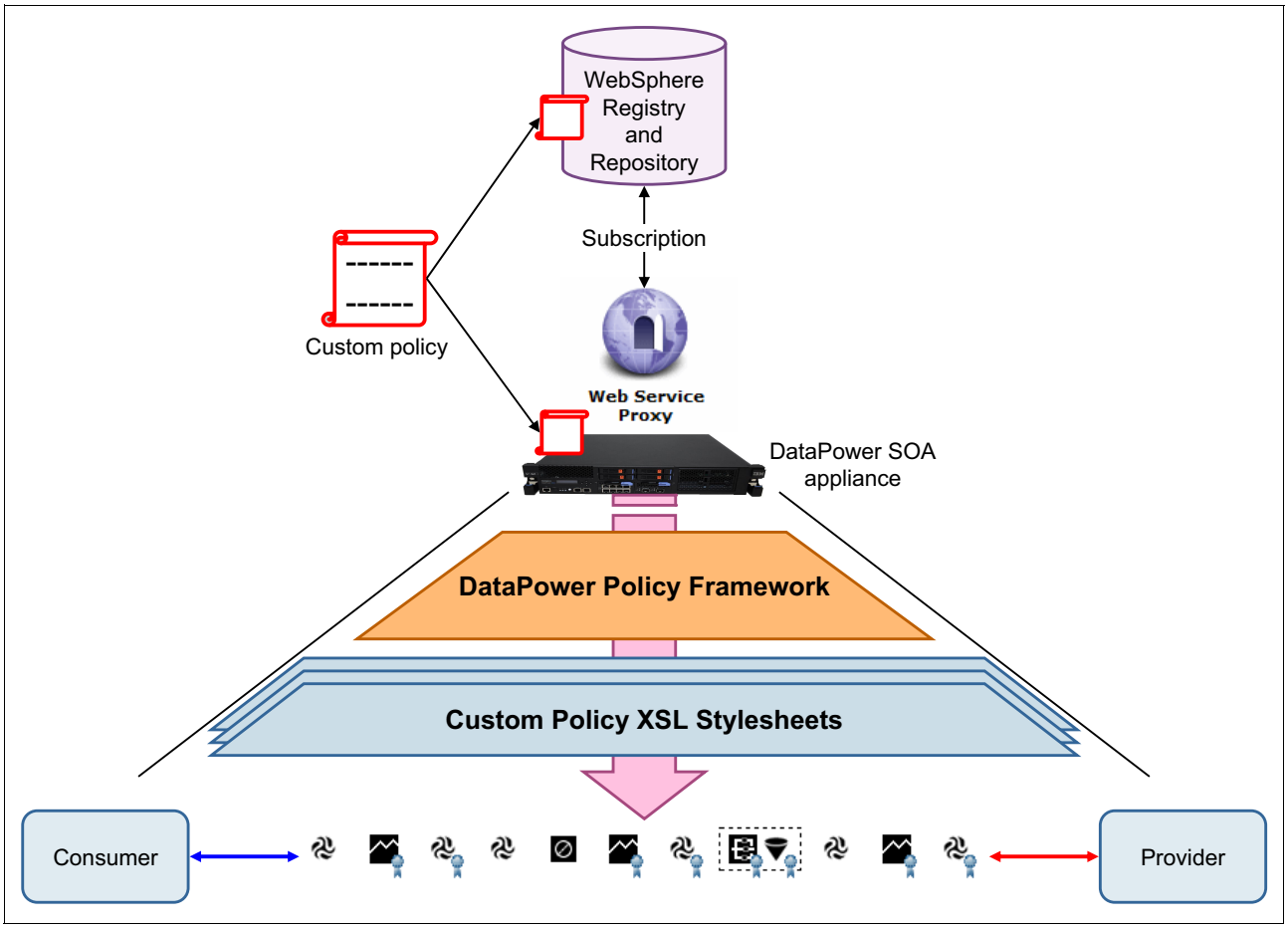


Figure 6-3 Generation of configuration artifacts based on a custom policy with DataPower

The custom policy can reside on WSRR or directly on the DataPower appliance. The best approach is for the custom policy to be exported from DataPower and reside on WSRR so that the policy can be attached to any services in WSRR, as shown in 3.5, “Attaching policies” on page 78.Examples of these two distinct management modes are demonstrated in 6.5, “Attach custom policy to all services for an organization” on page 220.

The custom style sheet is loaded on the DataPower appliance in the default domain, in the `store:///polices/custom` directory.

6.4.1 Creating the DataPower configuration artifacts

Before creating the custom policy style sheet, the best approach is to create the configuration artifacts that must be generated by the style sheet itself. After these artifacts are created, you can use their configurations as an example (or pattern) for the custom security policy style sheet that must be coded.

The AAA action has the following main properties:

- The extracted identity is the consumer identifier of the consumer application.
- The consumer identifier is extracted from a specific element of the SOAP header, as defined on the SLD object in WSRR, and shown in Figure 6-4.

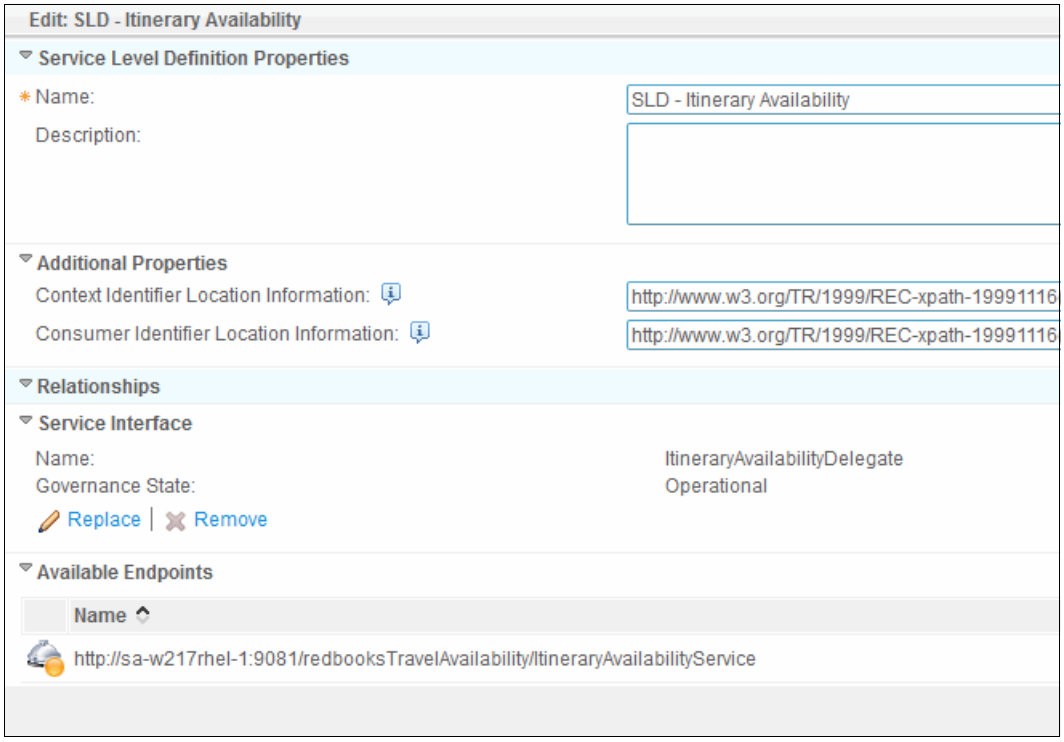


Figure 6-4 Consumer identifier location information in WSRR

This information is provided at the SLD level because the service provider is responsible for defining the location of the consumer indentifier.

The following XPath query expression extracts the consumer identifier:

```
/*[local-name()='Envelope']/*[local-name()='Header']/*[local-name()='ConsumerIdentifier']
```

- The authentication is enforced based on an XML parameter file (from the DataPower `AAAInfo` file).

Example 6-4 shows the content of this file.

Example 6-4 AAA Info file for the ITSO Redbooks Travel Company custom security policy

```
<?xml version="1.0" encoding="utf-8"?>
<AAAInfo xmlns="http://www.datapower.com/AAAInfo">
   <FormatVersion>1</FormatVersion>
   <Filename>AAAInfo.xml</Filename>
   <Summary>Repository of authenticated consumers of the ITSO Redbooks Travel
Company.</Summary>

<Authenticate>
   <!-- List of authenticated consumers -->
   <CustomToken>RESCLIENT</CustomToken>
   <OutputCredential>RESCLIENT</OutputCredential>

   <CustomToken>PRICLIENT</CustomToken>
   <OutputCredential>PRICLIENT</OutputCredential>

   <!-- Add CustomToken elements for the different authenticated consumers
   <CustomToken>...</CustomToken>
   <OutputCredential>...</OutputCredential>
   -->
</Authenticate>

</AAAInfo>
```

The two authenticated consumers are those that use the following consumer identifiers:

- RESCLIENT
- PRICLIENT

The name of this XML parameter file for authentication is `AAAInfo.xml`.

- Every authenticated consumer is authorized to access back-end services (no authorization step is required in the AAA policy).
- No post-processing must be configured on the AAA policy.

The following procedure are the configuration steps for creating this AAA policy on a DataPower device, based on the previous requirements:

1. Connect to a DataPower appliance on a development domain by using your credentials (login and password).
2. On the control panel, find the search field on the top left, as shown in Figure 6-5.

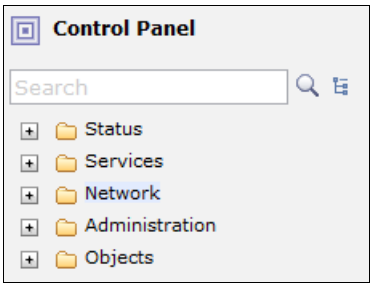


Figure 6-5 Search field on DataPower

3. Type the pattern `AAA` in the field, as shown in Figure 6-6.

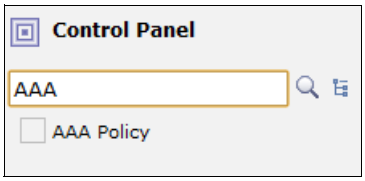


Figure 6-6 AAA policy search

4. Click the **AAA Policy** link to access the Configure AAA Policy user interface, as shown in Figure 6-7.

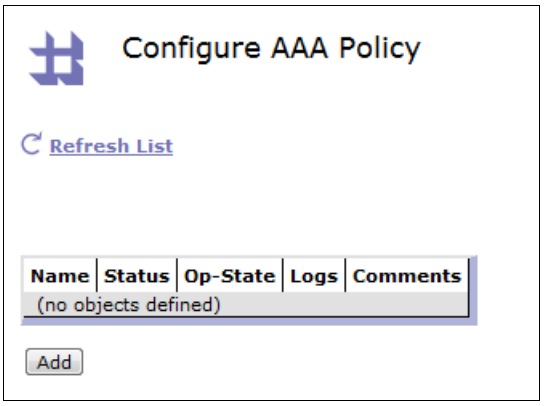


Figure 6-7 Configure AAA policy panel

5. Click **Add** to create a new AAA policy.
6. Enter a name for the AAA policy, for instance `AAAPolicyExample`, as shown in Figure 6-8.

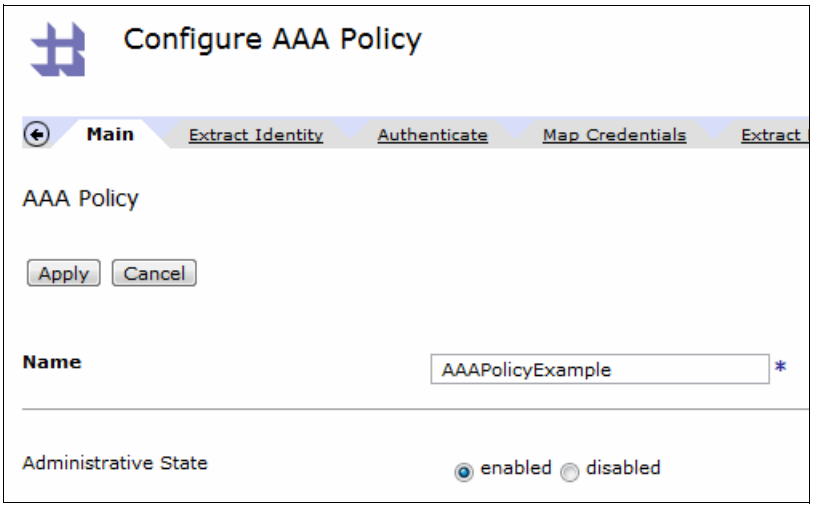


Figure 6-8 Naming a AAA policy

7. Click the **Extract Identity** tab, and select the **Token Extracted from the Message** method of identity data to extract, as shown in Figure 6-9.

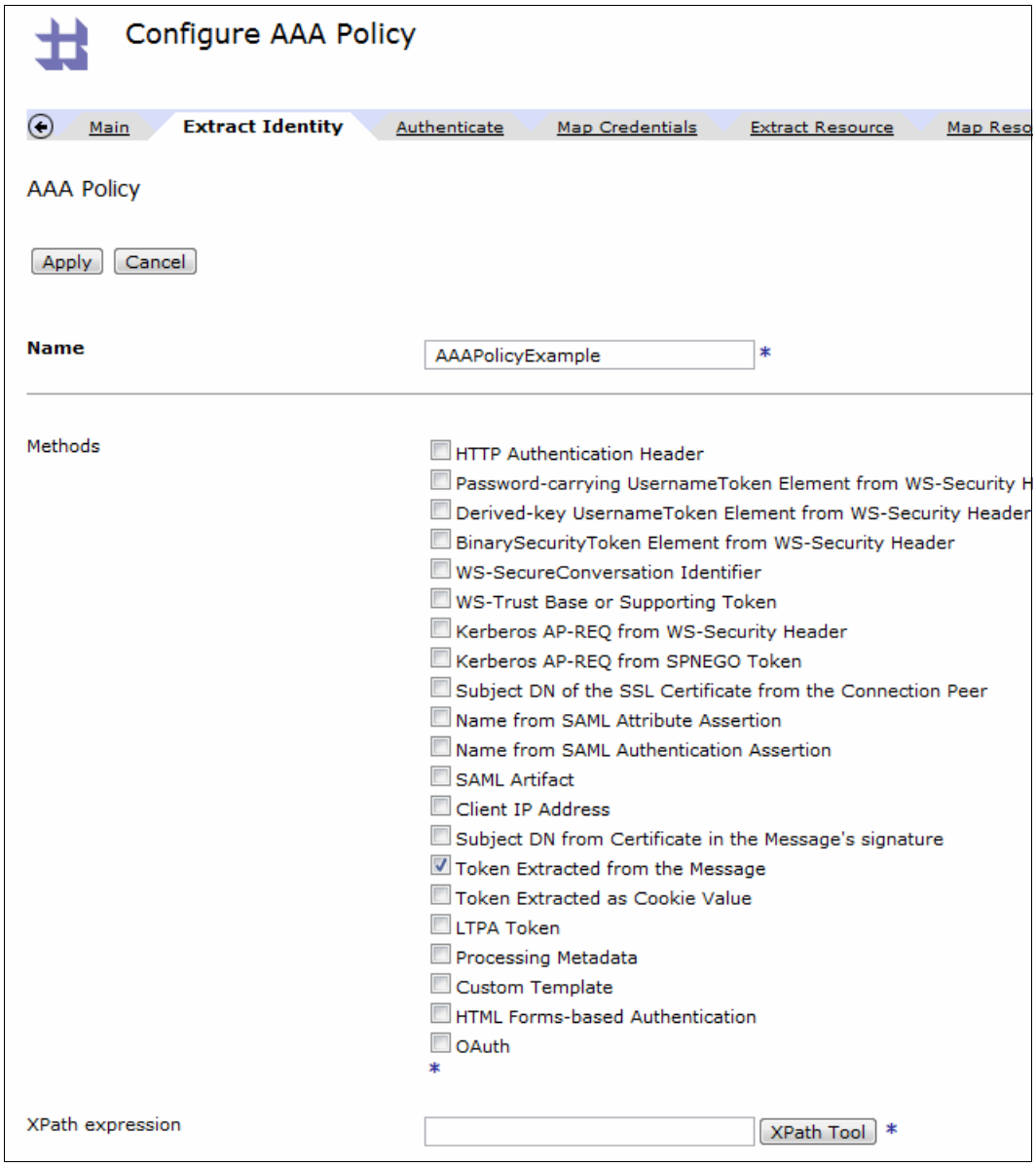


Figure 6-9 Extract identity step of the AAA policy

8. In the Xpath expression text field, enter the following value:

```
/*[local-name()='Envelope']/*[local-name()='Header']/*[local-name()='ConsumerIdentifier']
```

The identity you want to use for authentication is the consumer identifer. The XPath expression is only a way to extract this identity from the incoming SOAP messages.

9. Click the **Authenticate** tab of the AAA policy and select the **Use DataPower AAA Info File** authentication method, as shown in Figure 6-10.

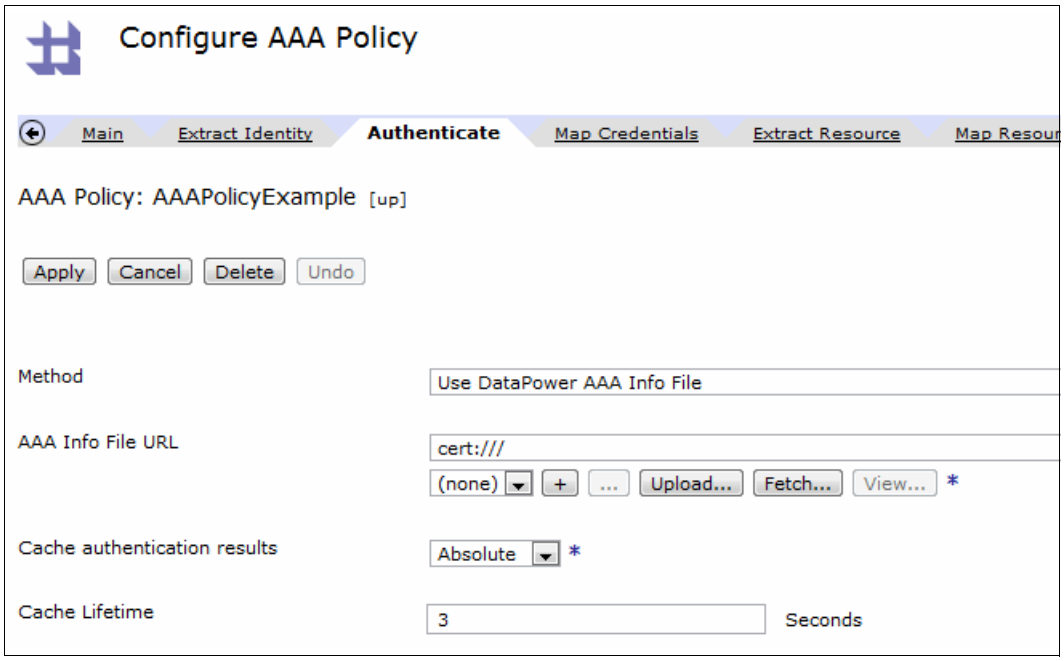


Figure 6-10 Authentication method of the AAA policy

10. To specify the AAA Info File URL property, select the **`local:///`** URL protocol and click **Upload**, as shown in Figure 6-11.

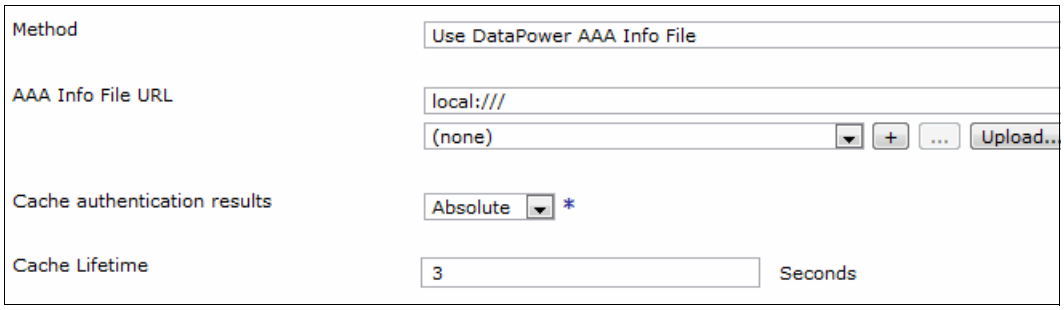


Figure 6-11 Select local directory to upload the AAAInfo.xml file

The `AAAInfo.xml` file on the `local:///` directory of the current domain is uploaded, as shown in Figure 6-12.

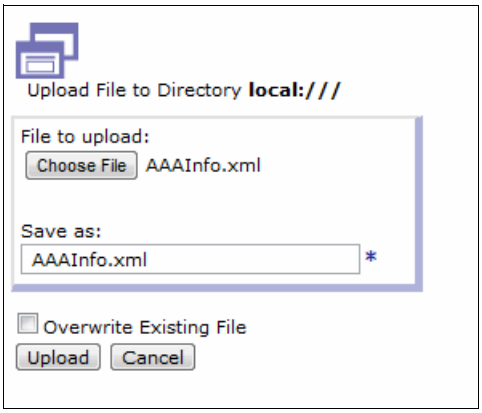


Figure 6-12 Browse to select the AAAInfo.xml file

11. Click the **Extract Resource** tab of the AAA policy and select one of the items used to identify the resource. For instance, select **Local Name of Request Element** and then click **Apply**. The resource is used during the authorization step of the policy. Because we do not do any authorization, we can select any proposed values.
12. After the AAA policy is created, it is displayed in the list of the AAA policies of the current domain, as shown in Figure 6-13.

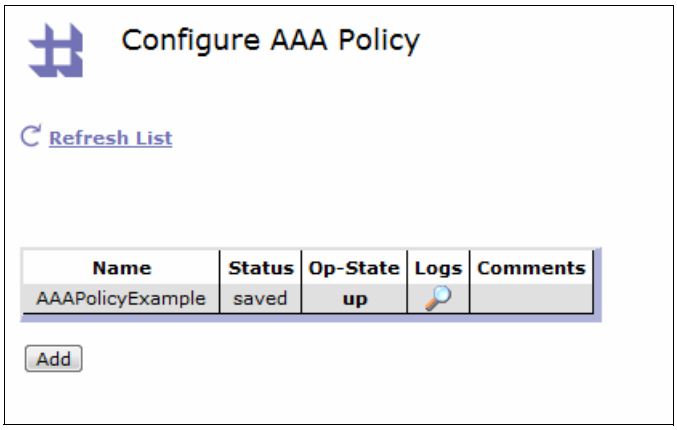


Figure 6-13 List of configured AAA policies

6.4.2 Exporting the DataPower configuration artifacts

After the AAA policy is configured, export it to integrate its configuration in the custom policy style sheet. The `export.xml` file of the archive (result of the export process) contains the AAA configuration pattern that can be included in the custom style sheet.

Use the following steps to create a downloadable archive of the AAA policy action (created in 6.4.1, “Creating the DataPower configuration artifacts” on page 198):

1. Connect to a DataPower appliance with your credentials (login and password). Select the domain on which the AAA policy example is defined.
2. Click the **Export Configuration** icon, shown in Figure 6-14 on page 204.

Figure 6-14 Export configuration icon

3. Select **Export configuration and files of the current domain** from the list of current exportation capabilities, as shown in Figure 6-15.

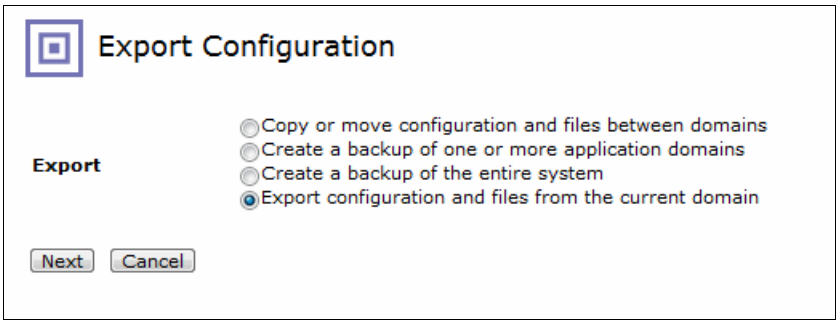


Figure 6-15 Export configuration and files of the current domain

4. Click **Next**.
5. Modify the export file name and set it to `export_AAAPolicy` name.

   Use the following steps to finalize the exportation process:

   a. In the list of configuration objects to export, select **AAA Policy** and then select the `AAAPolicyExample` policy.

   b. Click the right angle (>) button to move the `AAAPolicyExample` into the list of selected objects, as shown in Figure 6-16.

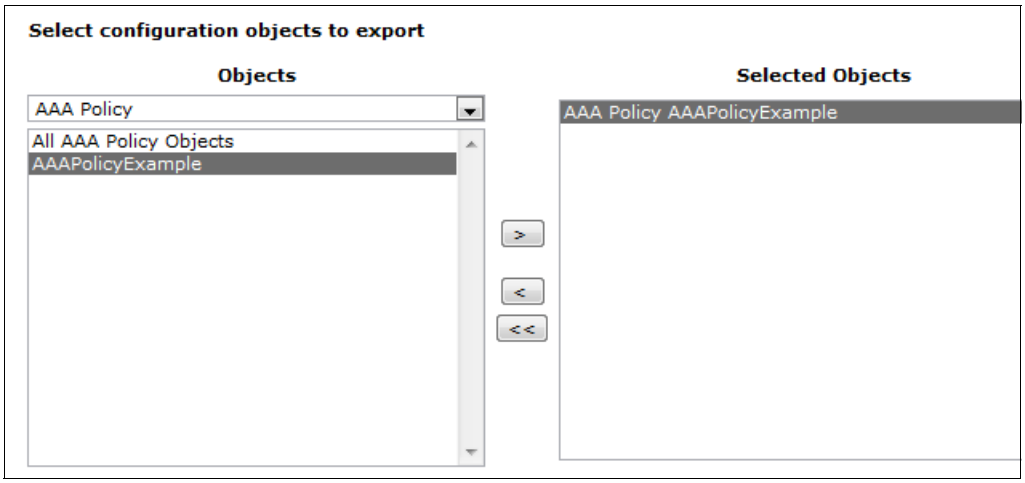


Figure 6-16 Select the AAA policy to export

   c. On the Export Files property, select **Export files referenced by selected objects**.

   d. Click **Next**.

6. Click **Download** to download the export file, as shown in Figure 6-17.

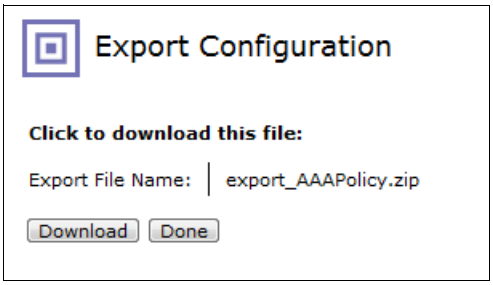


Figure 6-17 Download the export file

7. Click **Done** when the download task completes.
8. Extract the `export.xml` file from the `export_AAAPolicy.zip` archive and open it with an XML editor or your integrated development environment.

The element of interest in the `export.xml` file in the archive, is the `<AAAPolicy name="AAAPolicyExample"...>` element.

Example 6-5 is a complete configuration element, based on the AAA policy created in 6.4.1, “Creating the DataPower configuration artifacts” on page 198.

Example 6-5 AAAPolicy configuration element

```
<AAAPolicy name="AAAPolicyExample" xmlns:env="http://www.w3.org/2003/05/soap-envelope"
          xmlns:dp="http://www.datapower.com/schemas/management">
          <mAdminState>enabled</mAdminState>
          <ExtractIdentity>
             <EIBitmap>
                <http-basic-auth>off</http-basic-auth>
                <wssec-username>off</wssec-username>
                <wssec-derived-key>off</wssec-derived-key>
                <wssec-binary-token>off</wssec-binary-token>
                <ws-secure-conversation>off</ws-secure-conversation>
                <ws-trust>off</ws-trust>
                <kerberos>off</kerberos>
                <kerberos-spnego>off</kerberos-spnego>
                <client-ssl>off</client-ssl>
                <saml-attr-name>off</saml-attr-name>
                <saml-authen-name>off</saml-authen-name>
                <saml-artifact>off</saml-artifact>
                <client-ip-address>off</client-ip-address>
                <signer-dn>off</signer-dn>
                <token>on</token>
                <cookie-token>off</cookie-token>
                <ltpa>off</ltpa>
                <metadata>off</metadata>
                <custom>off</custom>
                <html-forms-auth>off</html-forms-auth>
                <oauth>off</oauth>
             </EIBitmap>
             <EICustomURL />

<EIXPath>/*[local-name()='Envelope']/*[local-name()='Header']/*[local-name()='ConsumerIdent
ifier']</EIXPath>
             <EISignerDNValcred />
             <EICookieName />
```

```
      <EIBasicAuthRealm>login</EIBasicAuthRealm>
      <EIUseWSSec>off</EIUseWSSec>
      <EIMetadata />
      <EIAllowRemoteTokenReference>off</EIAllowRemoteTokenReference>
      <EIRemoteTokenProcessService />
      <EIPasswordRetrievalMechanism>xmlfile</EIPasswordRetrievalMechanism>
      <EIPasswordRetrievalCustomURL />
      <EIPasswordRetrievalAAAInfoURL />
      <EISSLProxyProfile />
      <EIFormsLoginPolicy />
      <EIOAuthClientGroup />
   </ExtractIdentity>
   <Authenticate>
      <AUMethod>xmlfile</AUMethod>
      <AUCustomURL />
      <AUMapURL>local:///AAAInfo.xml</AUMapURL>
      <AUHost />
      <AUPort>1234</AUPort>
      <AUSSLValcred />
      <AUCacheAllow>absolute</AUCacheAllow>
      <AUCacheTTL>3</AUCacheTTL>
      <AUKerberosPrincipal />
      <AUKerberosPassword />
      <AUClearTrustServerURL />
      <AUClearTrustApplication />
      <AUSAMLArtifactResponder />
      <AUKerberosVerifySignature>on</AUKerberosVerifySignature>
      <AUNetegrityBaseURI />
      <AUSAMLAuthQueryServer />
      <AUSAMLVersion>1.1</AUSAMLVersion>
      <AULDAPPrefix>cn=</AULDAPPrefix>
      <AULDAPSuffix />
      <AULDAPLoadBalanceGroup />
      <AUKerberosKeytab />
      <AUWSTrustURL />
      <AUSAML2Issuer />
      <AUSignerValcred />
      <AUSignedXPath />
      <AUSSLProxyProfile />
      <AUNetegrityConfig />
      <AULDAPBindDN />
      <AULDAPBindPassword />
      <AULDAPSearchAttribute>userPassword</AULDAPSearchAttribute>
      <AULTPATokenVersionsBitmap>
         <LTPA>off</LTPA>
         <LTPA2>on</LTPA2>
         <LTPADomino>off</LTPADomino>
      </AULTPATokenVersionsBitmap>
      <AULTPAKeyFile />
      <AULTPAKeyFilePassword />
      <AULTPAStashFile />
      <AUBinaryTokenX509Valcred />
      <AUTAMServer />
      <AUAllowRemoteTokenReference>off</AUAllowRemoteTokenReference>
      <AURemoteTokenProcessService />
      <AUWSTrustVersion>1.2</AUWSTrustVersion>
      <AULDAPSearchForDN>off</AULDAPSearchForDN>
      <AULDAPSearchParameters />
      <AUWSTrustRequireClientEntropy>off</AUWSTrustRequireClientEntropy>
      <AUWSTrustClientEntropySize>32</AUWSTrustClientEntropySize>
```

```
    <AUWSTrustRequireServerEntropy>off</AUWSTrustRequireServerEntropy>
    <AUWSTrustServerEntropySize>32</AUWSTrustServerEntropySize>
    <AUWSTrustRequireRSTC>off</AUWSTrustRequireRSTC>
    <AUWSTrustRequireAppliesToHeader>off
    </AUWSTrustRequireAppliesToHeader>
    <AUWSTrustAppliesToHeader />
    <AUWSTrustEncryptionCertificate />
    <AUZOSNSSConfig />
    <AULDAPAttributes />
    <AUSkewTime>0</AUSkewTime>
    <AUTAMPACReturn>off</AUTAMPACReturn>
</Authenticate>
<MapCredentials>
    <MCMethod>none</MCMethod>
    <MCCustomURL />
    <MCMapURL />
    <MCMapXPath />
    <MCTFIMEndpoint />
</MapCredentials>
<ExtractResource>
    <ERBitmap>
        <target-url>off</target-url>
        <original-url>off</original-url>
        <request-uri>off</request-uri>
        <request-opname>on</request-opname>
        <http-method>off</http-method>
        <XPath>off</XPath>
        <metadata>off</metadata>
    </ERBitmap>
    <ERXPath />
    <ERMetadata />
</ExtractResource>
<MapResource>
    <MRMethod>none</MRMethod>
    <MRCustomURL />
    <MRMapURL />
    <MRMapXPath />
    <MRTAMMap>WebSEAL</MRTAMMap>
    <MRTAMInstancePrefix />
    <MRTAMWebSEALDynURLFile />
</MapResource>
<Authorize>
    <AZMethod>anyauthenticated</AZMethod>
    <AZCustomURL />
    <AZMapURL />
    <AZHost />
    <AZPort>1234</AZPort>
    <AZLDAPGroup />
    <AZValcred />
    <AZSAMLURL />
    <AZSAMLType>any</AZSAMLType>
    <AZSAMLXPath />
    <AZSAMLNameQualifier />
    <AZCacheAllow>absolute</AZCacheAllow>
    <AZCacheTTL>3</AZCacheTTL>
    <AZNetegrityBaseURI />
    <AZNetegrityOpNameExtension />
    <AZClearTrustServerURL />
    <AZSAMLVersion>1.1</AZSAMLVersion>
    <AZLDAPLoadBalanceGroup />
```

```
      <AZLDAPBindDN />
      <AZLDAPBindPassword />
      <AZLDAPGroupAttribute>member</AZLDAPGroupAttribute>
      <AZSSLProxyProfile />
      <AZNetegrityConfig />
      <AZLDAPSearchScope>subtree</AZLDAPSearchScope>
      <AZLDAPSearchFilter>(objectClass=*)</AZLDAPSearchFilter>
      <AZXACMLVersion>2.0</AZXACMLVersion>
      <AZXACMLPEPType>deny-biased</AZXACMLPEPType>
      <AZXACMLUseOnBoxPDP>on</AZXACMLUseOnBoxPDP>
      <AZXACMLPDP />
      <AZXACMLExternalPDPUrl />
      <AZXACMLBindingMethod>custom</AZXACMLBindingMethod>
      <AZXACMLBindingObject />
      <AZXACMLBindingXSL />
      <AZXACMLCustomObligation />
      <AZXACMLUseSAML2>off</AZXACMLUseSAML2>
      <AZTAMServer />
      <AZTAMDefaultAction>T</AZTAMDefaultAction>
      <AZTAMActionResourceMap />
      <AZXACMLUseSOAP>off</AZXACMLUseSOAP>
      <AZZOSNSSConfig />
      <AZSAFDefaultAction>r</AZSAFDefaultAction>
      <AZLDAPAttributes />
      <AZSkewTime>0</AZSkewTime>
      <AZOAuthValidationEndpointType>tfim</AZOAuthValidationEndpointType>
      <AZTFIMEndpoint />
      <AZOAuthEnforceScope>off</AZOAuthEnforceScope>
      <AZOAuthExportHeaders>on</AZOAuthExportHeaders>
      <AZTAMPACReturn>off</AZTAMPACReturn>
      <AZTAMPACUse>off</AZTAMPACUse>
   </Authorize>
   <PostProcess>
      <PPEnabled>off</PPEnabled>
      <PPCustomURL />
      <PPSAMLAuthAssertion>off</PPSAMLAuthAssertion>
      <PPSAMLServerName>XS</PPSAMLServerName>
      <PPSAMLNameQualifier />
      <PPKerberosTicket>off</PPKerberosTicket>
      <PPKerberosClient />
      <PPKerberosClientPassword />
      <PPKerberosServer />
      <PPWSTrust>off</PPWSTrust>
      <PPTimestamp>on</PPTimestamp>
      <PPTimestampExpiry>0</PPTimestampExpiry>
      <PPAllowRenewal>off</PPAllowRenewal>
      <PPSAMLVersion>2.0</PPSAMLVersion>
      <PPSAMLSendSLO>off</PPSAMLSendSLO>
      <PPSAMLSLOEndpoint />
      <PPSSLProxyProfile />
      <PPWSUsernameToken>off</PPWSUsernameToken>
      <PPWSUsernameTokenPasswordType>Digest
      </PPWSUsernameTokenPasswordType>
      <PPSAMLValidity>0</PPSAMLValidity>
      <PPSAMLSkew>0</PPSAMLSkew>
      <PPWSUsernameTokenIncludePwd>on</PPWSUsernameTokenIncludePwd>
      <PPLTPA>off</PPLTPA>
      <PPLTPAVersion>LTPA2</PPLTPAVersion>
      <PPLTPAExpiry>600</PPLTPAExpiry>
      <PPLTPAKeyFile />
```

```
                <PPLTPAKeyFilePassword />
                <PPLTPAStashFile />
                <PPKerberosSPNEGOToken>off</PPKerberosSPNEGOToken>

<PPKerberosBstValueType>http://docs.oasis-open.org/wss/oasis-wss-kerberos-token-profile-1.1
#GSS_Kerberosv5_AP_REQ
                </PPKerberosBstValueType>
                <PPSAMLUseWSSec>off</PPSAMLUseWSSec>
                <PPKerberosClientKeytab />
                <PPUseWSSec>off</PPUseWSSec>
                <PPActorRoleID />
                <PPTFIMTokenMapping>off</PPTFIMTokenMapping>
                <PPTFIMEndpoint />
                <PPWSDerivedKeyUsernameToken>off</PPWSDerivedKeyUsernameToken>
                <PPWSDerivedKeyUsernameTokenIterations>1000
                </PPWSDerivedKeyUsernameTokenIterations>
                <PPWSUsernameTokenAllowReplacement>off
                </PPWSUsernameTokenAllowReplacement>
                <PPTFIMReplaceMethod>all</PPTFIMReplaceMethod>
                <PPTFIMRetrieveMode>CallTFIM</PPTFIMRetrieveMode>
                <PPHMACSigningAlg>hmac-sha1</PPHMACSigningAlg>
                <PPSigningHashAlg>sha1</PPSigningHashAlg>
                <PPWSTrustHeader>off</PPWSTrustHeader>
                <PPWSSCKeySource>random</PPWSSCKeySource>
                <PPSharedSecretKey />
                <PPWSTrustRenewalWait>0</PPWSTrustRenewalWait>
                <PPWSTrustNewInstance>off</PPWSTrustNewInstance>
                <PPWSTrustNewKey>off</PPWSTrustNewKey>
                <PPWSTrustNeverExpire>off</PPWSTrustNeverExpire>
                <PPICRXToken>off</PPICRXToken>
                <PPICRXUserRealm />
                <PPSAMLIdentityProvider>off</PPSAMLIdentityProvider>
                <PPSAMLProtocol>assertion</PPSAMLProtocol>
                <PPSAMLResponseDestination />
                <PPResultWrapup>wssec-replace</PPResultWrapup>
                <PPSAMLAssertionType>
                   <authentication>on</authentication>
                   <attribute>on</attribute>
                   <authorization>off</authorization>
                </PPSAMLAssertionType>
                <PPSAMLSubjectConfirm>bearer</PPSAMLSubjectConfirm>
                <PPSAMLNameID>on</PPSAMLNameID>
                <PPSAMLNameIDFormat />
                <PPSAMLRecipient />
                <PPSAMLAudience />
                <PPSAMLOmitNotBefore>off</PPSAMLOmitNotBefore>
                <PPOneTimeUse>off</PPOneTimeUse>
                <PPSAMLProxy>off</PPSAMLProxy>
                <PPSAMLProxyAudience />
                <PPSAMLProxyCount>0</PPSAMLProxyCount>
                <PPSAMLAuthzAction>AllHTTP</PPSAMLAuthzAction>
                <PPSAMLAttributes />
                <PPLTPAInsertCookie>on</PPLTPAInsertCookie>
                <PPTAMPACPropagate>off</PPTAMPACPropagate>
                <PPTAMHeader>iv-creds</PPTAMHeader>
                <PPTAMHeaderSize>0</PPTAMHeaderSize>
             </PostProcess>
             <SAMLSigningHashAlg>sha1</SAMLSigningHashAlg>
             <SAMLSigningAlg>rsa</SAMLSigningAlg>
             <LDAPsuffix />
```

```
    <LogAllowed>on</LogAllowed>
    <LogAllowedLevel>info</LogAllowedLevel>
    <LogRejected>on</LogRejected>
    <LogRejectedLevel>warn</LogRejectedLevel>
    <PingIdentityCompatibility>off</PingIdentityCompatibility>
    <DoSValve>3</DoSValve>
    <LDAPVersion>v2</LDAPVersion>
    <EnforceSOAPActor>on</EnforceSOAPActor>
    <WSSecActorRoleID />
</AAAPolicy>
```

6.4.3 Writing the custom policy style sheet for security

Remember, the custom policy style sheet is responsible for generating the AAA policy artifacts, based on the custom security policy in Example 6-2 on page 194.

To write this custom mapping, you can use the style sheet in Example 13-1 on page 374, as a starting point.

This style sheet is inspired of the `jk-example.xsl` custom policy style sheet, which is provided in the `store:///policies/custom` directory of any DataPower appliance that uses firmware version 5.0.0 or later. The XSL style sheet that is created here has the following name:

`itso.customPolicy.example.xsl`

Important: The matching templates, other than the following one, must not be modified:

```
<xsl:template match="myNamespace:MyAssertion" mode="assertion">
```

New matching and named templates, and also extension functions and DataPower Policy parameters, may also be created.

Use the following steps to complete the coding of the style sheet so it can be used to generate DataPower configuration artifacts based on the custom security policy shown in Example 6-2 on page 194.

1. Declare the `itso` prefix, bound to the namespace of the custom security policy domain.
2. Add the namespace declaration into the `</dppolicy:domain>` element.
3. Create a `getAAAPolicy()` function in the style sheet. This function is used to generate a valid AAA policy configuration, as required by the custom security policy.
4. Create a matching template, which matches the qualified element of the custom security policy:

   `{http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11}ControlConsumerAccess`

   The result of this matching template is the generation of a `<dppolicy:config>` element.

   The `<dppolicy:config>` element contains the following artifacts:

   a. The configuration of the required AAA policy, as the matching template uses the `getAAAPolicy()` function.

   b. The configuration of a `<StylePolicyAction>` element, which references the AAA policy, generated in 6.4.1, “Creating the DataPower configuration artifacts” on page 198.

   c. Alternatively, the `<StylePolicyAction>` that is created in this template can also reference a fictitious predefined AAA policy that already exists in the same DataPower domain. In this case, step 3 and step 4a must not be executed.

d. The direction of the processing rule, on which the AAA policy must be created is indicated through the `direction` attribute of the `<dppolicy:config>` element. In this case, the processing rule is a request rule; therefore we use the `request-rule` value. For the list of available values of the `direction` attribute, see Chapter 13, “Creating and using custom policies” on page 367.

   The matching template also contains XSL directives that are used to log several important steps of the creation process. An XML file, which prefixed with the `MyAssertion-config-` pattern, is generated in the `temporary:///` directory in the `default` domain. A good practice is to modify the `MyAssertion` pattern with the name of the assertion that is related to the current matching template.

5. The authored custom policy XSL style sheet is now ready to be loaded on the DataPower device used to enforce the custom security policy.

   The final version of the `itso.customPolicy.aaa.xsl` is shown in Example 6-6.

Example 6-6 Custom policy XSL style sheet for AAA policy configuration artifacts generation

```
<?xml version="1.0"?>
<!-- +
     |*****************************************************
     |*** Author: ITSO Redbooks Travel Company - gauci@fr.ibm.com
     |*** file: itso.customPolicy.aaa.xsl
     |*** Description: Custom policy XSL styleheet for aaa policy enforcement
     |*** Revision: 1.0: Initial version
     |*****************************************************
    +-->

<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:func="http://exslt.org/functions"
        xmlns:dpe="http://www.datapower.com/extensions"
        xmlns:dppolicy="http://www.datapower.com/policy"
        xmlns:dpconfig="http://www.datapower.com/param/config"
        xmlns:dpfunc="http://www.datapower.com/extensions/functions"
        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:itso="http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11"
        extension-element-prefixes="dpe"
        exclude-result-prefixes="dpe dpconfig dppolicy wsp itso">

    <!-- ******************************************************* -->
    <!-- (1) Declare the policy domain this stylesheet implements -->
    <!-- ******************************************************* -->

    <dpe:summary xmlns="">
        <!-- Comment the following line to prevent the policy domain from being
processed -or- remove the comment if you want to process the policy domain. -->

<dppolicy:domain>http://itso.ibm.com/ibmredbooks-travelcompany/aaa/2012-11</dppolicy:dom
ain>
        <operation>xform</operation>
        <description>Implements policy assertions for IBM Redbooks Travel
Company</description>
    </dpe:summary>

    <!-- the policy domain namespace the stylesheet is executed for -->
    <xsl:variable name="seqno"
select="/dppolicy:request/dppolicy:header/dppolicy:SequenceNo"/>
    <xsl:variable name="nsuri"
select="/dppolicy:request/dppolicy:sequence/DomainNamespace[position()=$seqno]/@uri"/>
```

```
    <!-- The following global variables represent the input document -->
    <!-- header with aux information -->
    <xsl:variable name="header" select="/dppolicy:request/dppolicy:header"/>
    <!-- configured policy bindings defined as dpe:param above (Policy Parameters)-->
    <xsl:variable name="bindings" select="/dppolicy:request/dppolicy:bindings"/>
    <!-- ws-policy alternative -->
    <xsl:variable name="policy" select="/dppolicy:request/dppolicy:policy"/>
    <!-- previously generated configuration for this policy alternative -->
    <xsl:variable name="configuration"
select="/dppolicy:request/dppolicy:configuration"/>
    <!-- general notepad to pass information between processing steps of one alternative
-->
    <xsl:variable name="notepad" select="/dppolicy:request/dppolicy:notepad"/>

    <!-- helper variables -->
    <xsl:variable name="rD" select="$header/dppolicy:RequestDomain"/>
    <xsl:variable name="lT" select="$header/dppolicy:LogType"/>
    <xsl:variable name="lC" select="$header/dppolicy:LogClass"/>
    <xsl:variable name="rO" select="$header/dppolicy:RequestObject"/>

    <!--+
       |*****************************
       |*** Matching Template
       |*** Element: ROOT
       |*****************************
       +-->
    <xsl:template match="/">

    <!-- Log -->
      <xsl:message dpe:priority="error">Entering the root template</xsl:message>

        <!-- Get the Input context name -->
        <xsl:choose>
            <xsl:when test="string-length($notepad/shared-context/in-context) > 0">
                <dpe:set-local-variable name="'input-context'"
value="$notepad/shared-context/in-context"/>
                <xsl:message dpe:priority="debug" dpe:domain="{$rD}" dpe:type="{$lT}"
dpe:class="{$lC}" dpe:object="{$rO}">Input context=<xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:message>
            </xsl:when>
            <xsl:otherwise>
                <xsl:message terminate="yes" dpe:priority="error" dpe:domain="{$rD}"
dpe:type="{$lT}" dpe:class="{$lC}" dpe:object="{$rO}">Cannot find the input
context</xsl:message>
            </xsl:otherwise>
        </xsl:choose>

        <!-- Get the Output context name -->
        <xsl:choose>
            <xsl:when test="string-length($notepad/shared-context/out-context) > 0">
                <dpe:set-local-variable name="'output-context'"
value="$notepad/shared-context/out-context"/>
                <xsl:message dpe:priority="debug" dpe:domain="{$rD}" dpe:type="{$lT}"
dpe:class="{$lC}" dpe:object="{$rO}">Output context=<xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:message>
            </xsl:when>
            <xsl:otherwise>
```

```
                <xsl:message terminate="yes" dpe:priority="error" dpe:domain="{$rD}"
dpe:type="{$lT}" dpe:class="{$lC}" dpe:object="{$rO}">Cannot find the output
context</xsl:message>
            </xsl:otherwise>
        </xsl:choose>

         <!-- process single alternative -->
         <xsl:apply-templates select="$policy/*[local-name()='All']"/>

        <!-- Logs -->
      <xsl:message dpe:priority="error">Exiting the root template aaa</xsl:message>
    </xsl:template>

   <!--+
       |*****************************
       |*** Matching Template
       |*** Element: All
       |*****************************
       +-->
   <!-- process each assertion in the alternative -->
    <xsl:template match="*[local-name()='All']">
        <xsl:apply-templates mode="assertion"/>
    </xsl:template>

     <!-- my domain assertions -->

     <!--+
       |*****************************
       |*** Matching Template
       |*** Element: itso:ControlConsumerAccess
       |*** Mode: assertion
       |*****************************
       +-->
    <xsl:template mode="assertion" match="itso:ControlConsumerAccess">
        <xsl:message dpe:priority="error">Enter template ControlConsumerAccess:
PolicyID=<xsl:value-of select="$header/dppolicy:PolicyID"/></xsl:message>

        <!-- ************************************************************** -->
        <!-- (5) The configuration for assertion ControlConsumerAcces         -->
        <!-- ************************************************************** -->
        <!-- generate processing action to execute (assertionNo = order of processing)
-->
        <xsl:variable name="config">
            <dppolicy:config uri="{$nsuri}" assertionNo="{position()}"
direction="{'request-rule'}">

                <!-- AAA Policy configuration can be large so it is a good idea to
generate separate from the Actions -->
                <xsl:copy-of select="dpfunc:GetAAAPolicy()"/>

                <!-- StylePolicyAction. -->
                <xsl:variable name="actionNameValidate"
select="concat($header/dppolicy:PolicyID,'-call-', position(),'-EnforceAAA')"/>
                <xsl:element name="StylePolicyAction">
                    <xsl:attribute name="name"><xsl:value-of
select="$actionNameValidate"/></xsl:attribute>
                    <xsl:element name="Type"><xsl:text>aaa</xsl:text></xsl:element>
                    <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
```

```
                    <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                    <xsl:element
name="NamedInOutLocationType"><xsl:text>default</xsl:text></xsl:element>
                    <xsl:element name="AAA">
                        <xsl:attribute
name="class"><xsl:text>AAAPolicy</xsl:text></xsl:attribute>
                        <!-- reference the name of the AAA policy to be used.-->
                      <xsl:text>AAAPolicyExample</xsl:text>

                    </xsl:element>
                </xsl:element>
        </dppolicy:config>
        <!-- Log -->
           <xsl:message dpe:priority="error">Exit template
ControlConsumerAccess</xsl:message>
      </xsl:variable>

        <!-- Send the config to the console, whatever will fit (roughly 3k) -->
        <xsl:message dpe:priority="error">Exit template ControlConsumerAccess - Config:
           <xsl:copy-of select="$config"/>
        </xsl:message>

        <!-- Send the config to the temporary://  -->
        <xsl:variable name="filename"
select="concat('ControlConsumerAccess-config-',$header/dppolicy:PolicyID, '.xml')"/>
        <dpe:dump-nodes file="$filename" nodes="$config"/>

        <!-- Send the config to output -->
        <xsl:copy-of select="$config"/>

        <!-- Log -->
      <xsl:message dpe:priority="error">Exiting the ControlConsumerAccess
template</xsl:message>

    </xsl:template>

    <!--+
       |*****************************
       |*** Matching Template
       |*** Element: text()
       |*****************************
       +-->
    <xsl:template match="text()"/>


    <!--+
       |*****************************
       |*** Function
       |*** Name: dpfunc:GetAAAPolicy
       |*****************************
       +-->
    <func:function name="dpfunc:GetAAAPolicy">

    <xsl:variable name="result">
          <!-- AAA policy configuration, as exported from the export.xml file -->
          <AAAPolicy name="AAAPolicyExample">
             <mAdminState>enabled</mAdminState>
             <ExtractIdentity>
             <EIBitmap>
```

```
                    <http-basic-auth>off</http-basic-auth>
                    <wssec-username>off</wssec-username>
                    <wssec-derived-key>off</wssec-derived-key>
                    <wssec-binary-token>off</wssec-binary-token>
                    <ws-secure-conversation>off</ws-secure-conversation>
                    <ws-trust>off</ws-trust>
                    <kerberos>off</kerberos>
                    <kerberos-spnego>off</kerberos-spnego>
                    <client-ssl>off</client-ssl>
                    <saml-attr-name>off</saml-attr-name>
                    <saml-authen-name>off</saml-authen-name>
                    <saml-artifact>off</saml-artifact>
                    <client-ip-address>off</client-ip-address>
                    <signer-dn>off</signer-dn>
                    <token>on</token>
                    <cookie-token>off</cookie-token>
                    <ltpa>off</ltpa>
                    <metadata>off</metadata>
                    <custom>off</custom>
                    <html-forms-auth>off</html-forms-auth>
                    <oauth>off</oauth></EIBitmap>
                    <EICustomURL/>

<EIXPath>/*[local-name()='Envelope']/*[local-name()='Header']/*[local-name()='ConsumerId
entifier']</EIXPath>
                    <EISignerDNValcred/>
                    <EICookieName/>
                    <EIBasicAuthRealm>login</EIBasicAuthRealm>
                    <EIUseWSSec>off</EIUseWSSec>
                    <EIMetadata/>
                    <EIAllowRemoteTokenReference>off</EIAllowRemoteTokenReference>
                    <EIRemoteTokenProcessService/>
                    <EIPasswordRetrievalMechanism>xmlfile</EIPasswordRetrievalMechanism>
                    <EIPasswordRetrievalCustomURL/>
                    <EIPasswordRetrievalAAAInfoURL/>
                    <EISSLProxyProfile/>
                    <EIFormsLoginPolicy/>
                    <EIOAuthClientGroup/></ExtractIdentity>
                    <Authenticate>
                    <AUMethod>xmlfile</AUMethod>
                    <AUCustomURL/>
                    <AUMapURL>local:///AAAInfo.xml</AUMapURL>
                    <AUHost/>
                    <AUPort>1234</AUPort>
                    <AUSSLValcred/>
                    <AUCacheAllow>absolute</AUCacheAllow>
                    <AUCacheTTL>3</AUCacheTTL>
                    <AUKerberosPrincipal/>
                    <AUKerberosPassword/>
                    <AUClearTrustServerURL/>
                    <AUClearTrustApplication/>
                    <AUSAMLArtifactResponder/>
                    <AUKerberosVerifySignature>on</AUKerberosVerifySignature>
                    <AUNetegrityBaseURI/>
                    <AUSAMLAuthQueryServer/>
                    <AUSAMLVersion>1.1</AUSAMLVersion>
                    <AULDAPPrefix>cn=</AULDAPPrefix>
                    <AULDAPSuffix/>
                    <AULDAPLoadBalanceGroup/>
                    <AUKerberosKeytab/>
```

```
<AUWSTrustURL/>
<AUSAML2Issuer/>
<AUSignerValcred/>
<AUSignedXPath/>
<AUSSLProxyProfile/>
<AUNetegrityConfig/>
<AULDAPBindDN/>
<AULDAPBindPassword/>
<AULDAPSearchAttribute>userPassword</AULDAPSearchAttribute>
<AULTPATokenVersionsBitmap>
<LTPA>off</LTPA>
<LTPA2>on</LTPA2>
<LTPADomino>off</LTPADomino></AULTPATokenVersionsBitmap>
<AULTPAKeyFile/>
<AULTPAKeyFilePassword/>
<AULTPAStashFile/>
<AUBinaryTokenX509Valcred/>
<AUTAMServer/>
<AUAllowRemoteTokenReference>off</AUAllowRemoteTokenReference>
<AURemoteTokenProcessService/>
<AUWSTrustVersion>1.2</AUWSTrustVersion>
<AULDAPSearchForDN>off</AULDAPSearchForDN>
<AULDAPSearchParameters/>
<AUWSTrustRequireClientEntropy>off</AUWSTrustRequireClientEntropy>
<AUWSTrustClientEntropySize>32</AUWSTrustClientEntropySize>
<AUWSTrustRequireServerEntropy>off</AUWSTrustRequireServerEntropy>
<AUWSTrustServerEntropySize>32</AUWSTrustServerEntropySize>
<AUWSTrustRequireRSTC>off</AUWSTrustRequireRSTC>
<AUWSTrustRequireAppliesToHeader>off</AUWSTrustRequireAppliesToHeader>
<AUWSTrustAppliesToHeader/>
<AUWSTrustEncryptionCertificate/>
<AUZOSNSSConfig/>
<AULDAPAttributes/>
<AUSkewTime>0</AUSkewTime>
<AUTAMPACReturn>off</AUTAMPACReturn></Authenticate>
<MapCredentials>
<MCMethod>none</MCMethod>
<MCCustomURL/>
<MCMapURL/>
<MCMapXPath/>
<MCTFIMEndpoint/></MapCredentials>
<ExtractResource>
<ERBitmap>
<target-url>off</target-url>
<original-url>off</original-url>
<request-uri>off</request-uri>
<request-opname>on</request-opname>
<http-method>off</http-method>
<XPath>off</XPath>
<metadata>off</metadata></ERBitmap>
<ERXPath/>
<ERMetadata/></ExtractResource>
<MapResource>
<MRMethod>none</MRMethod>
<MRCustomURL/>
<MRMapURL/>
<MRMapXPath/>
<MRTAMMap>WebSEAL</MRTAMMap>
<MRTAMInstancePrefix/>
<MRTAMWebSEALDynURLFile/></MapResource>
```

```
<Authorize>
<AZMethod>anyauthenticated</AZMethod>
<AZCustomURL/>
<AZMapURL/>
<AZHost/>
<AZPort>1234</AZPort>
<AZLDAPGroup/>
<AZValcred/>
<AZSAMLURL/>
<AZSAMLType>any</AZSAMLType>
<AZSAMLXPath/>
<AZSAMLNameQualifier/>
<AZCacheAllow>absolute</AZCacheAllow>
<AZCacheTTL>3</AZCacheTTL>
<AZNetegrityBaseURI/>
<AZNetegrityOpNameExtension/>
<AZClearTrustServerURL/>
<AZSAMLVersion>1.1</AZSAMLVersion>
<AZLDAPLoadBalanceGroup/>
<AZLDAPBindDN/>
<AZLDAPBindPassword/>
<AZLDAPGroupAttribute>member</AZLDAPGroupAttribute>
<AZSSLProxyProfile/>
<AZNetegrityConfig/>
<AZLDAPSearchScope>subtree</AZLDAPSearchScope>
<AZLDAPSearchFilter>(objectClass=*)</AZLDAPSearchFilter>
<AZXACMLVersion>2.0</AZXACMLVersion>
<AZXACMLPEPType>deny-biased</AZXACMLPEPType>
<AZXACMLUseOnBoxPDP>on</AZXACMLUseOnBoxPDP>
<AZXACMLPDP/>
<AZXACMLExternalPDPUrl/>
<AZXACMLBindingMethod>custom</AZXACMLBindingMethod>
<AZXACMLBindingObject/>
<AZXACMLBindingXSL/>
<AZXACMLCustomObligation/>
<AZXACMLUseSAML2>off</AZXACMLUseSAML2>
<AZTAMServer/>
<AZTAMDefaultAction>T</AZTAMDefaultAction>
<AZTAMActionResourceMap/>
<AZXACMLUseSOAP>off</AZXACMLUseSOAP>
<AZZOSNSSConfig/>
<AZSAFDefaultAction>r</AZSAFDefaultAction>
<AZLDAPAttributes/>
<AZSkewTime>0</AZSkewTime>
<AZOAuthValidationEndpointType>tfim</AZOAuthValidationEndpointType>
<AZTFIMEndpoint/>
<AZOAuthEnforceScope>off</AZOAuthEnforceScope>
<AZOAuthExportHeaders>on</AZOAuthExportHeaders>
<AZTAMPACReturn>off</AZTAMPACReturn>
<AZTAMPACUse>off</AZTAMPACUse></Authorize>
<PostProcess>
<PPEnabled>off</PPEnabled>
<PPCustomURL/>
<PPSAMLAuthAssertion>off</PPSAMLAuthAssertion>
<PPSAMLServerName>XS</PPSAMLServerName>
<PPSAMLNameQualifier/>
<PPKerberosTicket>off</PPKerberosTicket>
<PPKerberosClient/>
<PPKerberosClientPassword/>
<PPKerberosServer/>
```

```
                    <PPWSTrust>off</PPWSTrust>
                    <PPTimestamp>on</PPTimestamp>
                    <PPTimestampExpiry>0</PPTimestampExpiry>
                    <PPAllowRenewal>off</PPAllowRenewal>
                    <PPSAMLVersion>2.0</PPSAMLVersion>
                    <PPSAMLSendSLO>off</PPSAMLSendSLO>
                    <PPSAMLSLOEndpoint/>
                    <PPSSLProxyProfile/>
                    <PPWSUsernameToken>off</PPWSUsernameToken>
                    <PPWSUsernameTokenPasswordType>Digest</PPWSUsernameTokenPasswordType>
                    <PPSAMLValidity>0</PPSAMLValidity>
                    <PPSAMLSkew>0</PPSAMLSkew>
                    <PPWSUsernameTokenIncludePwd>on</PPWSUsernameTokenIncludePwd>
                    <PPLTPA>off</PPLTPA>
                    <PPLTPAVersion>LTPA2</PPLTPAVersion>
                    <PPLTPAExpiry>600</PPLTPAExpiry>
                    <PPLTPAKeyFile/>
                    <PPLTPAKeyFilePassword/>
                    <PPLTPAStashFile/>
                    <PPKerberosSPNEGOToken>off</PPKerberosSPNEGOToken>

<PPKerberosBstValueType>http://docs.oasis-open.org/wss/oasis-wss-kerberos-token-profile-
1.1#GSS_Kerberosv5_AP_REQ</PPKerberosBstValueType>
                    <PPSAMLUseWSSec>off</PPSAMLUseWSSec>
                    <PPKerberosClientKeytab/>
                    <PPUseWSSec>off</PPUseWSSec>
                    <PPActorRoleID/>
                    <PPTFIMTokenMapping>off</PPTFIMTokenMapping>
                    <PPTFIMEndpoint/>
                    <PPWSDerivedKeyUsernameToken>off</PPWSDerivedKeyUsernameToken>

<PPWSDerivedKeyUsernameTokenIterations>1000</PPWSDerivedKeyUsernameTokenIterations>
                    <PPWSUsernameTokenAllowReplacement>off</PPWSUsernameTokenAllowReplacement>
                    <PPTFIMReplaceMethod>all</PPTFIMReplaceMethod>
                    <PPTFIMRetrieveMode>CallTFIM</PPTFIMRetrieveMode>
                    <PPHMACSigningAlg>hmac-sha1</PPHMACSigningAlg>
                    <PPSigningHashAlg>sha1</PPSigningHashAlg>
                    <PPWSTrustHeader>off</PPWSTrustHeader>
                    <PPWSSCKeySource>random</PPWSSCKeySource>
                    <PPSharedSecretKey/>
                    <PPWSTrustRenewalWait>0</PPWSTrustRenewalWait>
                    <PPWSTrustNewInstance>off</PPWSTrustNewInstance>
                    <PPWSTrustNewKey>off</PPWSTrustNewKey>
                    <PPWSTrustNeverExpire>off</PPWSTrustNeverExpire>
                    <PPICRXToken>off</PPICRXToken>
                    <PPICRXUserRealm/>
                    <PPSAMLIdentityProvider>off</PPSAMLIdentityProvider>
                    <PPSAMLProtocol>assertion</PPSAMLProtocol>
                    <PPSAMLResponseDestination/>
                    <PPResultWrapup>wssec-replace</PPResultWrapup>
                    <PPSAMLAssertionType>
                    <authentication>on</authentication>
                    <attribute>on</attribute>
                    <authorization>off</authorization></PPSAMLAssertionType>
                    <PPSAMLSubjectConfirm>bearer</PPSAMLSubjectConfirm>
                    <PPSAMLNameID>on</PPSAMLNameID>
                    <PPSAMLNameIDFormat/>
                    <PPSAMLRecipient/>
                    <PPSAMLAudience/>
                    <PPSAMLOmitNotBefore>off</PPSAMLOmitNotBefore>
```

```
            <PPOneTimeUse>off</PPOneTimeUse>
            <PPSAMLProxy>off</PPSAMLProxy>
            <PPSAMLProxyAudience/>
            <PPSAMLProxyCount>0</PPSAMLProxyCount>
            <PPSAMLAuthzAction>AllHTTP</PPSAMLAuthzAction>
            <PPSAMLAttributes/>
            <PPLTPAInsertCookie>on</PPLTPAInsertCookie>
            <PPTAMPACPropagate>off</PPTAMPACPropagate>
            <PPTAMHeader>iv-creds</PPTAMHeader>
            <PPTAMHeaderSize>0</PPTAMHeaderSize></PostProcess>
            <SAMLSigningHashAlg>sha1</SAMLSigningHashAlg>
            <SAMLSigningAlg>rsa</SAMLSigningAlg>
            <LDAPsuffix/>
            <LogAllowed>on</LogAllowed>
            <LogAllowedLevel>info</LogAllowedLevel>
            <LogRejected>on</LogRejected>
            <LogRejectedLevel>warn</LogRejectedLevel>
            <PingIdentityCompatibility>off</PingIdentityCompatibility>
            <DoSValve>3</DoSValve>
            <LDAPVersion>v2</LDAPVersion>
            <EnforceSOAPActor>on</EnforceSOAPActor>
            <WSSecActorRoleID/>
         </AAAPolicy>
    </xsl:variable>

        <func:result select="$result"/>
    </func:function>

</xsl:stylesheet>
```

Tip: When writing the `dpfunc:GetAAAPolicy()` function, an important step is to add any static integer value (`1234` in the example) to the following elements of the AAA policy configuration:

- `<AUPort>`
- `<AZPort>`

The DataPower policy framework requires a value for these two elements.

The directory that contains this custom mapping is the `store:///polices/custom` directory, as shown in Figure 6-18.

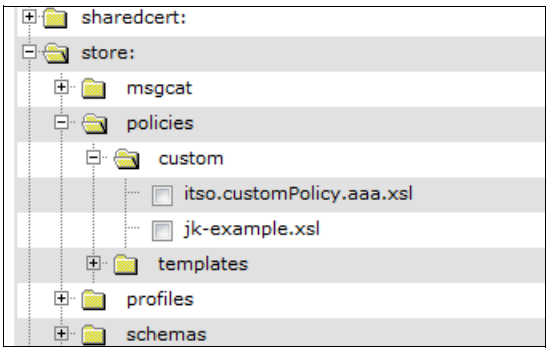


Figure 6-18 Directory for custom policy XSL style sheet

6. To verify whether the custom security policy style sheet is parsed properly, check the `Policy Domains Supported` interface. You can access this interface through the Status

menu: **Status** → **Web-Service** → **Policy Domains Supported**. This status interface provides the information shown in Figure 6-19.

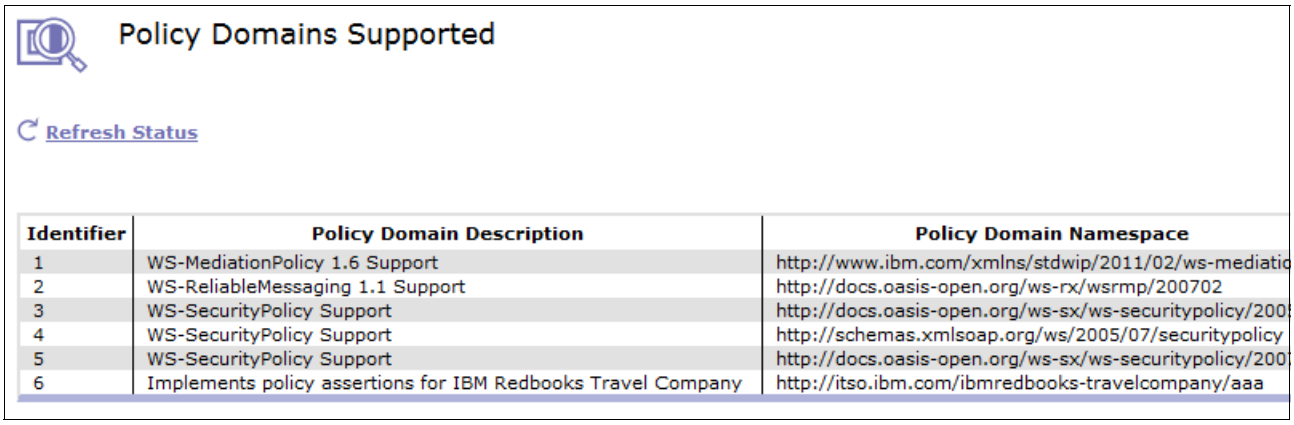


Figure 6-19 Policy Domains Supported interface

6.5 Attach custom policy to all services for an organization

The custom security policy can reside and be used on two separate components:

- WebSphere Service Registry and Repository (WSRR)

  This is the better option. It allows the user to use the WSRR policy attachment queries to manage the attachment of the custom policy to all of the services that need that policy. It also provided visibility in the policy administration point (which is the policy function of WSRR) as to which services are using the custom policy.

- The DataPower appliance itself

  If the custom security policy is loaded on the DataPower device, it must be placed in the `default` domain (`store:///` directory) or in the `local:///` directory where the Web Service Proxy that enforces the custom policy is configured.

After the location of the security policy is decided, attaching it to a service or a set of services for an organization is possible.

6.5.1 Attaching the custom security policy in WSRR at design-time

This section describes the capability of DataPower to export a policy, which is created on DataPower, to an external system, in this case WSRR. WSRR, as the policy administration point then provides capabilities to attach the custom security policy to a service or a set of services that match certain criteria. WSDL files and the custom security policy are then retrieved from WSRR to DataPower through the standard subscription process, as shown in Figure 6-20 on page 221.

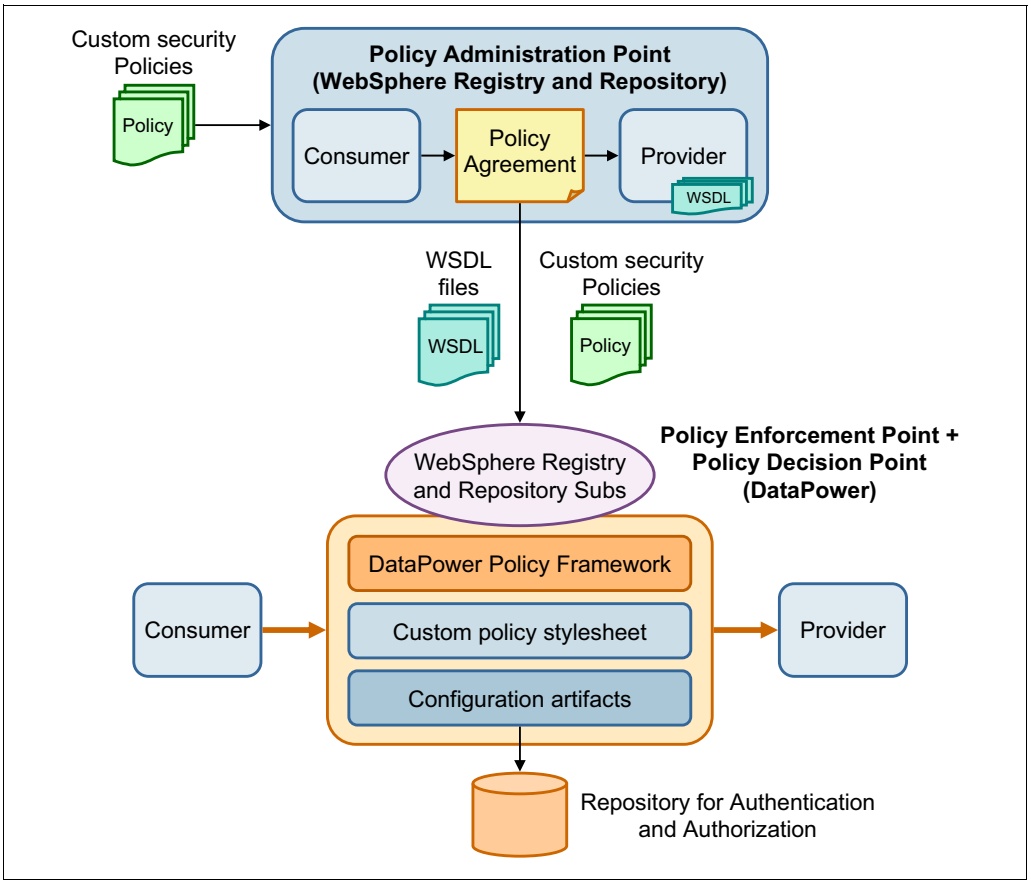


Figure 6-20 Custom security policy managed in WSRR

During policy enforcement, the DataPower policy framework accesses the appropriate custom policy style sheet and transforms the custom policy assertions into valid DataPower configuration artifacts. These artifacts consist of the following DataPower objects:

- An AAA policy, based on an `AAAInfo` file. Its name is `AAAPolicyExample`.
- A style policy action (or processing action), which references the `AAAPolicyExample` AAA policy.
- A request processing rule, which integrates the processing action. This request processing rule is bound to the service for which the custom security policy is attached.

In this example, the custom security policy is managed in WSRR. Assume that a Web Service Proxy was configured based on WSDL files that were retrieved from WSRR. This example describes the custom policy security attachment process.

You can also assume that the `AAAInfo.xml` file is already loaded on the correct DataPower domain. Again, this file is a referential of the authenticated consumers for these examples. In many companies, an LDAP server is usually the external source of the DataPower AAA framework and in this case, no artifacts must be loaded on the appliance.

The service to which the custom security policy is attached is as follows:

```
ItineraryReservationService
```

This chapter describes how to attach a custom security policy on several services, at the SLD level.

Use the following steps to attach a custom security policy to a service in WSRR:

1. Connect to the WSRR Governance Master business space by using your credentials.
2. Switch to the space named `IBM Redbooks Travel Service Registry for Operations`.
3. Click the **Load Documents** link in the Service Registry Actions widget, as shown in Figure 6-21.

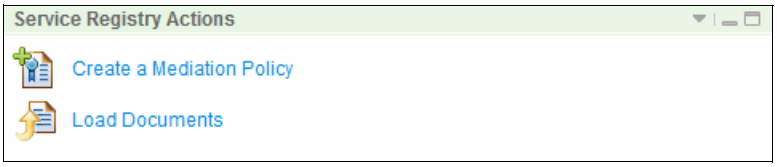


Figure 6-21 Load a custom policy document in WSRR

4. In the Load Documents window, click **Browse** to locate and select the custom policy that must be loaded, and then select the document type **Policy**, as shown in Figure 6-22.

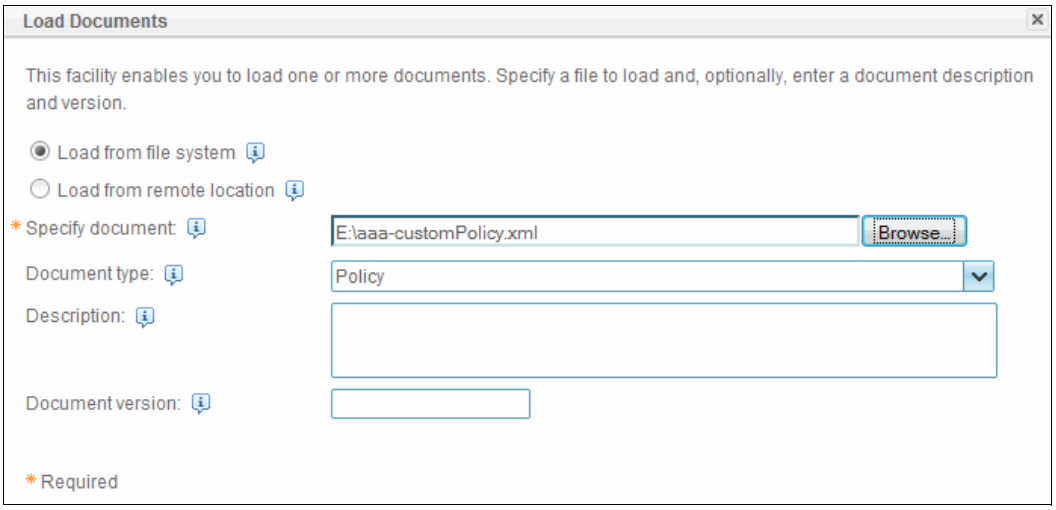


Figure 6-22 Document type specification when loading a custom policy document into WSRR

5. Enter a description and a version for the security policy that is loaded, as shown in Figure 6-23.

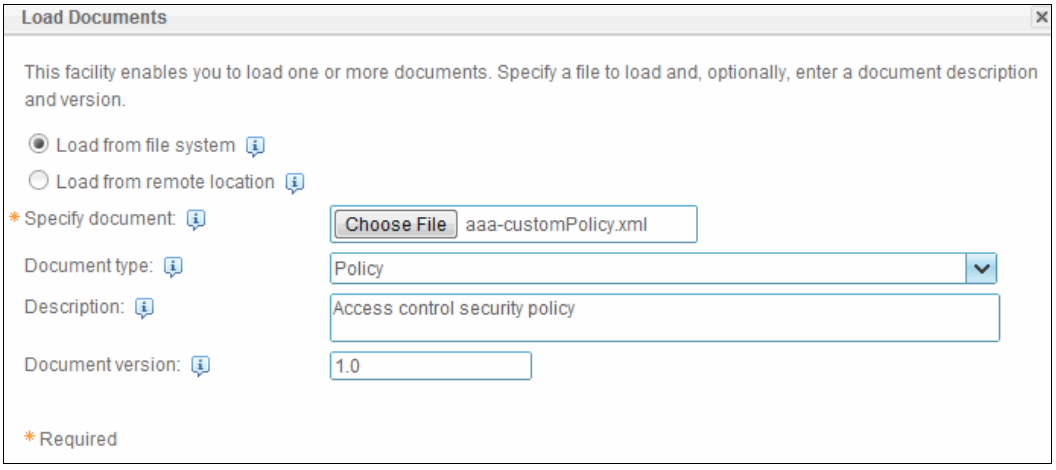


Figure 6-23 Description and version of a custom security policy in WSRR

6. To attach the custom security policy, follow the steps in 13.4, “Custom policy attachment in WSRR” on page 383. Attach the policy to a single service:
   - The name of the custom security policy is `aaa-CustomPolicy.xml`.
   - The SLD on which the custom security policy must be attached is the SLD of the itinerary reservation service.

   The properties of the `SLD - Itinerary Reservation Service` SLD are shown in Figure 6-24.

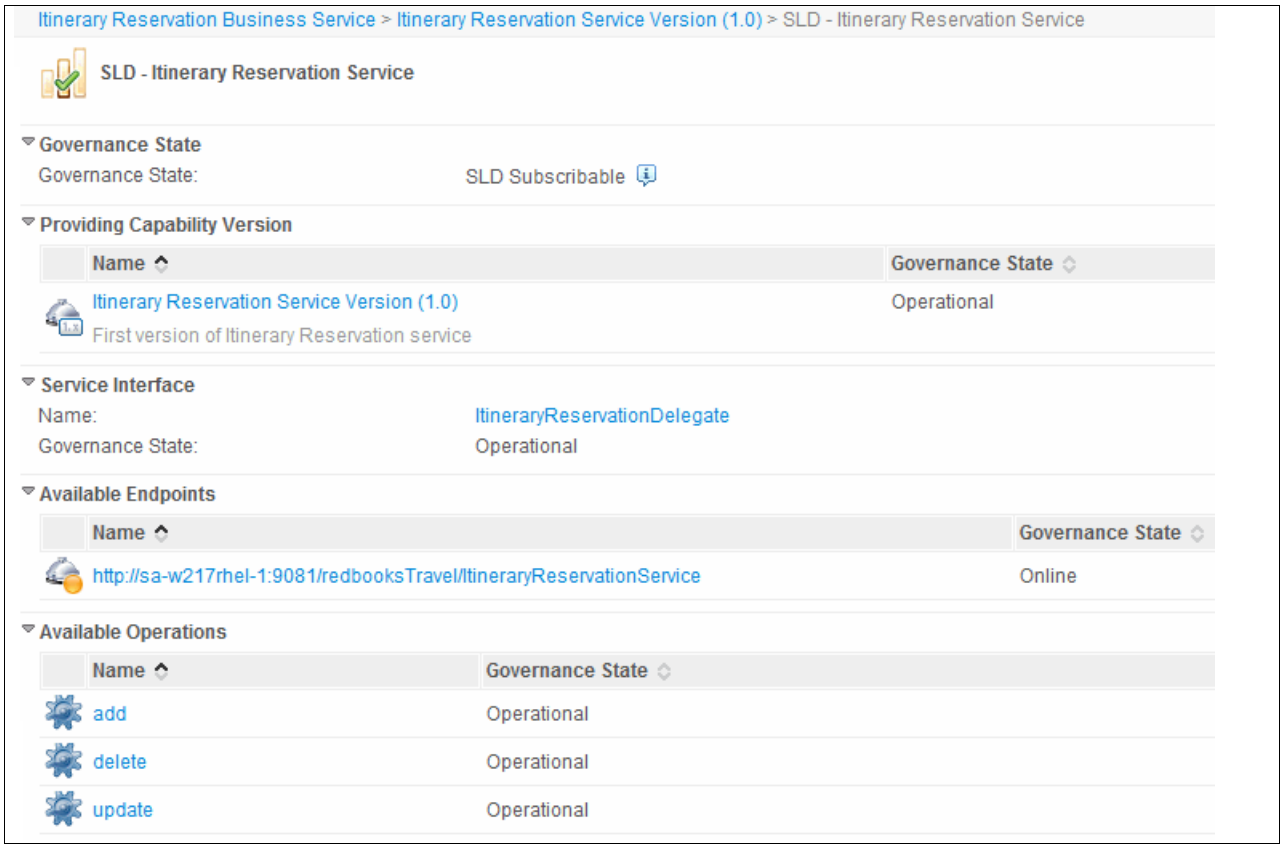


Figure 6-24 SLD property available for add, delete and update operations

As the figure shows, properties of the `SLD - Itinerary Reservation Service SLD` are bound to only three operations of the Itinerary service,:

- `add`
- `delete`
- `update`

The `read` operation is not associated to this SLD configuration. Therefore, the custom security policy that is attached to the SLD level of `SLD - Itinerary Reservation Service` is enforced only during the processing of the `add`, `delete`, and `update` operations of the Itinerary Reservation service, allowing anyone to perform a read function.

An overview of the custom security policy attachment is in Figure 6-25.

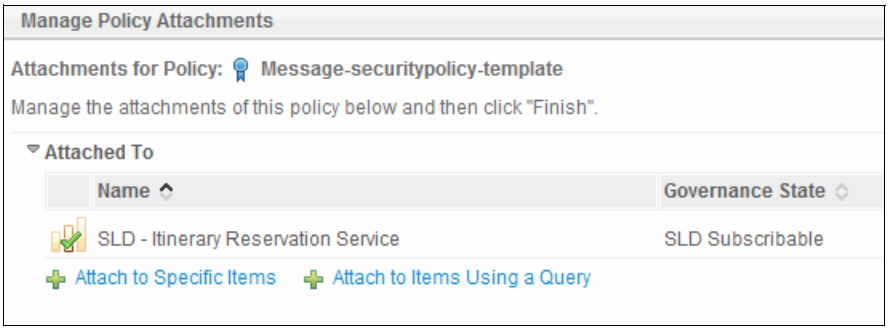


Figure 6-25 Versioning policy attachment at the SLD of the Itinerary Reservation Service

7. An important step is to deploy this policy in production, so the versioning policy attachment can affect the DataPower production appliance.
8. Select the **Itinerary Reservation Service Version 1.0**, accessible through the Itinerary Reservation Business Service, as shown in Figure 6-26.

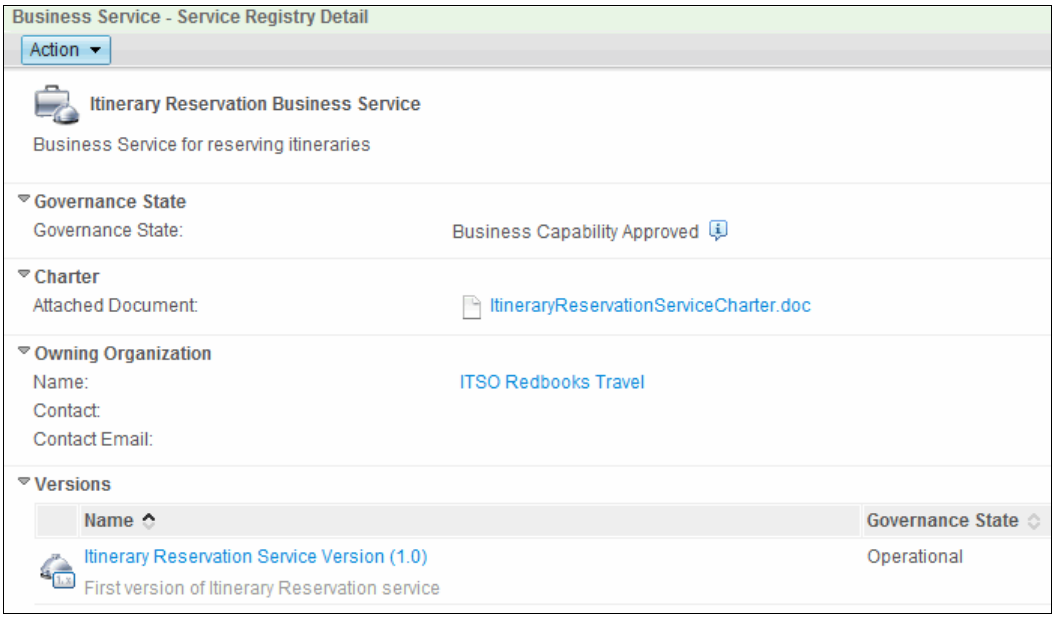


Figure 6-26 Itinerary Reservation Business Service

9. Click the **Itinerary Reservation Service Version (1.0)** link on the Service Registry Detail widget.
10. Select **Action** → **Redefine**
11. Click **OK** to go on the production deployment process.
12. Select **Action** → **Propose Production Deployment.**
13. Click **OK** to go on the production deployment process.
14. Select **Action** → **Approve Production Deployment.**
15. Click **OK** to validate the production deployment process.

At this stage, the custom versioning policy is deployed on the WSRR production instance.

You must create a Web Service Proxy on the DataPower side to retrieve the following elements:

- The WSDL file of the Itinerary Reservation service
- The custom security policy that is attached at the SLD level
- Information about the attachment of the custom security policy

Configuration

Use the following steps to enforce a custom security policy governed in WSRR:

1. Connect to a DataPower appliance by using your credentials (login and password). Select a production domain on which the `ITSORedbooksTravel_WSP` Web Service Proxy expose the `ItineraryReservationService` that is already configured.
2. Because the authentication is based on an XML file (`AAAInfo.xml`), this file must be loaded on the appliance (in the `local:///` directory of the current domain). An example of this file is in Example 6-4 on page 199.
3. From the Control Panel, click the **Web Service Proxy** icon, as shown in Figure 6-27.

Figure 6-27 Accessing the Web Service Proxy configuration panel

4. Select the Web Service Proxy service, which exposes the `ItineraryReservationService: ITSORedbooksTravel_WSP`.
5. Click the **WSDL files** tab of the Web Service Proxy.
6. Verify that the WSDL Status is set to `Okay` (Figure 6-28) at the `ITSORedbooks_WSDLs` subscription level, and click this **Okay** link.

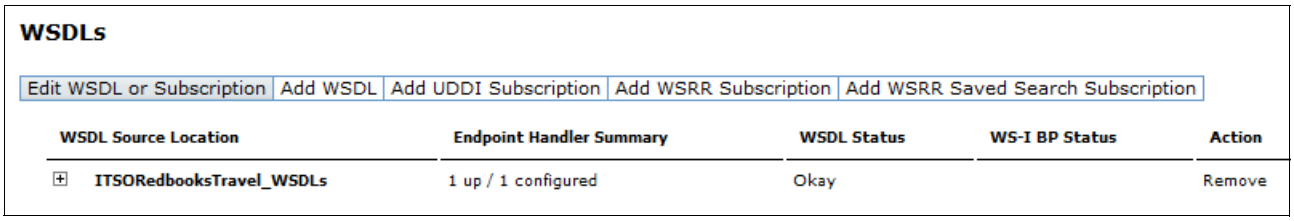


Figure 6-28 WSDL status of a Web Service Proxy

7. Click **Synchronize** in front of the `ITSORedbooksTravel_WSDLs` WSRR saved search subscription and confirm the synchronization.
8. The Web Service Proxy WSDL files panel indicates a synchronization state (`Synchronizing`) in the WSDL Status property, as shown in Figure 6-29 on page 226.

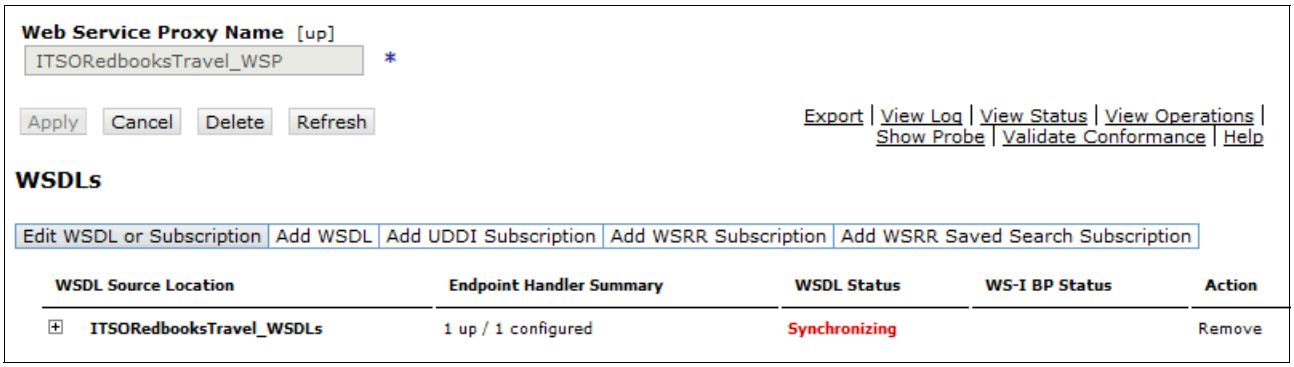


Figure 6-29 Synchronization of a subscription to WSRR from DataPower

9. After the synchronization process completes, click the **SLA Policy Details** tab of the Web Service Proxy.
10. From the SLA Table section of the panel, select the **port-operation** level of the `ItineraryReservationService` WSDL file, as shown in Figure 6-30.

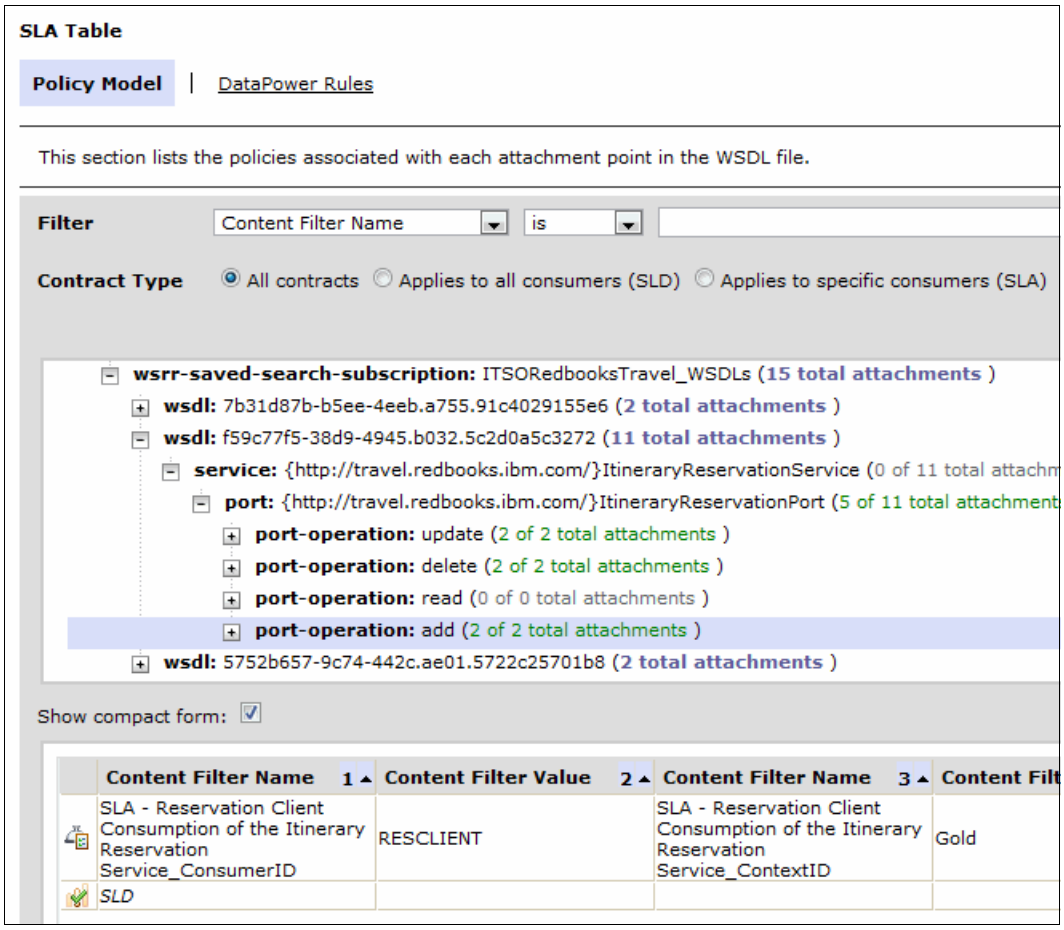


Figure 6-30 SLA Table, Policy Model

SLD properties are bound only to the update, delete, and add operations. Therefore you must select one of these three operations. Do not select the read operation.

11. Click the **SLD** link displayed at the bottom of the panel. The `ControlConsumerAccess` custom security policy must be displayed, as shown in Figure 6-31 on page 227.

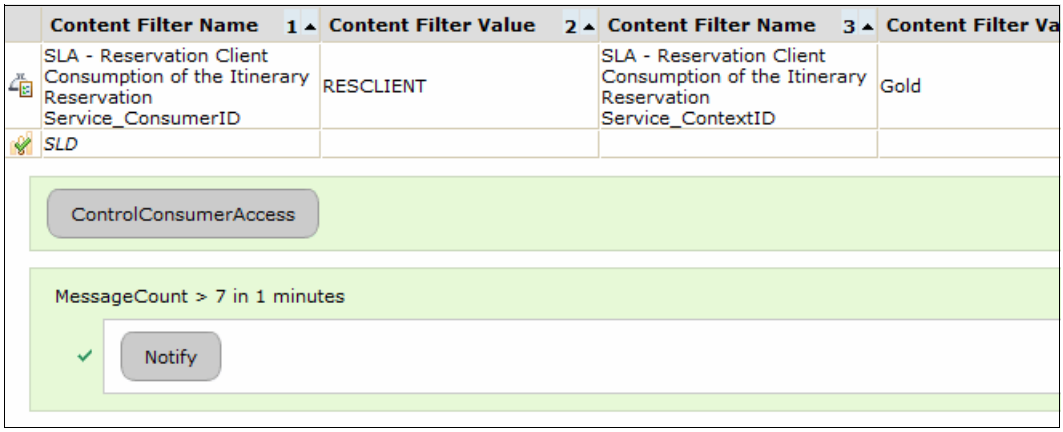


Figure 6-31 Custom security policy enforced at the SLD level

This display means that the ControlConsumerAccess policy (custom security policy) is enforced at the SLD level, that is, for all consumers that want to access the ItineraryReservationService service. This custom security policy is enforced only on the update, delete, and add operations.

The presence of a mediation policy is also attached at the SLD level.

12. From the SLA Table section of the panel, click the **DataPower Rules** link and select the **port-operation** level of the `ItineraryReservationService` WSDL file, as shown in Figure 6-32.

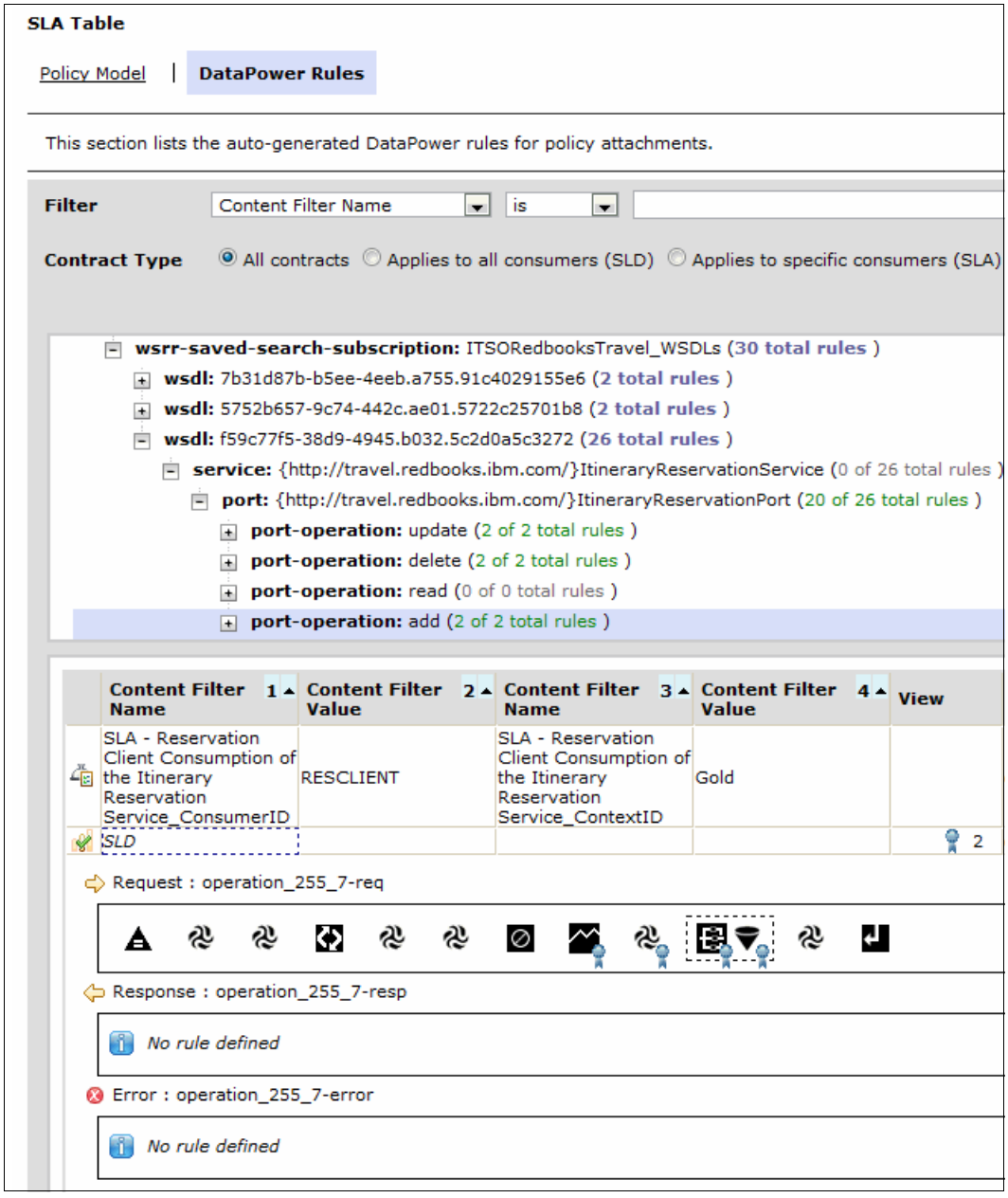


Figure 6-32 SLA tAble, DataPower Rules

13. Click one of the **SLD** links listed at the bottom of the panel. The configuration artifacts, which are created by the DataPower policy framework and the custom security policy style sheet, must display, as shown in Figure 6-33.

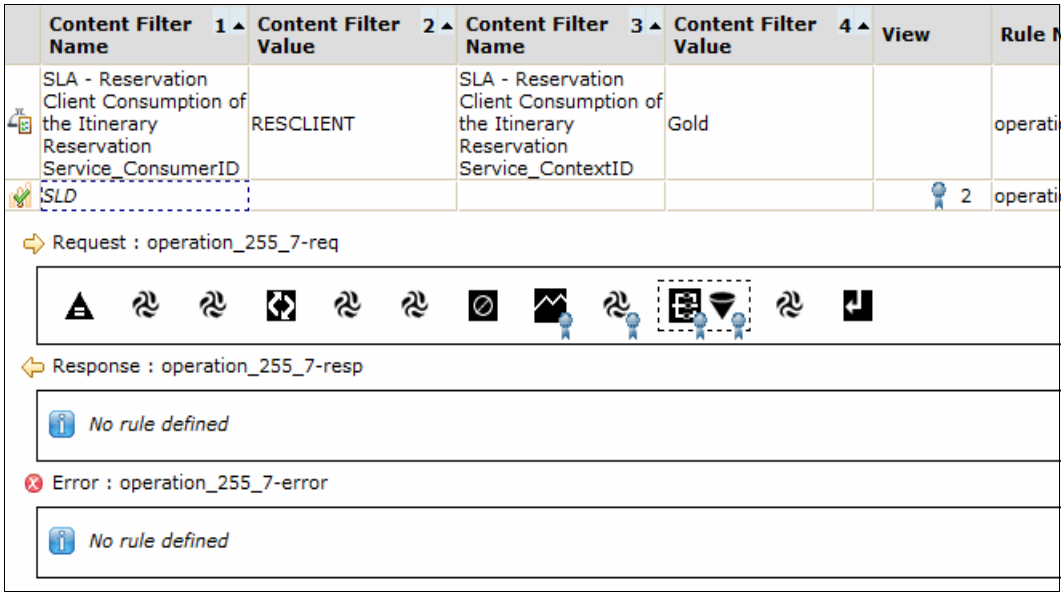


Figure 6-33 DataPower request rule with the required AAA policy

This request rule is the same for the three operations (update, add, delete) that are bound to the SLD properties.

A request processing rule is configured and contains an AAA policy action on it. This action is the `AAAPolicyExample` AAA policy, as shown in Figure 6-34.

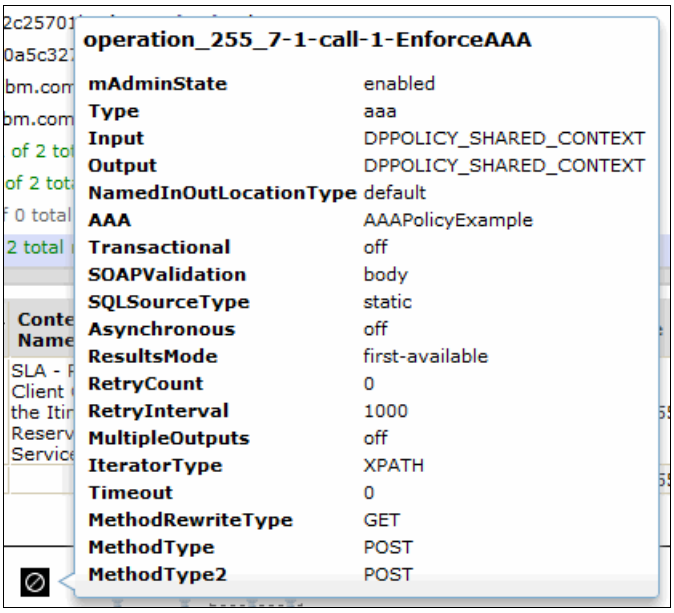


Figure 6-34 AAA action details

14. To verify that the `AAAPolicyExample` AAA policy was created, go to the Configure AAA Policy interface by selecting **Objects** → **XML Processing** → **AAA Policy.** The new `AAAPolicyExample` is created, as shown in Figure 6-35.

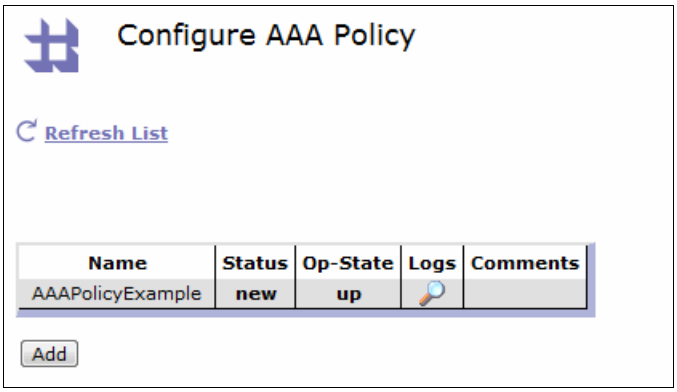


Figure 6-35 List of AAA policies in the current domain

Testing the access control security policy

This section presents a test used to prove that the access control security policy is enforced for consumers accessing the `ItineraryReservationService` service.

Use the following steps to perform the access control security policy enforcement test.

1. Connect to the ITSO Redbooks Travel Company test client interface, as shown in Figure 6-36.

Figure 6-36 ITSO Redbooks Travel Company test client home page

2. Click the **Itinerary Reservation Web Services Test Client** link to access the Reservation test client and enter required information to test the add operation of the ItineraryReservationService, as shown in Figure 6-37 on page 231.

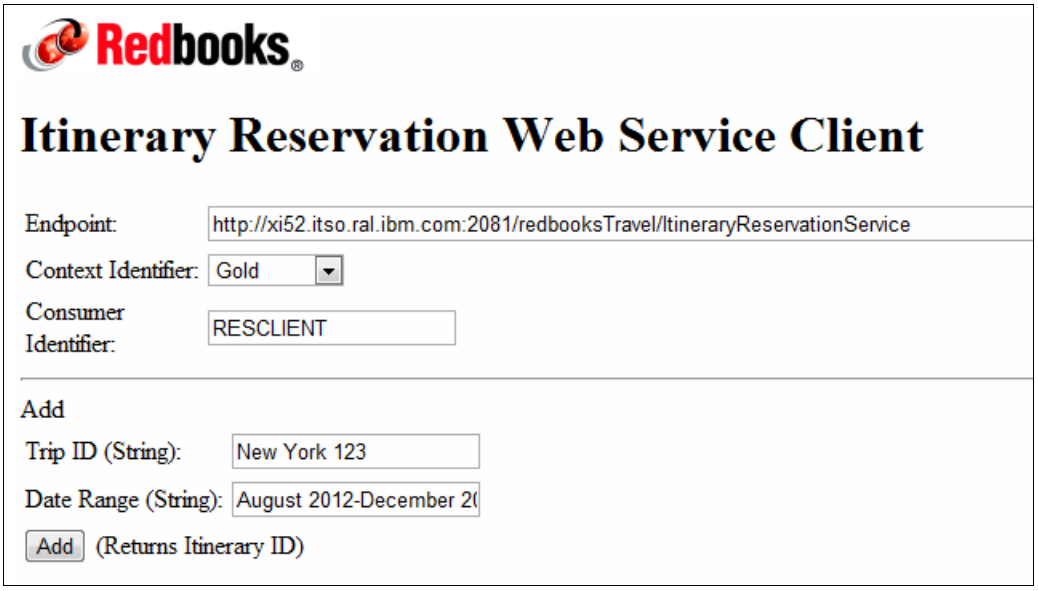


Figure 6-37 Testing the add operation with an authenticated consumer identifier

The value of the consumer identifier must be `RESCLIENT`, which is the value of the authenticated consumer identifier.

3. Click **Add** to invoke the service that is exposed on the DataPower appliance. The response can be captured in the DataPower probes, shown in Figure 6-38.

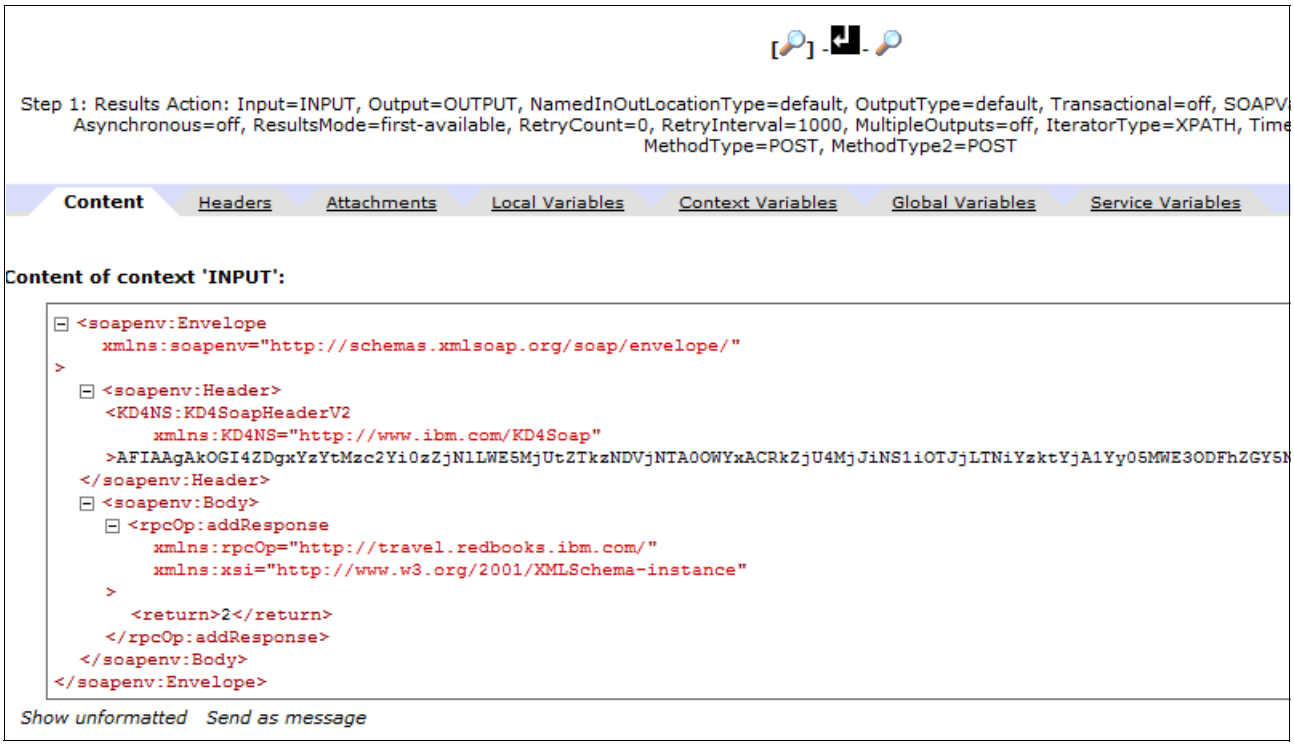


Figure 6-38 Itinerary reservation service valid response

On the probes, you can confirm that the AAA security policy enforcement allows the RESCLIENT consumer to access the service, as shown in Figure 6-39.

Figure 6-39 Authentication successful during the AAA security policy enforcement

4. Execute the same test with the same consumer identifier (RESCLIENT) but change the value of RESCLIENT in the AAAINfo.xml file, in order the RESCLIENT consumer not to be authenticated anymore, as shown in Figure 6-40.

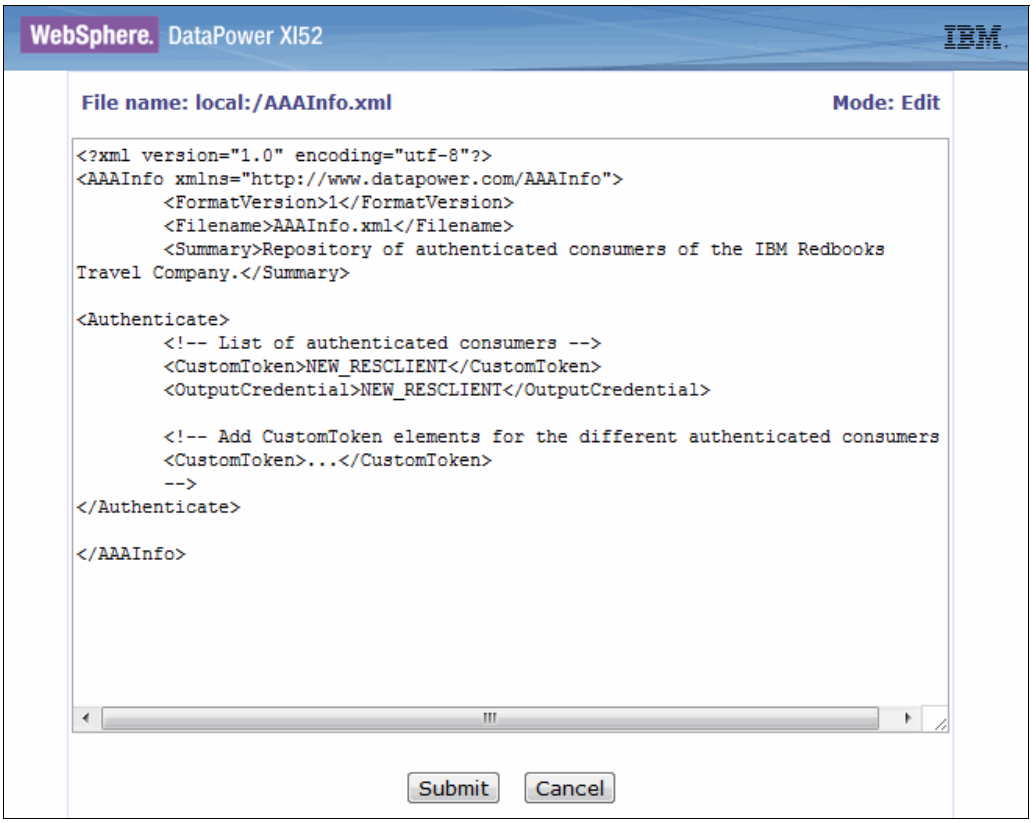


Figure 6-40 AAAInfo.xml file with a new consumer identifier value

The client request does not change, as shown in Figure 6-41.

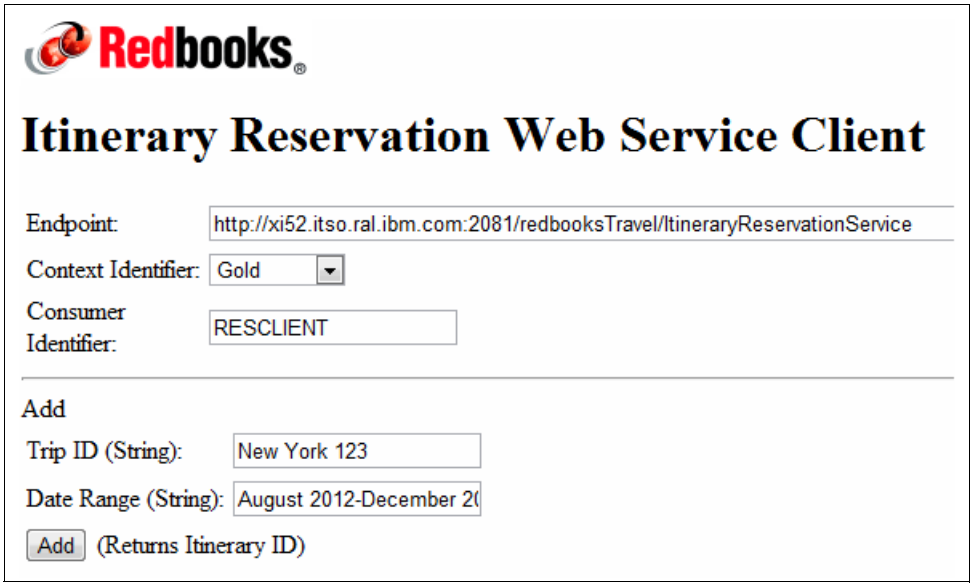


Figure 6-41 Testing the add operation with an unauthenticated consumer identifier

5. The message is rejected, as mentioned in the result at the bottom of the interface:

   `Rejected by policy. (from client).`

6. The DataPower probes show that the request is rejected during AAA security policy enforcement, as shown in Figure 6-42.

Figure 6-42 Testing the add operation with an unauthenticated consumer identifier

6.5.2 Attaching a custom security policy loaded on the DataPower device

DataPower can attach a custom security policy on a service. In this example, the custom security policy is located on the DataPower appliance itself. Only WSDL files are retrieved from WSRR through a subscription, as shown in Figure 6-43.

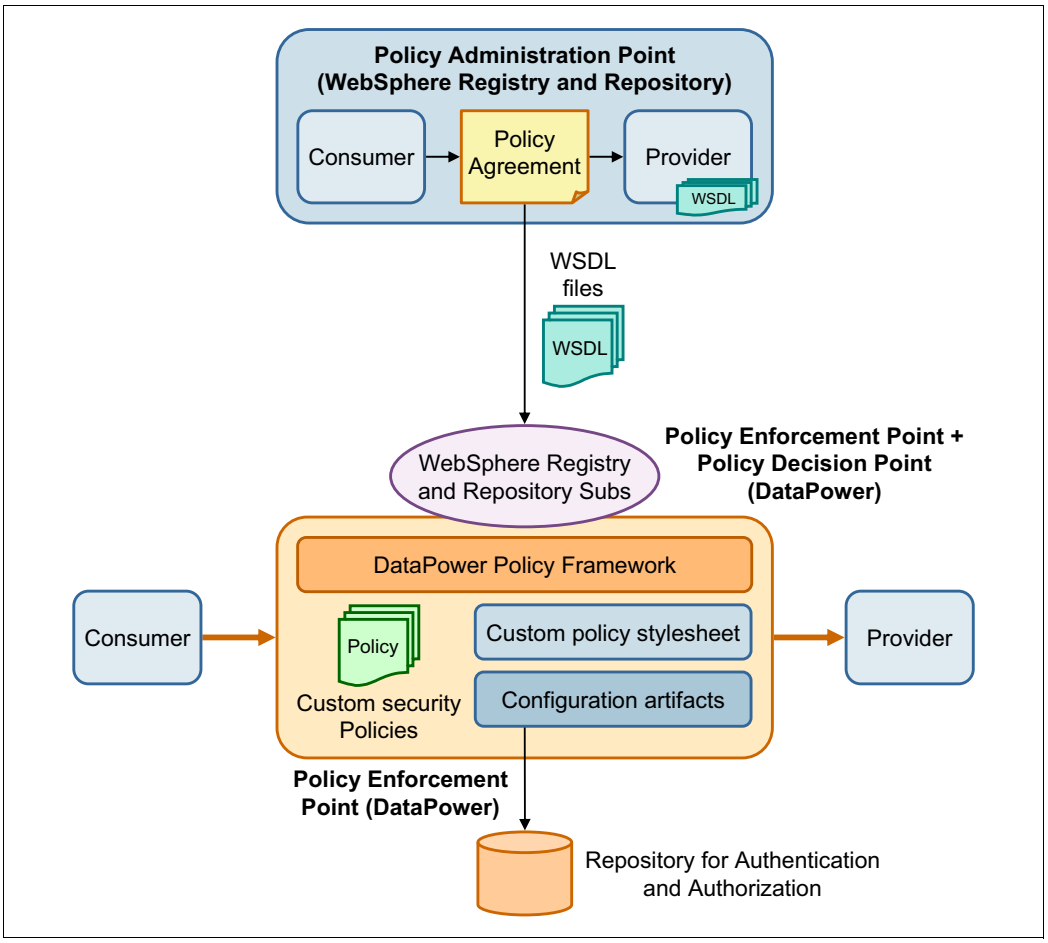


Figure 6-43 Custom security policy loaded on the DataPower device

When attaching the policy to a service, the DataPower policy framework, based on the correct custom policy style sheet, transforms the custom policy assertions into valid DataPower configuration artifacts.

Based on the custom security policy example, these artifacts consist of the following DataPower objects:

- An AAA policy (named `AAAPolicyExample`), based on an `AAAInfo` file.
- A style policy action (or processing action), which references the `AAAPolicyExample` AAA policy.
- A request processing rule, which integrates the processing action. This request processing rule is bound to the service for which the custom security policy is attached.

For more details about custom policy attachment in DataPower, see 13.6, “Custom policy attachment in DataPower” on page 386.

7

Policy monitoring

This chapter has information and examples for the creation of monitoring policies and their use in monitoring the services. The chapter describes the typical use of IBM Tivoli Composite Application Manager (ITCAM) as the PMP of service-oriented architecture (SOA) policies.

IBM Tivoli Composite Application Manager for SOA (ITCAM for SOA) provides monitoring and management of services and mediations in an SOA environment. ITCAM for SOA monitors a variety of metrics on many application server runtime environments and enterprise services buses. Paired with WSRR, ITCAM can use the governance policies that are defined in WSRR and implement them as monitoring situations, performing as the policy monitoring point (PMP) of the Policy governance pattern.

This chapter contains the following topics:

- 7.1, “Monitoring services” on page 236
- 7.2, “Displaying services in ITCAM” on page 237
- 7.3, “Monitoring examples” on page 240

© Copyright IBM Corp. 2013. All rights reserved.

7.1 Monitoring services

Monitoring a service consists of two distinct stages:

- Creating a policy, which from a monitoring point of view, is the declaration of a threshold which would cause an action to take place upon violation. The threshold can be numeric (message count may not be above a certain value) or textual (error message may not be ERR14); Boolean functions can be used to create complex thresholds (above two and under five). At this stage, the policy is simply a template that can be reused.
- After the policy is created, it must be attached to a specific SLD. Attaching the policy to a specific SLD takes the template and creates a concrete situation policy that is monitored in ITCAM. A single policy may be attached to multiple SLDs, so that each attachment will create a new ITCAM situation.

The creation of monitoring policies is more detailed in Chapter 14, "ITCAM as policy monitoring point" on page 389 and in the developerWorks article at the following website (you must have an IBM ID and password to access the article):

http://www.ibm.com/developerworks/wikis/display/tivolimediagallery/Enhancements+to+integration+with+WSRR+in+ITCAM+for+SOA+version+7.1.1+Fix+Pack+3

However, ITCAM situations are created from WSRR monitoring policy when the following prerequisites are followed:

- ITCAM and WSRR integration is configured correctly:
  - WSRR and ITCAM exchanged certificates
  - The ITCAM SDMS database was configured
  - A subscription for policies was created in WSRR
  - WSRR notifier and scheduler was configured.
- The SLD is in the governance *Subscribable* state and is attached to online endpoints.
- The monitoring policy is in the correct governance state:
  - For WSRR 7.5, policy is the *Monitored* state of the *Policy Lifecycle*
  - For WSRR 8.x, policy is the *Approved* state of the *SOA Policy Lifecycle*
- The classifications of the policy and SLD match those of the ITCAM and WSRR configuration.

Assuming the standard GEP policy was used to create the policies and the SDMS template file was used to configure the integration, then there should be no need to modify classifications.

In standard implementations of WSRR, all governance work (including creation of services, SLDs, and policies) are done on a Governance Master WSRR and ITCAM integration is done on a runtime WSRR (with multiple WSRRs for multiple environments, as needed). Therefore, another implicit step in the creation of the situation in ITCAM is the promotion of the service version to which the SLD is attached.

Figure 7-1 shows a standard implementation of WSRR-ITCAM integration.

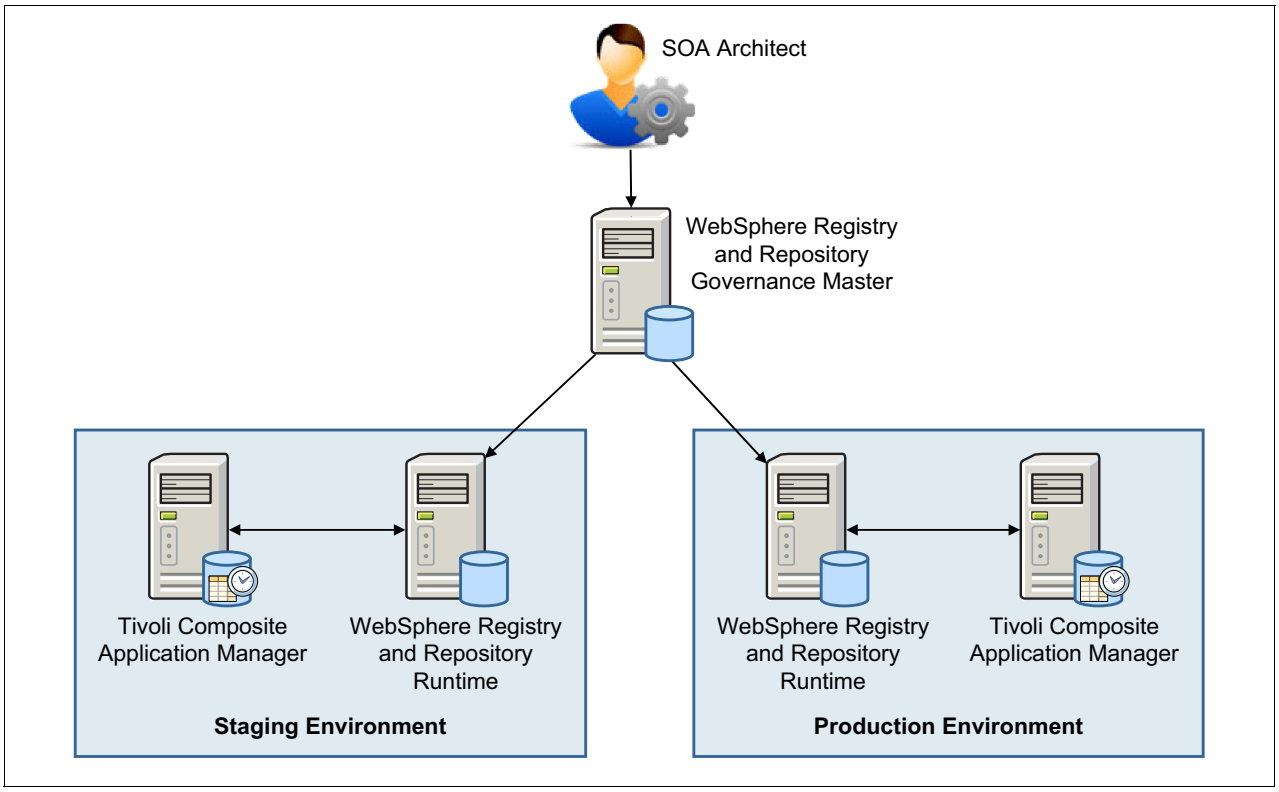


Figure 7-1 WSRR-ITCAM integration

After the SLD and policies are defined correctly, the WSRR-ITCAM integration is managed by the WSRR scheduler. It sends a notification to ITCAM for every relevant iteration.

7.2 Displaying services in ITCAM

ITCAM does more than monitor the policies that are defined in WSRR”

- It offers integrated management tools for web and enterprise infrastructures to aid SOA lifecycle availability and performance.
- It speeds and simplifies identification and resolution of SOA problems through a services topology view that provides actual views into service flows by displaying service-to-service relationships, including drill-down to service status and metrics/
- It uses smooth integration with other IBM Tivoli and WebSphere products to provide a comprehensive application management solution for complex environments.

To fully understand ITCAM for SOA, become familiar with IBM Tivoli Monitoring terminology. ITCAM for SOA is an advanced component of Tivoli Monitoring and uses the same terminology as all other monitoring agents.

At the least, you should know how to do these tasks:

- Log in to Tivoli Monitoring, through the basic client, the web client, or the webstart client.
- Find agents in the basic navigator views.
- Navigate through portal workspaces.
- View situations in the Active Event List.
- Create situations with the Situation Editor.

This level of knowledge of Tivoli Monitoring and ITCAM is necessary so that you can successfully work with WSRR and Tivoli Monitoring together. A deeper knowledge of Tivoli Monitoring will enable you to implement other advanced capabilities, which are described in the following chapters:

- Chapter 9, “WebSphere Service Registry and Repository for monitoring policy” on page 277
- Chapter 14, “ITCAM as policy monitoring point” on page 389.

Further details about these topics and basic training material are in the following location:

`http://pic.dhe.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=%2Fcom.ibm.itcamsoa.doc%2Fkd4inmst08.htm`

Figure 7-2 shows an ITCAMforSOA agent in the Tivoli Monitoring navigator tree.

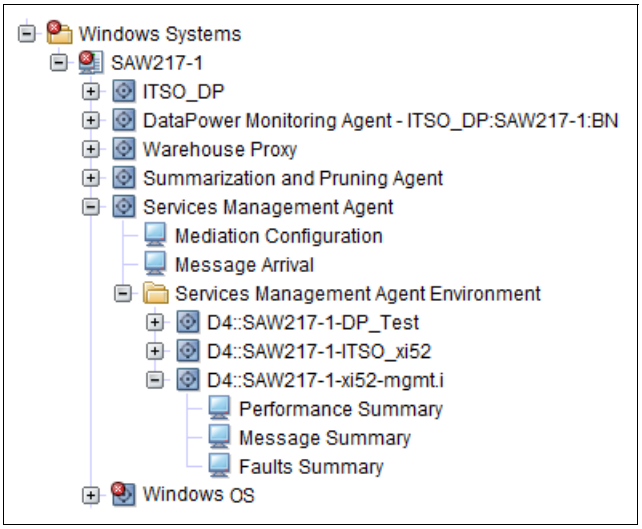
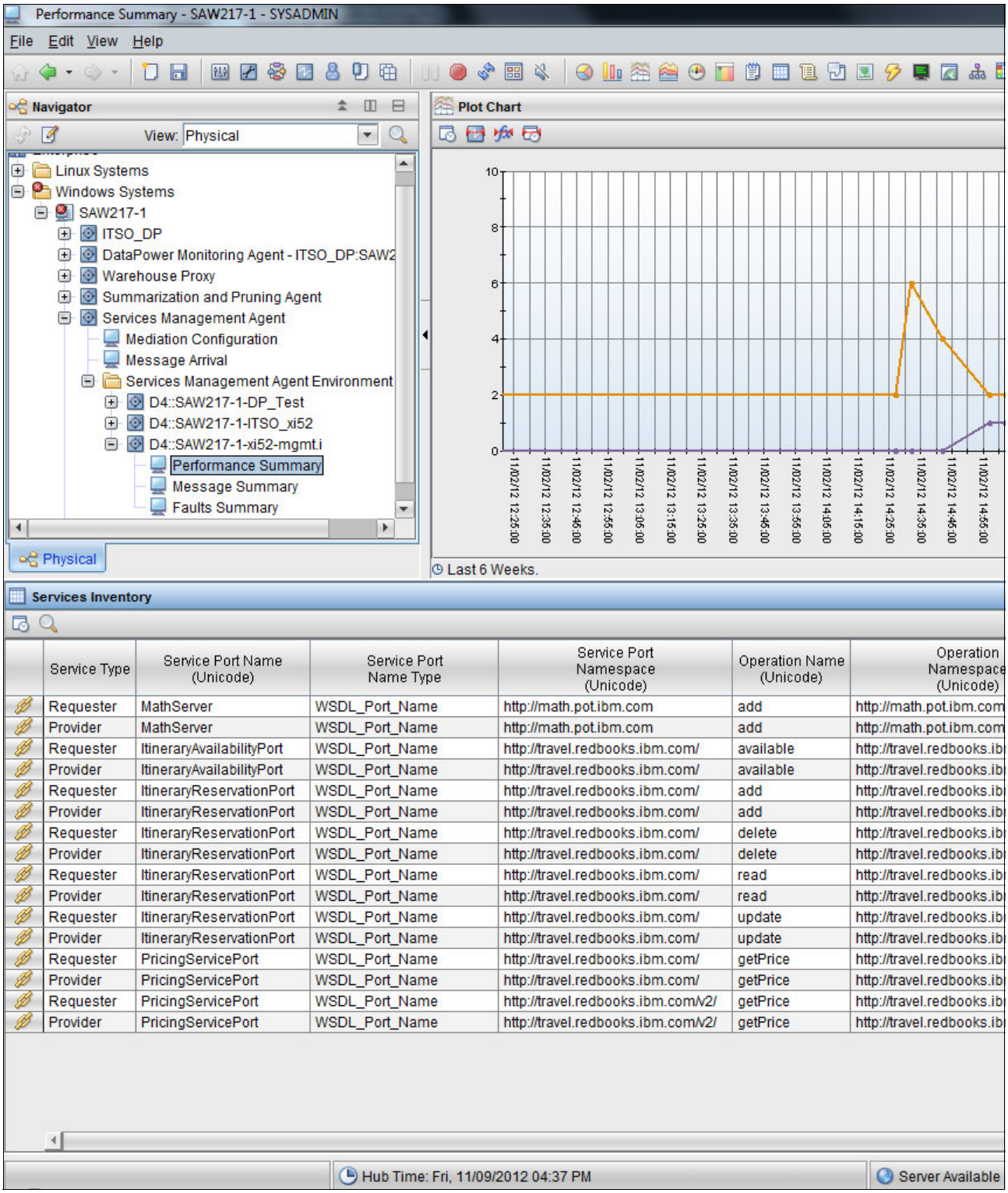


Figure 7-2 Tivoli Monitoring navigator tree with ITCAMforSOA agent

The SOA agent, named `Services Management Agent`, contains two global nodes, `Mediation Configuration` and `Message Arrival`. Each SOA runtime environment from which the agent collects information, in this case three domains of a DataPower, exists as a subnode of the `Services Management Agent Environment`.

For each SOA runtime environment, the agent collects performance metrics and displays them as tables and graphs, as shown in Figure 7-3.



	Service Type	Service Port Name (Unicode)	Service Port Name Type	Service Port Namespace (Unicode)	Operation Name (Unicode)	Operation Namespace (Unicode)
	Requester	MathServer	WSDL_Port_Name	http://math.pot.ibm.com	add	http://math.pot.ibm.com
	Provider	MathServer	WSDL_Port_Name	http://math.pot.ibm.com	add	http://math.pot.ibm.com
	Requester	ItineraryAvailabilityPort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	available	http://travel.redbooks.ib
	Provider	ItineraryAvailabilityPort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	available	http://travel.redbooks.ib
	Requester	ItineraryReservationPort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	add	http://travel.redbooks.ib
	Provider	ItineraryReservationPort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	add	http://travel.redbooks.ib
	Requester	ItineraryReservationPort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	delete	http://travel.redbooks.ib
	Provider	ItineraryReservationPort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	delete	http://travel.redbooks.ib
	Requester	ItineraryReservationPort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	read	http://travel.redbooks.ib
	Provider	ItineraryReservationPort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	read	http://travel.redbooks.ib
	Requester	ItineraryReservationPort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	update	http://travel.redbooks.ib
	Provider	ItineraryReservationPort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	update	http://travel.redbooks.ib
	Requester	PricingServicePort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	getPrice	http://travel.redbooks.ib
	Provider	PricingServicePort	WSDL_Port_Name	http://travel.redbooks.ibm.com/	getPrice	http://travel.redbooks.ib
	Requester	PricingServicePort	WSDL_Port_Name	http://travel.redbooks.ibm.com/v2/	getPrice	http://travel.redbooks.ib
	Provider	PricingServicePort	WSDL_Port_Name	http://travel.redbooks.ibm.com/v2/	getPrice	http://travel.redbooks.ib

Figure 7-3 SOA graphs and tabular metrics in ITCAM

Further details are in the *ITCAM for SOA User Guide* at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_user_guide_and_help/ws_about.html

7.3 Monitoring examples

This section describes example policies and the situations that are created from them.

These examples all assume that WSRR-ITCAM integration was configured. Details about how to configure the integration are in 14.4, “Configuring for integration of WSRR and ITCAM for Applications” on page 394.

See the information center for further information:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/WSRR_Integration_Guide_72/soawsrr.html

7.3.1 Steps to create a policy

This section describes how to create a simple policy. Further information is in 9.2.2, “Creating a monitoring policy” on page 282.

Assuming you already have an SLD (see Chapter 3.3.2, “Creating service level definitions” on page 60 to create a SLD) use the following steps to create a policy that monitors the response time of the web service:

1. Log in to WSRR Service Registry (not the Business Space).
2. Use either the Administrator or the SOA Governance perspective.
3. Under Service Documents, click the **Policy Documents** link. A list of existing policies opens, as Figure 7-4 on page 241 shows.

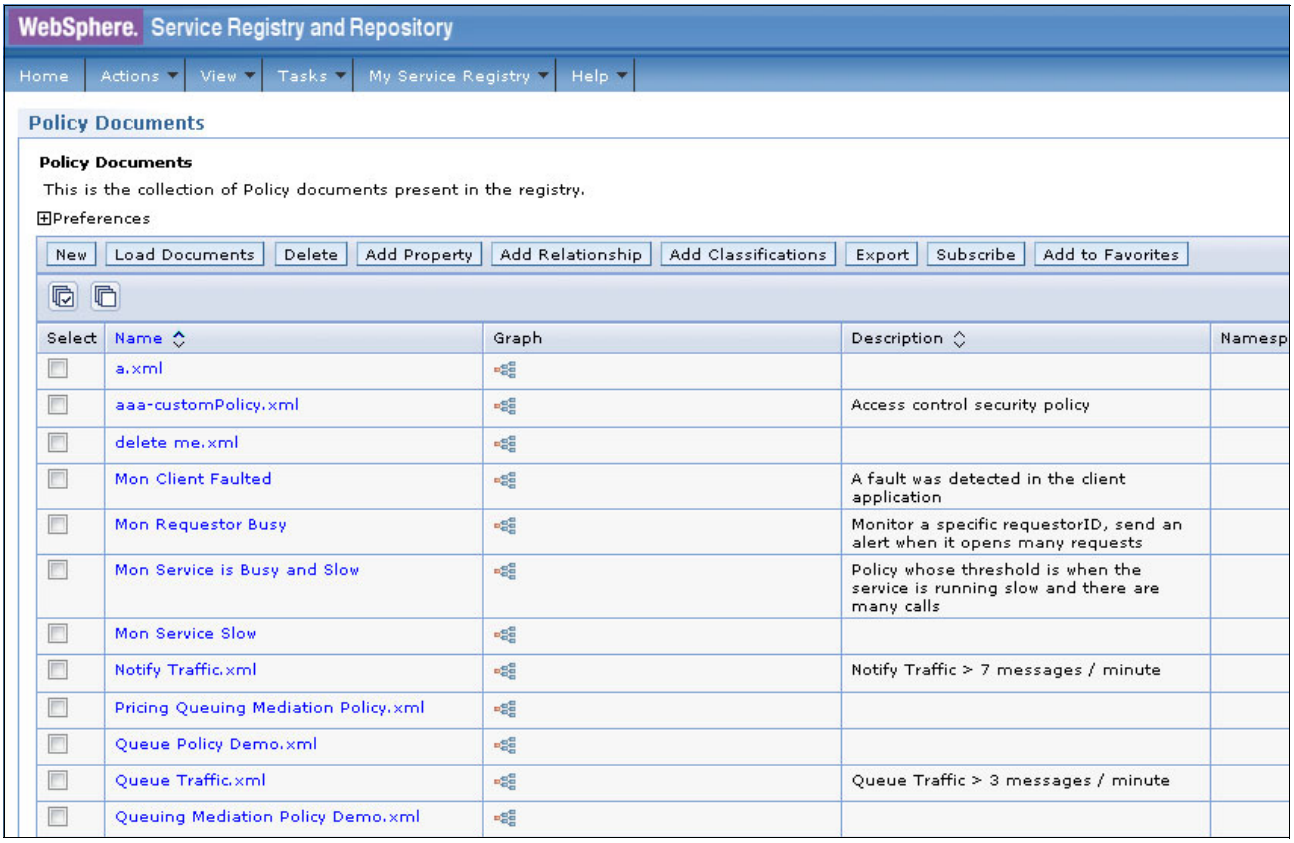


Figure 7-4 List of policies

4. Click **New**.
5. Select the **WS Policy Framework 1.5** option.
6. Click **Next**.
7. Enter the following values:
   - Policy name: `Mon service is slow`
   - Version: `1.0`
   - Description: `Monitors the response time of a web service`
8. Click **Select Policy Domain**.
9. Under Policy Domains, select **Service Level Monitoring Policy**.
10. Click **Apply**.
11. Click **Select Policy Type**.
12. Click **Select**. Web services monitoring policy is autoselected, as the only option.
13. Under Details, select **Add Property**.
14. Under Optional Properties, select **Policy Identifier**, and then click **Add**.
15. Under Policy Identifier, add the following name: `urn:mon_service_slow`
16. Under Policy Contents, click **Select Assertion**, beside the Situation table selector.
17. Select **Services inventory table**.
18. Click **Add**.

19. Click **Select Assertion**, beside Services inventory grouping selector.
20. Select **AllOf assertion** under Assertion Options.
21. Click **Add**.
22. Click **Add Assertion** beside AnyOf assertion.
23. Select **Average elapsed message round trip time** from the list.
24. Click **Add**.
25. Enter a value (for example, 10000) and select the **Greater than or equal** expression.
26. Click **Select Assertion**, beside **Notify selector**.
27. Under Assertion Options, select **IBM Tivoli Monitoring**.
28. Click **Add**.
29. Click **Add property**.
30. Select **State selector**, and click **Add**.
31. Select **Critical** state.
32. Click **Add Property**.
33. Select **Expert advice**, and click **Add**.
34. Enter the resolution advice for the policy, for example, enter the following information:

```
Service is slow. Follow standard protocol to resolve.
```

35. Click **Add Property**, and select **System Command**.
36. Click **Add**.
37. Enter a command that can run automatically when the policy threshold is passed. For example enter `echo TestCommand`.
38. Click **Publish**.

You are returned to the list of existing policies. Complete the following steps

1. Select the policy that was just created.
2. Select the **Governance** tab.
3. Depending on the version, do one of the following steps:
   - For WSRR 8.x:

     Click **Transition** until Governance State changes to **Approved Identified** → **Specification Review** → **Approved**.
   - For WSRR 7.5:
     i. Under Initial state transitions**,** select I**nitiate Policy Lifecycle**.
     ii. Click **Govern**.
     iii. Click **Transition** until Governance State changes to **Monitor Author** → **Transform** → **Enforce** → **Monitor**.

Now, the policy can be attached to an SLD in the same way that a Mediation Policy can be attached.

Complete the following steps in the Service Registry:

1. From the Perspective menu, click **SOA Governance**.
2. From the SOA Governance window, click **Service Level Definitions**.
3. Click the SLD to which you want to attach a Policy (SLD - Itinerary Availability).
4. Click the **Policy** tab.
5. Click **Edit Policy Attachments**.
6. Click **Attach Policy**.
7. In the Search window enter the following information: `urn:mon_service_slow`

   You may use a partial name and asterisk (*) to get a list of policies.
8. Select the policy to attach.
9. Click **Finish**.

7.3.2 Attaching to an SLD and creating a situation

Policies may also be attached to SLDs by using the business spaces as detailed in Chapter 3, “Policy traffic management, provider only, with operations” on page 35.

If the WSRR-ITCAM integration is configured on another WSRR run time, the Service Version of the SLD (in our example, Itinerary Availability) must be promoted to that run time so that the WSRR can notify the ITCAM.

WSRR view of policy

WSRR now has two policy documents:

- The policy itself: `Mon service is slow`
- The attachment between the policy and the SLD: `SLD - Itinerary Availability_GenericObject_SLD - Itinerary Availability_urn:mon_service_slow.xml`

From ITCAM’s point of view, the attachment is what becomes the situation.

Viewing the policy’s content by clicking the **Policy** tab shows details similar to Figure 7-5.

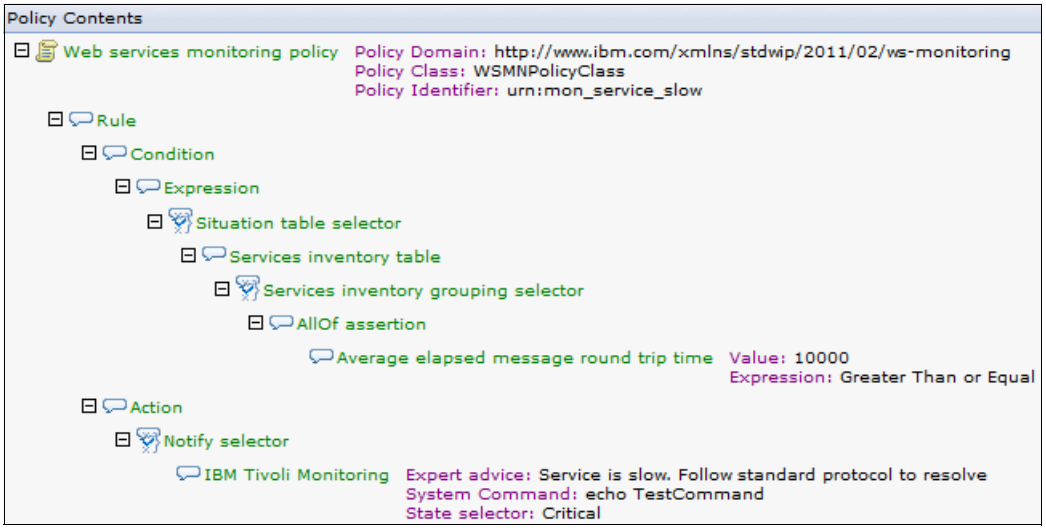


Figure 7-5 Policy contents

ITCAM view of policy

In ITCAM, view the situation by clicking **Situation Editor** and expanding the **Services Management Agent Environment** node. The situation has the following name, as the name of the policy attachment (see Figure 7-6):

```
SLD - Itinerary Availability - urn_mon_service_slow
```

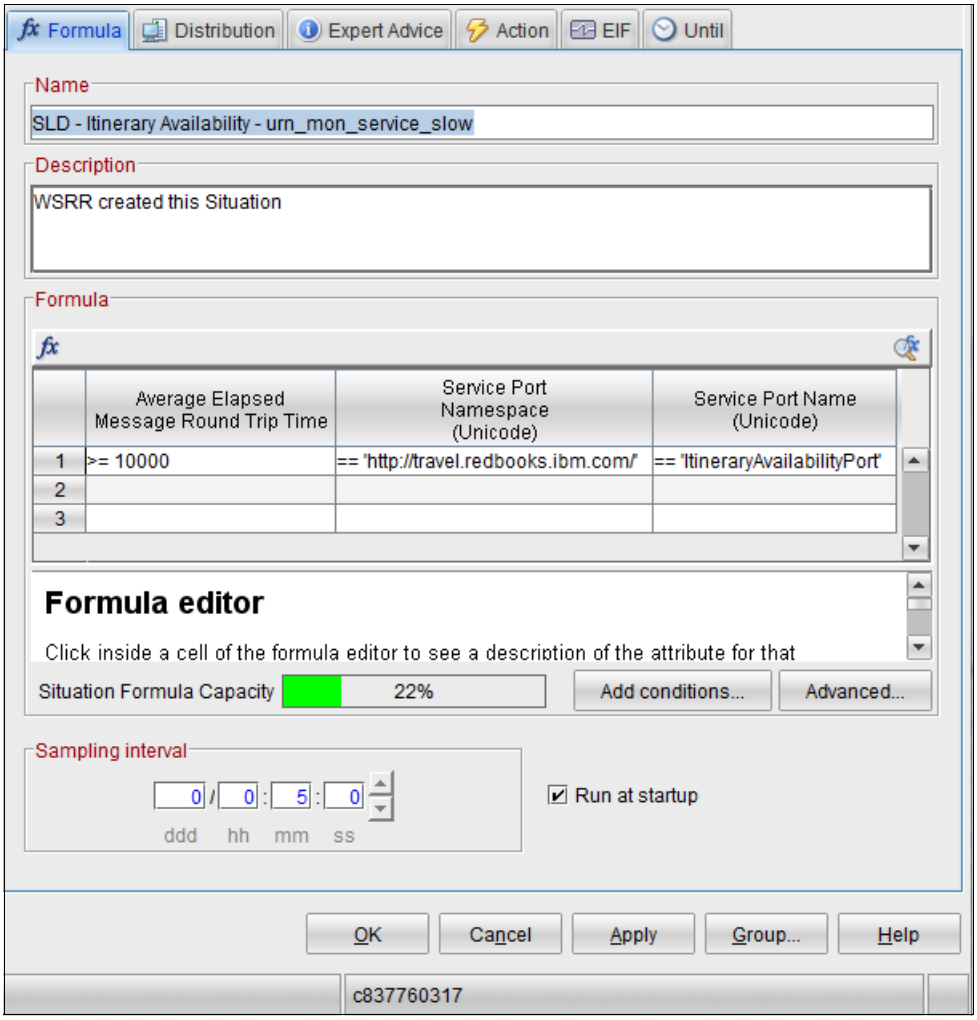


Figure 7-6 Situation formula

Consider the following information:

- The situation formula includes both the policy assertion and the service version endpoints.
- The sampling interval is the default of five minutes. See the following topics:
  - To change this value for a specific policy, see “Defining ITCAM specific properties” on page 291.
  - To change the default value, see 14.4, “Configuring for integration of WSRR and ITCAM for Applications” on page 394.
- The description is a boilerplate description for all situations that are created by WSRR. to Change the default value, see the information center:

  http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/WSRR_Integration_Guide_72/situationparameters.html
- The situation identifier is in the box in the lower right corner (c837760317).

By clicking the formula button (upper right corner of the table) the situation formula is displayed, as in Figure 7-7.

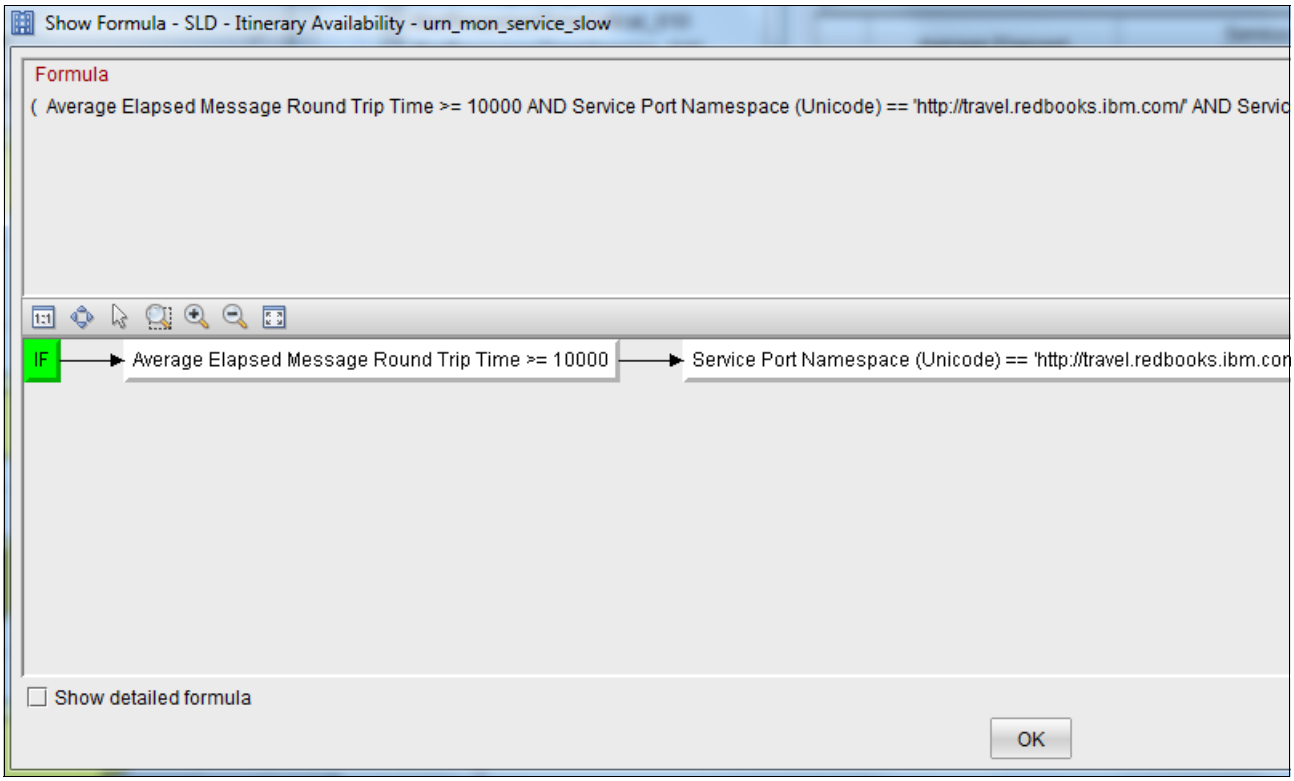


Figure 7-7 Situation formula: graphical view

The Expert Advice tab holds the information we entered in the Expert Advice field, as shown in Figure 7-8.

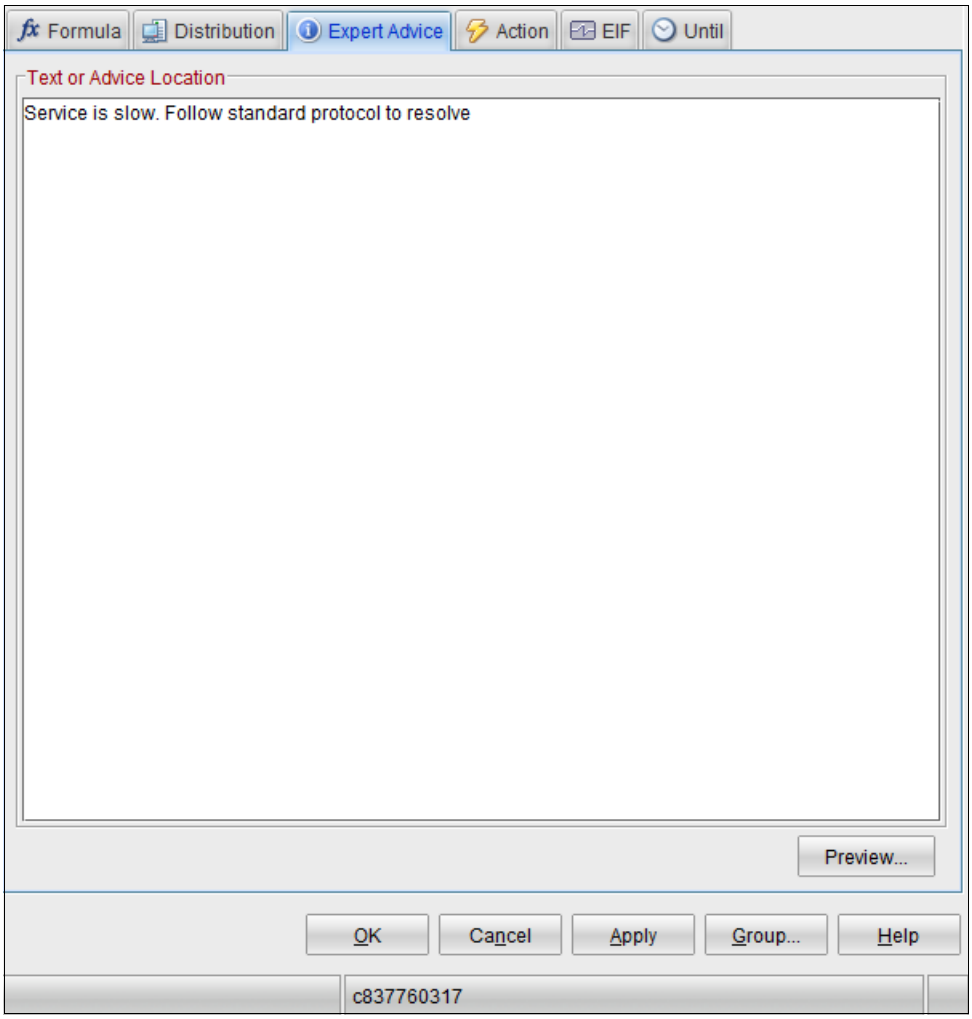


Figure 7-8 Situation expert advice

The Action tab shows the command that runs automatically when the situation fires because of the policy threshold being passed, as shown in Figure 7-9.

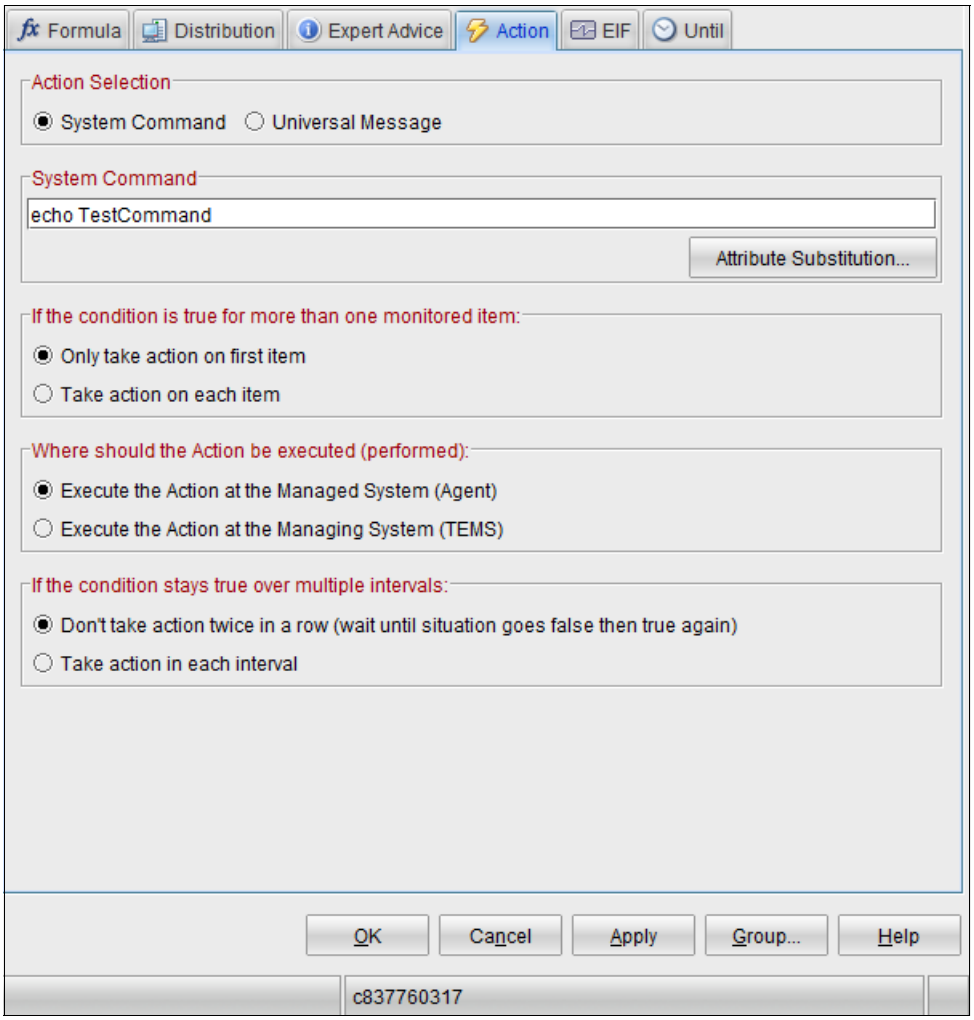


Figure 7-9 Situation action

There are a number of parameters in this tab beyond the system command itself. These parameters are default values, set during the WSRR-ITCAM integration configuration.

The EIF tab defines where ITCAM alerts will be sent. The values are defined during the WSRR-ITCAM integration configuration. See Figure 7-10.

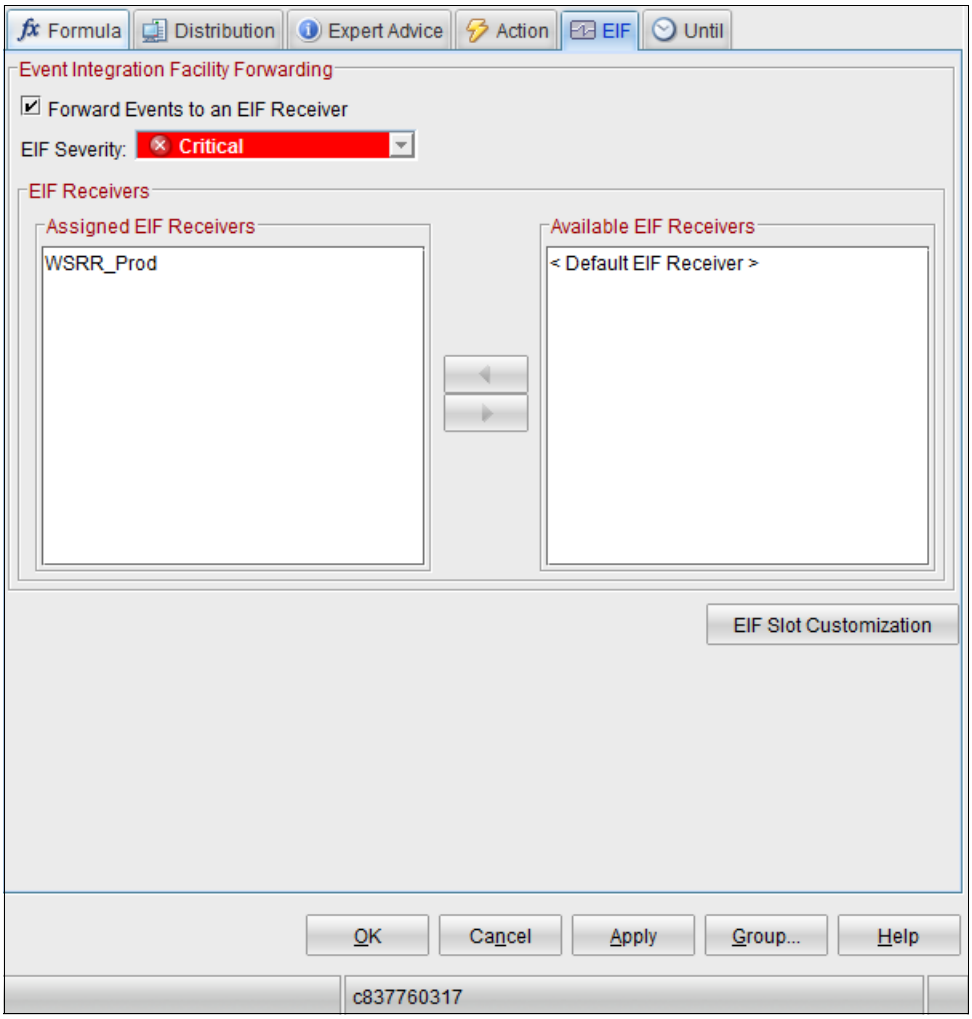


Figure 7-10 Situation: EIF

When the policy threshold is crossed, the ITCAM console displays the event, as shown in Figure 7-11.

Figure 7-11 ITCAM: single event in Situation Event Console

The Situation Event Console displays the situation information:

- Severity: From the policy's State selector
- Status: Open, Closed, Acknowledge by operator
- Owner: The operator who acknowledged the event
- Situation Name: Policy attachment name

- Display Item: An identifier to differentiate between events (For example, endpoints with the same namespace are otherwise identical to ITCAM.)
- Source: The ITCAM agent that performed the monitoring
- Situation ID: The unique identifier shared with WSRR
- Opened: When the event occurred
- Age: How long ago the event occurred

Right-clicking on the situation displays the menu in Figure 7-12.

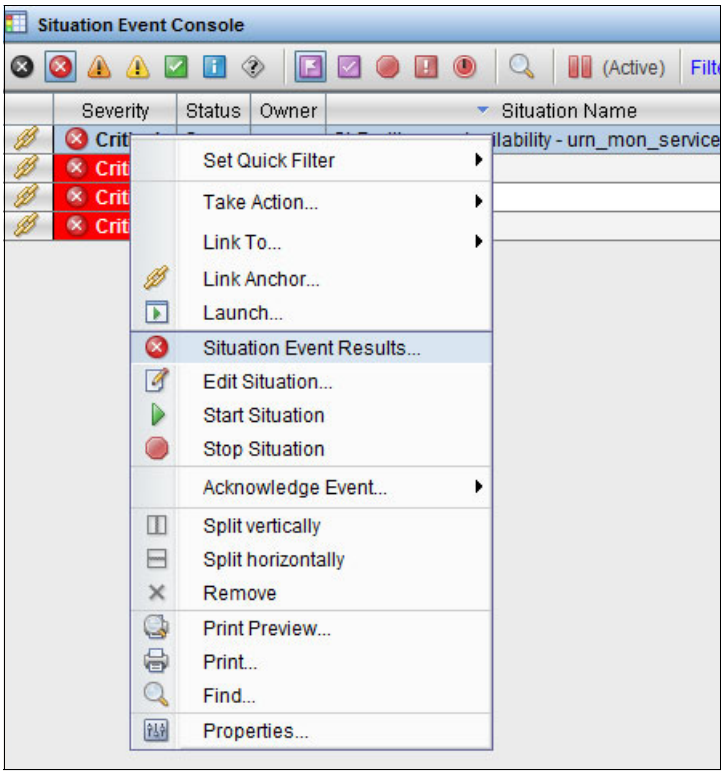


Figure 7-12 ITCAM: right-click Situation

Choosing **Situation Event Results** gathers further information about the event, as shown in Figure 7-13.

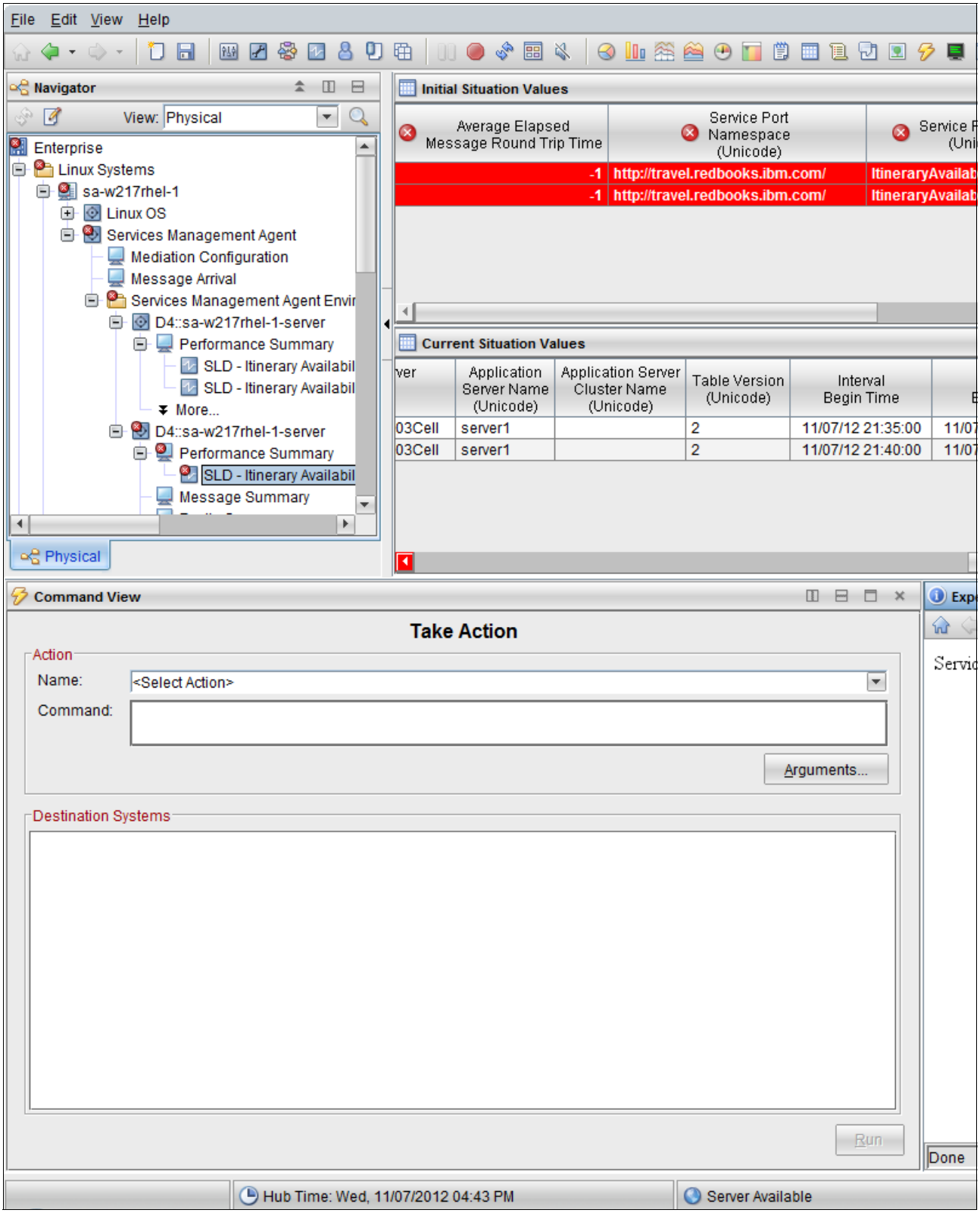


Figure 7-13 ITCAM: Situation event result

In this window, further information about the event is displayed, such as initial measurement, the current value, the expert advice as described in the WSRR policy, and more.

For further details, see 9.3.1, “Operator’s monitor” on page 296, and see the Tivoli Monitoring user’s guide:

http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3/nav_intro_tep.htm

7.3.3 Various situation tables

ITCAM monitors numerous parameters, which are represented by various situation tables in WSRR. Within a single situation table, the AllOf and AnyOf assertions can group other assertions together, but such joins cannot be performed between situation tables. For example, you can create a policy that measures both message count and average elapsed message round-trip time, but *not* also Fault code. The only difference in use between the situation tables is that during assertion selection, when after step 19 on page 242, different options are available in the drop-down list.

To create these examples, modify steps 7 on page 241 and 15 on page 241 to suit the name you will give to the policy and replace the steps after 16 on page 241.

Services Inventory Example table

This table contains data about current service inventory, including the following information:

- The minimum, maximum, average, and standard deviation of the response time of each SOA service
- Number of messages
- Number of faults
- Additional information too

A full list of parameters is in Appendix B, “ITCAM monitoring attribute tables” on page 421.

A busy service that is running slow, but not dangerously slow: When the average response time of the service is larger than or equal to two seconds, and under five seconds, a Warning alert to the operators is displayed, unless there are fewer than 10 messages.

The following policy creation demonstrates the use of Boolean conditions:

1. Select **Services inventory table**.
2. Click **Add**.
3. Click **Select Assertion** beside Services inventory grouping selector.
4. Select **AnyOf assertion** under Assertion Options.
5. Click **Add**.
6. Click **Add Assertion** beside AnyOf assertion.
7. Select **AllOf assertion** under Assertion Options.
8. Click **Add**.
9. Select **Average elapsed message round trip time** from the list.

10. Click **Add**.
11. Enter 2000 and select the **Greater than or equal** expression.
12. Click **Add Assertion** next to AllOf Assertion above the current line. See Figure 7-14.

Figure 7-14 Policy creation: adding an AND clause

13. Select **Average elapsed message round trip time** from the list.
14. Enter 5000 and select the **Less than** expression.

15. Click **Add Assertion** next to AnyOf on the line under Services inventory grouping selector. See Figure 7-15.

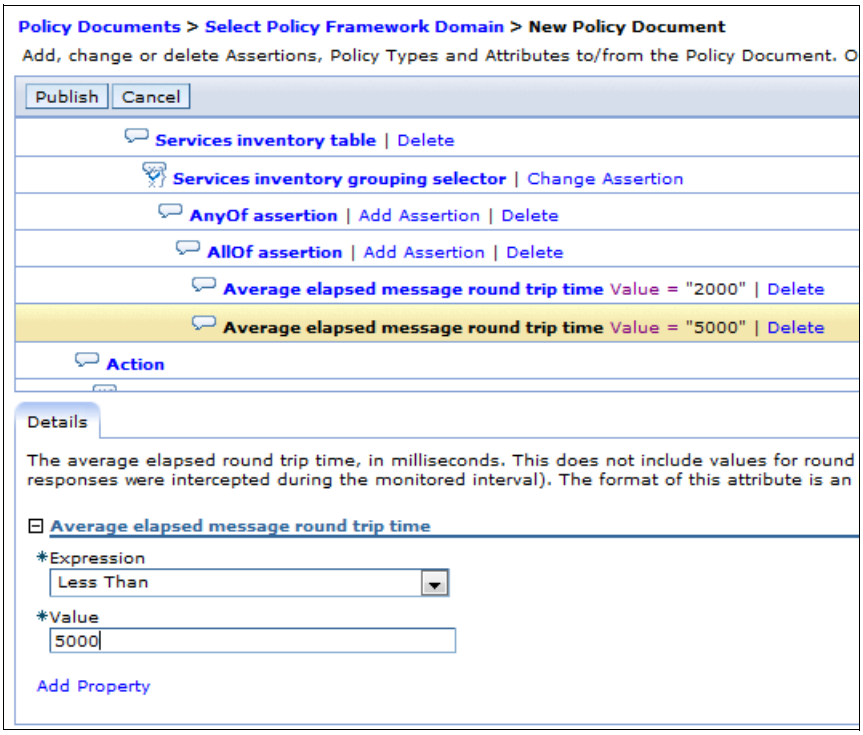


Figure 7-15 Policy creation adding an OR clause

16. Select **Message Count**.
17. Enter 10 and select the **Greater than or equal** expression.
18. Click **Add**.
19. Click **Select Assertion** beside Notify selector.
20. Under Assertion Options, select **IBM Tivoli Monitoring**.
21. Click **Add**.
22. Click **Add property**.
23. Select **State selector** and click **Add**.
24. Select state **Warning**.
25. Click **Publish**.

26. Select the **Policy** tab of the policy. The window in Figure 7-16 opens.

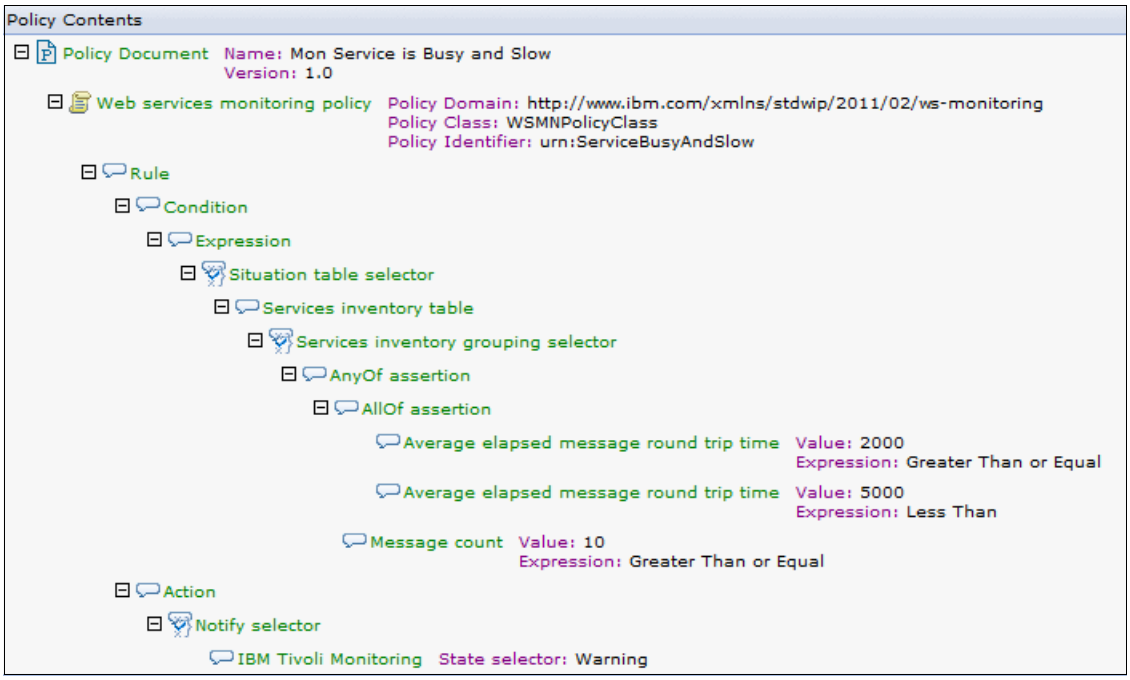


Figure 7-16 Policy details showing complex formula

After attaching the policy to `SLD - Itinerary Availability Service`, the situation named `SLD - Itinerary Availability - urn Busy service is slow` is created in ITCAM.

Figure 7-17 shows the ITCAM formula.

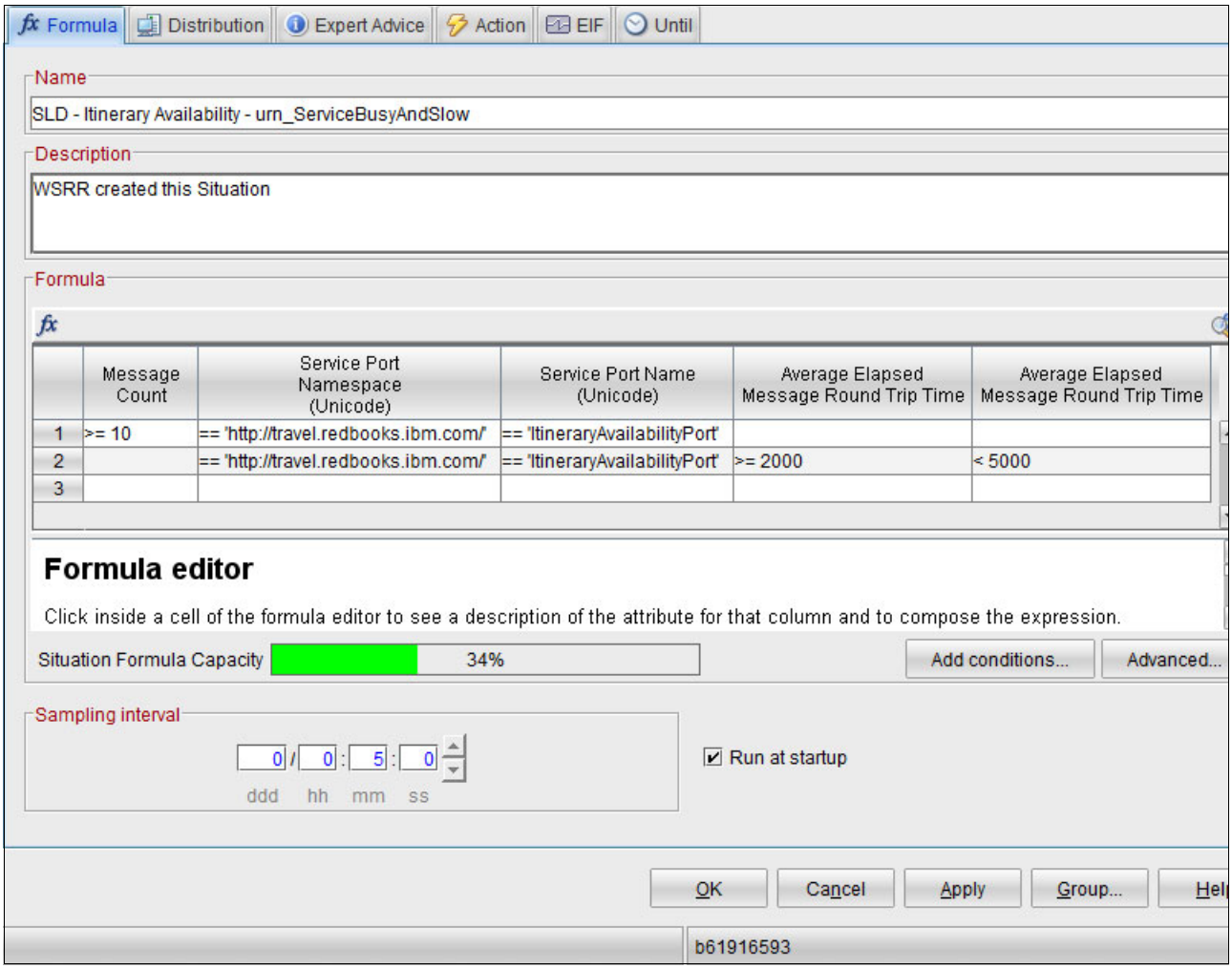


Figure 7-17 Situation formula containing multiple lines for OR condition and multiple columns for AND condition

Displaying the formula as a flow can make it simpler, as shown in Figure 7-18.

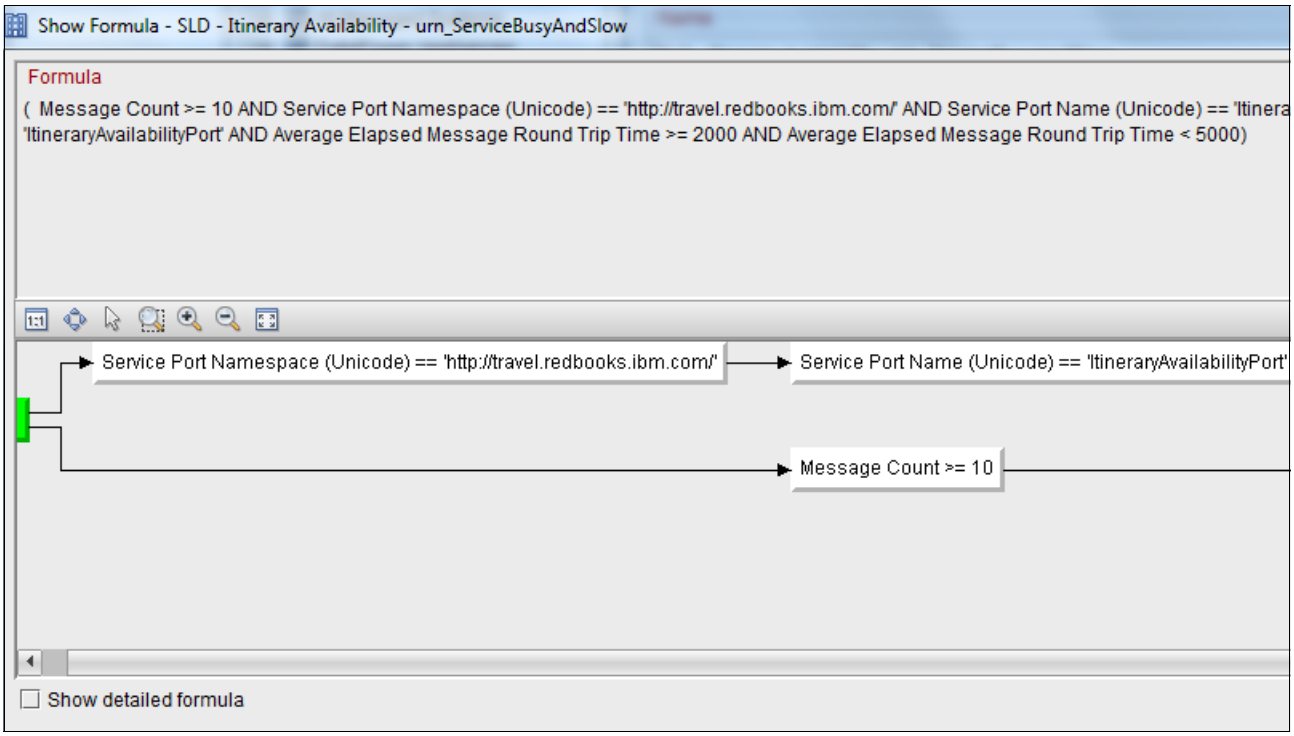


Figure 7-18 Complex situation formula as a flowchart

A service that is running slow, but not dangerously slow: When the average response time of the service is larger than or equal to two seconds, and under five seconds, a Warning alert to the operators is displayed, unless there are fewer than 10 messages.

Fault Log Table Example table

This table contains the SOAP errors that ITCAM collected. A full list of parameters is in Appendix B. When a fault with the `env:Client` ID occurs, a Critical alert to the operators is displayed.

Create a policy to demonstrate the use of textual conditions (which are similar to numeric conditions):

1. Select **Fault table**.
2. Click **Add**.
3. Click **Select Assertion**, beside the Services inventory grouping selector.
4. Select **AllOf assertion**, under Assertion Options.
5. Click **Add**.
6. Select **Fault ID** from the list.
7. Click **Add**.
8. Enter `env:Client` and select the **Equal to** expression.
9. Click **Select Assertion**, beside Notify selector.

10. Under Assertion Options, select **IBM Tivoli Monitoring**.
11. Click **Add**.
12. Click **Add property**.
13. Select **State selector**, and click **Add**.
14. Select state **Critical**.
15. Click **Publish**.
16. Click the **Policy** tab of the policy. The window in Figure 7-19 opens.

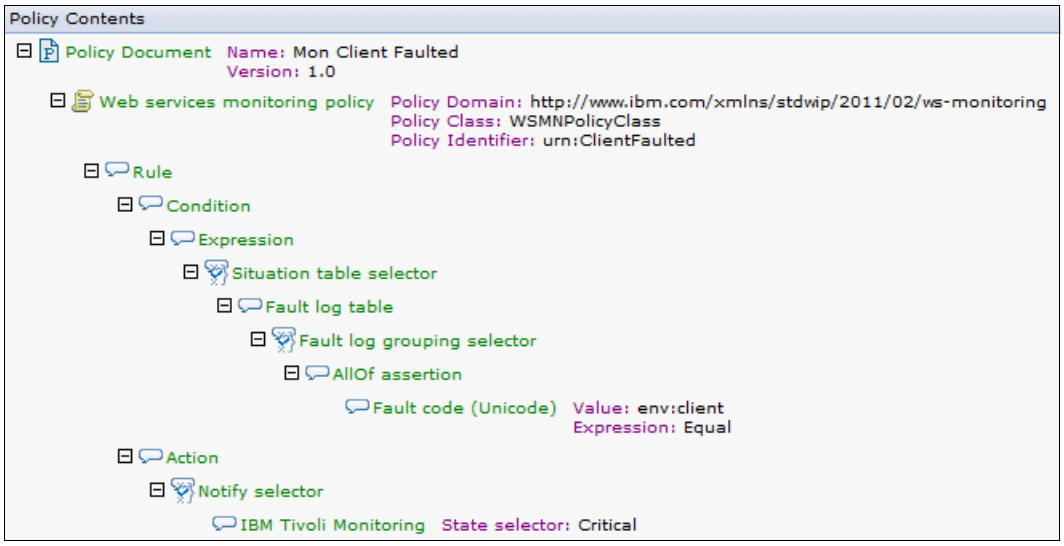


Figure 7-19 Policy contents

17. After attaching the policy to `SLD - Itinerary Availability Service`, the situation named `SLD - Itinerary Availability - urn client faulted` is created in ITCAM.

Figure 7-20 shows the ITCAM formula.

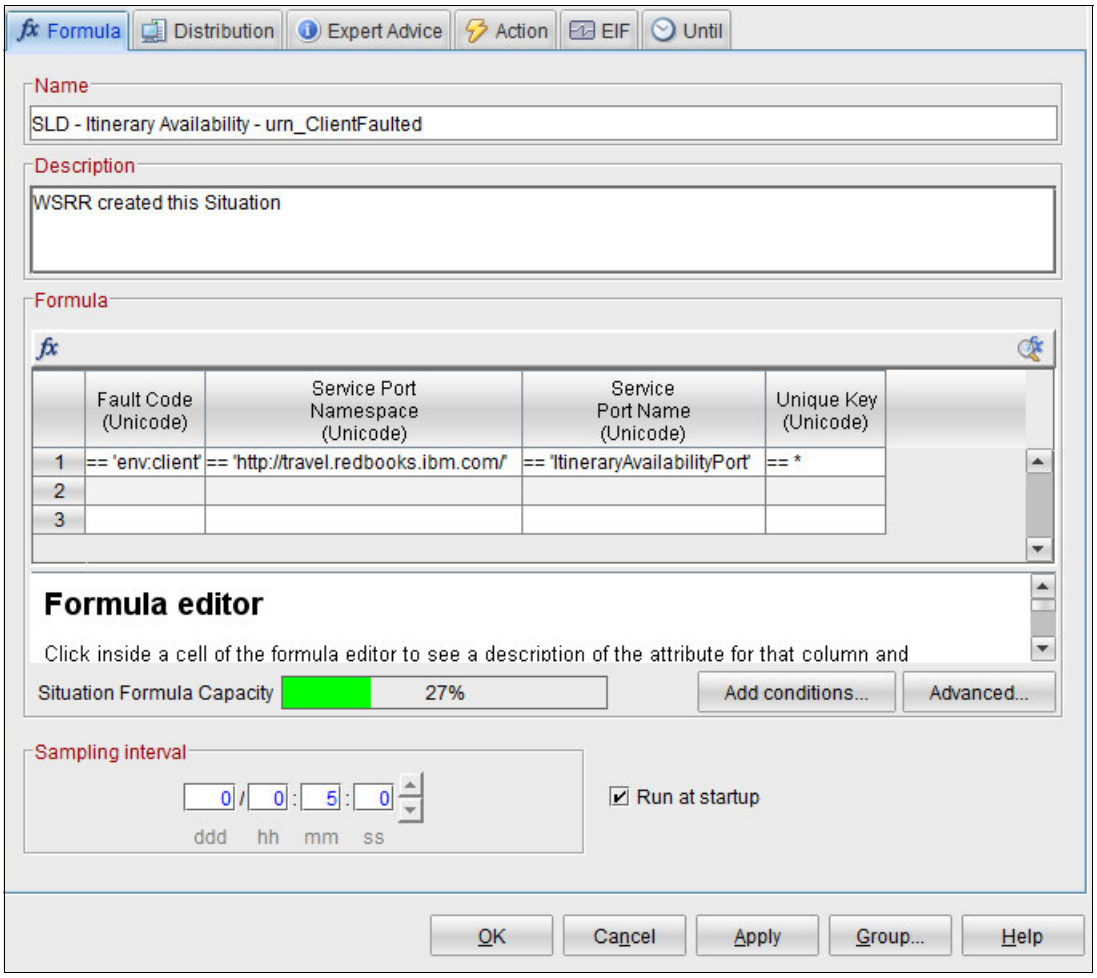


Figure 7-20 Situation on Fault log table

Services Inventory Requestor table

This table lists the requester identities that are attempting to access the services and various metrics of them. A full list of parameters is in Appendix B, “ITCAM monitoring attribute tables” on page 421.

Unique among the tables, Services Inventory Requestor has a required field; you must select **Requestor ID** as one of the assertions.

Figure 7-21 shows a created policy.

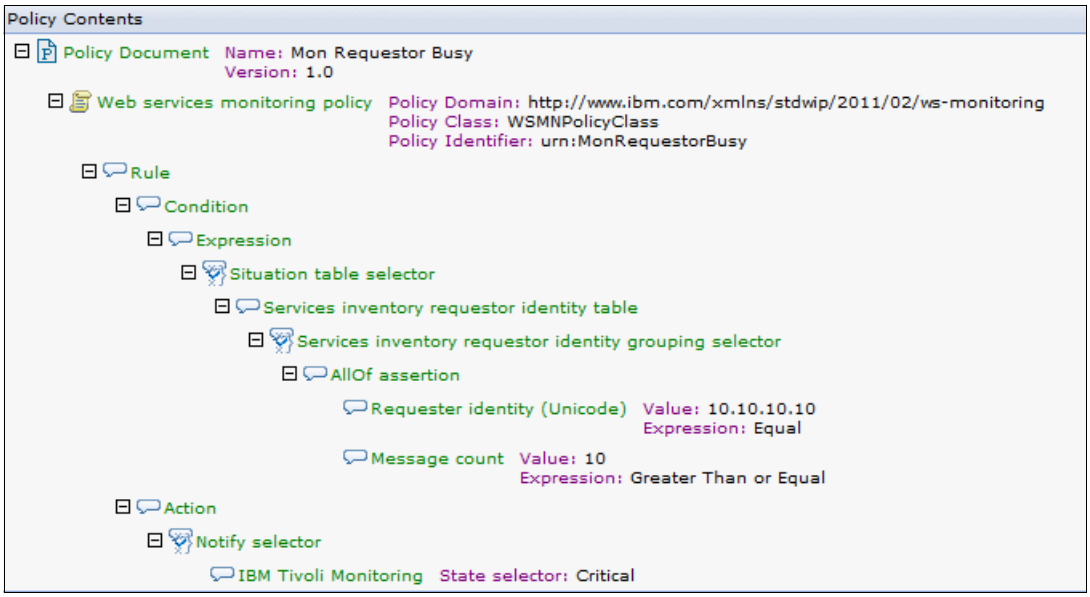


Figure 7-21 Policy contents

Figure 7-22 shows the ITCAM formula.

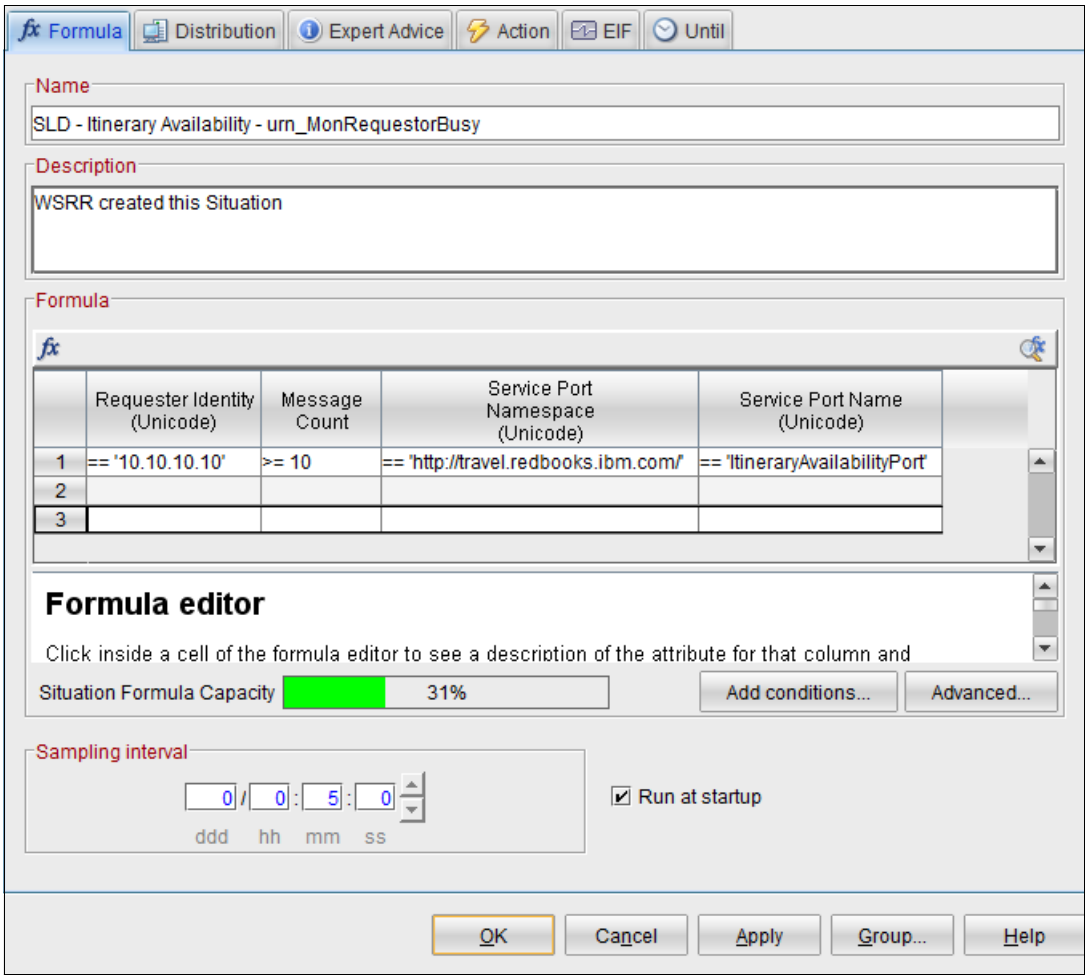


Figure 7-22 ITCAM situation with Requestor ID table

Message Arrival Threshold table

This table contains the computed data table such as the number of messages that arrive during a specified time interval. WSRR can create situations for the table only with ITCAM versions 7.2 and later. A full list of parameters is in Appendix B, “ITCAM monitoring attribute tables” on page 421.

Endpoint Inventory table

This table, which exists only in ITCAM versions 7.2 and later, contains metrics of the endpoints that are monitored by ITCAM. A full list of parameters is in Appendix B, “ITCAM monitoring attribute tables” on page 421.

Policy Enforcement table

This table is a WSRR artifact and does not represent any ITCAM table. You cannot create a valid policy using this selection.

Part 3

Policy administration point

This part contains the following chapters:

- Chapter 8, “WebSphere Service Registry and Repository for traffic management” on page 263
- Chapter 9, “WebSphere Service Registry and Repository for monitoring policy” on page 277
- Chapter 10, “Attaching a policy to a service” on page 313
- Chapter 11, “Policy administration point utilities” on page 333

© Copyright IBM Corp. 2013. All rights reserved.

8

WebSphere Service Registry and Repository for traffic management

This chapter describes WebSphere Service Registry and Repository (WSRR) as a policy administration point (PAP) for traffic management and mediation policies, which are then enforced by WebSphere DataPower as a policy enforcement point (PEP). It is meant to be used as a reference for creating, reading, updating, or deleting (CRUD) of policies in the PAP.

WSRR provides authoring, governance and attachment of policies to services. These polices are subsequently enforced by DataPower as specified in WSRR.

This chapter contains the following topics:

- 8.1, “Overview of WSRR as the PAP” on page 264
- 8.2, “Creating a mediation policy” on page 264
- 8.3, “Viewing a mediation policy” on page 273
- 8.4, “Updating a mediation policy” on page 273
- 8.5, “Deleting a mediation policy” on page 274
- 8.6, “Using Consumer ID and Context ID” on page 276

© Copyright IBM Corp. 2013. All rights reserved.

8.1 Overview of WSRR as the PAP

The WebSphere Service Registry and Repository Business Space UI provides a policy authoring tool that you can used to create and maintain policies and apply those policies to objects that are stored in the registry. Those objects are instances of service level definitions (SLDs) or service level agreements (SLAs). Policies can be attached to other objects but should be attached to either of these types only for automatic enforcement by DataPower.

8.2 Creating a mediation policy

WSRR V8.0 has built-in support for creating WS-MediationPolicy (referred to as *mediation policy*) documents with a dedicated user interface editor in the WSRR Business Space UI. With it, the user can quickly create a mediation policy by adding conditions and actions.

Before trying to author a mediation policy by using the WSRR Business Space UI, ensure that the system administrator configured the space for the appropriate role. The space must already be created, based on one of the following templates:

- Service Registry for Development
- Service Registry for Operations
- Service Registry for SOA Governance

Complete the following steps:

1. To start the process of authoring a new mediation policy in the WSRR Business Space UI, locate the Service Registry Actions widget on the Overview tab. This widget contains the action link, shown in Figure 8-1, that is used to start the authoring process.

Figure 8-1 Create a Mediation Policy action

2. Click **Create a Mediation Policy** to open the dialog in Figure 8-2 on page 265.

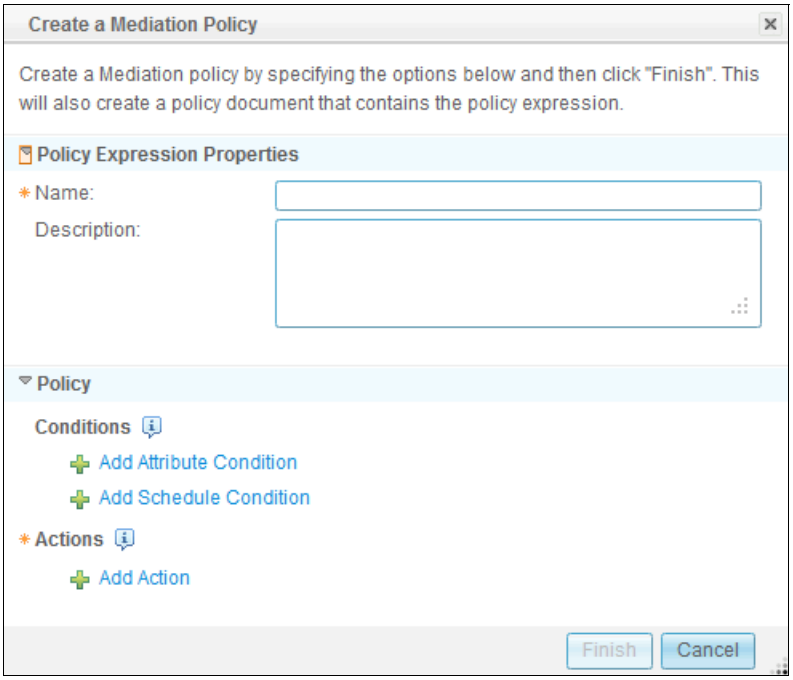


Figure 8-2 Create a Mediation Policy dialog

3. The Policy section represents a single rule element in the XML. To author multiple rules, you must create multiple mediation policies in WSRR. This is a limitation of the WSRR Business Space policy editor.

   When first entered, the dialog has the options to add an attribute condition, a schedule condition, and multiple actions. The actions added in the Actions section should be done when no conditions are present. After certain conditions are added, as shown in Figure 8-3, the actions section changes to allow the addition of actions to be done *if all conditions are true* or *if any conditions are false*.

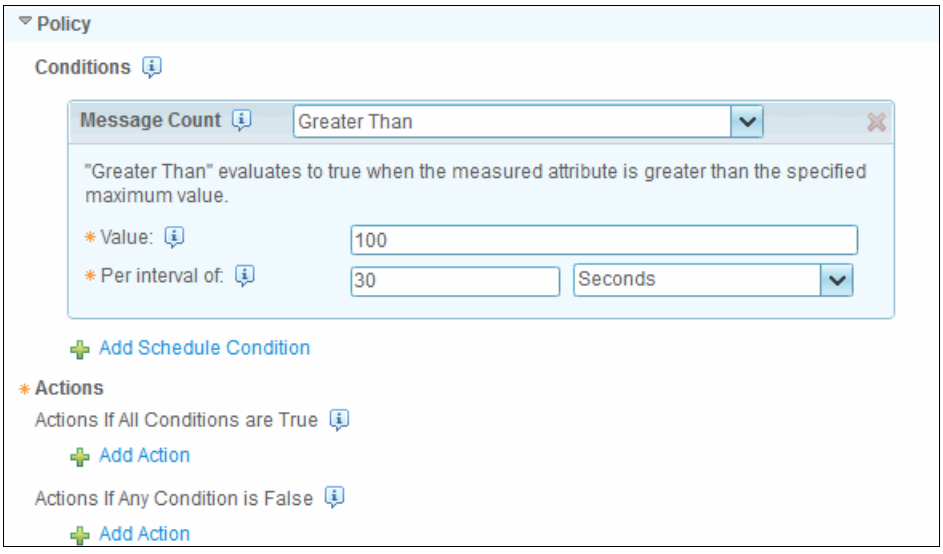


Figure 8-3 Mediation policy with one condition added

8.2.1 Attribute conditions

After clicking **Add Attribute Condition**, the area shown in Figure 8-4 opens where you can select the attribute that to which the condition applies; the area also displays a description of the attribute. After the attribute condition is added by clicking **Add**, the parameters of the condition may be set.

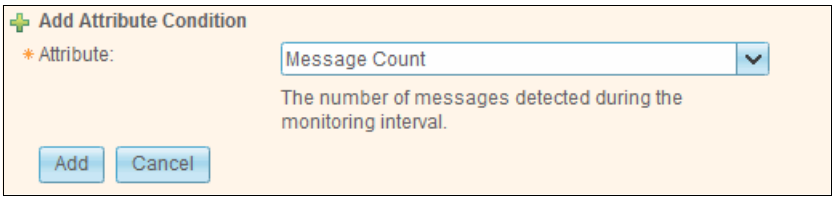


Figure 8-4 Adding an attribute condition

Use the latency conditions when setting up a policy for response time. Typically, use the Average Total Latency condition, but the following attribute conditions are available:

- Average Back-end Latency: The appliance-to-server latency in milliseconds
- Average Internal Latency: The internal DataPower latency (processing time) in milliseconds
- Average Total Latency: The sum of back-end and internal latency in milliseconds
- Error Count: The number of faults observed during this monitoring interval
- Message Count: The number of actual messages intercepted during the monitoring interval

The available operators for the conditions are as follows:

- Greater Than: A simple numeric algorithm that evaluates to true when the attribute is greater than the defined value.
- Greater Than, Allowing Bursts: A rate-based algorithm that allows bursting.

  The algorithm consists of a bucket with a maximum capacity of limit tokens. The bucket refills at a constant rate of value tokens per interval, while for each unit of the attribute specified, a token is removed. This algorithm evaluates to true when there are no tokens in the bucket, and evaluates to false otherwise. Here is an example to help explain the algorithm:

  a. Assume Limit=100, Value=5, Interval=1 second, and the Attribute=MessageCount.
  b. The bucket starts full with a maximum capacity of 100 tokens.
  c. When a message arrives, the algorithm checks whether the bucket holds any tokens.
  d. If it does, the algorithm evaluates to false and one token is removed from the bucket.
  e. If it does not, the algorithm evaluates to true.
  f. Continuously, every second, the algorithm adds 5 tokens back to the bucket, as room permits.
- Greater Than, Until Less Than: An algorithm that evaluates to true when the specified attribute reaches the high threshold specified as the value and then continues to evaluate to true until the attribute reaches the low threshold specified as the limit.
- Less Than: A simple numeric algorithm that evaluates to true when the attribute is less than the defined value.

Operator mapping:

- The `Greater Than, Allowing Bursts` operator is available only for the two count conditions. This operator maps to `TokenBucket`
- The `Greater Than, Until Less Than` operator maps to `HighLow` in the underlying policy.

Additional information is in a developerWorks article named “SOA governance using WebSphere DataPower and WebSphere Service Registry and Repository, Part 1: Leveraging WS-MediationPolicy capabilities,” which is at the following location:

`http://www.ibm.com/developerworks/websphere/library/techarticles/1204_burke/1204_burke.html`

The operators behave as their names suggest. As shown in Figure 8-3 on page 265, a condition with an attribute of `Message Count`, an operator of `Greater Than`, a `Value` of `100`, and a `Per interval` of 30 seconds was defined. This definition means that this condition will evaluate to true if the number of messages, measured over a 30-second interval, is greater than 100. After the number of messages drops below 100, measured over the same interval, then the condition will evaluate to false.

The fields that are available for input vary in number and name according to the attribute and operator chosen. In Figure 8-5, the `Greater Than, Allowing Bursts` operator is selected. This changes the first field to `Normal maximum` and we have an extra field named `Allowing bursts of up to`. In this example, the `Greater Than, Allowing Bursts` operator behaves like the `Greater Than` operator, except that a burst of 200 messages over a 30-second interval, or two consecutive bursts of 150 messages over two 30-second intervals, and so on, will cause the condition to evaluate to false.

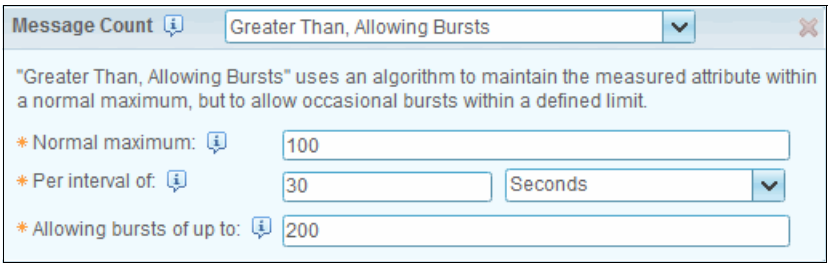


Figure 8-5 A condition that uses the Greater Than, Allowing Bursts operator

In Figure 8-6, `Greater Than, Until Less Than` operator is selected in the drop-down menu. This operator behaves like `Greater Than`. The exception is that after it begins to evaluate to true, it continues to evaluate to true until the number of messages drops below the value in the `Until less than` field.

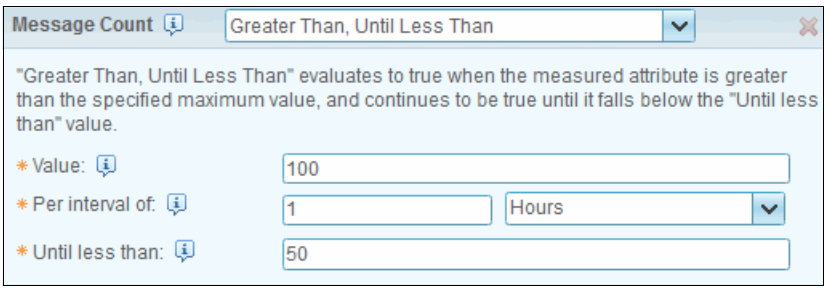


Figure 8-6 A condition that uses the Greater Than, Until Less Than operator

The `Less Than` operator presents the same fields as the `Greater Than` operator, but evaluates to false while the measured value is above or equal to the defined value, and true only when the measured value is below the value that is defined in the Value field.

8.2.2 Schedule conditions

By clicking **Add Schedule Condition**, a condition that evaluates to true only during a specific time period can be added. Dates for the period to start and end on, days of the week to include, and also restrictions on the time of day that is included in the schedule for each day can be specified. A simple schedule is shown in Figure 8-7. This schedule evaluates to true for the period of one year and one day starting on 5 November 2012. Note that the Start date and End date fields are inclusive. For the benefit of readers who are used to working with the XML Stop date, which is exclusive, it is indicated in the text that is displayed immediately below the end date.

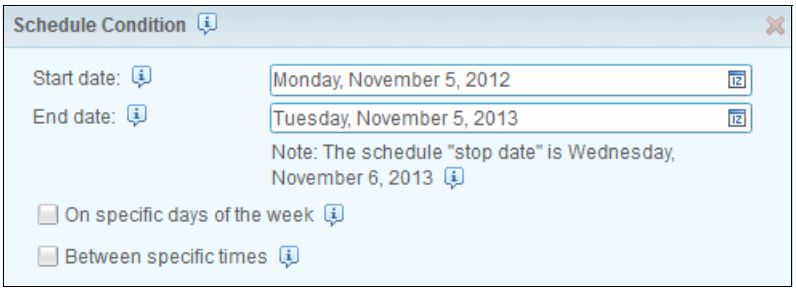


Figure 8-7 A simple schedule condition

When days of the week and time of the day are also specified, the schedule condition looks like the example shown in Figure 8-8.

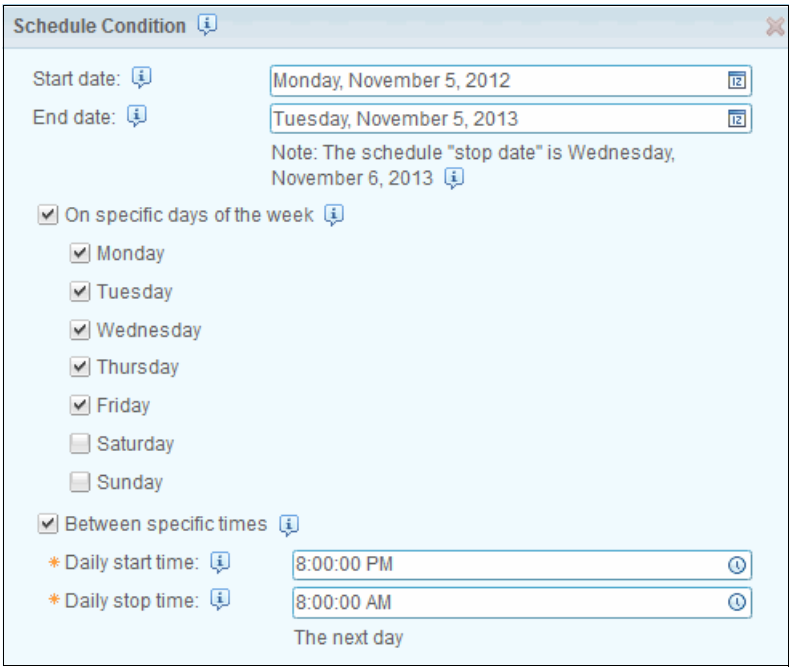


Figure 8-8 A complex schedule condition

This example evaluates to true for the period of one year and one day, starting on 5 November 2012, but only on Monday - Friday, from 8 p.m. in the evening until 8 a.m. in the

morning. In this example, because of the times of day that are specified are running over a day boundary, the schedule still evaluates to true on Saturday until 8 a.m., even though Saturday is not included in the schedule. In the figure, this information is indicated by the `The next day` text.

The schedule also still evaluates to true on Wednesday, 6 November, until 8 a.m., because the interval starting at 8 p.m. and running until 8 a.m. is started on Tuesday, 5 November. The schedule condition parameters may be specified in any combination. That is, you can specify only a Start date or only an End date, or only specific days of the week or only between specific times. Or, any combination of the four may be specified.

8.2.3 Actions

Add an action by clicking **Add Action**. If no conditions are specified for the policy, then the actions will always be executed. If conditions are specified for the policy, then the actions in the following relevant section will be executed based on the result of evaluating the conditions:

`Actions If All Conditions are True or Actions If Any Condition is False`

The actions available for use in the policy are as follows:

- Execute XSL Transformation: This action is used to transform message formats by using XSLT (Extensible Stylesheet Language Transformation), which is a language for transforming XML documents into other XML documents. Sometimes this action is used to execute XSLT style sheets that do not do transformation but accomplish some other type of task (as shown in Chapter 5, “Versioning with custom policy” on page 119 and Chapter 6, “Security using custom policy” on page 187).
- QueueMessage (not available in the Actions If Any Conditions are False section): This action causes the current transaction to be queued (wait) until such time as the conditions that cause the queue are not true.
- RejectMessage: This action is used to reject (throttle) the current transaction with a SOAP fault being returned.
- Notify: This action is used to write information to the DataPower log.
- RouteMessage: This action is used to set the back-end destination for the routing of the message to a secondary endpoint.
- ValidateMessage: This action is used to validate XML message structure against an XML Schema Document (XSD)

By clicking **Add Action**, an inline area opens (Figure 8-9) where you select an action to perform and a description of the action is displayed. After selecting the action, click **Add**. The parameters of the action may be set and more actions may also be added by subsequent clicking of **Add Action**.

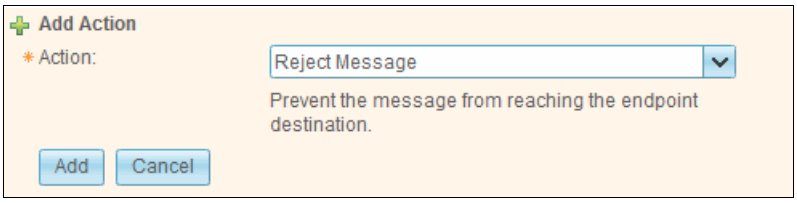


Figure 8-9 Adding an action area

The Queue Message, Reject Message, and Notify actions have no parameters, as Figure 8-10 shows.

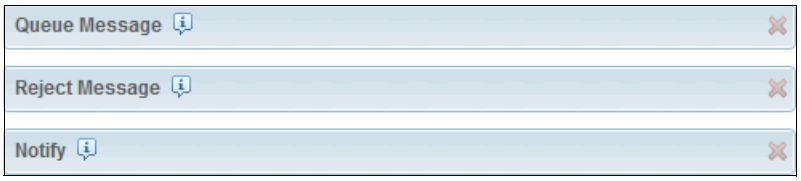


Figure 8-10 Actions with no parameters

Route Message (Figure 8-11) takes an endpoint as the parameter to which the message will be routed. This endpoint can be any textual input and is interpreted by the enforcement point at run time.

Figure 8-11 Route message action

Validate Message requires either an XSD or WSDL file to validate the message against. In the drop-down menu of the title bar of the action, select the type of file the action will use. Specify one of the following options (see Figure 8-12):

- The XSD document is in the registry: Select this option if the file is stored in WSRR, and to search for the file by name.
- Enter the XSD document URL: Select this option and type the URL that identifies the file. Then, specify the Validation Scope by selecting a buttons at the bottom of the action. Figure 8-12 shows a validation by an XSD that is identified by a URL.

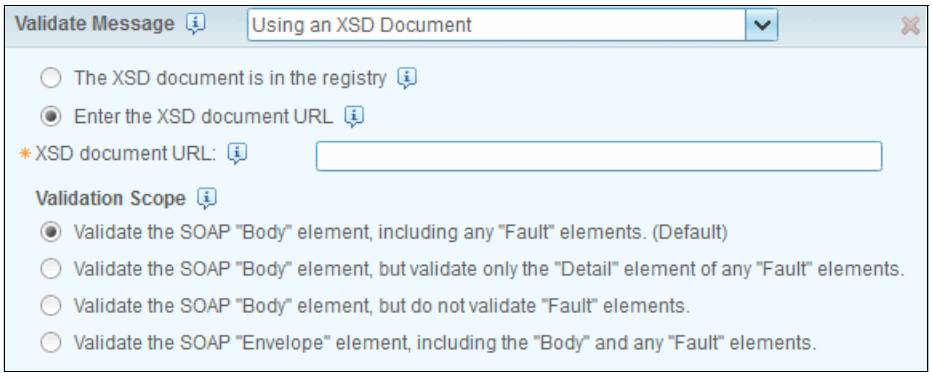


Figure 8-12 Validate message by XSD

Figure 8-13 shows a validation by a WSDL. The user is about to identify the WSDL file from WSRR that is using a suggestion given by the search field, after typing ItineraryA into the field. Note that the Validation Scope field is not present for the validation by WSDL.

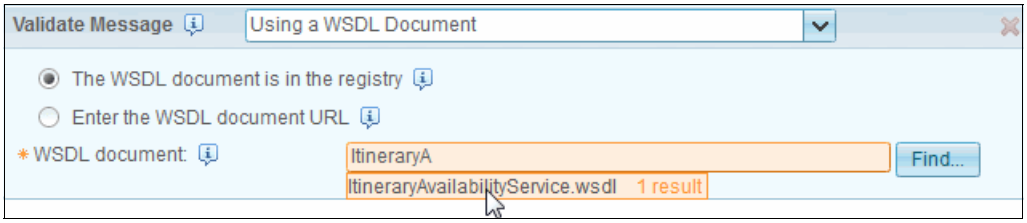


Figure 8-13 Validate message by WSDL

The Execute XSLT Transform action requires the URL of the style sheet to use for the transformation and, optionally, some parameters to pass to the style sheet. Figure 8-14 shows a simple example of this action, where there are no parameters needed.

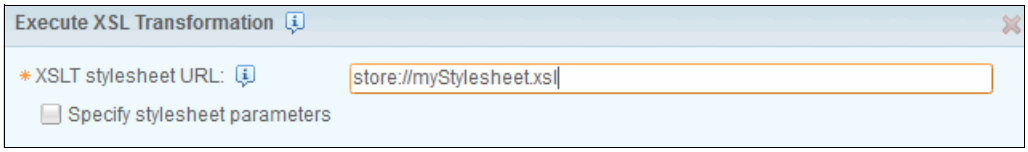


Figure 8-14 Execute XSLT Transform with no parameters

Note: The XSLT stylesheet URL must start with either `store://` or `local://`.

In Figure 8-15, the user selected the check box to specify several parameters. You can add name and value pairs that are passed to the style sheet when they are executed by the enforcement point. Add a new parameter by clicking **Add Parameter**. You can reorder these parameters by using the arrow icons at the left of each row or by dragging and dropping.

A parameter may have no value, because you can override a default value specified in the style sheet with a blank value. You can have empty rows in this section. The empty rows are factored out when the policy is saved. A parameter value without a parameter name is not valid, as indicated by the red highlight of the name field for the parameter with the `valueC` value.

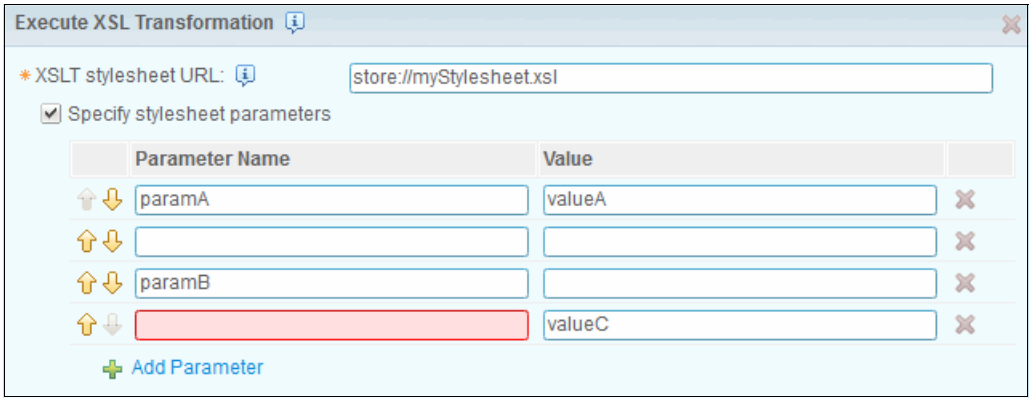


Figure 8-15 Execute XSLT Transform with some parameters

Important: When defining an XSL transformation in a mediation policy, you can specify a list of parameters for this transformation. Furthermore, the transformation must be loaded on a DataPower appliance in either the `local://` directory of a domain or in the `store://` directory of the `default` domain.

Saving and editing

After the policy is complete, click **Finish** in the dialog to save the policy to WSRR. A disabled **Finish** button means that a mandatory field is not complete or the value is invalid. A field that is incomplete has a red asterisk (*) to the left of the field name. A field that contains a value that is not valid has either a red border as shown in Figure 8-15 on page 271, or a Warning icon to the right of the field, as shown in Figure 8-16.

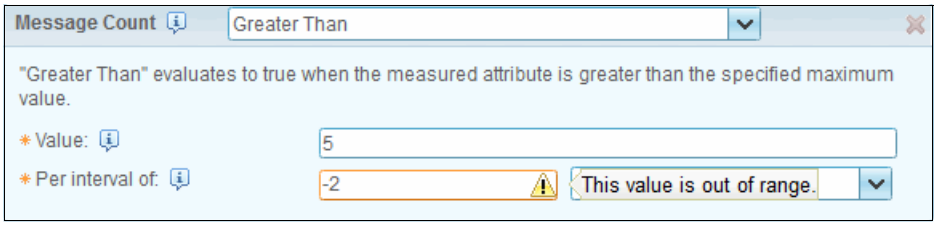


Figure 8-16 A field with an invalid value

After the policy is saved to WSRR, the Detail widget loads a view of the policy as shown in Figure 8-17.

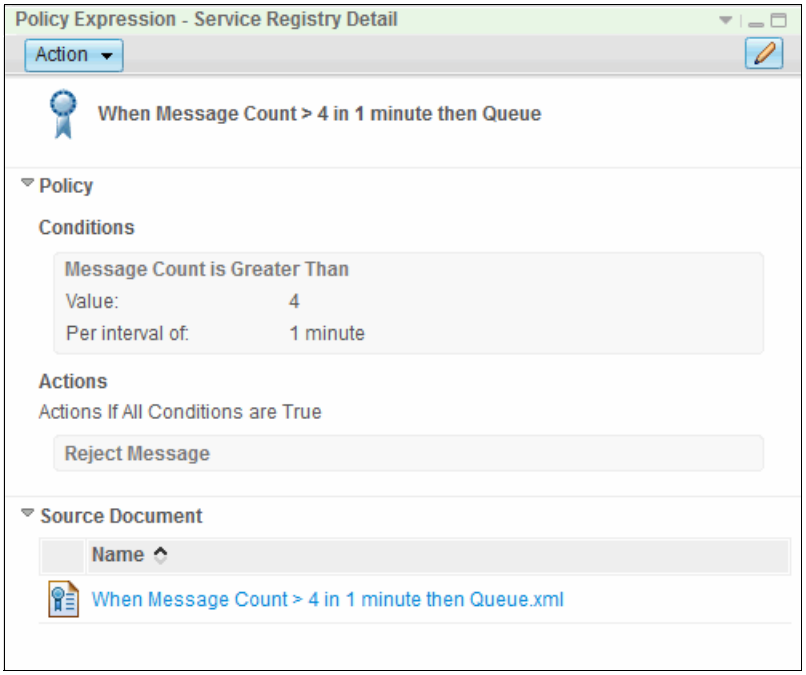


Figure 8-17 Completed mediation policy

When viewing the policy in the Detail widget, only the sections that have some content is shown. In this case, the Actions section, `If Any Condition is False`, is not displayed because no actions are contained within it.

In the example, a Source Document section is also shown. It is a link to the details of the document that WSRR created when the policy was saved. The document is the physical artifact that contains the XML that was generated by WSRR to represent the policy.

Detail view might differ: Details that are displayed when viewing the policy might differ if the system administrator modified the detail view.

8.3 Viewing a mediation policy

If you recently created a mediation policy, it is already displayed for you. Otherwise, search for it by using the Search widget as shown in Figure 8-18. In the drop-down menu, where `All Listed Types` is selected, select **Policy Expression**.

Figure 8-18 The Search widget

You can now begin typing the name of your policy or click **Search** to see a list of policy expressions. The results are displayed in the Collection Widget (Figure 8-19). If you are not already viewing the Browse page, then searching will switch to it. Use the controls at the bottom of the widget, if necessary, to find the policy you want to view and click on it. It will be displayed in the Detail widget.

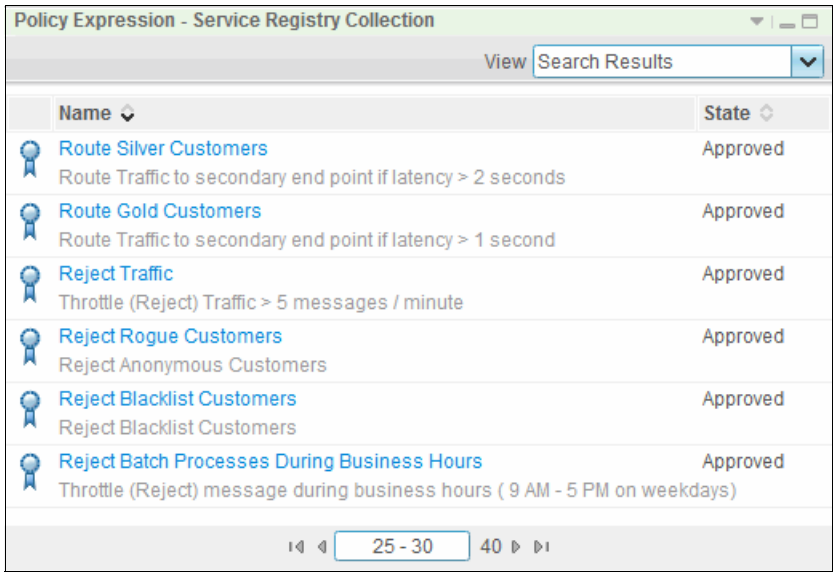


Figure 8-19 The Collection Widget showing results from a Policy Expression search

8.4 Updating a mediation policy

To a edit a policy, you must first view its details. To edit the policy from the Detail widget, click the **Pencil** icon in the top right corner of the widget (Figure 8-17 on page 272). You can then edit the policy in the same way as when it was created.

8.5 Deleting a mediation policy

Deleting a policy is similar to deleting any object in WSRR.

No explicit deletion: A policy expression cannot be deleted explicitly, but instead the policy document itself is deleted causing the deletion of the expression.

Complete the following steps to delete a policy:

1. Navigate to the policy (described in 8.3, “Viewing a mediation policy” on page 273) so that it is displayed in the Detail widget.
2. Click the name of the source document. In this example, click **When Message Count > 4 in 1 minute then Queue.xml** (shown in Figure 8-17 on page 272). Alternatively, you can search for the Policy Document directly.
3. Click **Action** → **Delete**, as shown in Figure 8-20. In the Delete confirmation dialog that opens, click **Yes**.

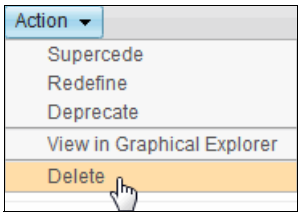


Figure 8-20 Selecting Delete from the Action menu

Missing option to delete: The option to delete a policy document is available by default on all spaces except those that are based on the Service Registry for Business space. If the option is missing from your space check with your system administrator.

4. After deleting the Policy Document, a dialog (Figure 8-21) indicates that deletion is successful.

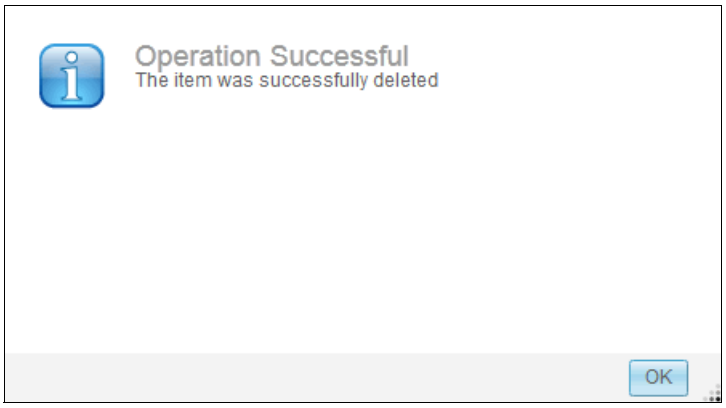


Figure 8-21 Successful deletion message

Certain criteria allow for the deletion of a policy. If these are not met, an error message displays similar to the message shown in Figure 8-22.

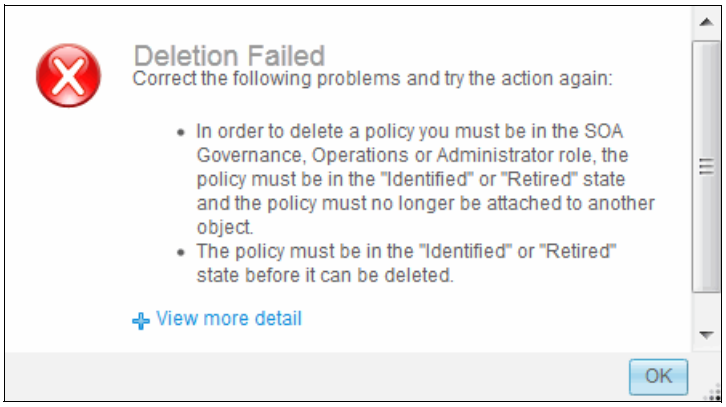


Figure 8-22 Deletion failed message

5. Click **View more detail** to view the error messages, as shown in Example 8-1. In this case the policy was in the incorrect lifecycle state.

Example 8-1 Deletion failed details

```
ERROR GSR1428E: The Governance Policy Validator has encountered 2 problems.
ERROR GSR1432E: AllOfAssertion: In order to delete a policy you must be in the
SOA Governance, Operations or Administrator role, the policy must be in the
"Identified" or "Retired" state and the policy must no longer be attached to
another object.. At least one of the assertions contained within the
AllOfAssertion failed ?[Assertion name:
SOAPolicyLifecycleUpdateRights-Delete.CheckStateRoleAndAttachments]?
ERROR GSR1424E: ClassificationAssertion: The policy must be in the "Identified"
or "Retired" state before it can be deleted.. Entity was not classified
correctly to match query
/WSRR/BaseObject/[classifiedByAnyOf('http://www.ibm.com/xmlns/prod/serviceregis
try/lifecycle/v6r3/LifecycleDefinition#SOAPolicyLifecycle_Identified','http://w
ww.ibm.com/xmlns/prod/serviceregistry/lifecycle/v6r3/LifecycleDefinition#SOAPol
icyLifecycle_Retired')] ?[Assertion name:
SOAPolicyLifecycleUpdateRights-Delete.CheckClassifications]?
```

The rules for policy deletion say that the user must embody at least one of the following WSRR roles:

- SOAGovernance
- WSRRAdmin
- Operations

The policy must also be in either of the following states:

- Identified
- Retired

Also ensure that the policy is not attached to any objects in WSRR, although it is not good practice to have a policy in either the Identified or Retired state attached. See the following website for more details about the roles in the governance enablement profile:

http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/rwsr_gep_roles.html

8.6 Using Consumer ID and Context ID

Consumer ID and Context ID can be easy to use if you understand when they make sense for your policy and how to complete the required information in WSRR. See Figure 8-23.

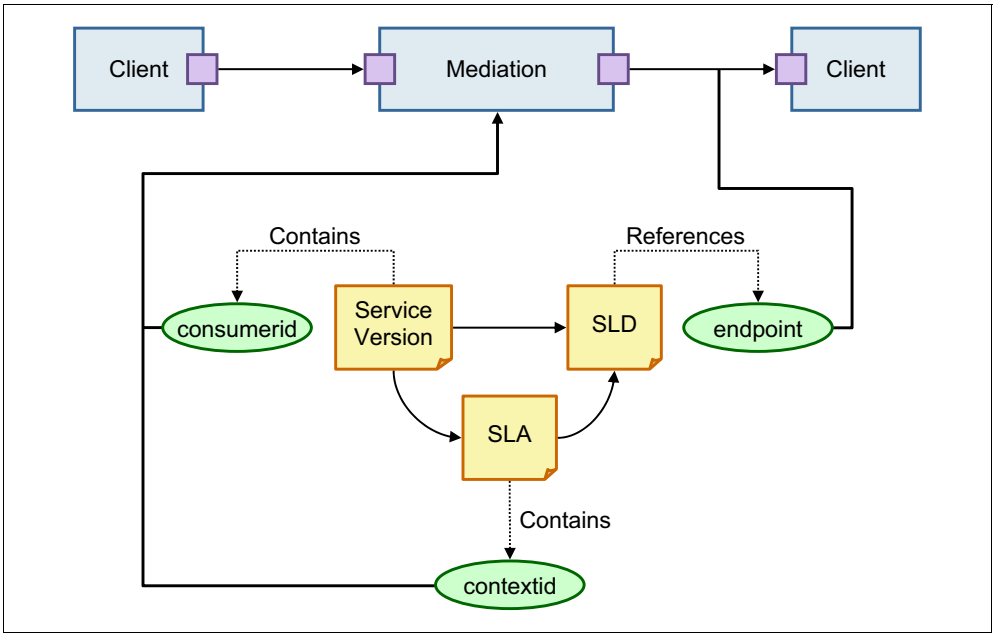


Figure 8-23 Where Consumer ID and Context ID go in WSRR

Use the Consumer ID when policy is being specified for a consumer-provider pair. The Consumer ID specifies the identity of the consumer that this policy is for and is specified in the WSRR Service Version for the SLA. During policy enforcement, the Consumer ID is passed in the transaction message header in a manner and location that is specified in the SLD.

Use the Context ID when policy is being specified for a consumer-provider pair *and* it is necessary to further differentiate the consumer transaction based upon certain criteria. An example of such criteria is customer differentiation, such as gold, silver, and bronze. The Context ID is specified in the SLA. During policy enforcement, the actual Context ID is passed in the transaction message header in a manner and location as specified in the SLD.

For further details, see Chapter 4, “Policy traffic management and consumer provider pairs” on page 87, which has several examples of using Consumer ID and Context ID.

9

WebSphere Service Registry and Repository for monitoring policy

This chapter serves as a reference for monitoring policy. It describes the interaction between IBM WebSphere Service Registry and Repository (WSRR) as the policy administration point (PAP) and IBM Tivoli Composite Application Manager (ITCAM) as the policy monitoring point (PMP), and their various roles.

WSRR is responsible for overall governance throughout the entire lifecyle of the business SOA solution and is oriented toward application and workflow. It is usually managed by the SOA architects with an aim towards the business goals of the products.

ITCAM is operations oriented and is often managed by the operations group who do not have an overall view of the business characteristics and requirements of the services, rather they are concerned with overall uptime and health of the systems.

The WSRR and ITCAM integration bridges this gap by enabling the service owners to define the Monitoring Policies within WSRR and provide transparency between the business-oriented WSRR and operations-oriented ITCAM.

This chapter contains the following topics:

- 9.1, “Why monitor” on page 278
- 9.2, “Monitoring options in WSRR” on page 279
- 9.3, “When a threshold is crossed” on page 296
- 9.4, “Advanced monitoring with ITCAM” on page 308

© Copyright IBM Corp. 2013. All rights reserved.

9.1 Why monitor

Today, companies focus on providing innovative services. To deliver these services, IT and operations departments must strive to guarantee compliance, security, and continuous uptime. These areas all play a part in helping to ensure these business services are effectively performed to support the organization's business goals.

9.1.1 Monitoring or IBM Service Management

A common practice is for companies with organizational silos and traditional implementations to become entrenched in managing aspects such as IT infrastructure technologies, single product revenues and expenses, and individual processes and organizational efficiencies, instead of managing the integrated solutions and business services that are delivered by the sum of all these components.

ITCAM is designed to help you manage your business. Because IBM understands that IT and operations are a part of your business, IBM created the alignment between ITCAM, which represents IT and WSRR, which represent the business side of SOA.

WSRR and ITCAM together create a common language that can be presented to application designers, architects, and users in WSRR, and for operators and technical staff in ITCAM.

All IBM solutions are based on IBM and industry best practices, such as the IT Infrastructure Library (ITIL), Control Objectives for Information and related Technology (COBIT) and enhanced Telecom Operations Map (eTOM). These best practices help users, operators, and managers ensure that IT and operational processes are consistently designed, automated, and executed, and are auditable for compliance adherence.

ITCAM helps you anticipate and plan for change by providing timely access to critical information.

9.1.2 Terminology

ITCAM and WSRR are products with a separate class of users in mind. ITCAM is often used by operators and development and operations (DevOps) teams with reports and dashboards for managers. WSRR is often used by developers and only occasionally by DevOps. Therefore, each product has its own terminology for capabilities that are often analogous.

WSRR, in its role as the policy administration point (PAP), provides for the creation, update, or deletion of monitoring policies. The policy monitoring point (PMP) is responsible for using the monitoring policy and creating a *situation policy* within the PMP. It is the situation policy that actually gets monitored in the PMP.

For the most part, WSRR terminology is used with specific exceptions when certain ITCAM capabilities are discussed and it would not make sense to mix WSRR and ITCAM terminology. Table 9-1 lists examples.

Table 9-1 Terminology

WSRR Term	ITCAM Term
Policy	Situation
Event	Event/Alert
Policy threshold exceeded	Situation fired

9.2 Monitoring options in WSRR

While you use WSRR as the primary location to define the monitoring policy, ITCAM does the monitoring itself. Therefore, to create monitoring policies you must understand the capabilities of ITCAM.

9.2.1 Basic ITCAM architecture

This section explains the portion of the ITCAM architecture that is relevant to monitoring options. The architecture described in this chapter is a simplified architecture that contains only the components that are necessary to explain the monitoring capabilities.

The overall architecture of the ITCAM environment is detailed in Chapter 14, “ITCAM as policy monitoring point” on page 389.

All work with ITCAM is done through the IBM Tivoli Monitoring infrastructure. Tivoli Monitoring contains both a GUI for viewing the current status of the systems monitored and creating monitoring threshold violation, and a web service interface. This interface enables interaction with WSRR for policy creation and upstream integration, for example, sending mail when a policy threshold is passed.

The ITCAM for SOA agent is an autonomous component that monitors the runtime environment and communicates with the Tivoli Monitoring infrastructure. The ITCAM for SOA agent has two components:

- A Tivoli Enterprise Monitoring Agent, which is a universal connection to the IBM Tivoli Monitoring infrastructure.
- A data collector, which connects to a specific runtime environment, for example, WebSphere Application Server, WebSphere Message Broker, DataPower, and so on.

When an ITCAM agent monitors both a DataPower and a WebSphere Message Broker, the environment will include a single Tivoli Enterprise Monitoring Agent and two data collectors.

Figure 9-1 shows a simplified logical representation of the architecture.

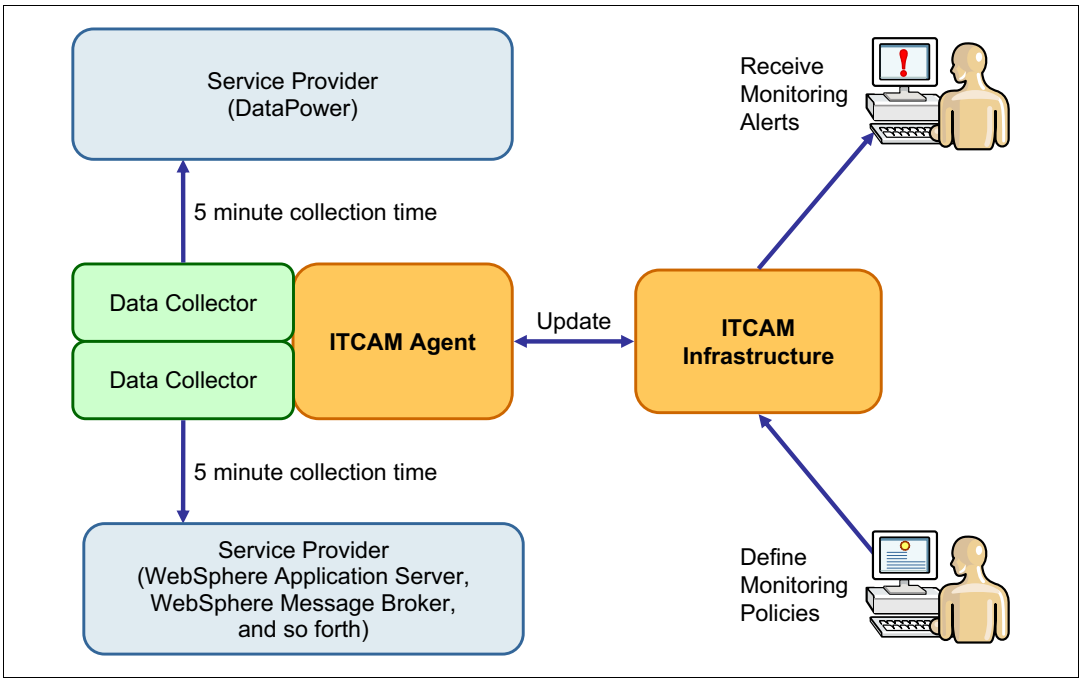


Figure 9-1 ITCAM architecture

The data collectors do not gather real-time data out of the runtime environments. Rather, they register with the run times and collect statistics every five minutes. The data collector does not gather information about every message that passes through the runtime environment. Instead, it gathers statistics and metrics such as the overall amount of messages and elapsed message round trip (average, maximum, minimum, standard deviation).

Figure 9-2 shows the Performance Summary workspace in ITCAM.

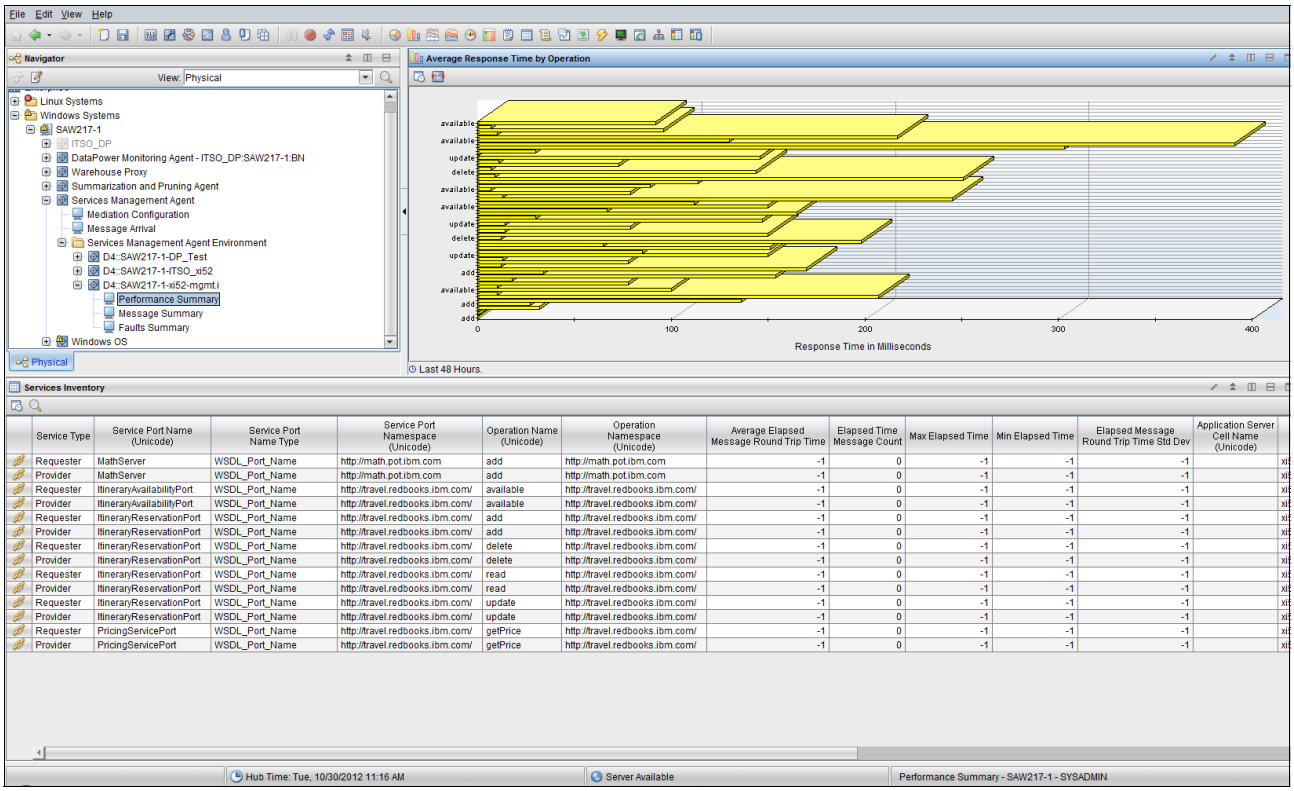


Figure 9-2 Performance Summary workspace

Figure 9-3 shows a zoomed in view of the Service Inventory table.

Service Port Name (Unicode)	Operation Name (Unicode)	Average Elapsed Message Round Trip Time	Elapsed Time Message Count	Max Elapsed Time	Min Elapsed Time	Elapsed Message Round Trip Time Std Dev	Interval Begin Time	Interval End Time
ItineraryReservationPort	delete	144	4	146	143	1	10/29/12 18:25:00	10/29/12 18:30:
ItineraryReservationPort	add	251	4	256	243	6	10/29/12 18:25:00	10/29/12 18:30:
ItineraryAvailabilityPort	available	89	5	92	86	2	10/29/12 18:25:00	10/29/12 18:30:

Figure 9-3 Service inventory table

As the figure shows, the delete and add operations both ran four times during the last monitoring cycle. Although the duration that was spent by the delete operation was the same to run each time, the add operation was much more variable, which might indicate stability issues with the operation.

A monitoring policy can be defined against each column that ITCAM displays. For example, a policy might be defined as *Elapsed Time Message Count > 100* that runs when more than 100 messages are received within five minutes. Another, more complex, policy might be *Elapsed Time Message Count > 100 and Max Elapsed Time > 250*.

In effect, a policy is performing an SQL query on the table of information that the ITCAM agent is collecting.

Tip: ITCAM checks the policy against the latest set of information that is collected by the agent. Therefore, if a policy threshold is passed, the operator's monitor reflects this. However, if, after the next collection interval, the policy is no longer true, the event will be closed and no longer display in the monitor.

9.2.2 Creating a monitoring policy

There are four stages in the creation of a new policy:

1. Initial creation and naming of the policy.
2. Defining the monitoring policy.
3. Defining ITCAM specific parameters.
4. Governing the policy.

After the policy is created, it still needs to be attached to an SLD and promoted to a runtime environment before an ITCAM situation is created.

Initial creation and naming of the policy

Use the following steps to create a new policy, access the New Policy Document, and select a policy type of monitoring:

1. From the SOA Governance perspective, click **View** → **Service Documents** → **Policy Documents**. The Policy Documents window opens, as Figure 9-4 shows.

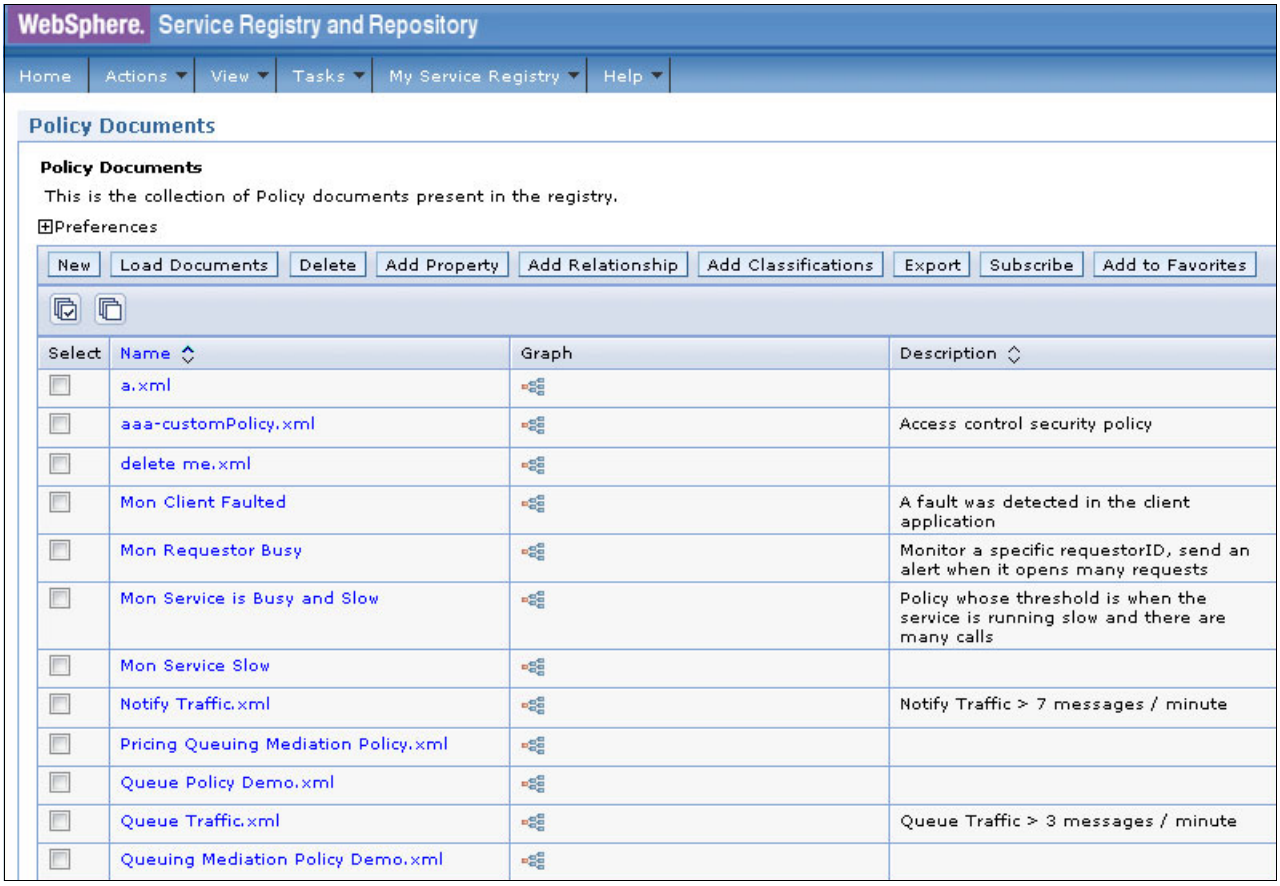


Figure 9-4 Policy Documents window

2. Click **New**, select **WS Policy Framework 1.5**, and click **Next**. See Figure 9-5.

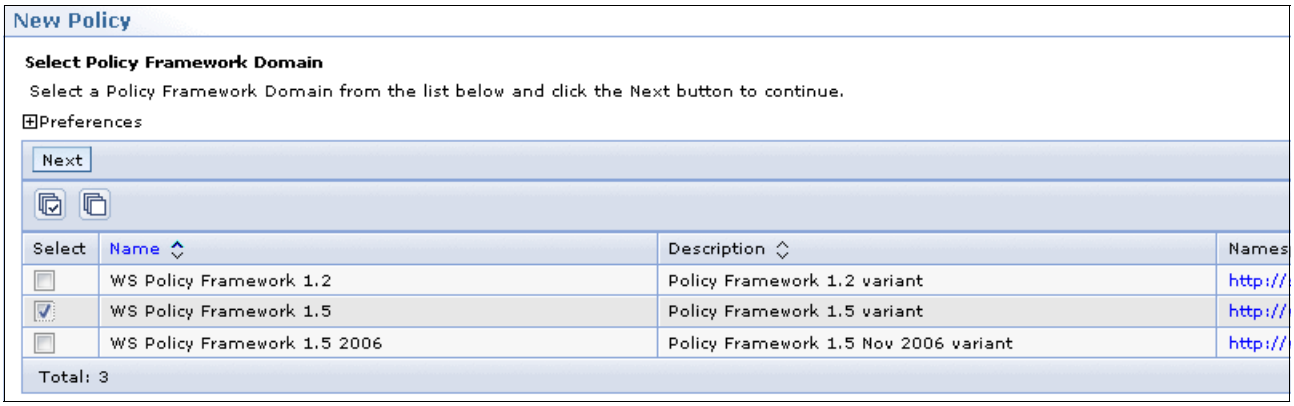


Figure 9-5 Select Framework

3. Enter a name, version, and description for the policy in the New Policy Document window (Figure 9-6). The name must be unique throughout the environment. The description and version are optional.

Figure 9-6 New policy name

Naming conventions: Your organization should have a naming convention for policies to help differentiate between monitoring and mediation policies and between policies that are specific to certain services or are global. A naming convention can be in the following format:

<PolicyType>_<ServiceName>_<PolicyName>

Examples are as follows:

- `Mon_All_MultipleFaults`
- `Mon_Pricing_TooManyMessages`

4. Click **Select Policy Domain** and choose **Service Level Monitoring Policy** from the drop-down list, as shown in Figure 9-7.

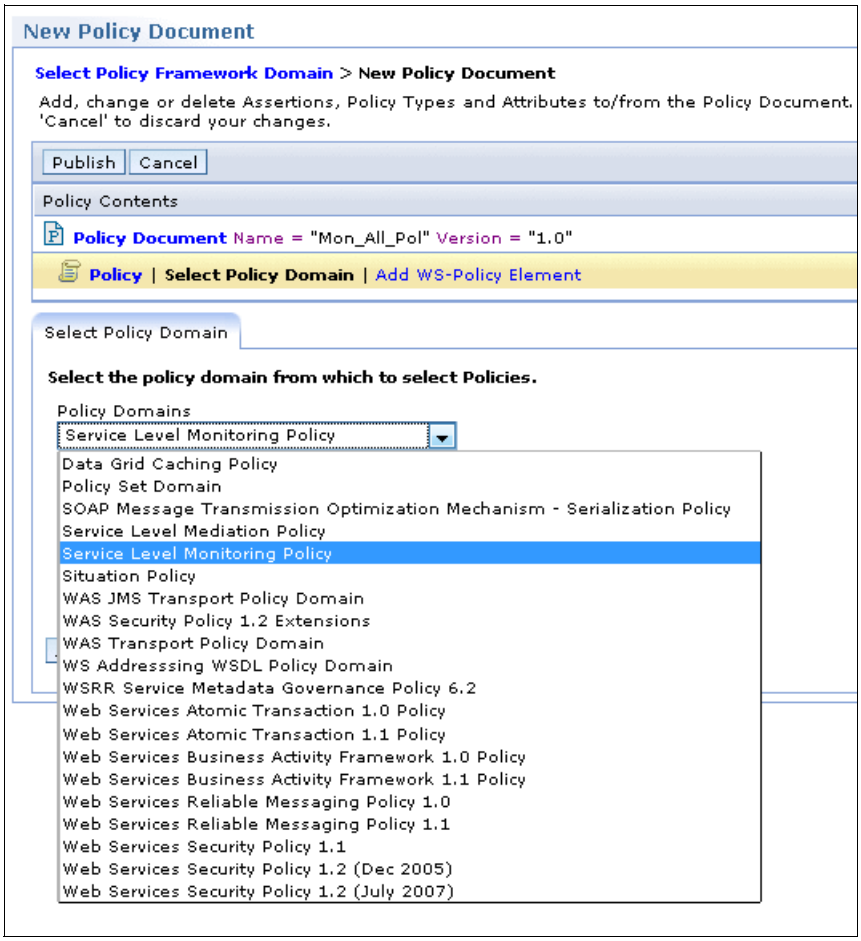


Figure 9-7 Policy Domain

5. Click **Apply** and then **Add Property.** Click **Add** again for the Policy Identifier. Enter any name you want, but a good idea is to use the name of the policy itself.

Tip: Do not remove the `urn:` text from the identifier.

At this stage, you have a policy that resembles the one in Figure 9-8.

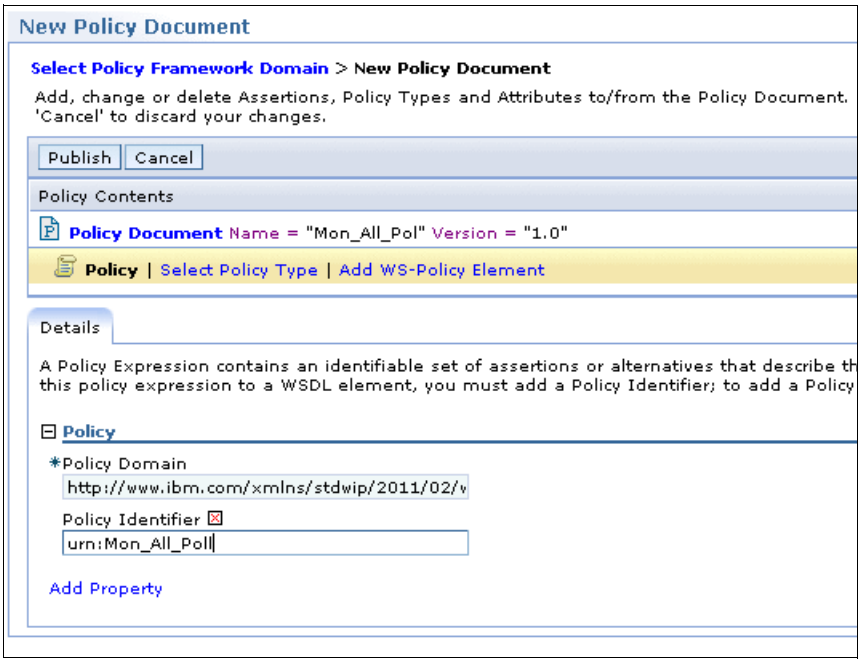


Figure 9-8 Policy Identifier

6. Click **Select Policy Type.** The only option in the drop-down menu is **Web services monitoring policy**, as shown in Figure 9-9.

Figure 9-9 Policy Type: only one option

7. Click **Select** to display the information, as shown in Figure 9-10.

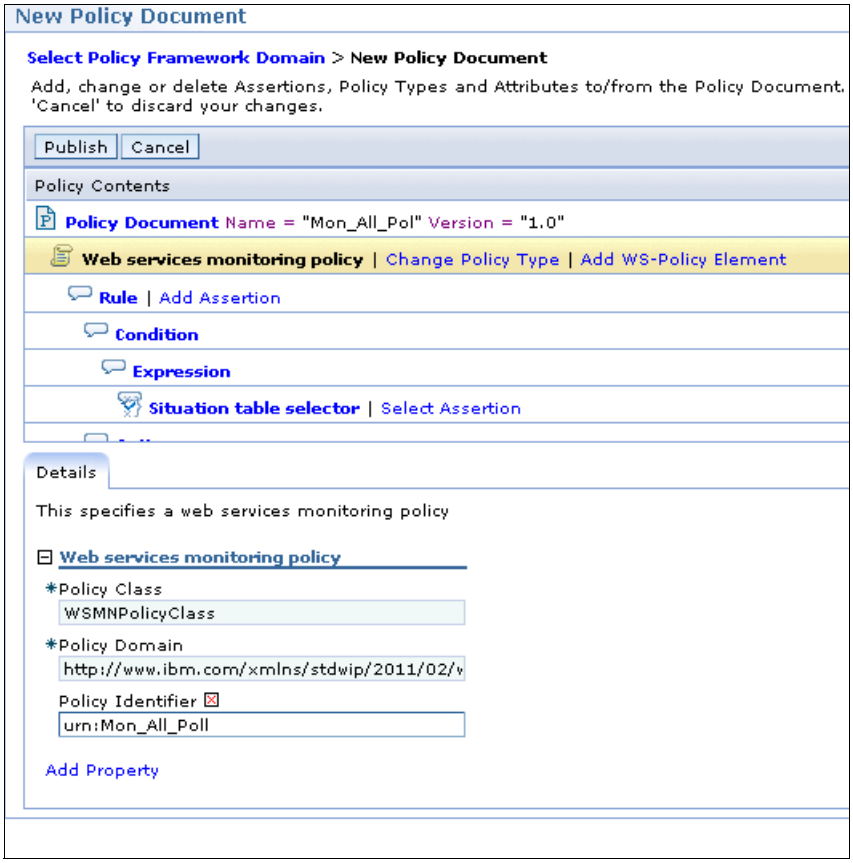


Figure 9-10 Policy ready for definitions

Defining the monitoring policy

Now that the set up for the policy to perform monitoring is complete, you must choose specifically what you want to monitor. The ITCAM paradigm is that of an SQL query on a table (see 9.2.1, “Basic ITCAM architecture” on page 279).

Click **Select Assertion** next to Situation table selector. The menu (Figure 9-11 on page 287) lists the following tables:

- Endpoint Inventory (only valid for ITCAM 7.2 and later)
- Fault log table (valid for ITCAM 7.1.1.3 and later)
- Message arrival threshold table (only valid for ITCAM 7.2 and later)
- Policy Enforcement (WSRR artifact, not in use)
- Services inventory requestor identity table (valid for ITCAM 7.1.1.3 and later; Unique ID is a mandatory field)
- Services inventory table (valid for ITCAM 7.1.1.3 and later)

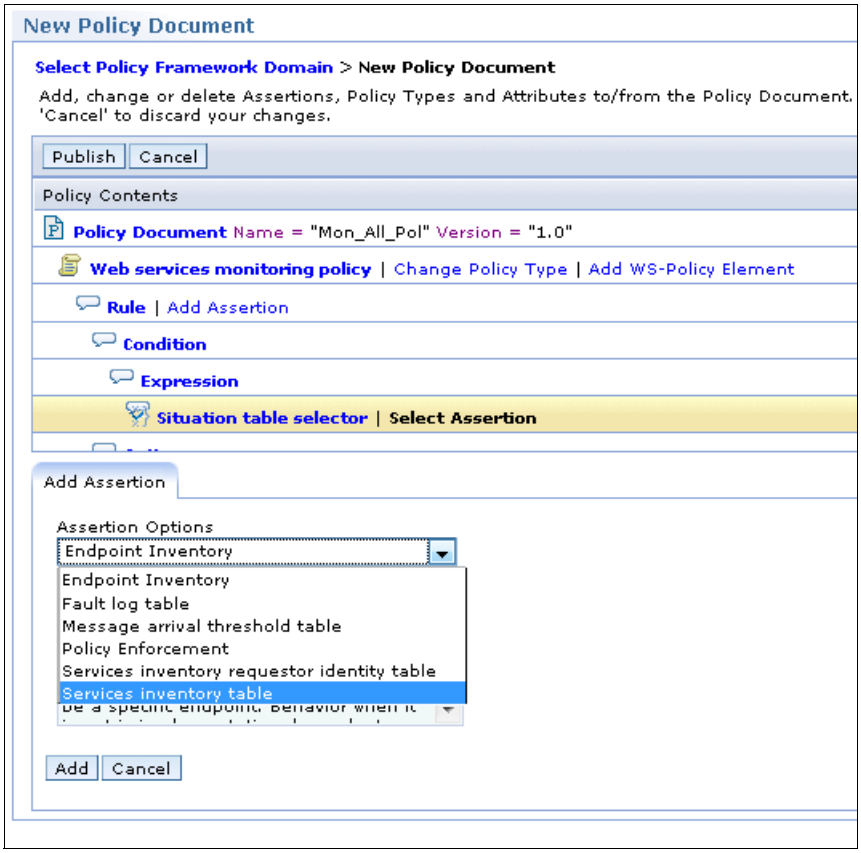


Figure 9-11 Select table

Each table represents the equivalent ITCAMforSOA attribute group and each possible assertion represents a field in the attribute group. Further details of the specific fields are in Appendix B, “ITCAM monitoring attribute tables” on page 421.

In addition to the data fields, assertions such as `AnyOf` and `AllOf` can be included; they enable the OR and AND functions and groupings.

The policy in Figure 9-12 on page 288 starts with an `AnyOf` assertion, built of a regular assertion and an `AllOf` assertion making a policy that is `A OR (B AND C)`.

Figure 9-12 shows how the monitoring policy will look in WSRR.

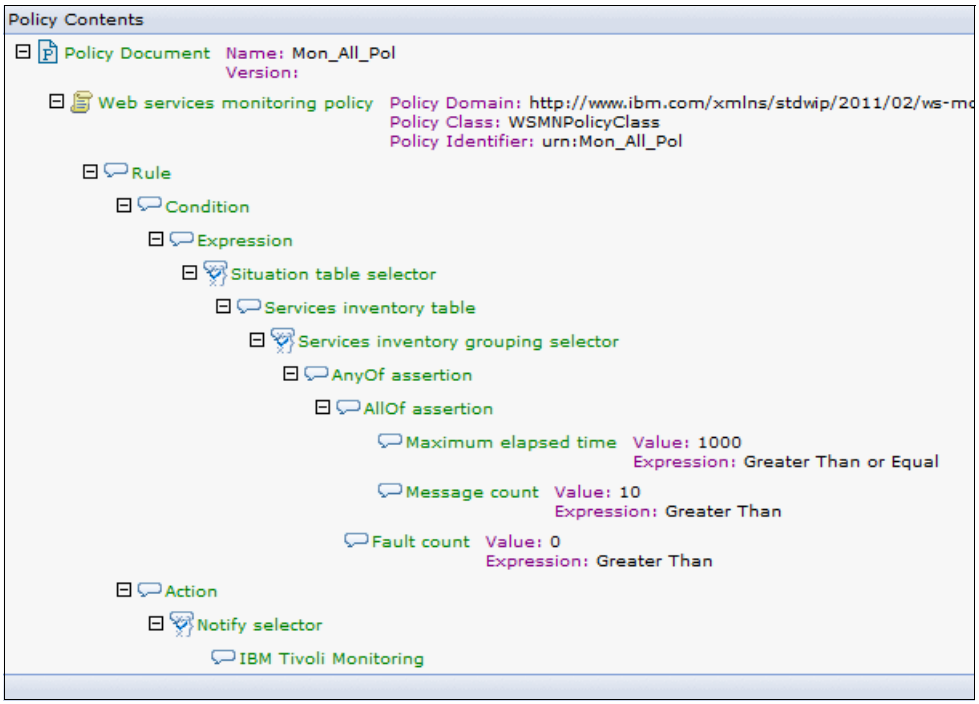
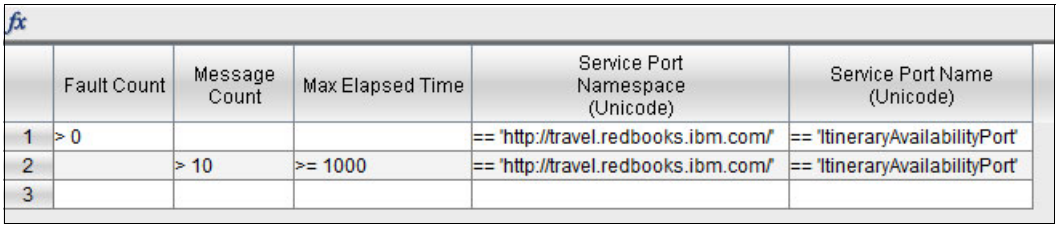


Figure 9-12 Policy contents

Figure 9-13 shows the monitoring policy in ITCAM after the process of creating the policy and synchronizing with ITCAM is complete.



	Fault Count	Message Count	Max Elapsed Time	Service Port Namespace (Unicode)	Service Port Name (Unicode)
1	> 0			== 'http://travel.redbooks.ibm.com/'	== 'ItineraryAvailabilityPort'
2		> 10	>= 1000	== 'http://travel.redbooks.ibm.com/'	== 'ItineraryAvailabilityPort'
3					

Figure 9-13 Situation contents

Two interesting artifacts in the ITCAM view are as follows:

- All fields on the same row have an AND relationship, but separate rows have an OR relationship between them.
- The ITCAM situation added the Service Port Namespace and Service Port Name, which represent the endpoint of the SLD to which the policy was attached.

Situation, threshold, endpoint: If several services are sharing the same name and namespace, ITCAM fires a situation for each of them when the threshold is crossed, without regard to the specific endpoint they are represented by in WSRR.

Figure 9-14 displays the same situation that fired a number of times because the situation distinguishes only between services with different Service Port Names and Service Port Namespaces. In this example, the DataPower and the WebSphere Application services have the same names and therefore a single situation monitors them all, even if the SLD is attached to only the DataPower endpoint.

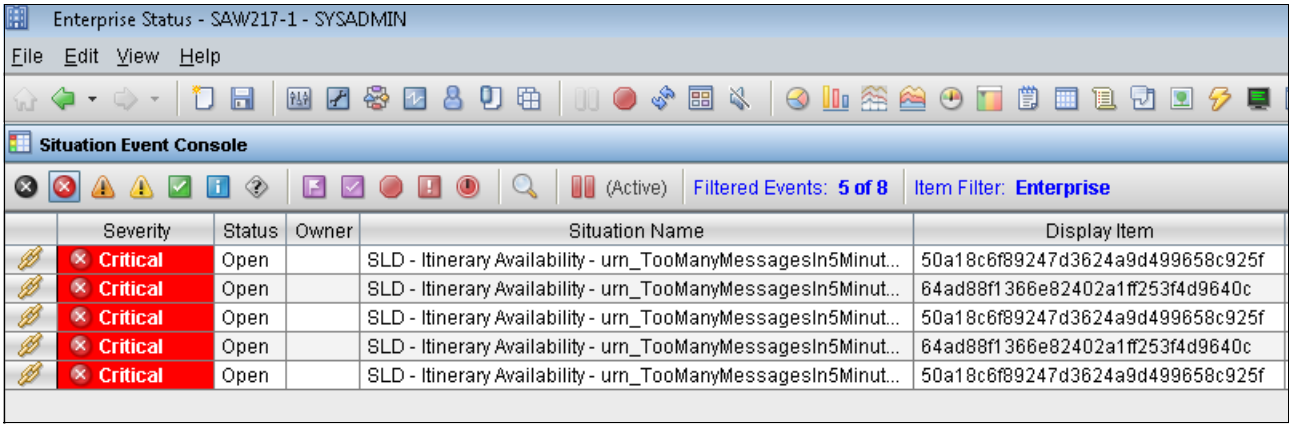


Figure 9-14 Situation Event Console

This behavior is normal and each runtime environment must decide whether the better approach is to reuse the names or to have unique names for the DataPower and WebSphere application services.

Within ITCAM, the policy definition can be represented as a table, as Figure 9-15 shows.

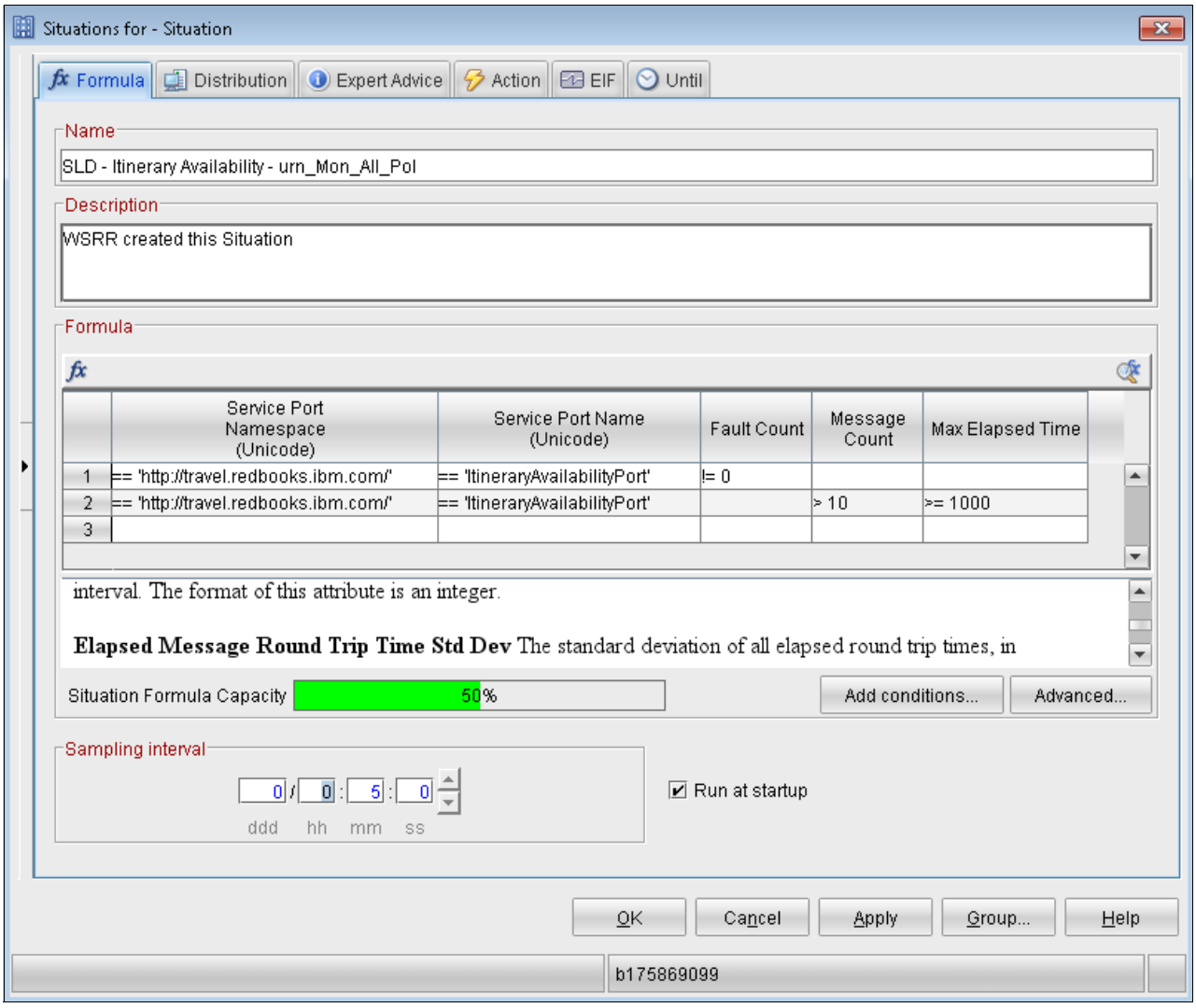


Figure 9-15 Situation Editor

Figure 9-16 shows the policy definition as a graphical flow chart and SQL query.

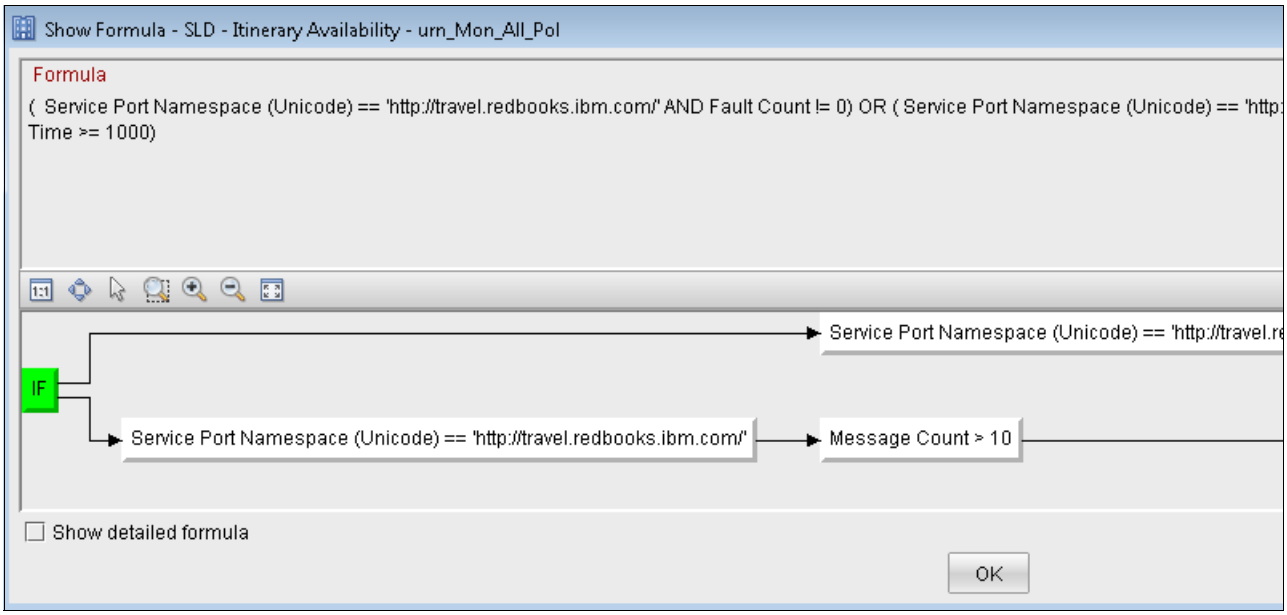


Figure 9-16 Situation Formula

Defining ITCAM specific properties

The final stage in the creation of a policy is linking the policy to ITCAM. Use the following steps to define the properties:

1. Click **Select Assertion** (under the Action tab and to the left of the Notify selector) and select **IBM Tivoli Monitoring** from the dropdown menu.
2. Click **Add**. If you do not do this step, the policy will be capable of being attached to an SLD and promoted, but will not create an ITCAM situation.

All further steps in this stage are optional and, if not chosen, will result in default or empty (but legitimate) values being entered in the ITCAM situation. Further steps are made by clicking **Add Property** under IBM Tivoli Monitoring notification.

Figure 9-17 shows example values for the properties.

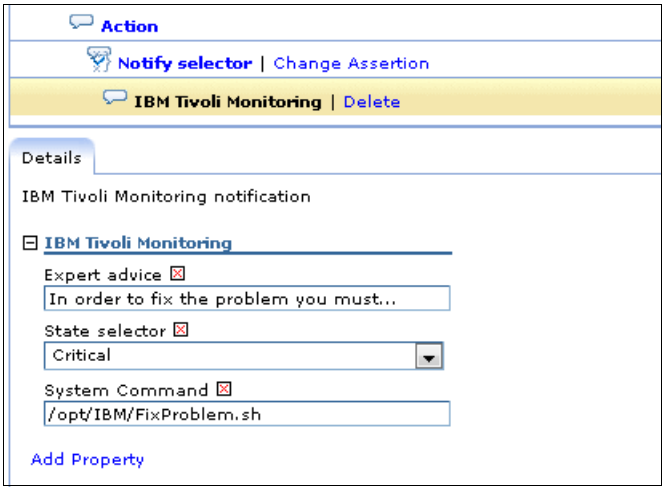


Figure 9-17 ITCAM specific properties

The fields are as follows:

- *Expert advice* is text that is used by operators to help solve the problem. It can include text such as pointers to documentation or contact information.
- *State selector* defines the severity of the policy. For example, a policy that is monitoring faults might raise a critical error when its threshold is passed, but a policy that is measuring response time does not raise more than a warning. Often, a number of policies are created that are responsible for a rising severity level:
  - Policy A threshold is over 500 milliseconds and equal to or less than 2,500, and it has a state of Warning.
  - Policy B threshold is over 2,500 milliseconds and it has a state of Critical.

  Tip: If two policies have overlapping thresholds, they might both be active at the same time and the operators will see multiple alerts. This might or might not be correct behavior, depending on the needs of the organization.

- *System command* is a command that is run by ITCAM automatically whenever the policy threshold is passed.

Two more parameters that are specific to ITCAM are Sampling Interval and Until. These parameters can be set by selecting **Add assertion** under Web services monitoring policy.

- *Sampling Interval* states how often ITCAM checks whether a policy threshold is crossed. The default value, 300 seconds, is set in the `srr_sdms_config.xml` file.
- *Until* states how long the situation will remain *alive* after the policy violation is detected. By default, a situation never closes until the policy is no longer in violation, for example, after 10 minutes ITCAM detects that the load on the runtime environment is not too high. In ITCAM, operators may mark the situation as *acknowledged*, meaning that it will not be displayed in high priority.

  Tip: This field must never be used, because a situation must never be closed automatically when the policy is still in violation. Closing a situation automatically when the policy is still in violation causes ITCAM not to recognize the correct status of the SOA environment. If a policy violation can be safely ignored, then either the policy must be changed or operators must use the tools in their OSS systems to mask the policy violations.

  More information about the field is at the following website:

  `http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3fp1/adminuse/situation_until_tep.htm`

See Chapter 14, “ITCAM as policy monitoring point” on page 389 for more information about setting the defaults for these values.

Governing the policy

Use the following steps to govern a policy:

1. Click **Publish** to create the policy. Figure 9-18 shows the created policy.

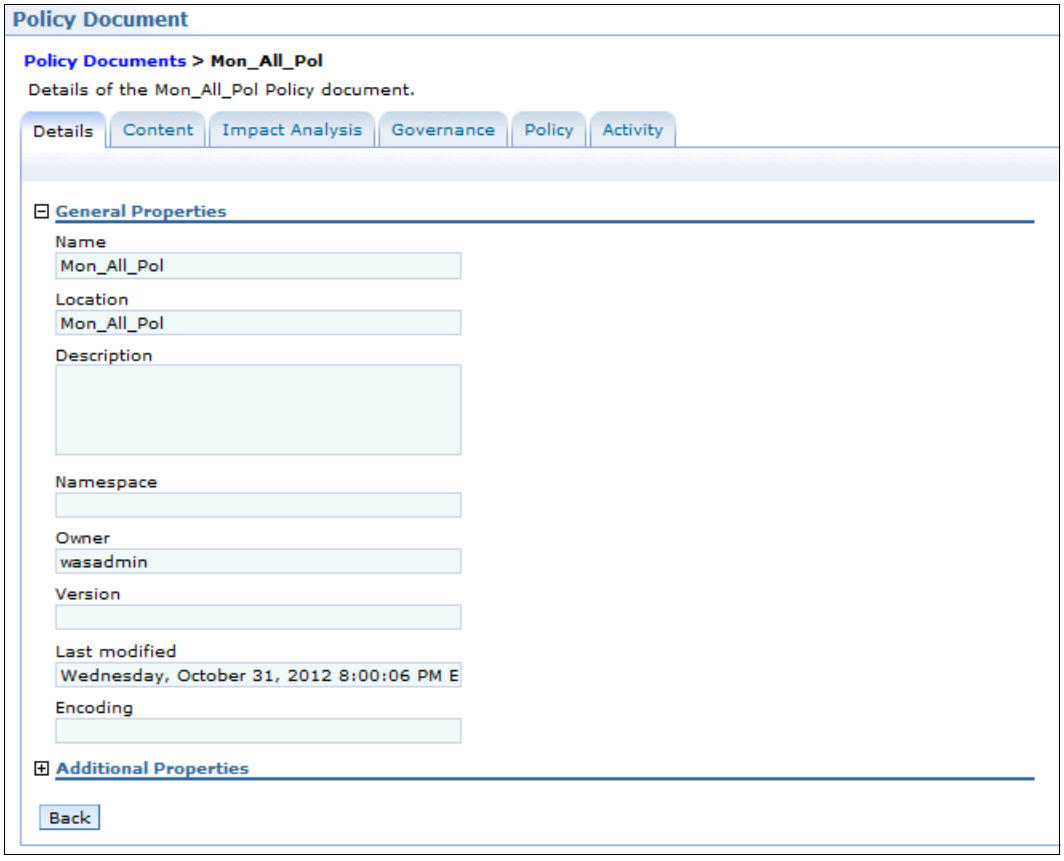


Figure 9-18 Policy after creation

2. After creating the policy, move it into the correct stage of the lifecycle so that you can attach it to an SLD and create situations.
3. Select the **Governance** tab. It displays the initial governance state as shown in Figure 9-19.

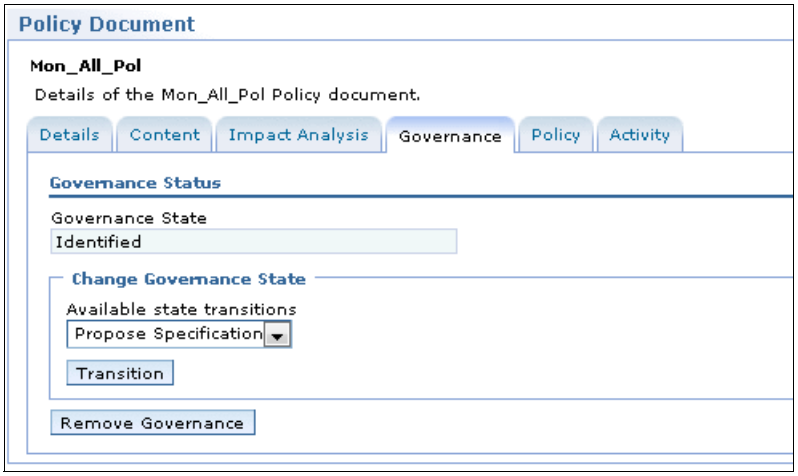


Figure 9-19 Initial governance state (GEP 8)

4. Transition the policy to the Approved state by clicking **Transition**, as shown in Figure 9-20. This state assumes that you are using a governance policy such as the GEP 8. If not, then the required state might be different (GEP 7 uses the Monitored state).

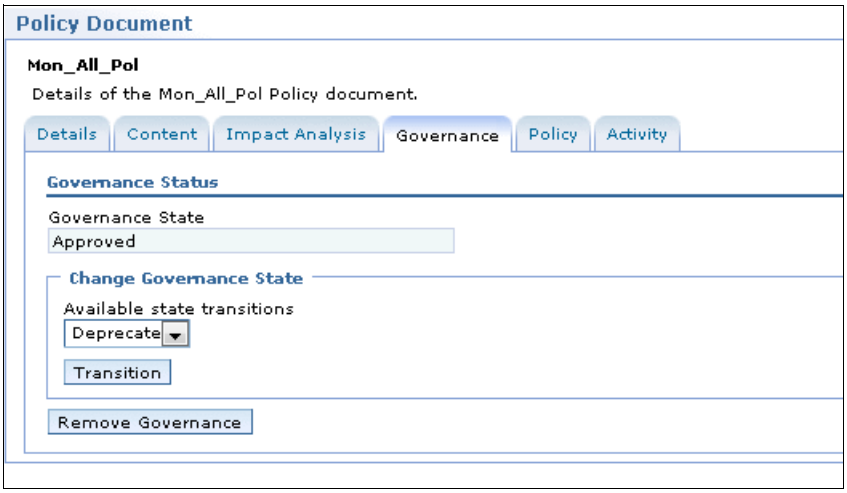


Figure 9-20 Final Governance state (GEP 8)

When the policy is attached to an SLD and promoted, a situation is created in ITCAM.

See Chapter 10, “Attaching a policy to a service” on page 313 for details about attaching and promotion in WSRR. See Chapter 14, “ITCAM as policy monitoring point” on page 389 for troubleshooting situation creation.

9.2.3 Modifying a Policy

Use the following steps to update or delete a policy:

1. From the SOA Governance perspective, click **View** → **Service Documents** → **Policy Documents**.
2. Select the policy you want to update and then click the **Policy** tab to view the policy details, as shown in Figure 9-21 on page 295.

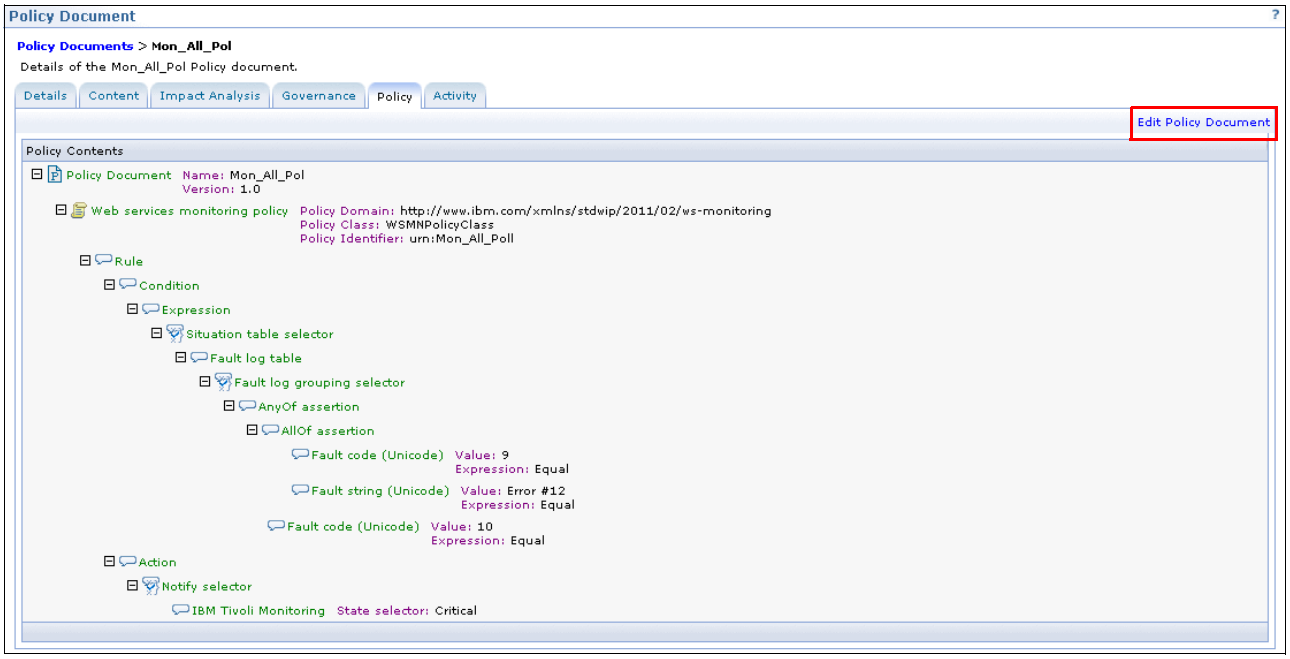


Figure 9-21 Policy details

3. If you want to return to the same display that you used to create the policy, select **Edit Policy Document** (Figure 9-22). You can now change, remove, or add any assertions or parameters.
4. Finish by clicking **Publish**. The policy will remain in the Approved governance state so there is no need to do any tasks beyond repromoting the SLD to which the policy is attached.

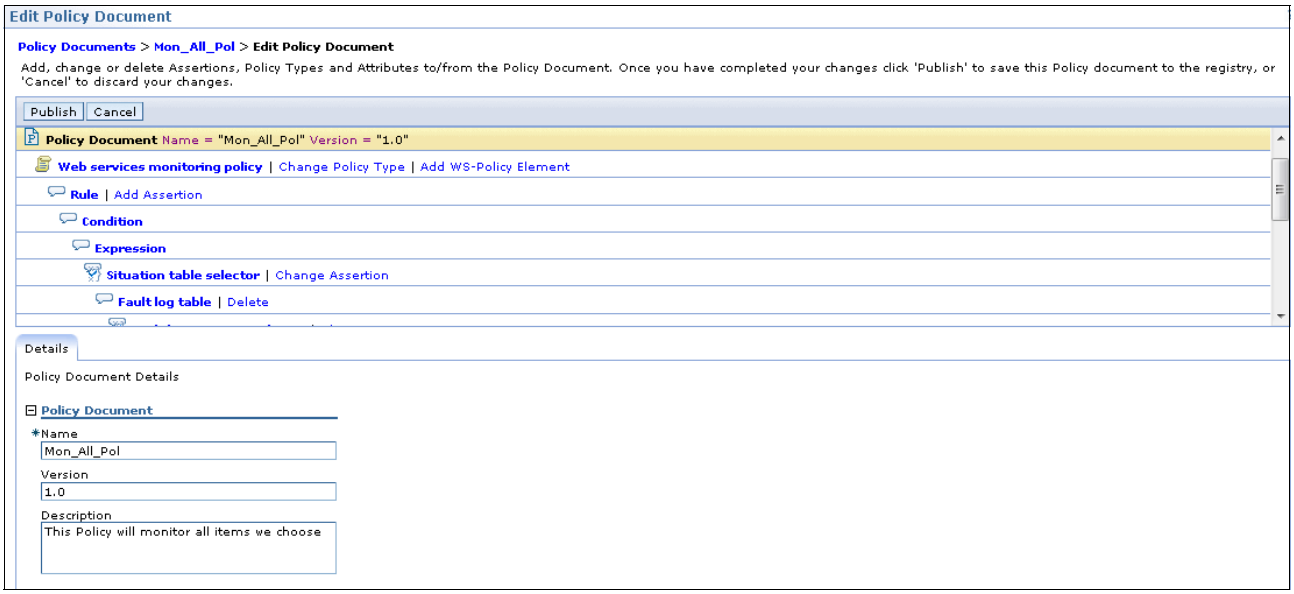


Figure 9-22 Edit policy document

Deleting a policy is done from the list of policies. When deleting a policy, all attachments to SLDs must be removed first, otherwise the deletion will fail.

9.3 When a threshold is crossed

The purpose of creating a monitoring policy is to respond when a threshold is crossed. Either the operator sees a new alert on the monitor and responds, or the system responds automatically.

9.3.1 Operator's monitor

Figure 9-23 on page 297 shows the main ITCAM window. The top left section has a hierarchical tree, which lists all agents that are installed in the environment. The top right section is a list of all current situations, for example, all policies that crossed their threshold.

The ITCAM main window includes all agents in the Tivoli Monitoring environment, both ITCAMforSOA and others. The window does not filter by agent, so events that are managed by WSRR and other events which were manually entered into Tivoli Monitoring (such as disk space, CPU load, database events, and so on) are displayed.

When the operator sees an event to investigate, the operation can select the relevant event line and right-click **Situation Event Results** to open a window with more information, as shown in Figure 9-23 on page 297.

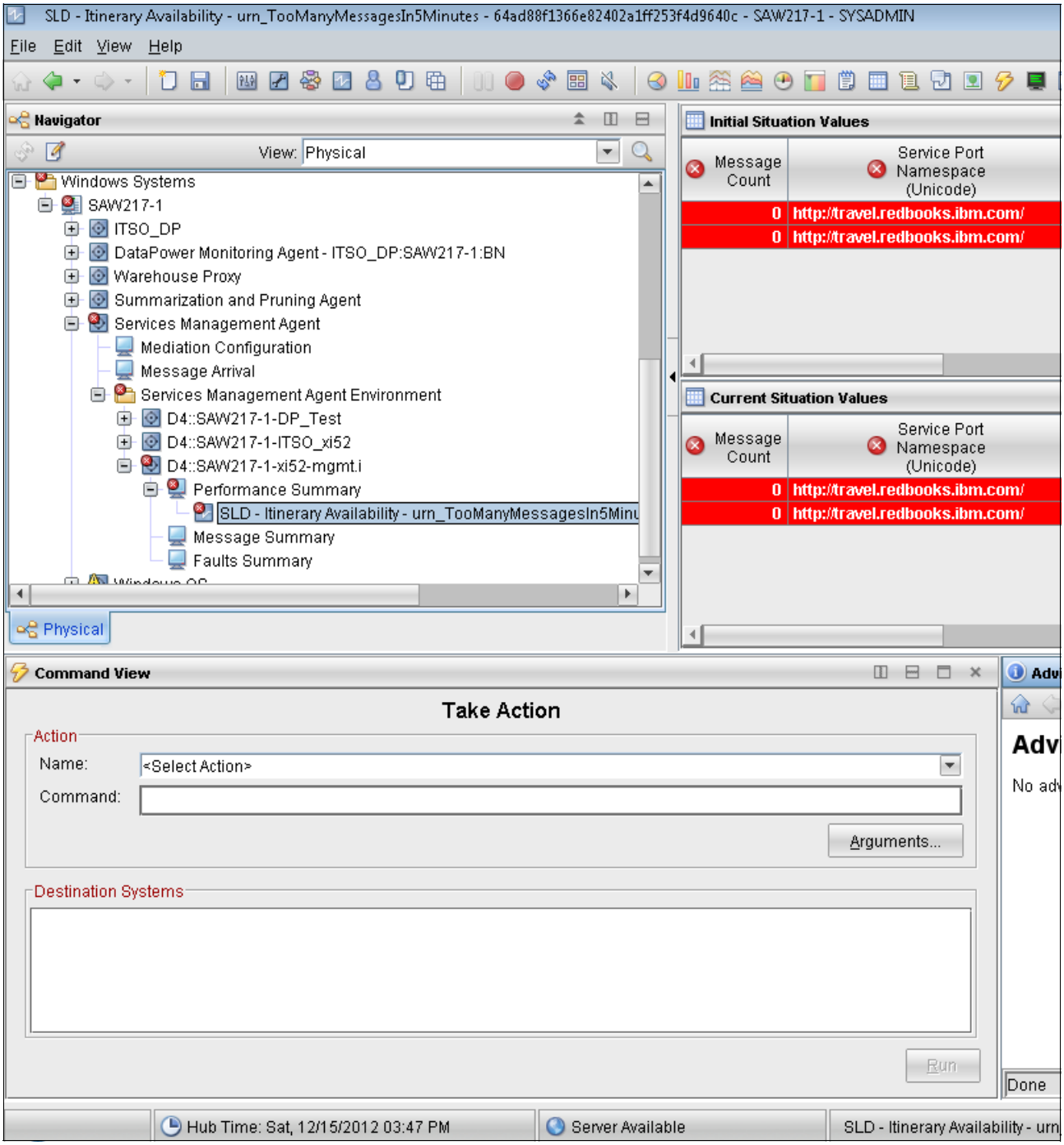


Figure 9-23 Situation event results

In the upper left corner is the tree of agents, opened at the specific agent which detected the event. If the agent has one data collector, the event will be linked to a subtree named `Service Management Agent Environment`. If an agent has a number of data collectors, each will be a named subtree under of the `Service Management Agent Environment` and the event will be linked to the relevant subtree.

The Initial Situation Values section contains the original measurement that caused the threshold violation. Under this section is the Current Situation Values section, which contains the values at this moment.

Consider the following information:

- The Current Value may be higher or lower than the Initial Value if the current measurement still exceeds (or is less than) the policy threshold.
- If the Current Value is empty, the current measurement does not violate the policy and the situation will close automatically.

The Expert Advice section contains information to help operators resolve the problem.

A section in the lower left of the window contains a list of actions that can be done to automate the problem solution.

The Tivoli Monitoring framework is capable of forwarding ITCAM events upstream to other Operations Support Software (OSS) applications which may display formatted messages or enrich the event data.

9.3.2 Automatic response by ITCAM

When creating a policy, you can select **Action** → **Notify Selector** → **IBM Tivoli Monitoring** → **System Command.** You can use this System Command action to define an automated response that activates when the policy threshold is passed.

Figure 9-24 shows a policy that runs the `/opt/IBM/fixProblem.sh` script when there are too many messages during the most recent monitoring period.

You may enter any command string in this field. WSRR performs no validation test and passes the text, as-is, to ITCAM.

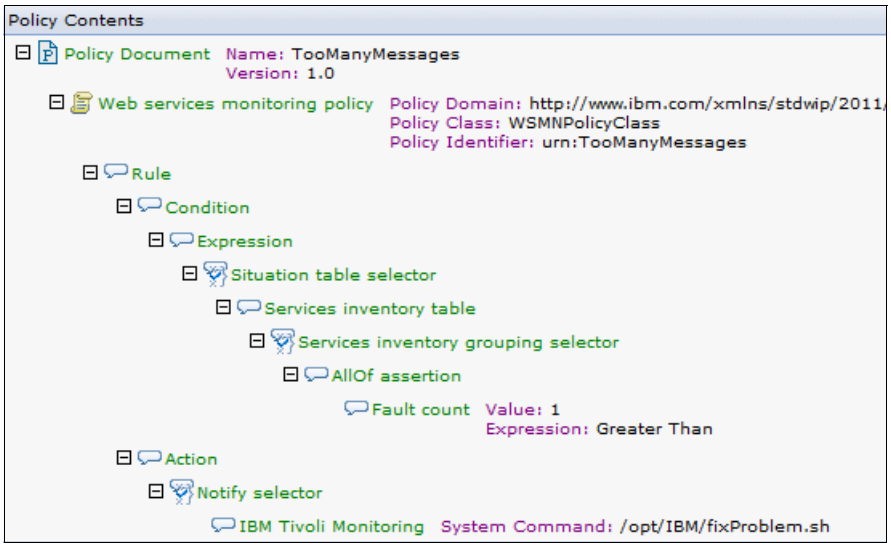


Figure 9-24 Situation automatic response

Be careful: When using operating system-specific scripts, if a policy is attached to two services of which one runs on Windows and the other runs on any type of UNIX, then the same System Command will run on both.

After attached properly to an SLD and promoted, the policy becomes an ITCAM situation.

Figure 9-25 shows that the resulting situation is open in the Situation Editor under the Action tab.

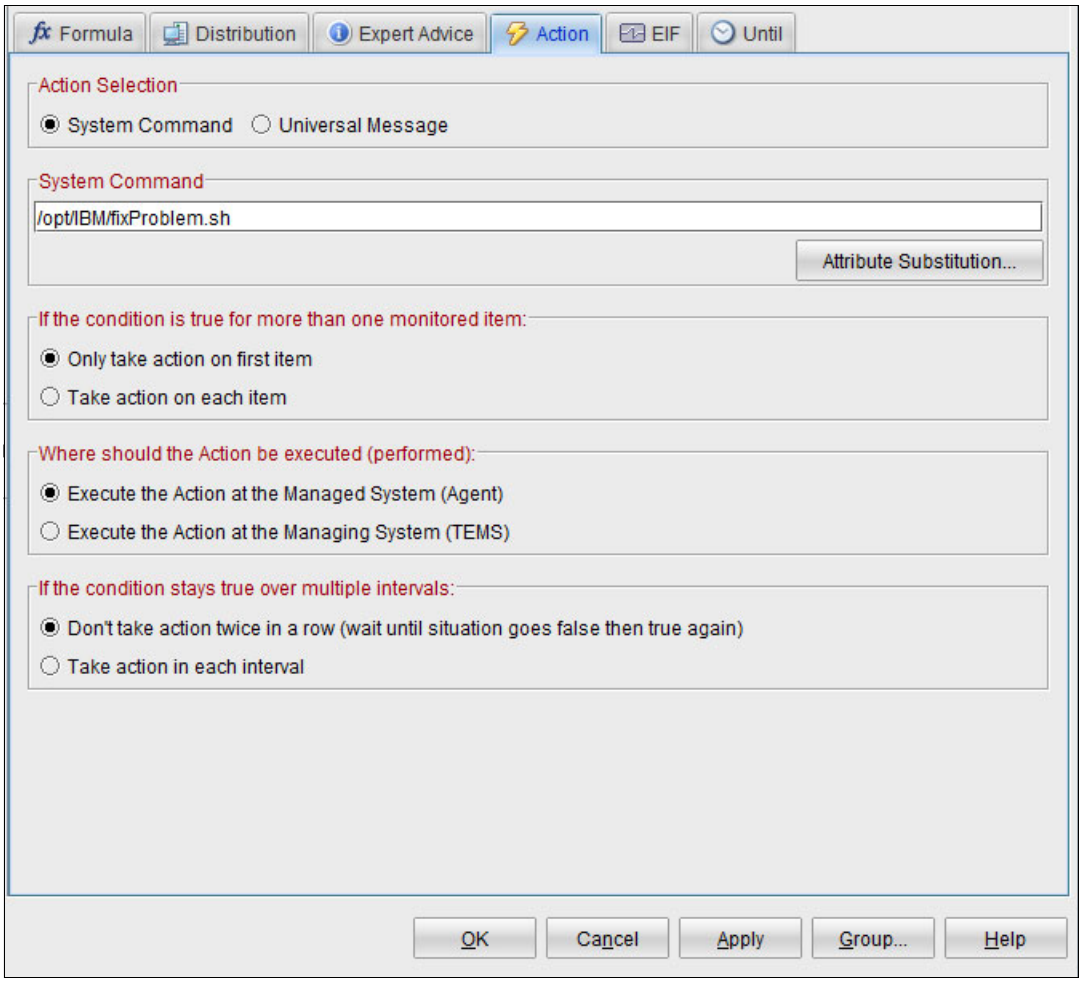


Figure 9-25 Situation action tab

The button options on this tab are defined in the `wsrr_sdms_config.xml` file. See Chapter 14, “ITCAM as policy monitoring point” on page 389 for further information about setting these values.

In the current example, when the situation fires, the `fixProblem` script will run. However, although ITCAM has further information (such as the name of the problematic endpoint, the number of messages, and so on), this information is not passed on to the script.

To pass information to the script, select **Attribute Substitution**. The window shown in Figure 9-26 opens.

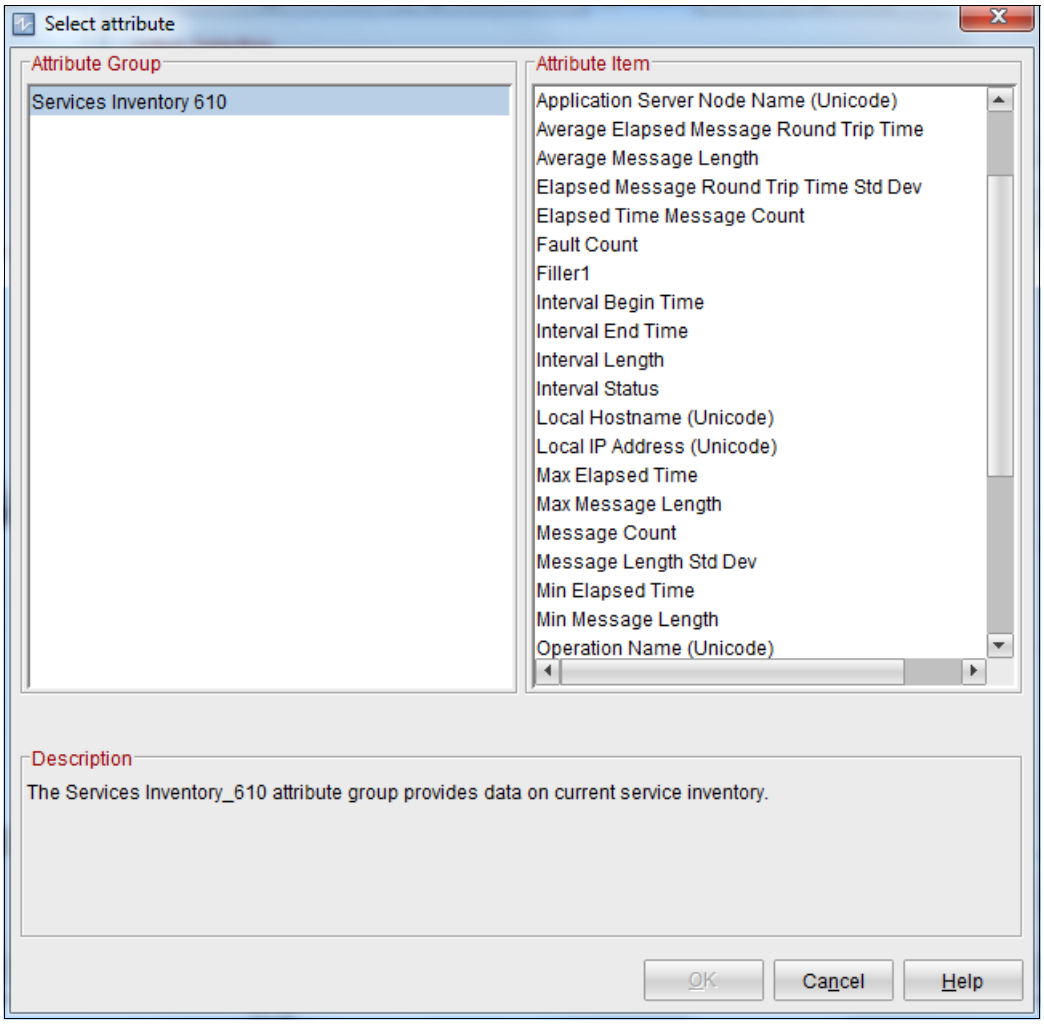


Figure 9-26 Situation Attribute window

In this window, you can insert placeholders for values that will be completed when the situation fires. Choosing an attribute inserts that attribute into the command line. For example, the command in Example 9-1 can be built.

Example 9-1 Situation action with attribute substitution

```
echo The operation &{Services_Inventory_610.Operation_Name_U} on
&{Services_Inventory_610.Service_Name_U} has too many messages
(&{Services_Inventory_610.Msg_Count} over the last 5 minutes) >> C:\ErrorLog.txt
```

This command results in the line that is shown in Example 9-2 being inserted into the `C:\Errorlog.txt` file every time the situation fired.

Example 9-2 Result of situation action

```
The operation available on ItineraryAvailabilityPort has too many messages (12
over the last 5 minutes)
```

You can create other actions by using script files that send email or run advanced commands to fix problems.

When using advanced scripts, consider whether to run the script:

- On the server with the ITCAM agent, which is the better option for running repair scripts locally on the runtime environment.
- On the ITCAM infrastructure server (Tivoli Enterprise Monitoring Server), which is the better option for running alerting scripts.

A further example is presented in Chapter 14, “ITCAM as policy monitoring point” on page 389.

9.3.3 Automatic response to WSRR (eventing)

As configured during the setup on the ITCAM-WSRR integration, ITCAM can send updates to WSRR when ITCAM detects a policy threshold being violated.

Although this behavior appears to be a closed loop (WSRR defines, ITCAM monitors, WSRR receives the response), it is actually two links (WSRR → ITCAM and ITCAM → WSRR) without any dependency between them.

The basic flow is as follows:

1. When the conditions of a policy resolve to true ITCAM generates an event. The event fired is an instance of an ITCAM situation.
2. ITCAM sends the event to WSRR and WSRR updates metadata in the registry based on the mapping defined.
3. WSRR takes an action (create, update, or delete) to the property on any of these service objects such as ServiceEndpoint

   Necessary configuration: WSRR event handler must be configured to process the ITCAM events.

4. ITCAM sends an event notification when a situation is cleared or triggered again.

From an ITCAM perspective, whether a situation updates WSRR depends only on whether the situation is configured to emit an Event Integration Facility (EIF) event. The EIF is a protocol that is used by Tivoli monitoring products to communicate and send events between them. The standard implementation is for ITCAM to send an event to a central Omnibus Manager-of-Managers. The Omnibus collects events from multiple sources and preforms advanced correlations on them as opposed to the ITCAM which collects performance data and uses situations to decide whether a threshold was crossed.

When a situation fires, ITCAM can display the event in its own windows or forward the event upstream, either to Omnibus, WSRR, or other IBM products.

Within the Situation Editor, the EIF tab controls the event's destination. Figure 9-27 shows a situation that is configured to send an event to WSRR.

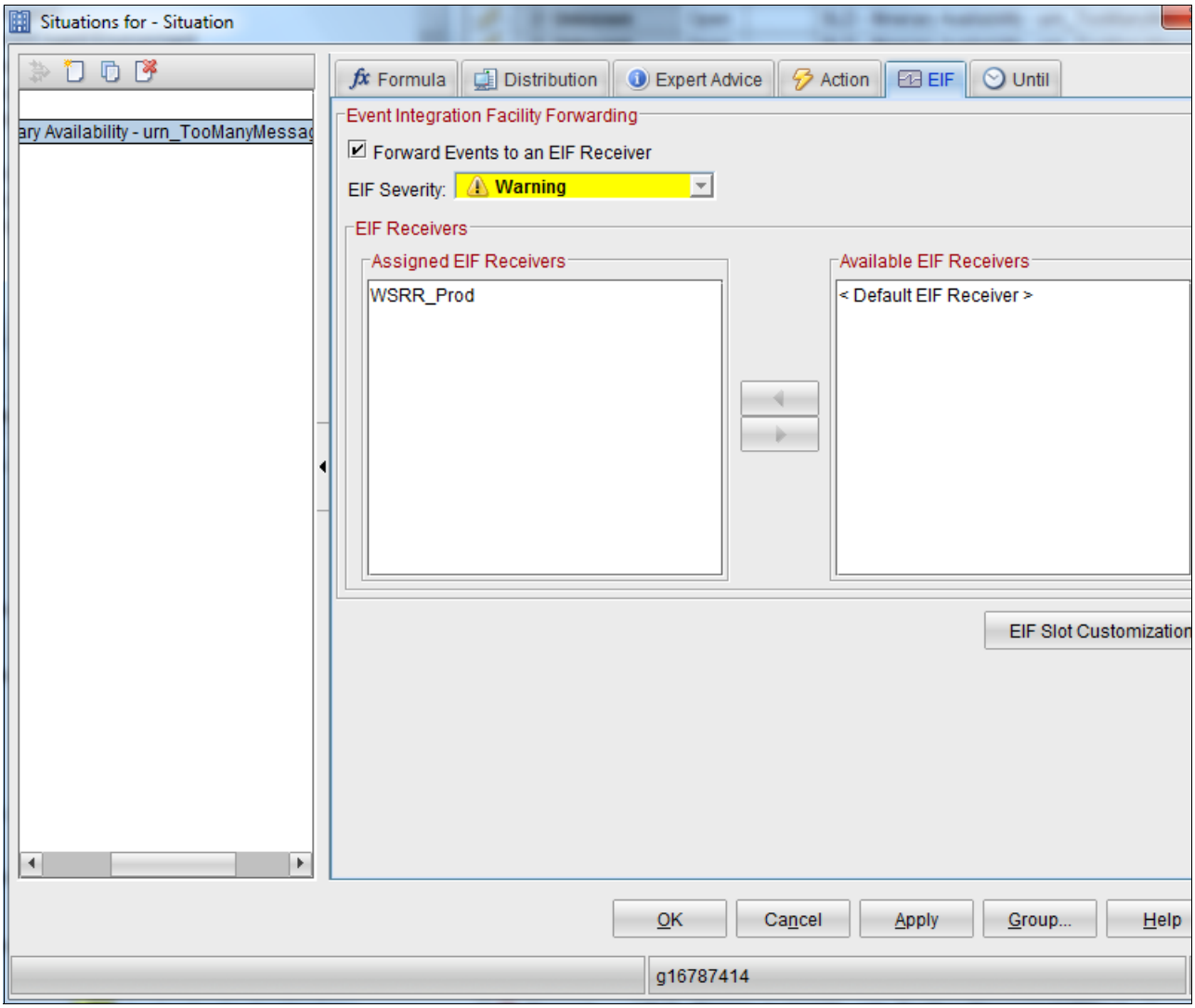


Figure 9-27 Event Integration Facility

From a WSRR perspective, whether a situation updates WSRR depends only on whether the ITCAM listener is configured to catch that specific situation.

Log in to WSRR Runtime Service Registry:

`https://sa-w217rhel-2.itso.ral.ibm.com:9443/ServiceRegistry`

Navigate to **Administrator Perspective** → **Home** → **Service Documents** → **Policy Documents** → **Choose a policy (**`SLD - Itinerary Availability_GenericObject_SLD - Itinerary Availability_urn:ClientFaulted.xml`**)** → **Policy Attachment**.

Figure 9-28 shows the `itmSituationIdentifier` property sent by ITCAM to WSRR. This property must go in the Event ID column for any new mapppings that are defined in Studio.

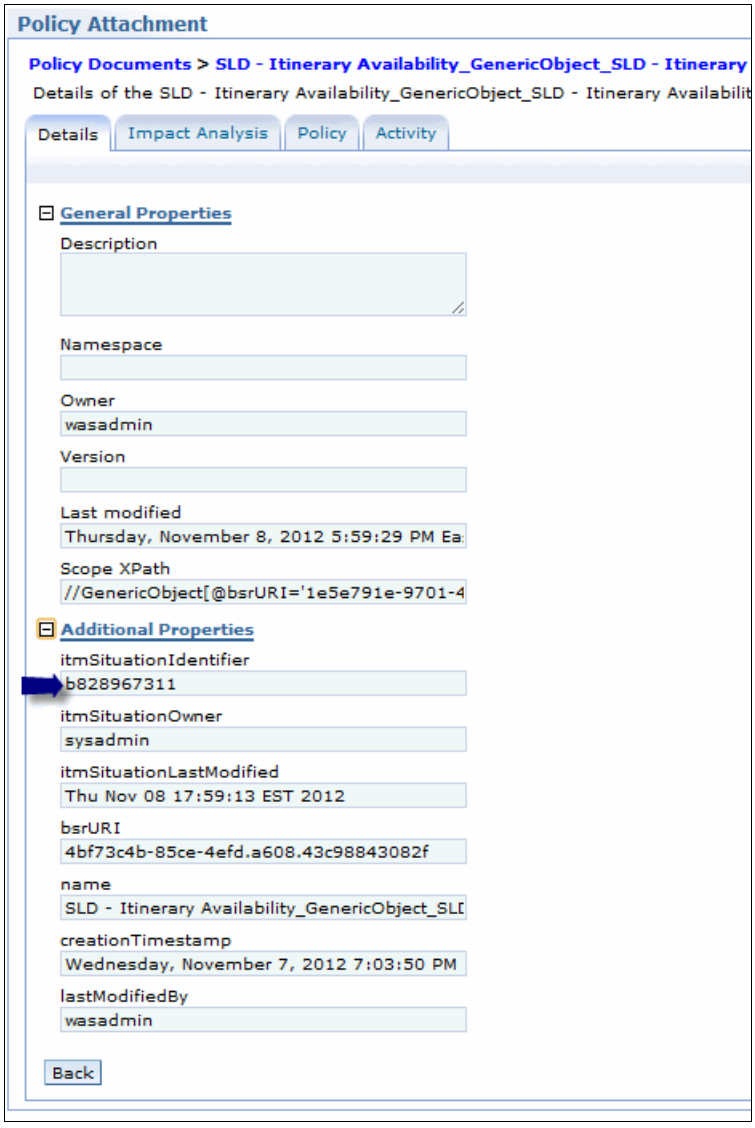


Figure 9-28 ITCAM Situation Identifier in WSRR

WSRR studio has ITCAM for SOA Integration Configuration Editor tool which can be used to manage WSRR ITCAM Event Handler configuration in an easier way. You can also edit the configuration manually in WSRR.

Launch **WSRR Studio** → **ITCAM for SOA Integration Configuration Editor** as shown in Figure 9-29.

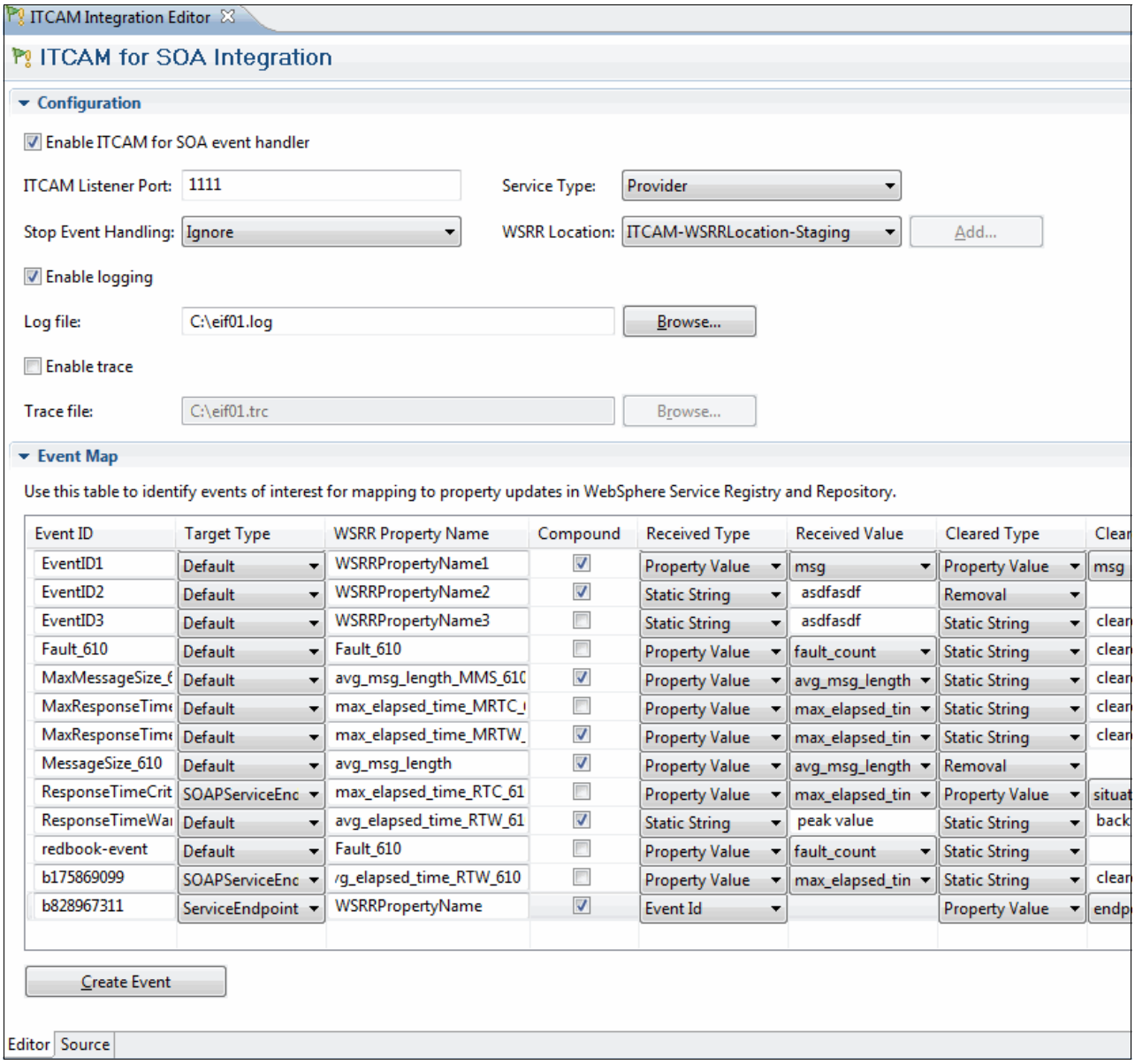


Figure 9-29 WSRR Studio Tool ITCAM for SOA Integration Configuration Editor

Event mappings can be created or updated with this utility and can be published to WSRR. The changes are reflected in the ITCAMEventHandlerConfiguration item in WSRR as shown in Figure 9-30 after synchronization occurs.

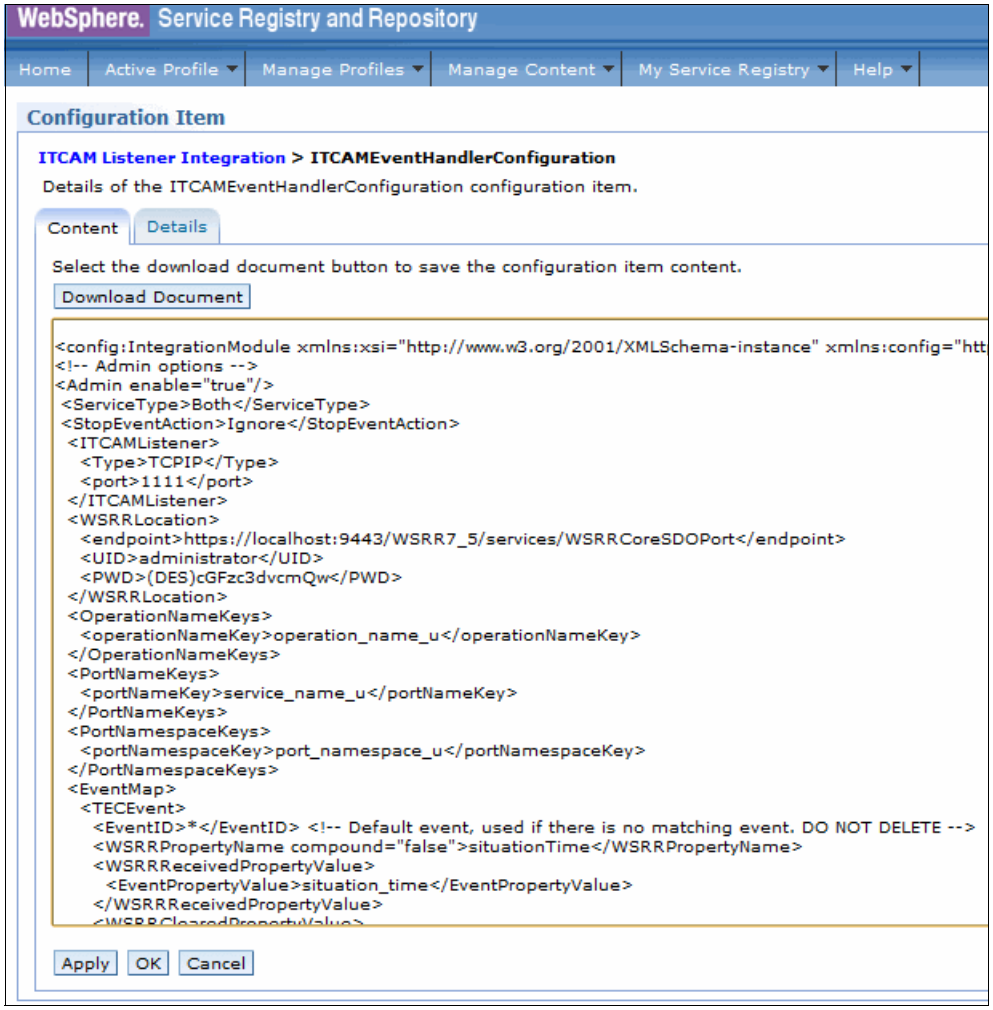


Figure 9-30 ITCAM Event Handler Configuration in WSRR

Log in to **WSRR Runtime Service Registry** → **Configuration Perspective** → **Active Profile** → **ITCAM Listener Integration** to view the ITCAMEventHandlerConfiguration item.

Find the eventID of a situation in two ways:

- From within ITCAM by viewing the situation in the Situation Editor and looking at the lower right of the frame, as shown in Figure 9-31.

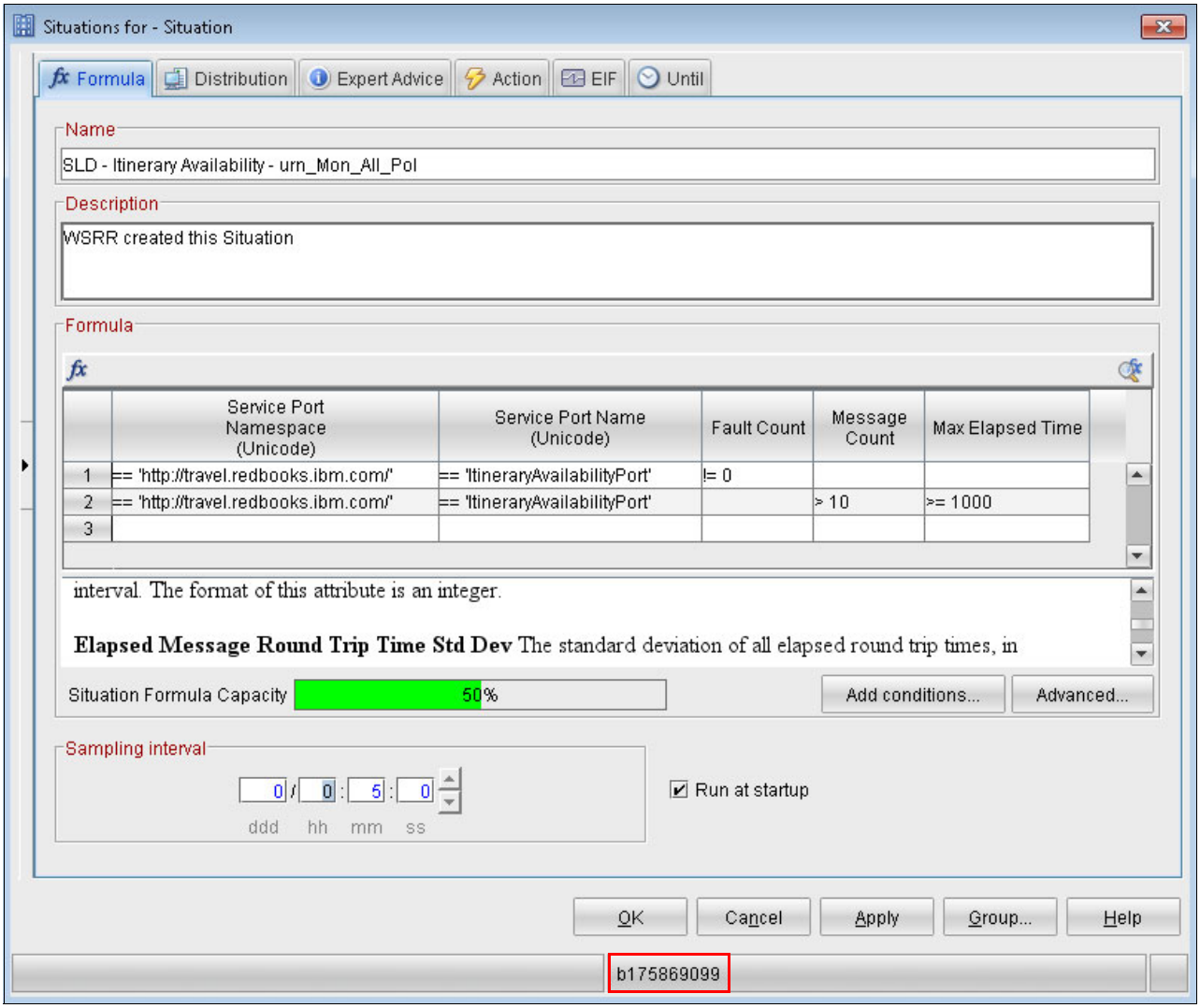


Figure 9-31 Situation Id in ITCAM

- From within WSRR by viewing the Additional Property named tmSituationIdentifier within the Policy Attachment of the Policy Document, as shown in Figure 9-32.

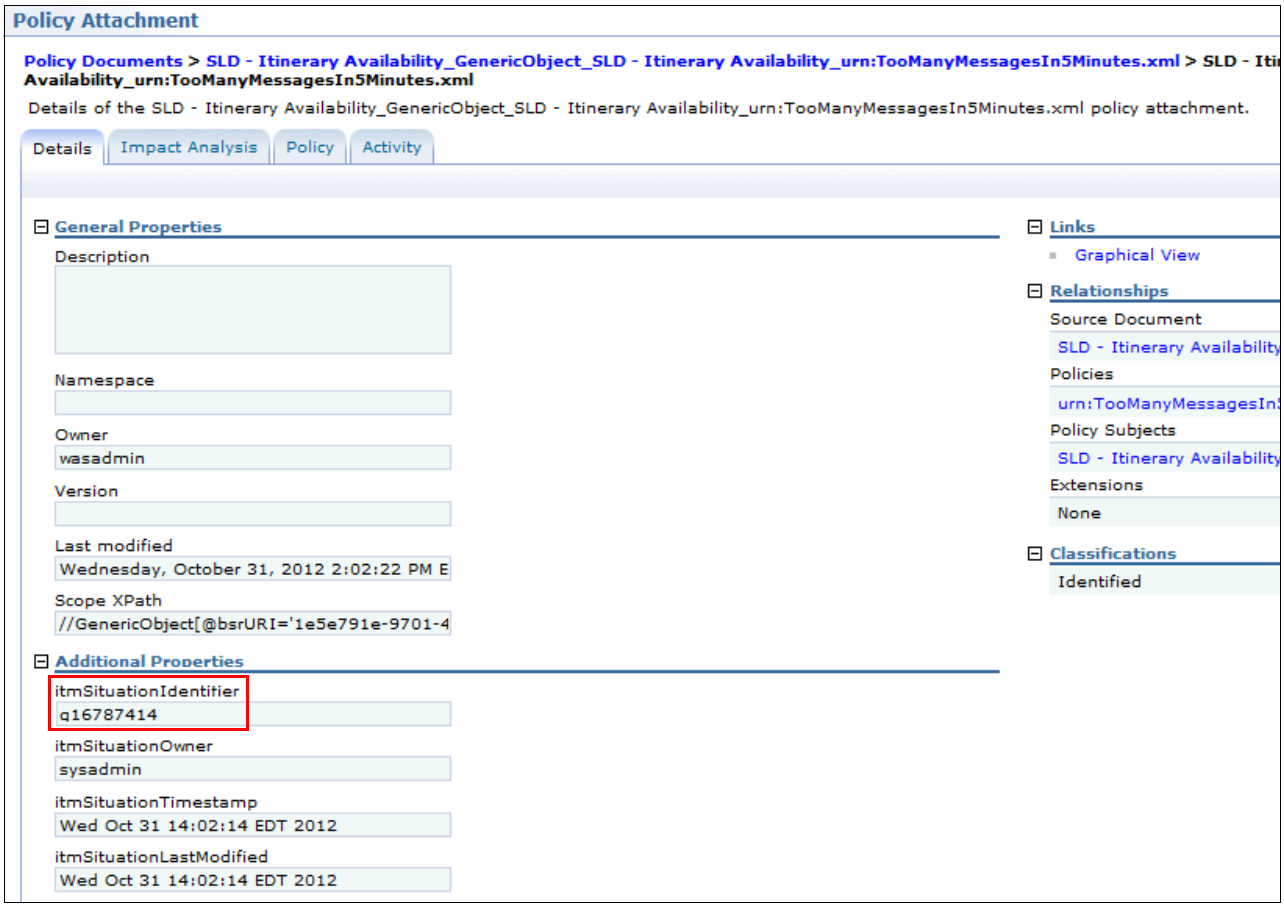


Figure 9-32 Situation ID in ITCAM

After the situation reaches the WSRR, it modifies metadata in accordance with the event data and the configuration of the listener.

The ITCAM event handler in WSRR uses the service port name and service port namespace event properties to identify the WSRR target object to update. If the event handler finds multiple matching target-type objects, it compares the application server host name and port from the event to the host name and port in the property of the target object. If it finds a match, it updates the metadata of the corresponding target object. If there are multiple matches, the event handler does not process the event further.

Further details are in the information center:

http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/cwsr_updatingwsrrmetadatawithitcamforsoaevents.html

Figure 9-33 shows the metadata update in WSRR.

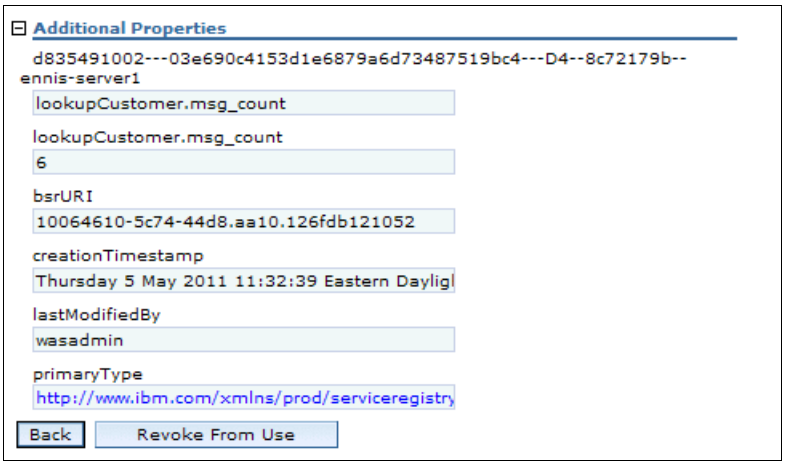


Figure 9-33 Metadata updated by ITCAM situation

For further details, see the article “Integration between IBM Tivoli Composite Application Manager for SOA and WebSphere Service Registry and Repository” at the developerWorks website:

http://www.ibm.com/developerworks/websphere/library/techarticles/0702_badlaney/0702_badlaney.html

Article is for older versions: The article is for older versions of WSRR and ITCAM. However, use the setup configuration described in this section.

9.4 Advanced monitoring with ITCAM

Many monitoring options are available to ITCAM that are not exposed through the WSRR menu. For example, you can set a threshold as a function, such as *10% above the weekly average*, and not as a simple condition, such as *larger than 10*. Another possibility is searching the Fault Log for a substring.

The only way to define these advanced situations is directly with ITCAM.

Tip: A full explanation of all ITCAM options is beyond the scope of this book. An assumption is that anyone who interested in using the full capabilities of the ITCAM and Tivoli Monitoring stack meets the following requirements:

- Has prior knowledge of Tivoli Monitoring and ITCAM functions
- Has access to Tivoli Monitoring and ITCAM specialists
- Can learn about these capabilities from documentation other than this book.

Start with the basic information center for the products:

http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3fp1/adminuse/situation_create_tep.htm

Then, continue with the following sites:

- https://www.ibm.com/developerworks/servicemanagement/
- https://www.ibm.com/developerworks/mydeveloperworks/groups/service/html/communityview?communityUuid=0587adbc-8477-431f-8c68-9226adea11ed

9.4.1 Creating situations without WSRR

Although the creation of policies in WSRR and propagating them as situations into ITCAM creates a closed loop environment, which has the benefit of WSRR behaving as a central portal for all policies, in certain cases, creating the situation directly in ITCAM makes sense.

ITCAM has several advanced functions that are not exposed through the WSRR interface. The functions are available for use from the ITCAM interface.

After the situation is created, the ITCAM listener configuration can be updated manually in WSRR to receive the event when the policy threshold is passed.

9.4.2 Creating situations in ITCAM

Full details of how to create situations in ITCAM are in the Tivoli Monitoring information center. Tivoli Monitoring 6.2.3.1 documentation is available at the following website:

http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3fp1/adminuse/situation_create_tep.htm

The simplest way to create a situation that WSRR will be able to accept is to take an existing situation that is created by WSRR, duplicate it, and change the necessary parameters.

For example, to create a situation that searches the Fault Log table for Fault Messages, which contain the string `ERR`, you can create a policy that searches the Fault Log table for Fault Messages, which are the exact string `ERR`.

The advantage of creating situations in this way is that items such as Service Port Namespace and Service Port Name will be filled in automatically, based on the endpoint of the SLD, to which the policy is attached.

After the situation is created, view it in the Situation Editor and right-click **Create Another** to create a new situation.

The original situation is named according to the naming convention of `SLD - policy`. The new situation can have any name, but the best approach is to stay within a regular naming

convention. The situation description can also be changed to reflect that it was created manually. Figure 9-34 shows a manually created situation.

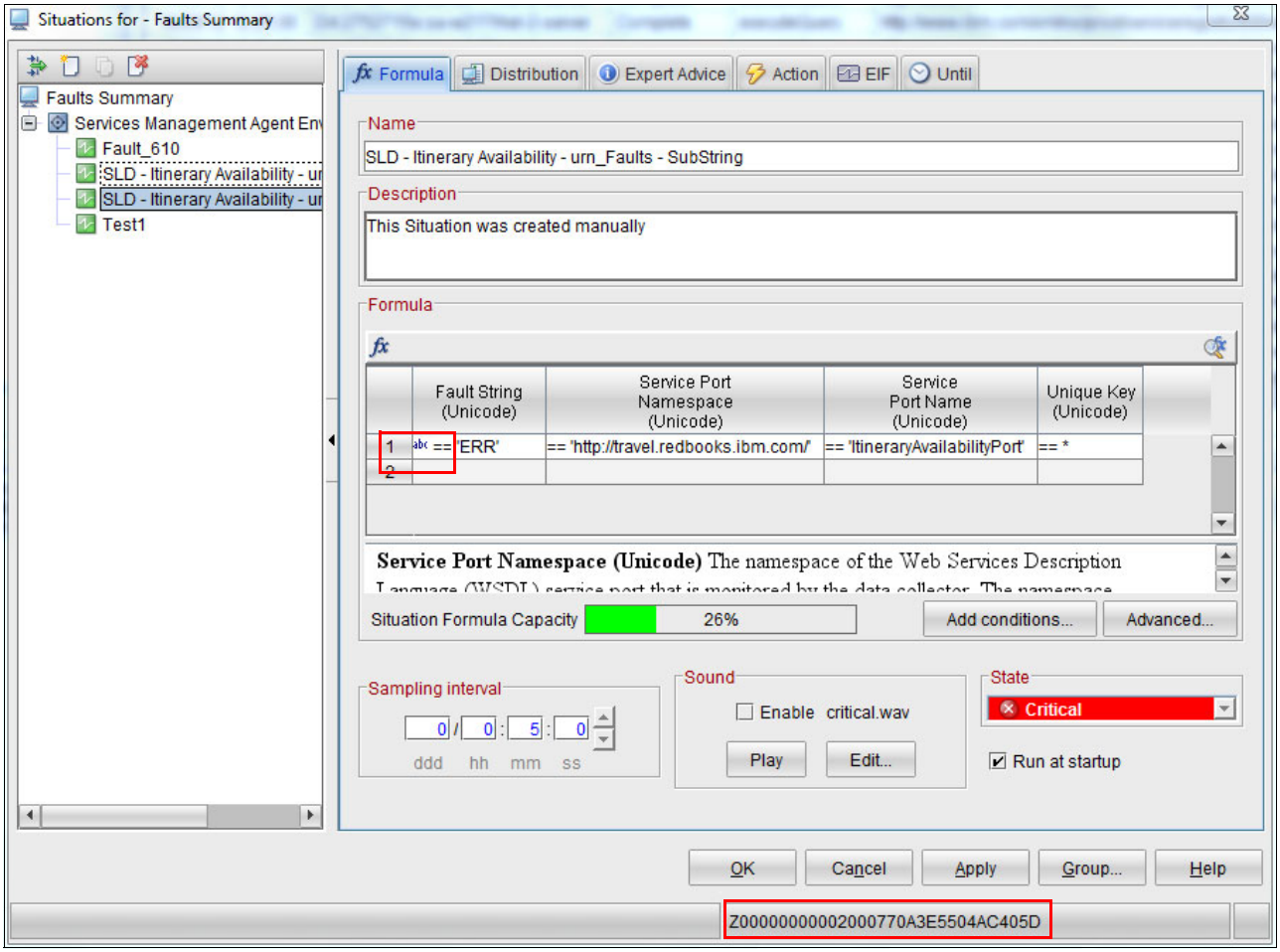


Figure 9-34 Situation created within ITCAM

Note the string search function that is used in the Fault String column.

The situation identifier is shown in the lower right. This value must be inserted into the EventID field of the ITCAM listener in WSRR in order to receive the event.

Because the situation duplicates the situation created by WSRR, there is no need to use the Distribution and EIF tabs in the Situation Editor. Depending on the specific details of the situation, the Expert Advice and Action tabs may be changed.

This situation will never be changed or deleted by WSRR. It exists outside of the lifecycle displayed in WSRR, despite the fact that WSRR is capable of receiving events when the situation is triggered.

9.4.3 Advanced situation functions

The following examples are of advanced situation functions that exist in ITCAM. These functions are common to all Tivoli Monitoring systems and subsystems and therefore do not usually have specific documentation for ITCAM for SOA. As stated previously, situations are essentially SQL statements that are run on tables that are generated by the ITCAM. The

Tivoli Monitoring infrastructure treats all tables equally, whether they are created by the base Tivoli Monitoring agents, ITCAM for SOA, or any other ITCAM agents.

String functions

Although the WSRR creates policies that compare exact strings, ITCAM can perform substring searches and regular expression matches, as shown at the following website:

http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3fp1/adminuse/formulafunction_regex_tep.htm

Periodic functions

ITCAM can modify expression values during specified periods of time to accommodate varying service level objectives. For example, you might apply change thresholds during weekends and holidays. This function is called *Dynamic Situation Thresholding* and is detailed at the following websites:

- http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3fp1/adminuse/situation_overrideintro_tep.htm
- http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3fp1/adminuse/dlg_sited_overridescheduleoptions.htm

Historical functions

ITCAM can be connected to a data warehouse that stores the performance data that is collected by the agents for long periods. Beyond the ability to analyze long-term behavior by using the Tivoli Monitoring portal windows and generating reports, this information can be used to create advanced situations. These situations can set a threshold such as *slow service* based on deviation from the average response time for that system. For further details, see the following website:

http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3fp1/adminuse/dlg_sited_overridemodel.htm

Predictive functions

One of the components in the IBM Tivoli Monitoring product is the Performance Analyzer. This component adds predictive capability to IBM Tivoli Monitoring. This capability helps operations and IT staff to understand resource consumption trends, identify problems, resolve problems quickly, and predict and avoid future problems. For example, you can use Tivoli Performance Analyzer, which is fully automated, to predict application bottlenecks and create alerts for potential service threats.

The Performance Analyzer models future behavior based on past behavior. If a service is gradually slowing, the Performance Analyzer can predict that it will pass a threshold in so many days in the future.

For further details about using the Performance Analyzer, see the user's guide:

http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3fp1/itpa/c_users_predmon.htm

10

Attaching a policy to a service

This chapter serves as a reference that summarizes and describes how to attach a policy to a service. It shows the specific options on where in IBM WebSphere Service Registry and Repository (WSRR) a policy can be attached, and the methods of editing the attachments.

This chapter contains the following topics:

- 10.1, “Attaching a policy to a service” on page 314
- 10.2, “Viewing and editing a policy attachment by using WSRR” on page 314
- 10.3, “Attaching a policy to a provider service using an SLD” on page 325
- 10.4, “Attaching policy to consumer-provider pair by using SLA” on page 327
- 10.5, “Specifying policies from unknown consumers” on page 330

© Copyright IBM Corp. 2013. All rights reserved.

10.1 Attaching a policy to a service

A policy, whether it is a mediation policy, a monitoring policy, or a custom policy is of little use by itself. A policy is most useful when attached to something, implying that the rules of the policy should be applied to the object to which the policy is attached. In WSRR, a policy may be attached to any object. However for the attachment of a mediation policy to mean something to DataPower or ITCAM for SOA, it must be attached to either a service level agreement (SLA) or a service level definition (SLD).

10.2 Viewing and editing a policy attachment by using WSRR

WSRR 8.0 updated its business space widgets to allow the attachment of policies to services in context. There are two ways that a policy can be attached to an object in WSRR:

- From the policy itself
- From a specific object

Each method is looked at separately, starting with attaching from the policy.

10.2.1 Dynamic attachment of a policy to a set of services

To attach a policy from the policy expression, you must first navigate to it. Follow the steps described in 8.3, “Viewing a mediation policy” on page 273 to view the policy.

Use the following steps to attach a policy:

1. After the policy is displayed, click the **Actions** drop-down menu in the top left corner of the detail widget for the Policy Expression, and then select **Manage Policy Attachments**. The Manage Policy Attachments dialog window opens (Figure 10-1 on page 315). Use this dialog to attach the policy to various objects within WSRR.

   The Manage Policy Attachments window shows any objects to which a policy is currently attached. As shown in Figure 10-1 on page 315. there are no objects to which this policy is currently attached, so the option that the dialog shows is to attach the policy to specific items and another that uses a query to attach to multiple items.

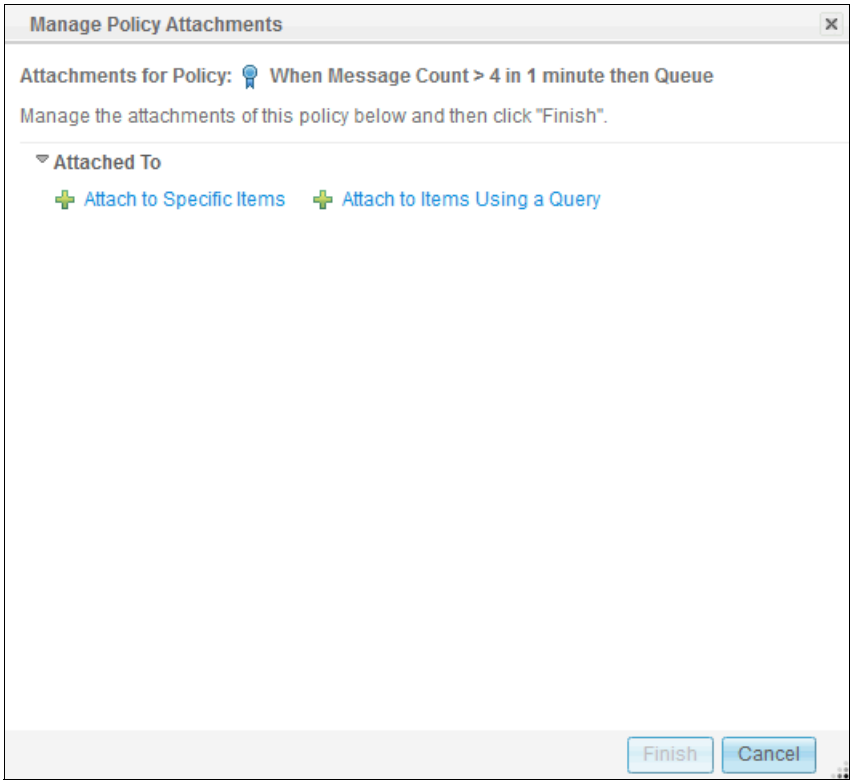


Figure 10-1 Manage Policy attachments dialog

2. Select **Attach to Specific Items**. A field is then available (Figure 10-2) where you select the specific item to be attached.

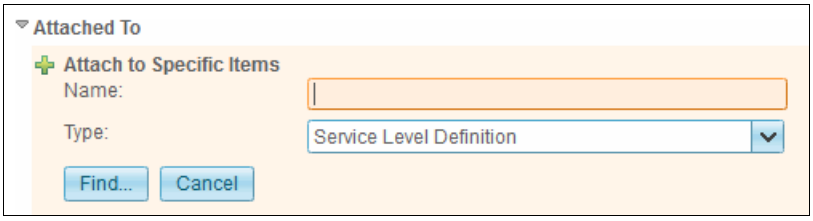


Figure 10-2 Attach to Specific Items dialog

You can select the type of object within WSRR that you are looking to attach the policy to. You can also type part of the name of the object (including wild cards) to filter the search.

If the WSRR auto-suggest capability is enabled, suggestions that match the search string are listed and can be selected, as shown in Figure 10-3.

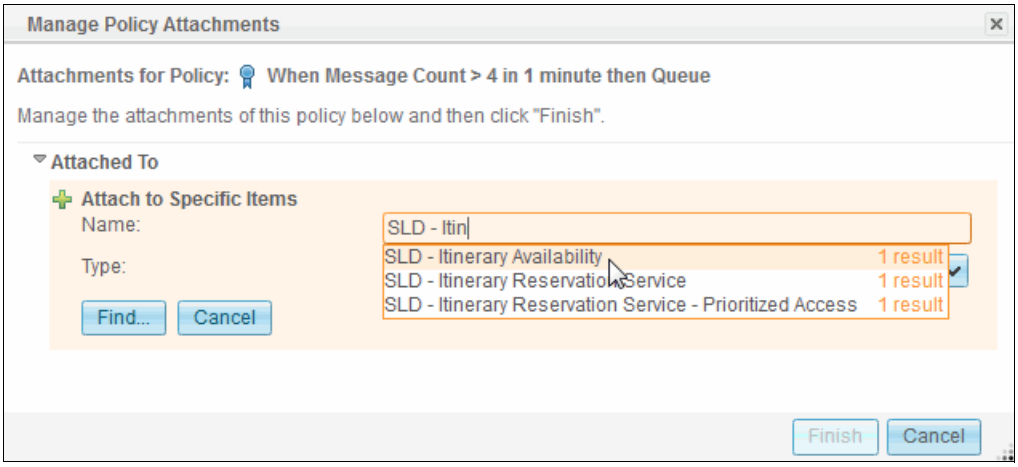


Figure 10-3 Attach to Specific Items dialog with auto suggestions

3. If you want to attach to more than one object of the specified type, click **Find**. A dialog opens to show all matching objects (Figure 10-4). You can choose multiple items by selecting the check box for each item and then clicking **Finish**.

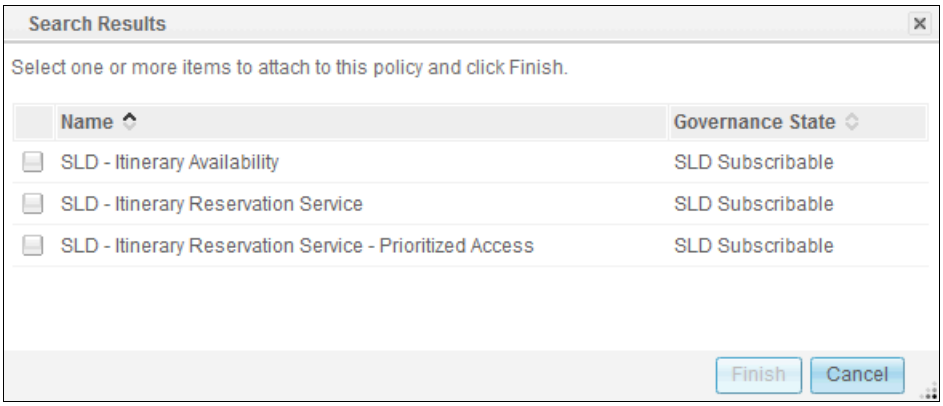


Figure 10-4 Attach to Specific Items Search Results dialog

4. Select **Attach to Items Using a Query** from the Manage Policy Attachments dialog window (Figure 10-1 on page 315) to open the dialog that is used to create the query (Figure 10-5).

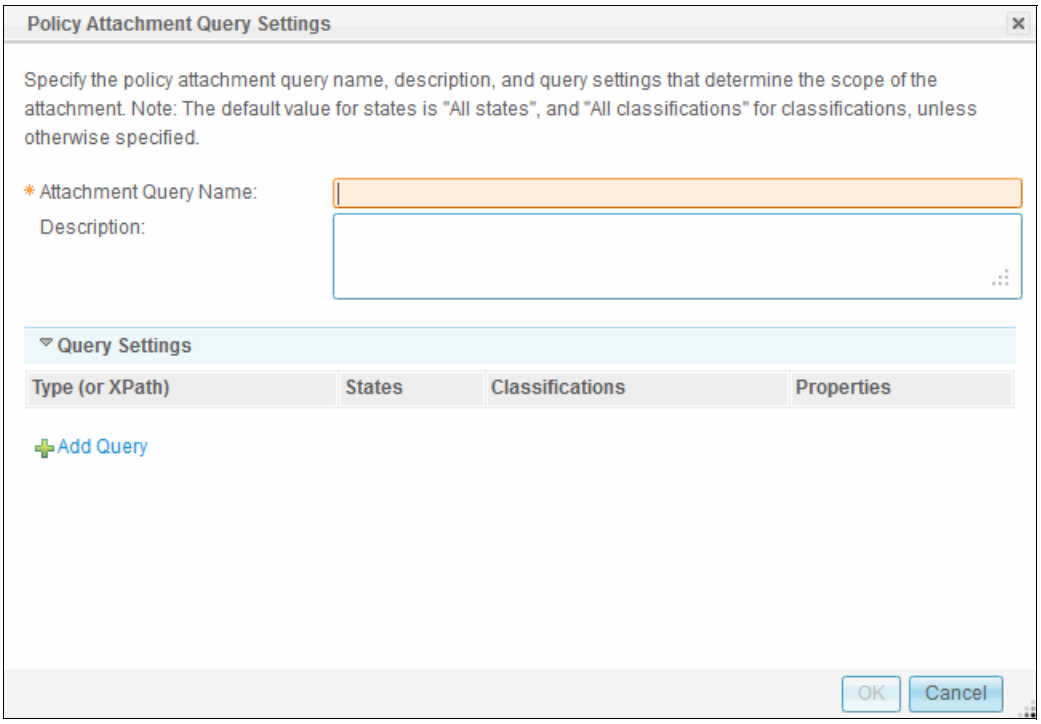


Figure 10-5 Policy Attachment Query Settings dialog

5. Enter a name and optional description for the attachment query, and then click **Add Query** to define the query, as shown in Figure 10-6. The name and description do not affect the query in any way; they are purely for reference.

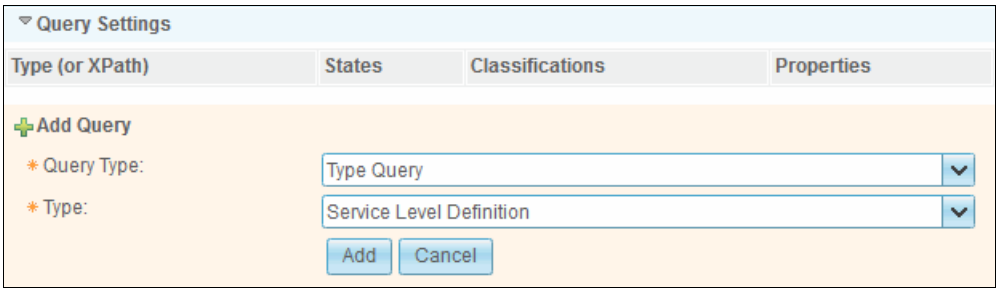


Figure 10-6 Policy Query Settings dialog

6. For the Query Type, the following options are available:
   - **Type Query** uses a simple wizard to help you generate the query.
   - **Custom Query** requires you to type a complete XPath expression as the query.

Tip: Use Custom Query only when you cannot get the required results from the Type Query wizard. If you use Custom Query, you need experience with XPath, and WSRR's system model and GEP models. The following section have examples of using XPATH:

- 3.5.2, “Attaching the policy to all services matching a name” on page 80
- 3.5.3, “Attaching policy to all services for an organization” on page 82

Use **Type Query** (the default) as the Query Type

The Type drop-down list shows object types that a policy can be attached to in WSRR. Select a type and click **Add**. The panel in Figure 10-7 opens.

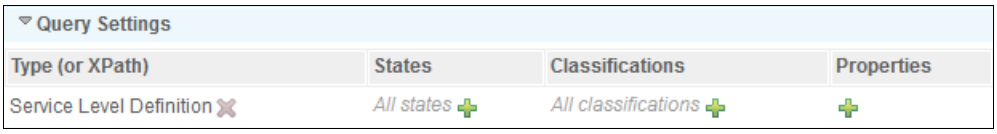


Figure 10-7 Policy Attachment Query Setting dialog with query specified

7. After the initial Query is added, you can include additional information to filter the items that the query will return. The additional information that you can filter on are Lifecycle states, Classifications, and Properties.

   To add one of these filter types, click the plus icon (**+**) next to the type you want to add. A window opens; it shows options that can be selected.

   Figure 10-8, Figure 10-9, and Figure 10-10 on page 319 show windows that open when various options are selected.

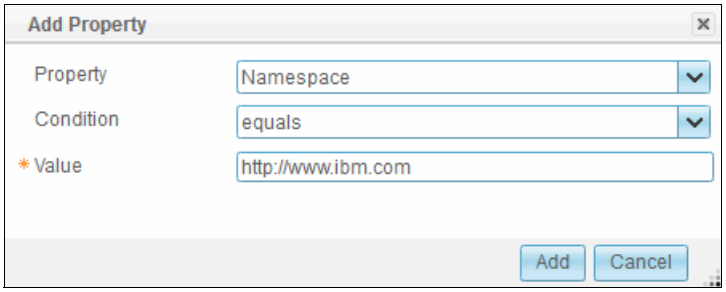


Figure 10-8 Add Property pop-up

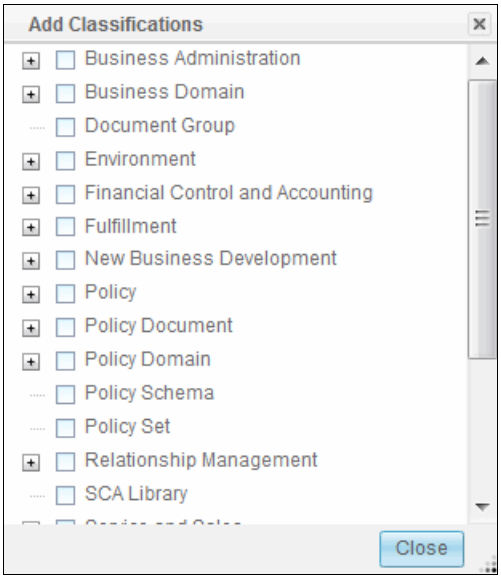


Figure 10-9 Add Classifications pop-up

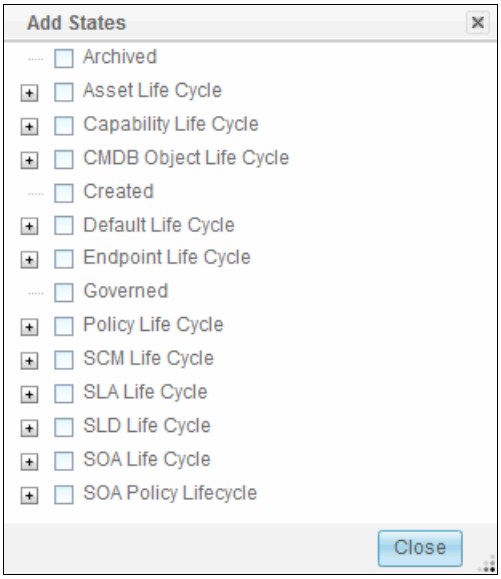


Figure 10-10 Add States pop-up

8. After the relevant additional filter information is added, the window shown in Figure 10-11 opens. In this example, the window shows the query is for a service level definition, with a lifecycle state of SLD Subscribable, a classification of Staging, and a Namespace property with the value `http://ww.ibm.com`.

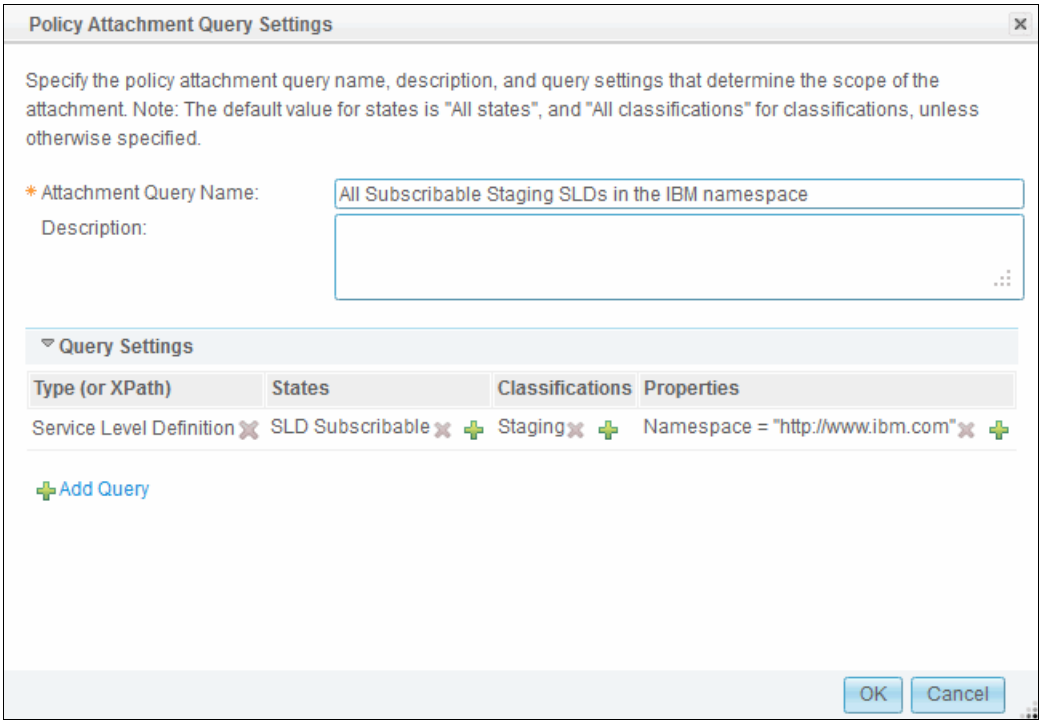


Figure 10-11 Policy Attachment Query Setting with completed query

9. You can add more queries by selecting the **Add Query** link again.

   After all queries are added, click **OK**. The Manage Policy Attachments dialog now shows the attachment query that was just created, as shown in Figure 10-12.

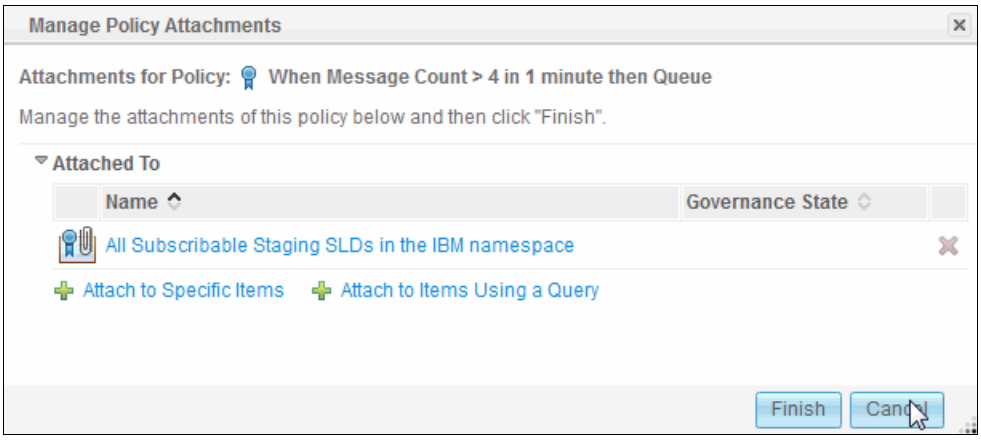


Figure 10-12 Manage Policy Attachments with Query

If an attachment was created by using the **Attach to Specific Items**, the Manage Policy Attachments dialog will show an attachment to the specific item, as in Figure Figure 10-13.

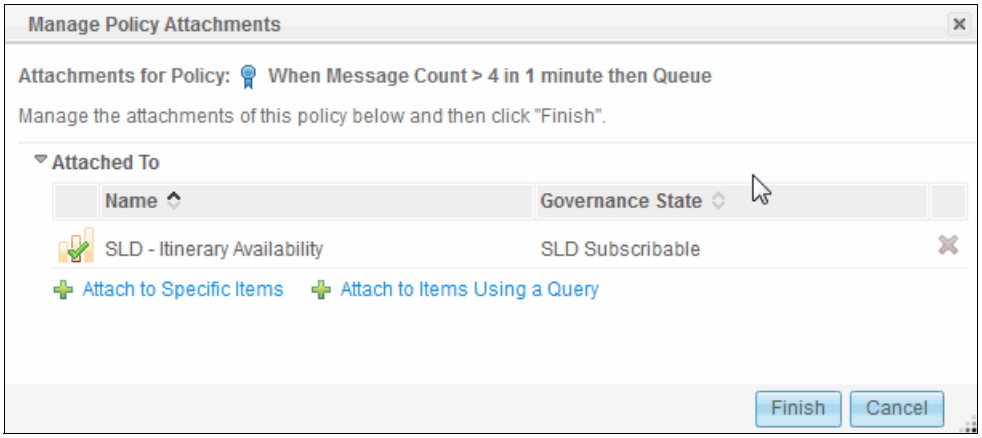


Figure 10-13 Manage Policy Attachments with Specific Item

10. At this stage no actual attachments are made, so new attachments can be deleted by clicking **Cancel**. After you make all of the attachments that are required for the policy, click **Finish** to actually create these attachments.
11. You can also detach the policy from an object to which it is already attached. This dialog displays all of the objects (or queries) to which the policy is attached. Select the plus sign (**+**) to the right of the item to remove it from the list and click **Finish** to detach the item.

> **Tip:** When you work in multiple environments (for example, Governance Master, Staging, and Production instances of WSRR), clicking **Finish** attaches the polices only on the Governance Master. The relevant objects must be promoted or repromoted to actually effect the changes in the pertinent run time or run times. See 4.1.7, “Promote the services to the WSRR run time” on page 95 for more detail about the objects that require repromotion.

After clicking **Finish** and the attachments or detachments are done, the detail view for the policy is again displayed, showing all of the objects to which the policy is attached and the query that is used to attach the policy. See Figure 10-14.

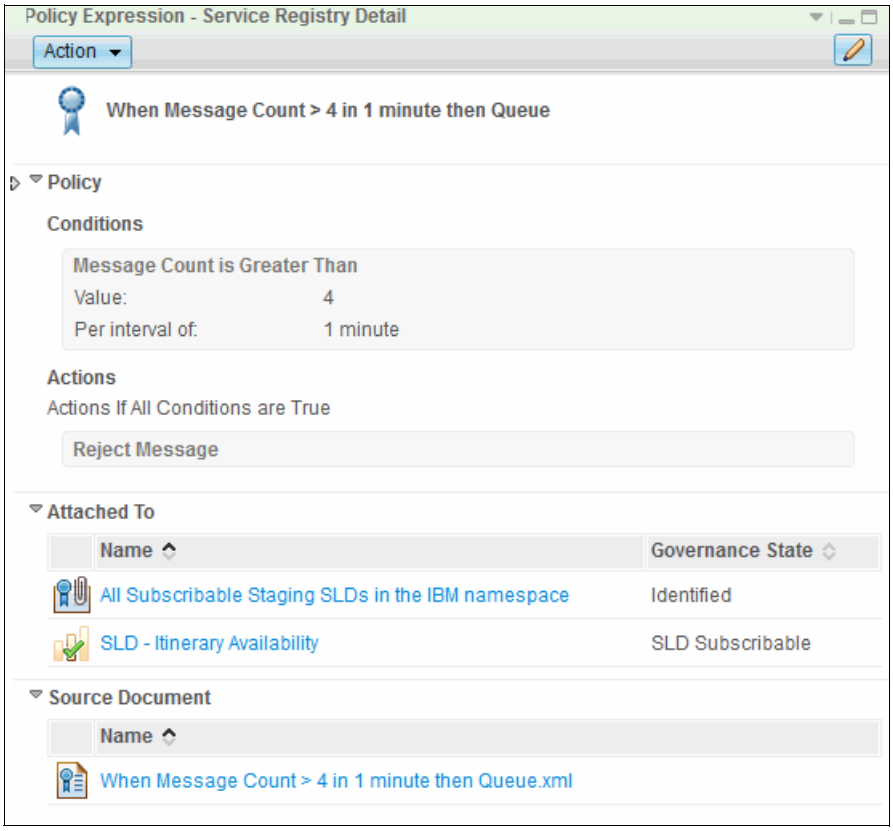


Figure 10-14 Policy Expression details view showing objects to which the policy is attached

If a policy was attached by using a query, the Attached To section shows the name of the query that was used but it does not show all of the objects that are currently in the system to which this policy is attached through the query. To see this information, click the link for the query.

For example, click **All Subscribable Staging SLDs in the IBM namespace** (Figure 10-14 on page 321) to display the detail view for the attachment query, as shown in Figure 10-15. The Applies To section in this view shows all objects to which this query relates, and thus all objects to which this query is attaching the policy.

Tip: Policies that are attached with the use of a query are attached dynamically. That is, if a new object has the specific criteria to match the query, it can gain the policy as an attachment. Similarly, if an object's metadata changes so that it no longer matches the specified criteria, it no longer has the policy attached.

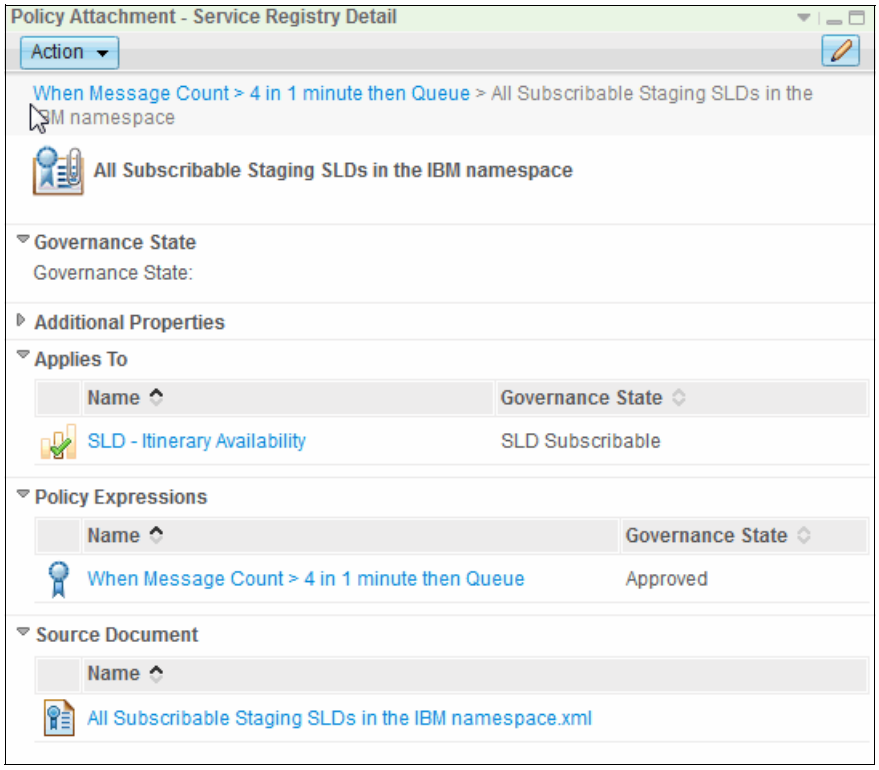


Figure 10-15 Policy Query Attachment Details view

10.2.2 Attaching a policy from a specific item

The second method of attaching a policy to an object is from a specific object. Navigate to the detail view of the object to which you want to attach the policy.

The following procedure selects the SLD for version 1.0 of the Itinerary Availability service, which is named `SLD - Itinerary Availability`, as shown in Figure 10-16.

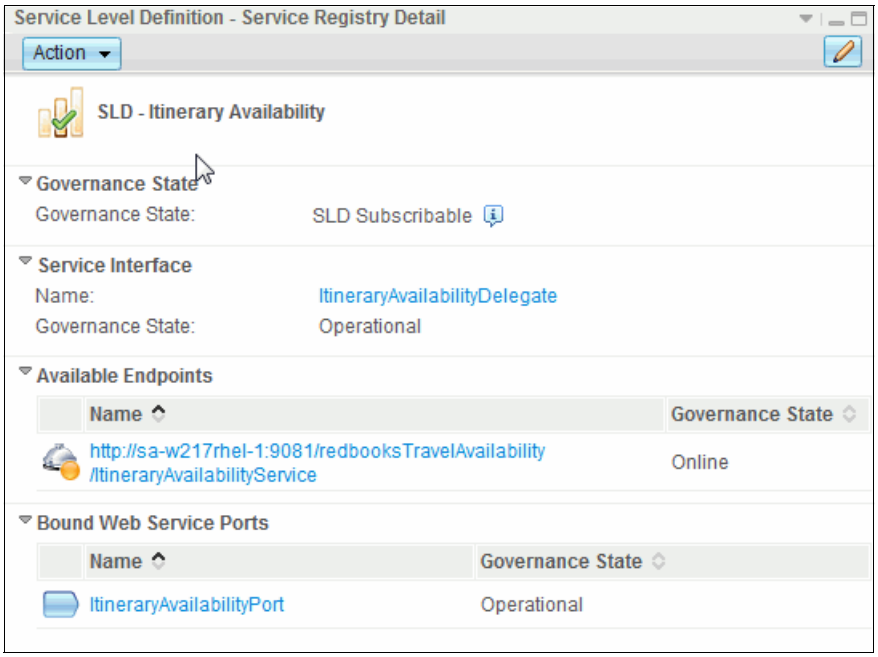


Figure 10-16 SLD – Itinerary Availability detail view

Use the following steps:

1. To attach a policy to this object, edit the object by clicking the **pencil** icon in the top right corner. The Edit panel for the object opens, as shown in Figure 10-17.

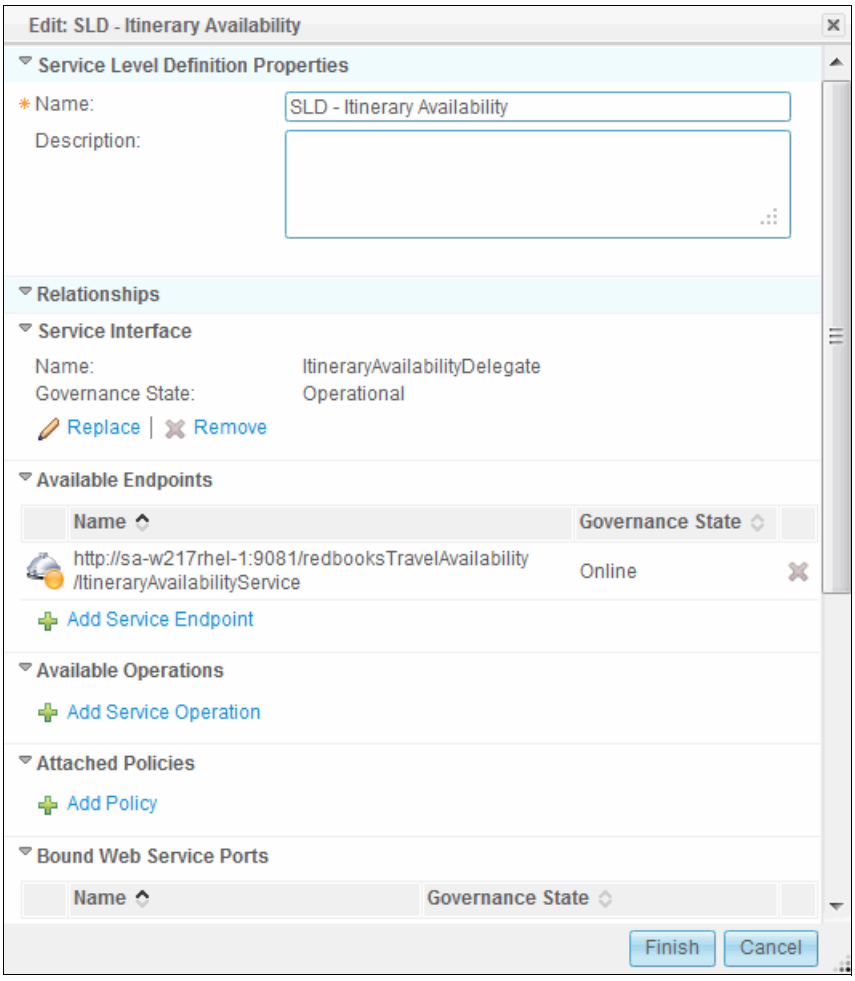


Figure 10-17 Edit pop-up for the SLD – Eligibility Service object

2. In the Attached Policies section, click the **Add Policy** link and then specify a name and type for the policy, as shown in Figure 10-18.

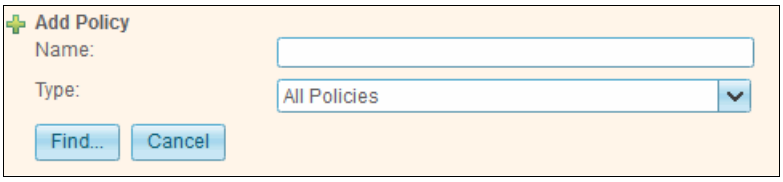


Figure 10-18 Add Policy section

3. Select the type of policy that you want to attach to this object. You can also type part of the name of the policy (including wildcard characters) to filter the search. If the WSRR auto-suggest capability is enabled, suggestions that match the search string are listed and can be selected, as shown in Figure 10-19.

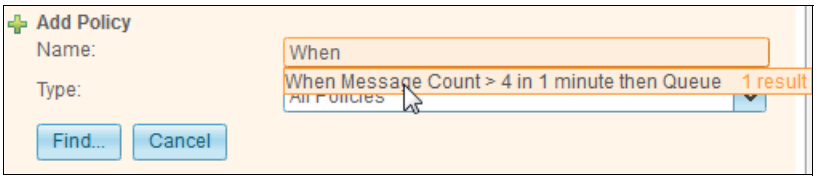


Figure 10-19 Policy Selection with auto-suggestions

4. When a policy is selected by using the suggestions, the policy is displayed as being attached to the object, as shown in Figure 10-20. Click **Finish** to attach the policy.

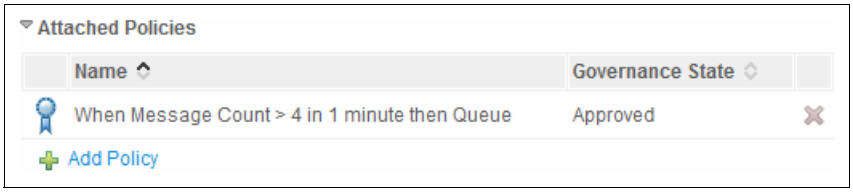


Figure 10-20 Policy Attachment from a object showing an attached policy

Tip: When you work in multiple environments (for example, Governance Master, Staging, and Production instances of WSRR), clicking **Finish** attaches the polices only on the Governance Master.The relevant objects must be promoted or repromoted to actually effect the changes in the pertinent run time or run times. See 4.1.7, “Promote the services to the WSRR run time” on page 95 for details about the objects that require repromotion.

10.3 Attaching a policy to a provider service using an SLD

To have a policy enforced on a service (or monitored for ITCAM for SOA), that service must have an SLD (see 3.3.2, “Creating service level definitions” on page 60). Only on the SLD (or SLA) can a policy be attached. A policy attached to an SLD applies to the corresponding provider service and is enforced for all consumers of the service.

10.3.1 Service

This service is the easiest case. The SLD must be created as specified in 3.3.2, “Creating service level definitions” on page 60 and must be transitioned into the SLD Subscribable state. After the policy is attached by using one of the methods that is specified earlier in this chapter, the policy will be enforced.

10.3.2 Operation

If a provider service has multiple operations and you want a policy to be enforced only for a subset of the operations, you must specify the operations on the SLD for which you want the policy to apply. Instead of only one SLD for the service, you should have as many as needed so that all operations to be consumed have at least one SLD; and also the operations must be

partitioned in such a way so you attach the necessary polices to each subset of operations. For example, if you want to apply one policy to a delete operation and another policy to the add and delete operations, you must have two separate SLDs.

When additional SLDs are required, repeat the steps described in 3.3.2, “Creating service level definitions” on page 60. Operations can be specified as described in the following procedure, after you navigate to the details for your SLD in the WSRR Business Space:

1. Click the **pencil** icon to edit the SLD.
2. In the Available Operations section, click **Add Service Operation** to display the section shown in Figure 10-21.

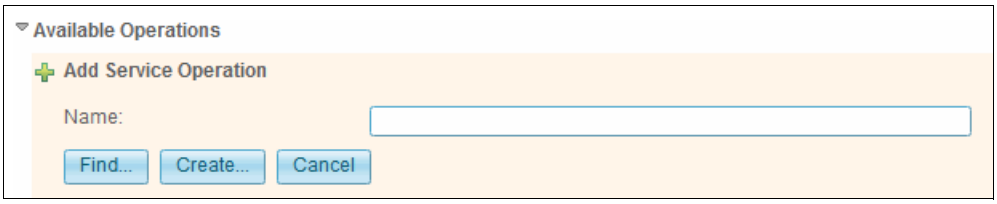


Figure 10-21 Add Service Operation in Edit dialog for SLD

3. Enter search text in the Name field and click **Find**, or use auto-suggest to select the operations you want to add. A panel similar to Figure 10-22 opens.

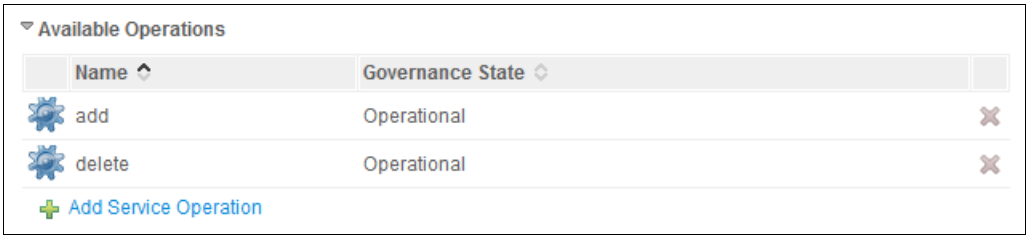


Figure 10-22 Example operations added to an SLD

4. After you add all required operations, click **Finish**.

You can now add any required policies to your SLD, as described in Chapter 10, “Attaching a policy to a service” on page 313; these are enforced only for the specified operations.

It is possible that an operation may be covered by more than one SLD. In this case, all policies that are attached to all of the SLDs for which the operation is specified are enforced. If there are, for example, three operations for a service, and an SLD exists for the service with one specified operation and another with no specified operation, the SLD with no specified operations still applies for all three operations.

10.3.3 Endpoint

As part of creating an SLD (as detailed in 3.3.2, “Creating service level definitions” on page 60) you specify the Available Endpoints. If a service has multiple endpoints and you want all endpoints to be treated the same, with respect to policies, then one SLD can be used for all of them and all endpoints should be specified under the Available Endpoints section of the SLD. However, if some endpoints apply different polices, then you must use more than one SLD, with (at least) one SLD for each set of endpoints that are to have the same policies. For more information about adding endpoints, see 3.3.2, “Creating service level definitions” on page 60.

10.4 Attaching policy to consumer-provider pair by using SLA

To have a policy enforced on a service consumer-provider pairing (or monitored for ITCAM for SOA) there must be a corresponding SLD-SLA pairing in WSRR. The provider service must have an SLD (see 3.3.2, “Creating service level definitions” on page 60) and that SLD must be specified as the agreed endpoint on an SLA. That SLA represents the contract between the consumer and provider. It is only on the SLA (or SLD) that a policy can be attached. A policy that is attached at the SLA level is enforced for only the specified consumer of the provider service. Of course, any policies that are attached to the SLD will be applied to all consumers of the corresponding provider service.

Tip: The policies that are applicable for a given request of a service are those on the SLA for the given consumer-provider pairing and those on *all* SLDs for the service that specifies the operation that is being invoked (or which specify no operations), not only the SLD that the relevant SLA consumes.

10.4.1 Service

As in 10.3.1, “Service” on page 325, this case is the easiest. The SLA must be created as described in 4.1.3, “Create a service level agreement” on page 91 to consume an SLD that is for the given service. The SLA must be in the SLA Active state and the SLD must be in the SLD Subscribable state. After the policy is attached, by using one of the methods that is described in previous sections of this chapter, the policy will be enforced for that consumer.

10.4.2 Operation

If a provider service has multiple operations and you want a policy to be enforced for only a subset of the operations, then you must specify the operations on the SLD for which you want the policy to apply.

An SLA consumes one or more SLDs. If SLDs do not exist so that the combined set of operations match those for which you want your policy to apply for your given consumer, then you must create a new SLD. If you want different policies for one operation of a service for a given consumer compared to other operations of the same server for the same consumer, you must have more than one SLA between the same consuming Capability Version and (through different SLDs) the providing endpoint.

Figure 10-23 shows an example with two SLAs between the same consumer and provider. It shows two SLDs for the same Service Version, each SLA consumers one of the SLDs. See 10.3.2, “Operation” on page 325 for how to create SLDs and specify operations.

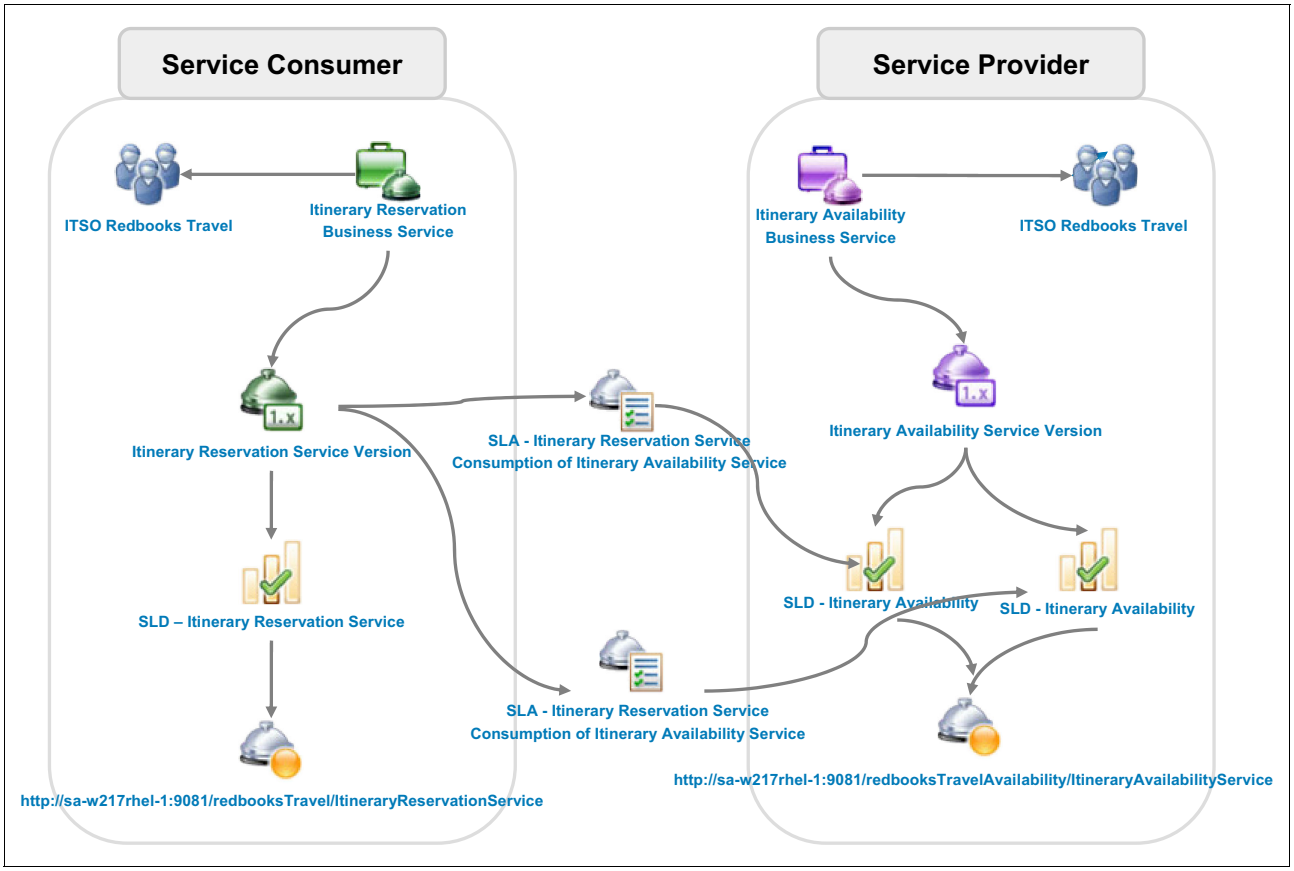


Figure 10-23 GEP Objects for a Service Consumer of a Service Provider with multiple SLAs and SLDs

10.4.3 Endpoint

If you want separate policies for one endpoint of a service for a given consumer versus another endpoint for the same consumer, then you must have a separate SLD for each endpoint and an SLA from the consuming Capability Version to each of the SLDs. The policies can then be attached to each SLA as required. Figure 10-24 shows how the governance objects might look for such an example.

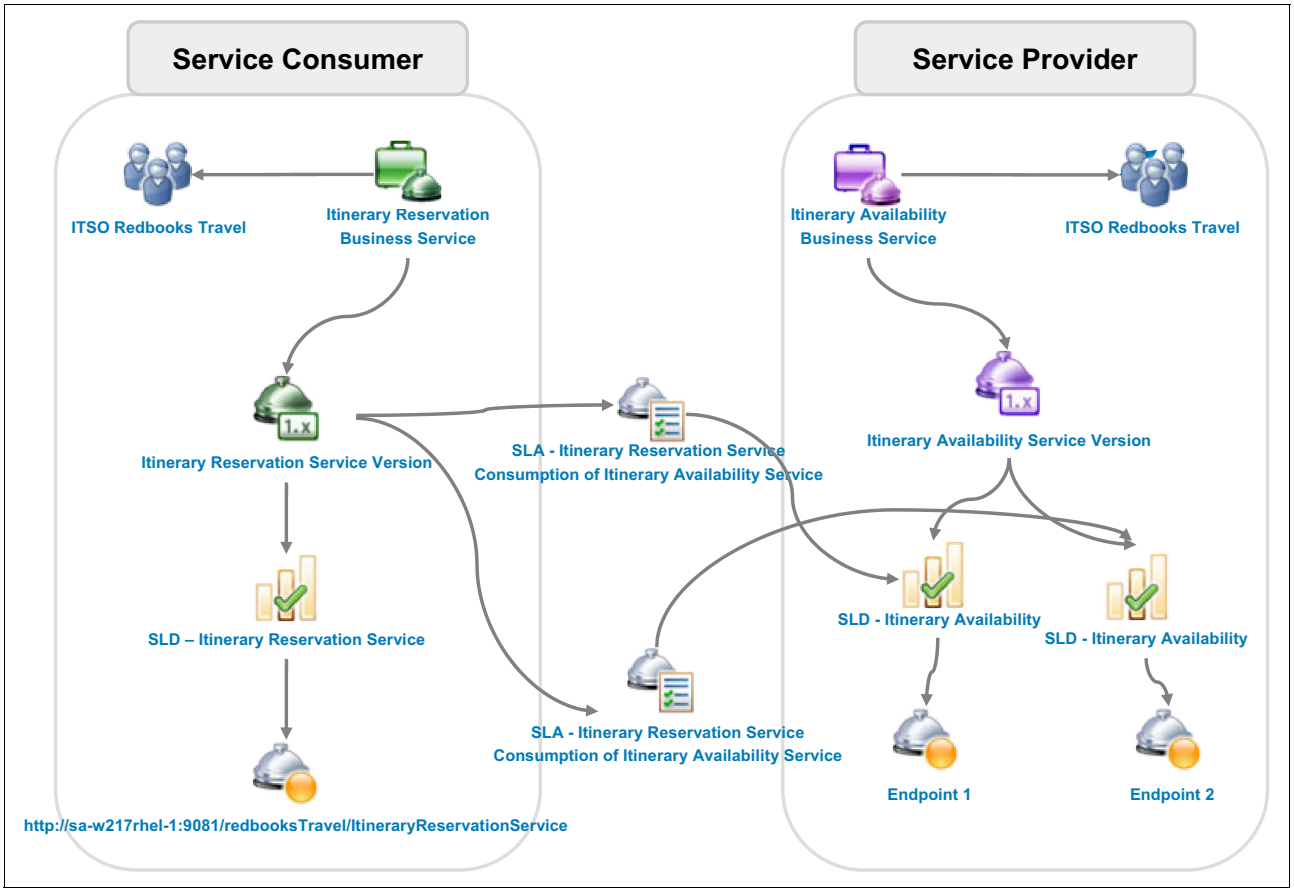


Figure 10-24 GEP Objects for a service consumer of a service provider with multiple endpoints, SLAs and SLDs

10.5 Specifying policies from unknown consumers

Now you understand how to attach policies to SLDs (for all consumers of a service) and to SLAs (for a specific consumer-provider pairing). However, in certain cases you might want to specify policies for all unknown clients of a service, that is, for those that do not have a matching SLA.

WSRR has the concept of an *anonymous* SLA. When DataPower picks up policies that are attached to an anonymous SLA, the policies are applied only to invocations of a service when no other SLAs are applicable.

Use the following procedure to create an anonymous SLA:

1. Open the SLD detail in the WSRR Business Space.
2. Click the **pencil** icon to edit the SLD.
3. Scroll to the bottom of the dialog and click **Add Service Level Agreement** in the Anonymous SLA section, as shown in Figure 10-25.

Figure 10-25 Add anonymous SLA

4. Click **Create**.
5. Enter a name for the SLA, as shown in Figure 10-26.

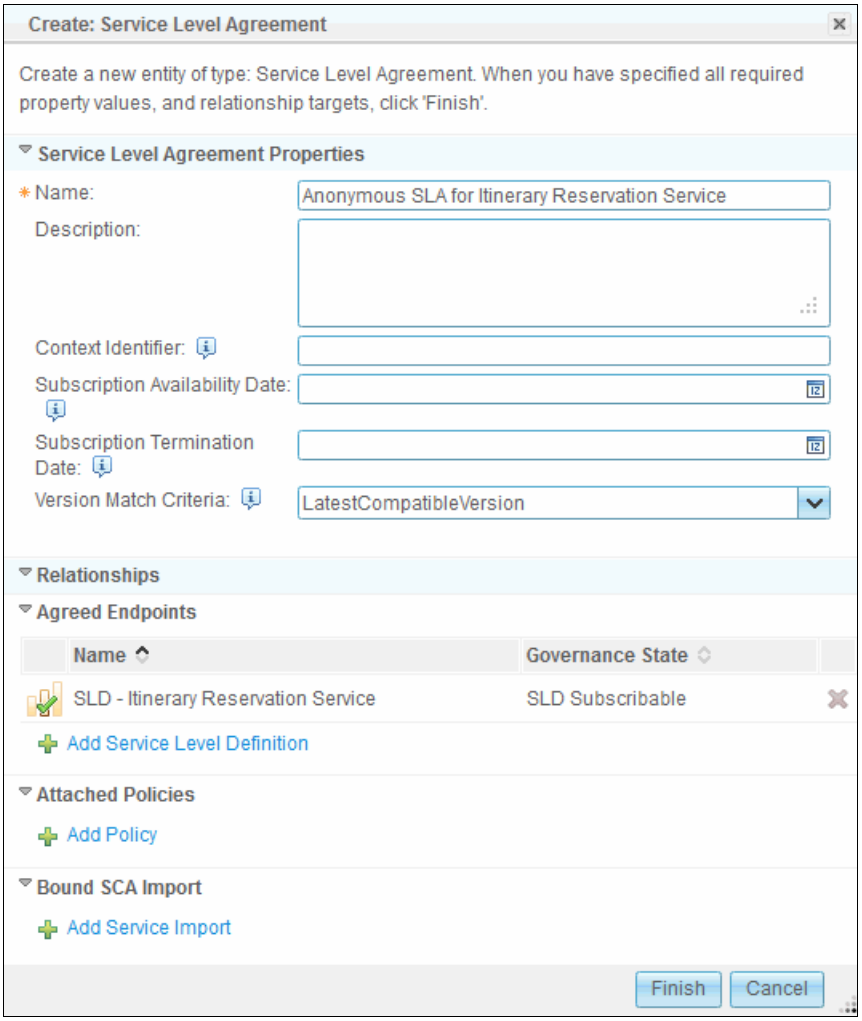


Figure 10-26 Creating an anonymous SLA

6. Under Agreed Endpoints, click **Add Service Level Definition** and link back to the same SLD from step 2. This link is required, because every SLA must consume an endpoint.
7. Click **Finish** to return to the Edit SLD dialog.
8. Click **Finish** again.

The Anonymous SLA is created and you are ready to attach policies by using whatever method you want to employ. The SLA must be transitioned to the SLA Active state (as with any other SLD) to be active.

11

Policy administration point utilities

This chapter serves as a reference to be used in understanding the various utilities provided for policy administration through the policy administration point (PAP), as implemented in WSRR.

This chapter contains the following topics:

- 11.1, “Understanding the policy lifecycle” on page 334
- 11.2, “Viewing policies attached to a service” on page 336
- 11.3, “Viewing services attached to a policy” on page 338
- 11.4, “Restricting policy access” on page 339
- 11.5, “Viewing activity log for a policy” on page 342
- 11.6, “Using policy events for management of monitoring policy results” on page 343

© Copyright IBM Corp. 2013. All rights reserved.

11.1 Understanding the policy lifecycle

Policy lifecycle in IBM WebSphere Service Registry and Repository (WSRR) has six possible governance states.

- Identified
- Specification Review
- Approved
- Superceded
- Deprecated
- Retired

Figure 11-1 shows SOA user actions (transitions) and WSRR governance states during a policy lifecycle.

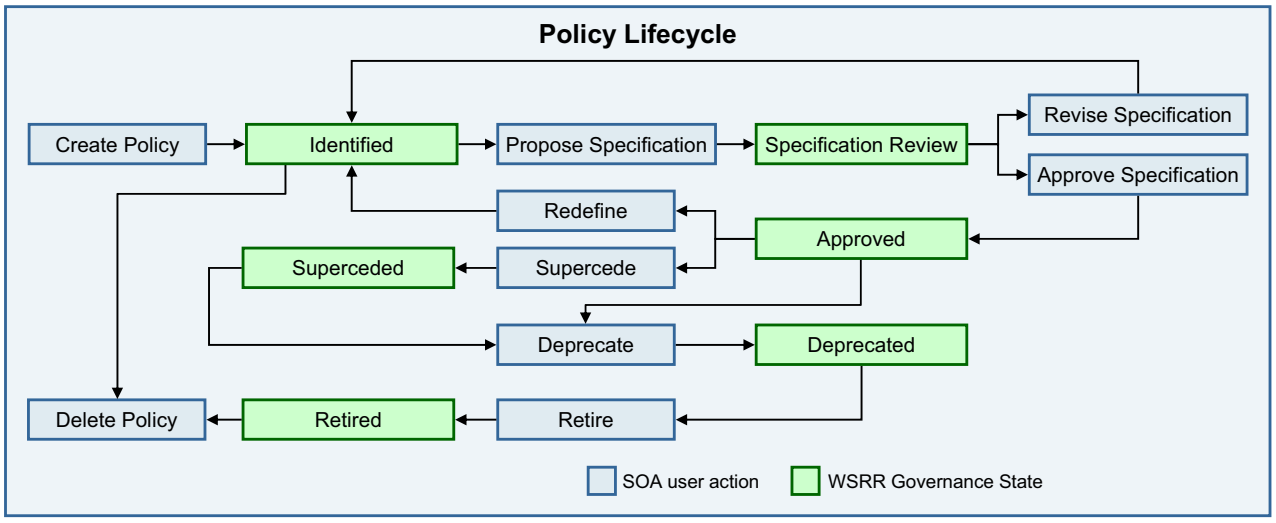


Figure 11-1 Policy lifecycle

The stages of a policy lifecycle (as shown in the figure) are as follows:

- When a policy is created, WSRR governs the policy to the *Identified* state.
- Meet the stakeholder requirements in the policy definition and propose the specification. WSRR governs the policy to the *Specification Review* state.
- When the specification review is complete, approve the specification to deploy to runtime environments.
  - If approved, WSRR governs the policy to the *Approved* state.
  - If the specification review does not meet the requirements, revise the specification to rework it; WSRR governs to the *Identified* state.
- After approval, if reworking the policy is necessary because of a specification change or any runtime errors, redefine the policy. On a redefine action, WSRR governs to the *Identified* state. If a later version of the policy is preferred to the current version, supercede the policy. On a supercede action, WSRR governs to the *Superceded* state. If the policy should no longer be used after approval or superceding and there is a possibility of removing the policy, deprecate the policy. WSRR governs to the *Deprecated* state.
- If the policy is deprecated and can be removed from the system, retire the policy. WSRR governs to the *Retired* state.
- A retired policy can be deleted only if the governance state is either *Identified* or *Retired*.

To perform any action on a policy, log in to the WSRR Business Space, and then select **Go to Spaces → IBM Redbooks Travel Registry for Governance → Overview → Search widget → Policy Expression → Choose Policy → Source Document → policydocument.xml → Action**.

Figure 11-2 shows the action on a sample policy called `ThrottleRogue`. Note that transitions cannot be made at the policy expression level. Navigate to the policy document level to make the transitions.

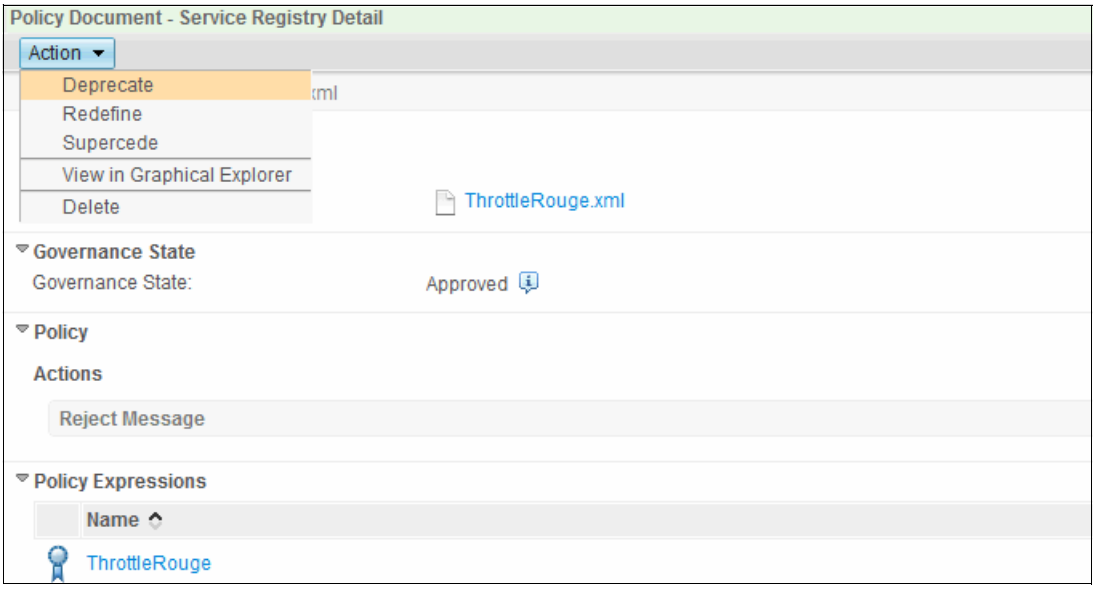


Figure 11-2 Policy Action

Figure 11-3 shows an example mediation policy transitions to various governance states.

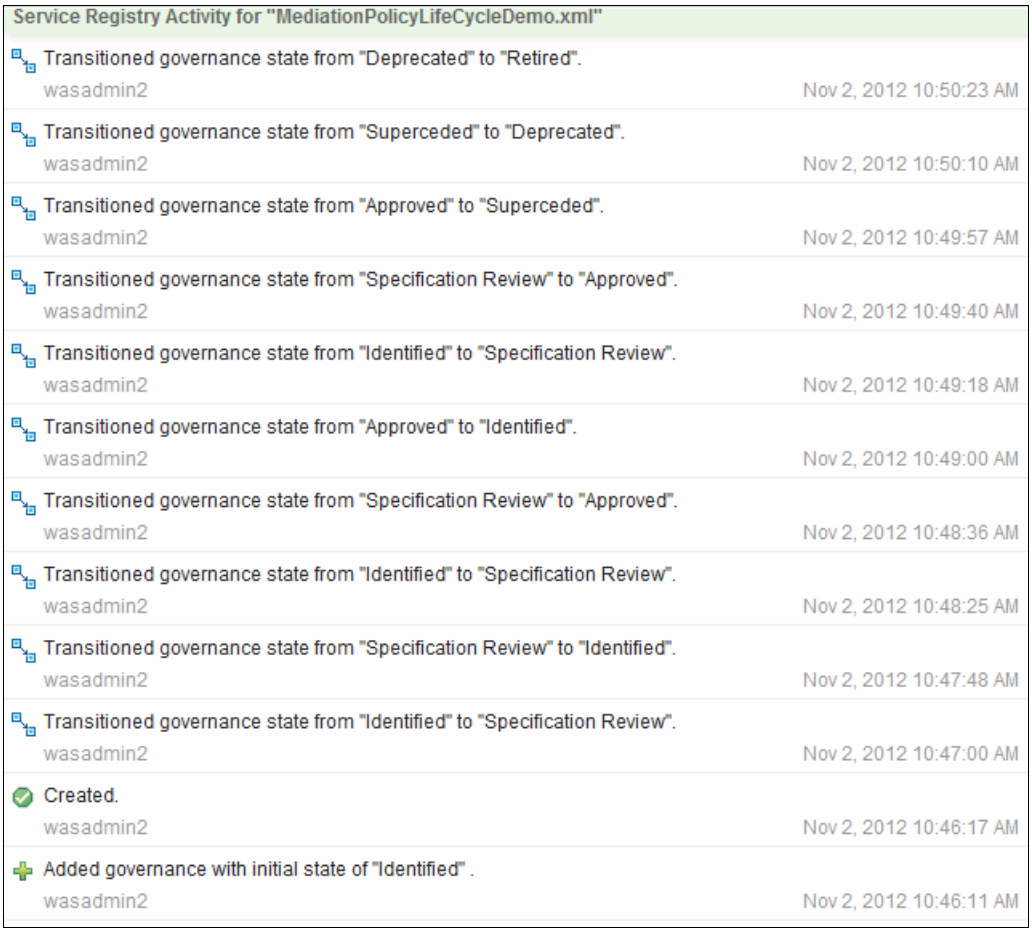


Figure 11-3 Policy transitions

11.2 Viewing policies attached to a service

To view the policies that are attached to a sample service, log in to WSRR Business Space, and then select **Go To Spaces** → **IBM Redbooks Travel Service Registry Governance Space** → **Overview** → **Search widget** → **Business Service** → **Itinerary Reservation Business Service**.

You can either navigate through links in the Service Registry Detail widget or use the graphical Service Registry Navigator widget.

Choose **Itinerary Reservation Business Service** → **Service Registry Detail widget** → **Itinerary Reservation Service Version (1.0)** → **Service Level Definitions** → **SLD - Itinerary Reservation Service - Prioritized Access** → **Attached Policies**.

Figure 11-4 shows the policies that are attached to SLDs that are associated with the Itinerary Reservation Service.

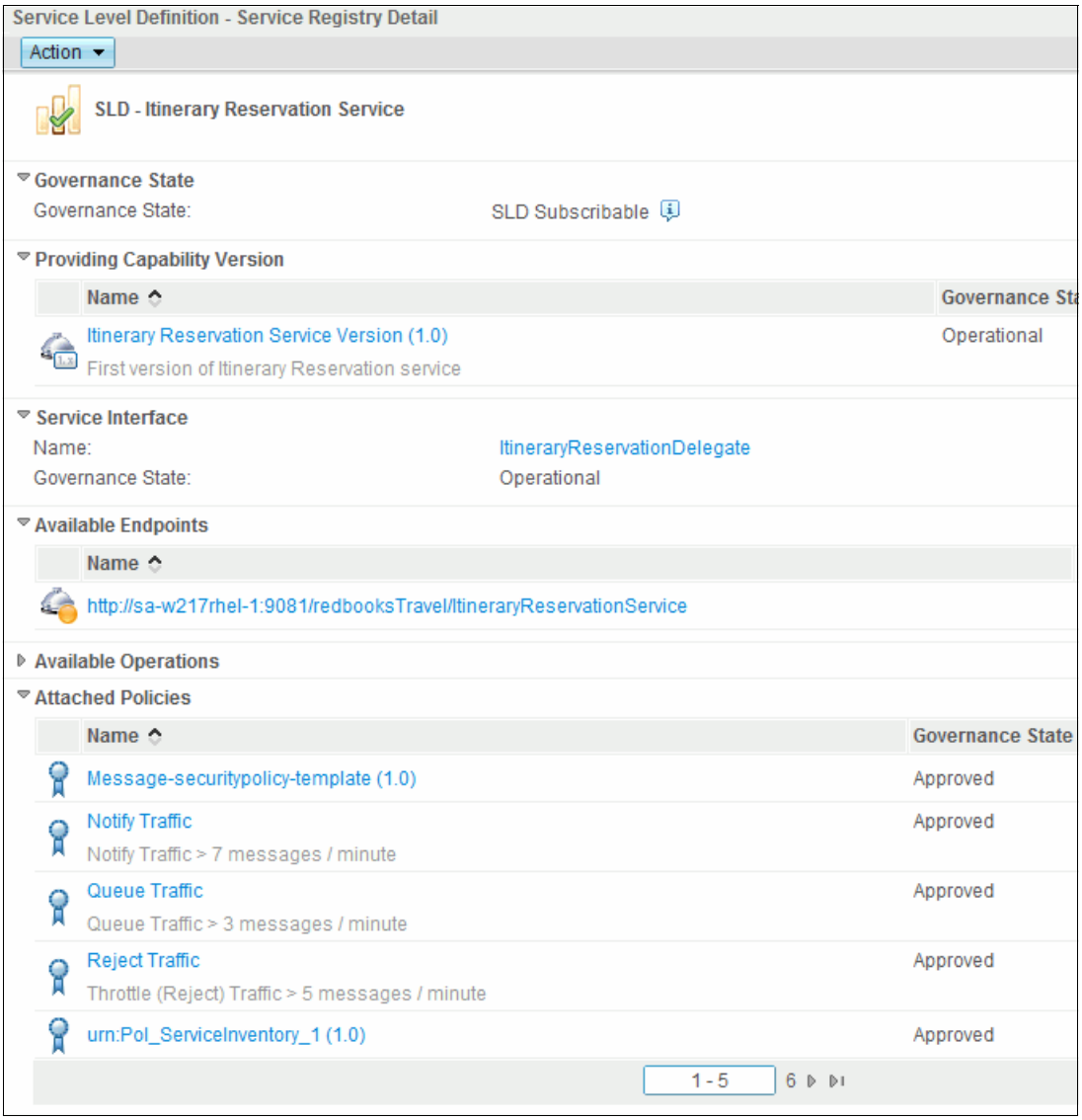


Figure 11-4 Viewing policies attached to a service

11.3 Viewing services attached to a policy

To view the services that are attached to a sample policy, log in to WSRR Business Space, and then select **Go To Spaces → IBM Redbooks Travel Service Registry Governance Space → Overview → Search widget → Policy Expression → Queue Traffic (policy name)**.

Figure 11-5 shows the SLD that is attached to the Queue Traffic policy. This example shows only one service attached to the Queue Traffic policy. If multiple services are attached, multiple SLDs are listed in the result.

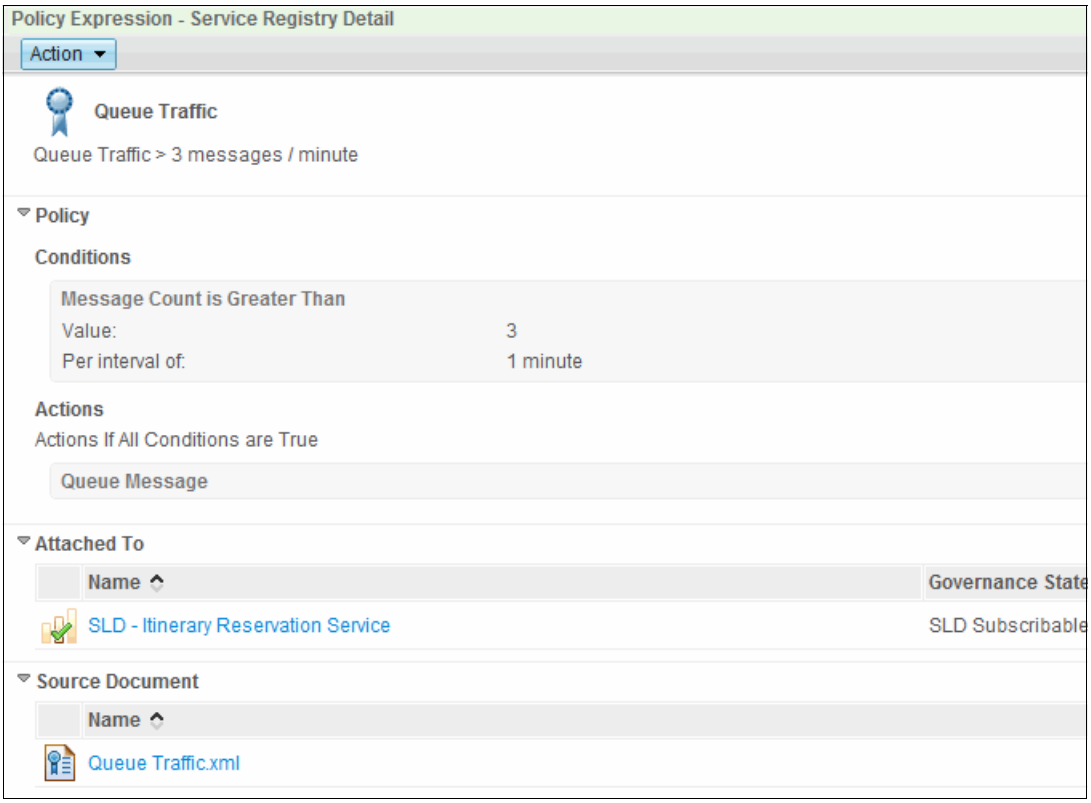


Figure 11-5 Viewing SLD related to a policy

You can either navigate through the SLD links in the Service Registry Detail widget or use the graphical Service Registry Navigator widget.

Navigate to **Queue Traffic → Attached To → SLD - Itinerary Reservation Service → Providing Capability Version**.

Figure 11-6 shows the capability version that is attached to the SLD. The capability version is the Service Version that is associated to the policy.

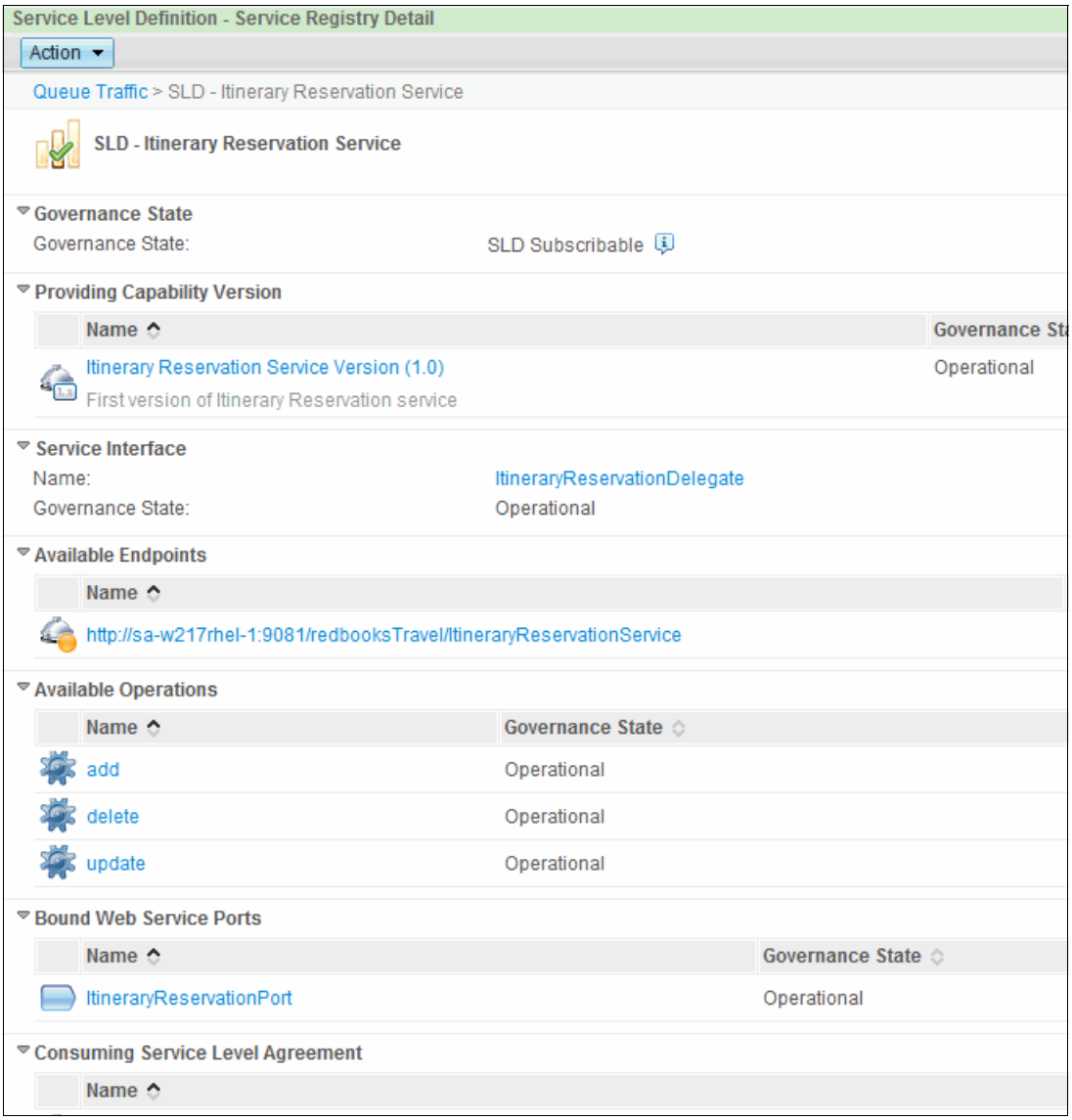


Figure 11-6 Viewing services related to SLD

11.4 Restricting policy access

Policy access can be restricted either directly in WSRR or with WSRR Studio's Access Control editor.

To launch the access control tool in WSRR Studio, start Studio, and then navigate to **Tools** → **WSRR Access Control Editor**. Be sure you created a new WSRR Configuration Profile if this time is the first time you are using Studio.

Figure 11-7 shows the Access Control Editor. There are two new roles (RedbooksTravelAdmin and RedbooksTravelDeveloper) and a new permission (DeletePolicy).

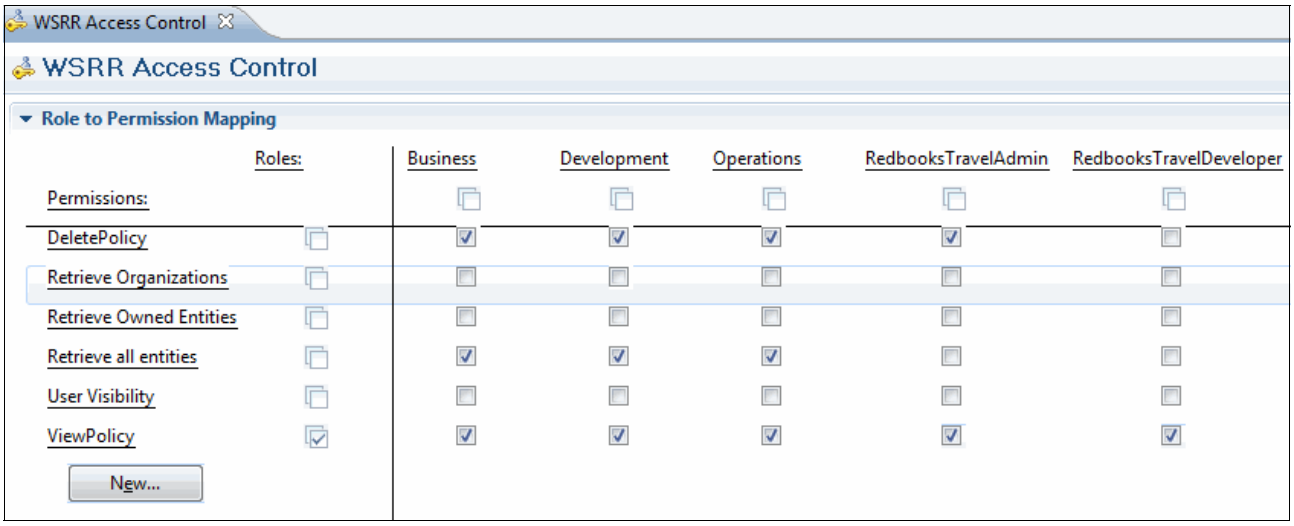


Figure 11-7 WSRR Studio's Access Control Editor

Figure 11-8 shows the DeletePolicy permission with the Delete action, and Policy Document as the SDO type.

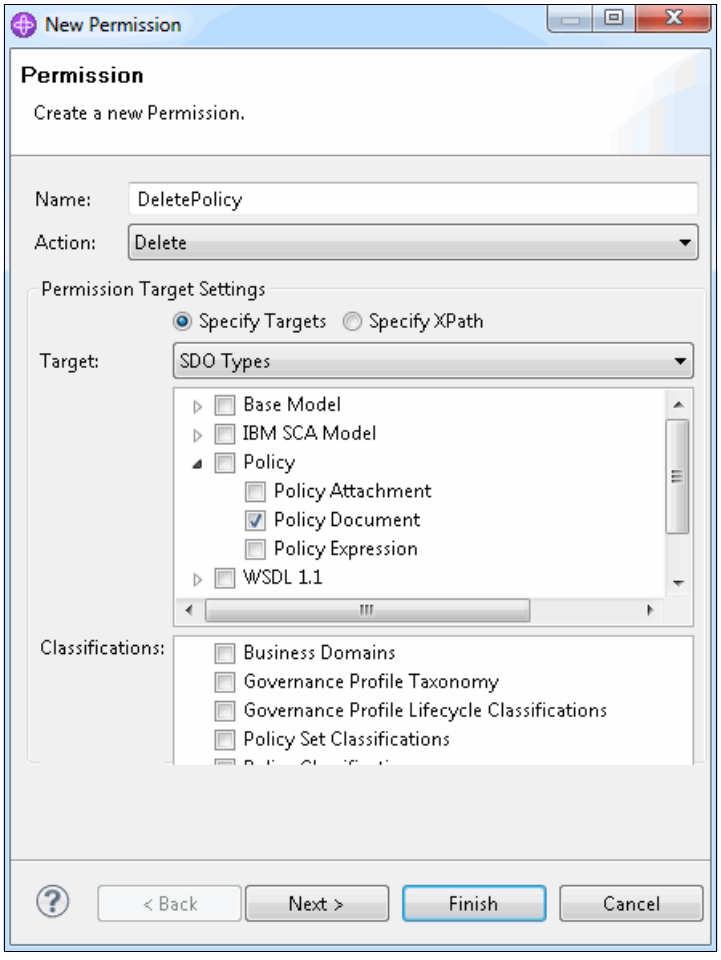


Figure 11-8 Creating a new permission

The consequences of the DeletePolicy permission and its mapping to the all roles except RedbooksTravelDeveloper are that all users can delete policies except those in the RedbooksTravelDeveloper role. Before the permission was created and mapped, all users were free to delete policies.

By using the WSRR configuration project in WSRR Studio, you can set up fine-grained access control to policies and can publish the changes to the WSRR servers. Generate all WSRR artifacts and the synchronize profile with WSRR to see the changes in the runtime environment.

Alternatively, you can specify access controls in the WSRR web UI. Log in to the web UI and navigate to **Configuration Perspective** → **Active Profile** → **Access Control**. This method is also the only method for mapping users to roles.

Role definitions and permission definitions can be viewed or updated from the access control editor. After the access changes are published from WSRR Studio, the updated roles and permission mappings can be seen in the WSRR Service Registry access control editor.

WSRR users are mapped to the new roles, as shown in Figure 11-9.

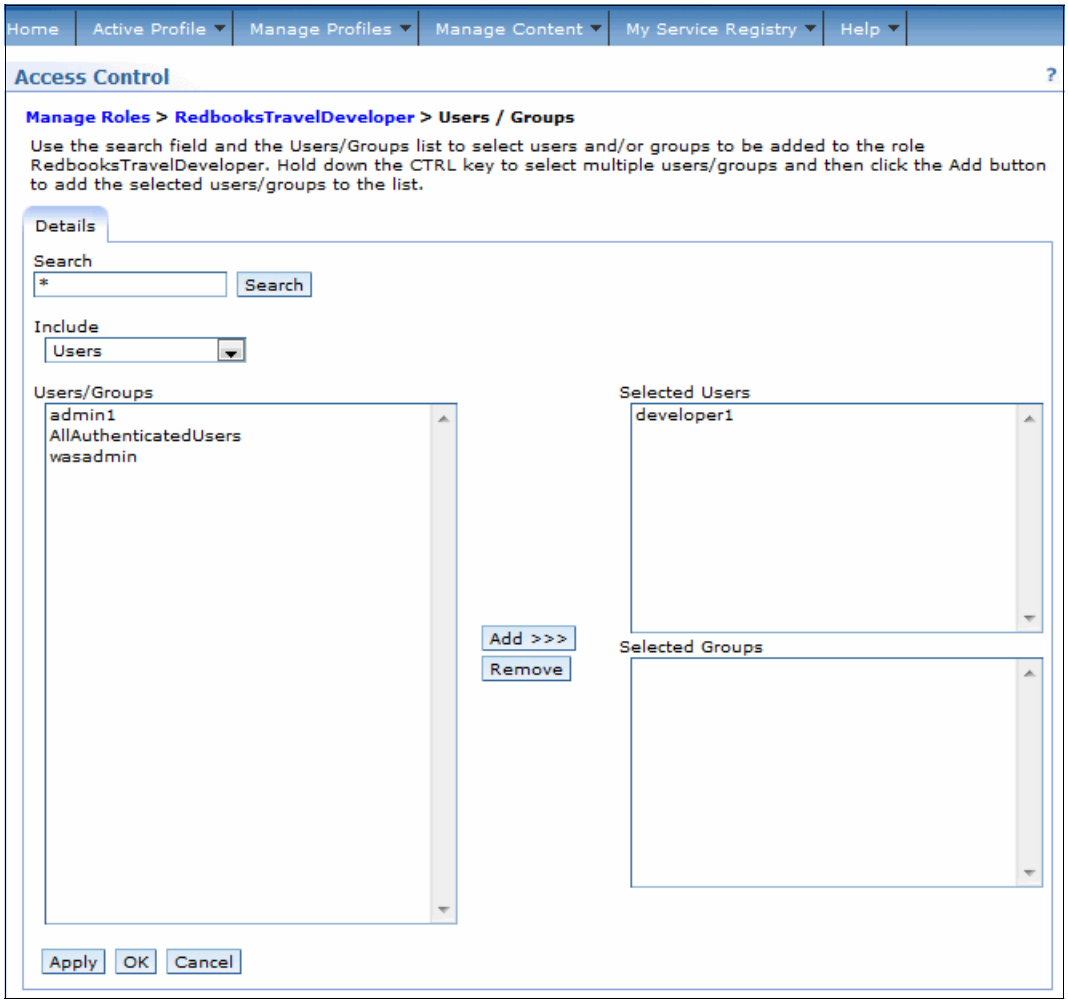


Figure 11-9 Mapping user to the RedbooksTravelDeveloper role in the WSRR web UI

Fine-grained access control for policies can be implemented with these procedures. In this way, organizations can establish proper governance and meet security requirements.

11.5 Viewing activity log for a policy

To view the history for a sample policy, log in to WSRR Business Space, and then select **Go To Spaces → IBM Redbooks Travel Service Registry Governance Space → Overview → Search widget → Policy Expression → Queue Traffic (policy name)**.

Figure 11-10 shows activities for an existing policy name Queue Traffic.

Figure 11-10 Viewing activities for a policy

Figure 11-11 shows the Queue Traffic policy in the Service Registry Navigator.

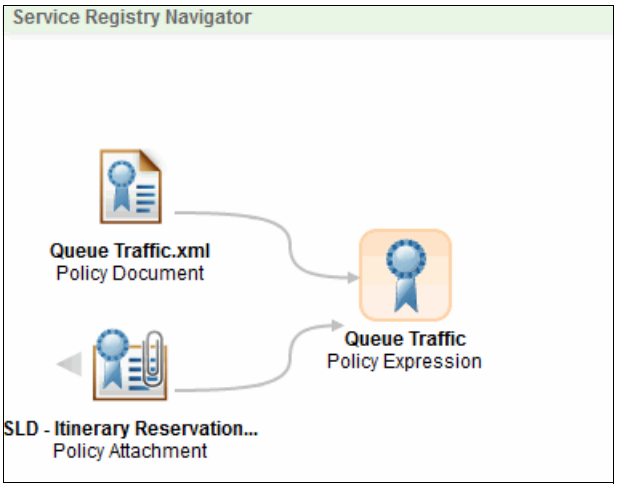


Figure 11-11 Viewing policy in the Service Registry Navigator

In the Service Registry Navigator, click **Policy attachment** to view the activities on policy attachments, as shown in Figure 11-12.

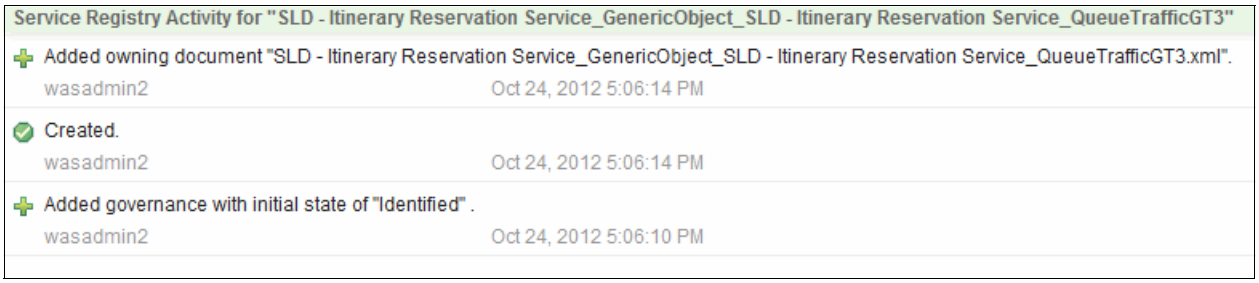


Figure 11-12 Viewing activities for policy attachments

11.6 Using policy events for management of monitoring policy results

ITCAM monitoring policy events can be used to dynamically manage the runtime environments to maintain the SLAs. ITCAM does this task by changing registry metadata with corresponding middleware products that receive this information and dynamically change the process flow. Further detail is beyond the scope of this book other than how ITCAM works this metadata eventing with WSRR. For details about how ITCAM sends events to WSRR, which changes the metadata, see 9.3.3, “Automatic response to WSRR (eventing)” on page 301. Also see the following information center for further details:

http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/cwsr_itcamforsoaintegration.html

Part 4

Policy enforcement point

This part contains the following chapters:

- Chapter 12, “DataPower policy enforcement point configuration” on page 347
- Chapter 13, “Creating and using custom policies” on page 367
- Chapter 14, “ITCAM as policy monitoring point” on page 389

© Copyright IBM Corp. 2013. All rights reserved.

12

DataPower policy enforcement point configuration

This chapter describes various concepts and tasks for configuring a DataPower device to serve as the policy enforcement point in the larger SOA policy architecture.

The DataPower device requires the configuration of the following objects to interoperate with IBM WebSphere Service Registry and Repository (WSRR) for policy enforcement:

- WSRR Server object, which identifies the WSRR server to use
- WSRR Subscription object, which identifies a subscription to a WSRR object
- WSRR Saved Search Query object, which identifies a saved search query for WSRR objects
- DataPower Web Service Proxy object, which is the primary service
- WS-Policy Parameter Set object, which provides additional input to policies

This chapter contains the following topics:

- 12.1, “Overview” on page 348
- 12.2, “WSRR server object” on page 349
- 12.3, “Web Service Proxy object” on page 350
- 12.5, “WSRR subscriptions and saved search subscription” on page 354
- 12.6, “Policy Parameter Set object” on page 356
- 12.7, “SLA policy details” on page 357
- 12.8, “Scope of a policy on one or more DataPower appliances” on page 362

© Copyright IBM Corp. 2013. All rights reserved.

12.1 Overview

In any scenario that involves service level agreement (SLA) policy implementations (for example, the limit of *x* transactions per second or a guaranteed response time), with or without an SOA policy mediation component (for example, all transactions must be authenticated), some entity must enforce the policies. This entity is known as the policy enforcement point (PEP). In the scenarios discussed in this book, a DataPower device serves as the PEP. For an illustration of a PEP, see 6.1, “Service security” on page 188.

DataPower provides Web Service Proxy services for enterprise applications running inside the enterprise firewalls. Enterprise application clients send requests to the DataPower device as though the device were the actual application. The Web Service Proxy service then applies any wanted policies to every transaction destined for the real application.

The DataPower service can be built easily by providing the WSDL for the enterprise application to the device and providing some endpoint information to set the IP address and port that are used to listen for requests. The device parses the WSDL, creates a default set of message processing policies, sets the back-end destination to the URL that is defined for the service in the WSDL, and automatically includes the necessary actions to use service level monitoring to enforce SLAs.

When a WSRR server is present in the architecture, the DataPower device can retrieve WSDLs from WSRR. Administrators can use the repository and governance capabilities of WSRR to manage the WSDLs that describe enterprise applications rather than uploading them to one or more DataPower devices. This way becomes particularly useful as those applications evolve and change, thus requiring new WSDL deployments. The WSRR server can be configured to automatically notify those DataPower devices that subscribed to a given WSDL when it changes. Upon receipt of the notification from WSRR, DataPower obtains the new WSDL and refreshes the corresponding Web Service Proxy.

Starting with version 4.0 of the DataPower firmware, in combination with version 7.5 or later of WSRR, it is possible to configure DataPower to read and enforce WS-SecurityPolicy attachments that are associated with a WSDL. These WS-SecurityPolicy attachments can be authored in WSRR and linked to one or more WSDLs, also stored in WSRR. When the DataPower device obtains the WSDL from WSRR, it can also obtain all related policy attachments and enforce those policies. For example, a WS-SecurityPolicy attachment might require a user name and password in every request. When the DataPower device receives this attachment to the WSDL, it automatically creates the necessary processing policy to enforce the rule. Messages that do not contain a user name and password are no longer accepted.

With the release of DataPower firmware version 5.0 and WSRR Server 8.5, these policy attachments can now extend to service level definition (SLD) and service level agreement (SLA) policies. These are expressed as WS-MediationPolicy attachments to a WSDL, and are associated with a WSDL in WSRR in much the same way as WS-SecurityPolicy attachments. Similarly, the DataPower device obtains these policy statements from WSRR as attachments and enforces them. In this case, the enforcement is primarily done through the automatic configuration of a DataPower service level management (SLM) action object.

See Figure 6-2 on page 193 for a representation of these relationships.

The combination of DataPower 5.0 and WSRR 8.5 together provide a clean and powerful system for the following items:

- Reverse proxy of enterprise applications
- Enforcement of SOA policies with minimal effort
- Enforcement of SLA policies with minimal effort
- Centralized management and governance of polices
- DMZ-quality throughput and protection

12.2 WSRR server object

The WSRR server object identifies a remote WSRR server, available to the DataPower device. The configuration inputs listed in Table 12-1 provide all of the necessary information to establish a connection.

Table 12-1 WSRR server object

Input	Value
Name	`production-wsrr` This value can be any arbitrary name. Choose a descriptive name for easy selection later.
SOAP URL	`https://sa-w217rhel-2.itso.ral.ibm.com:9443/WSRRCoreSDO/services/WSRRCoreSDOPortt` This URL is used to communicate with the Registry through WSRR SOAP API. A typical default looks like one of the following URLs, depending on whether WebSphere global security is enabled. - `https://192.18.1.120:9443/WSRRCoreSDO/services/WSRRCoreSDOPort` - `http://192.18.1.120:9080/WSRRCoreSDO/services/WSRRCoreSDOPort` This scenario employs a secure connection (SSL), so you must configure an SSL Proxy Profile to handle the transport security.

SSL Proxy Profile	`production-wsrr-sslproxy-profile`
	
	You must select SSL proxy profile if you are using SSL (HTTPS) to connect to the WSRR server. You can use an existing profile by selecting one from the drop-down list or you can create a new profile.
	
	This profile employs the certificate from the WSRR server for authentication. For information regarding the creation of SSL Proxy Profiles, see the DataPower 5.0 information center:
	
	`http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/index.jsp`

Username	`wasadmin` This user name is used to log in to the WSRR server. Leave this field empty if authentication is not required. In this scenario, an appropriate username is used.
Password	This password is used to log in to the WSRR server. Leave this field empty if authentication is not required. In this scenario, a password is used.
WSRR Server Version	This field indicates the WSRR server version, (must be 7.5 or later).

Figure 12-1 shows an example of this configuration.

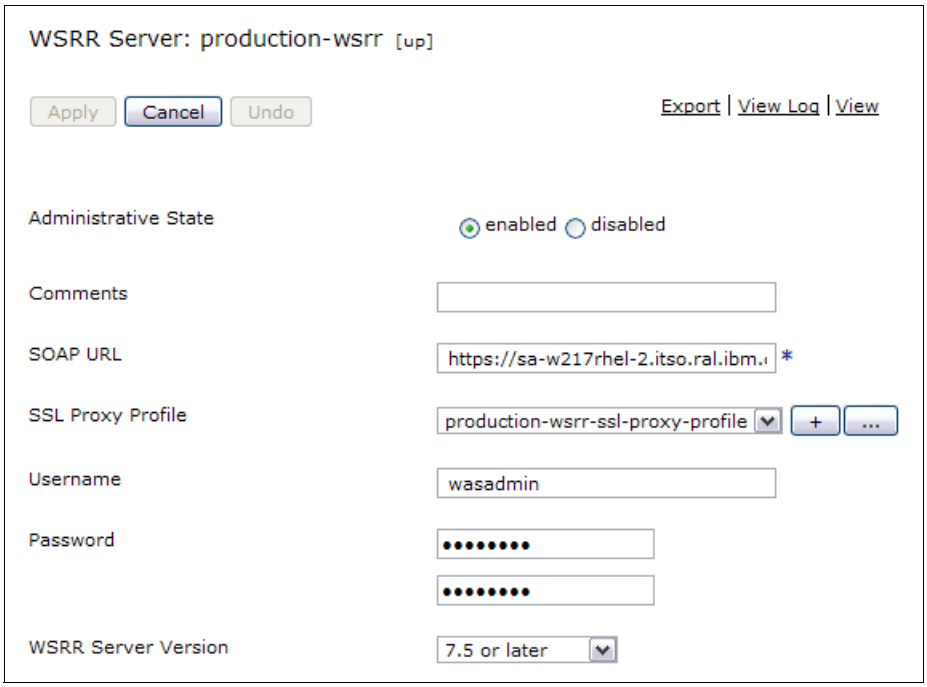


Figure 12-1 WSRR Server object

12.3 Web Service Proxy object

With DataPower Web Service Proxy service, users can proxy enterprise applications quickly and easily by simply providing the service with a WSDL.

In this scenario, Web Service Proxy service obtains the necessary WSDL files and policy attachments from a remote WSRR server. To arrive at a fully functional proxy, only a small number of the many configuration inputs must be set. For information about the full range of capabilities of this service, see the DataPower 5.0 information center:

http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/index.jsp

This section addresses the inputs that are described in Table 12-2.

Table 12-2 Web Service Proxy inputs

Input	Values
Web Service Proxy Name	`ITSORedbooksTravel_WSP` This value is an arbitrary name. Choose a descriptive name for easy reference later.
WSRR Server	`production-wsrr` If the server you want is not listed, click the plus icon (**+**) to create a new WSRR Server object.
Saved Search Name	`ITSORedbooksTravel_WSDLs` This value specifies the name of an existing saved search on the WSRR server that returns the subscribed-to resource. Click **Find a Saved Search** to see a list of all saved searches available from the selected WSRR server.

Saved Search Parameters	`None`
	
	This value specifies the parameters for the saved search that requires parameters. The parameters are used in the query that is sent to the WSRR server. A parameter can be up to 255 characters in length. You can define a maximum of 9 parameters for a query. If a saved search requires query parameters, you must specify the required parameters. Do not define parameters for a saved search that does not require parameters.
	
	See the WebSphere Service Registry and Repository information center for information about creating a saved search with parameters or modifying an existing saved search to accept parameters:
	
	`http://pic.dhe.ibm.com/infocenter/sr/v8r0/index.jsp`

Synchronization Method	`Automatic` (WSRR 7.5 or later) This selection causes the DataPower device to listen for notifications from the selected WSRR server that the subscription artifacts (such as a WSDL) changed. When DataPower receives this notice, it then issues a request for the new data. The polling method causes the DataPower device to issue requests to synchronize its version of the data with WSRR on a regularly scheduled basis. The manual method requires a user to click **Synchronize**, available on the WSRR Subscription Status panel, displayed by clicking **Okay** under WSDL Status. You can manually synchronize to the WSRR server at any time, regardless of the method selected here. This method is useful for rapid updates or debugging.

Input	Values
Fetch Policy Attachments	On If enabled, the subscription queries the registry for external WS-Policy PolicyAttachments that apply to the retrieved WSDLs. Retrieved policies will be processed by any Web Service Proxy that is using this subscription and is configured to allow external policy attachments.
Local Endpoint Handler	production-http-fsh Select the Front Side Handler to use to determine the IP address, port, and protocol. See 12.4, “Front Side Handler” on page 353 for configuration information.
All other inputs	Use the defaults for all other inputs. See the DataPower 5.0 information center: http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/index.jsp

Figure 12-2 shows an example of this configuration.

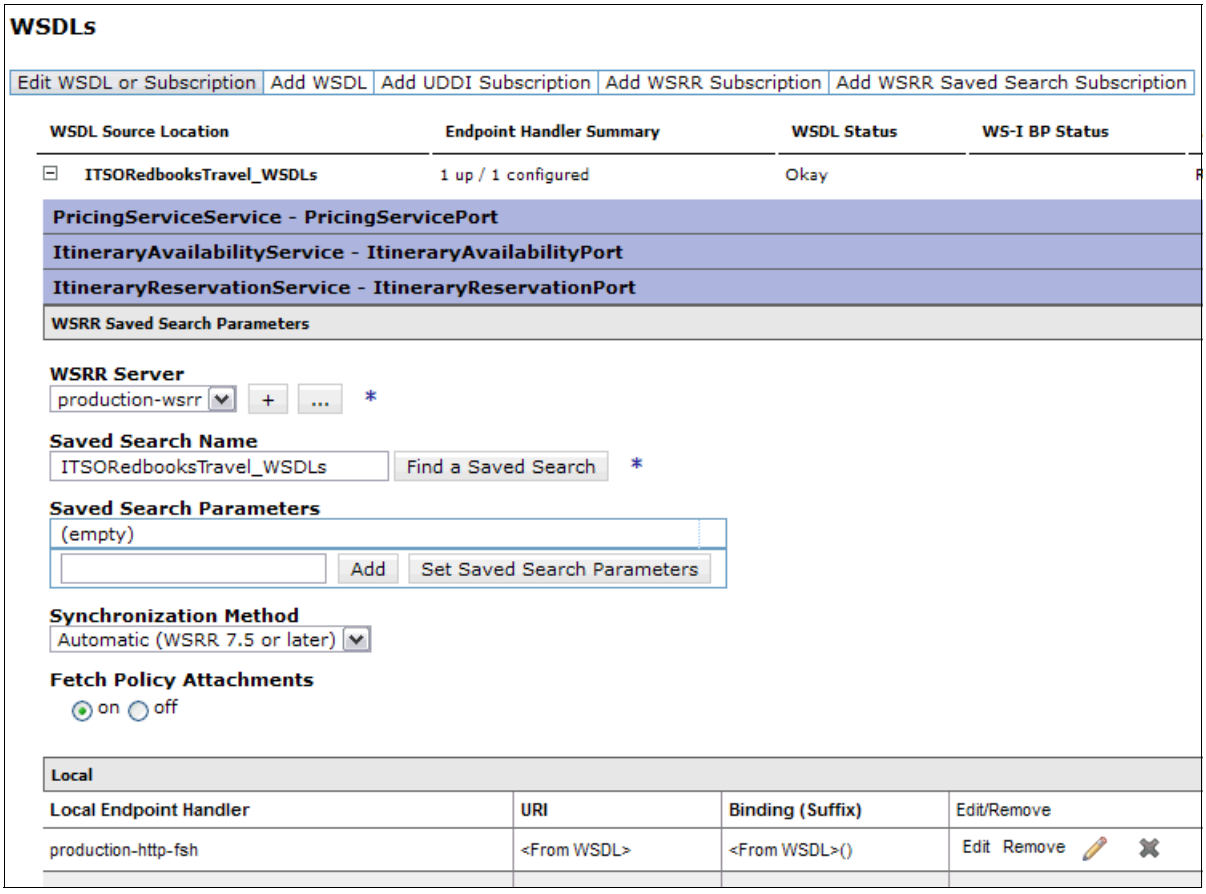


Figure 12-2 Web Service Proxy configuration

12.4 Front Side Handler

A Front Side Handler establishes network connections between the Web Service Proxy and any clients that requests services. The clients send requests to the IP address and port number shown in Figure 12-3 on page 354.

This section addresses the inputs that are described in Table 12-3.

Table 12-3 Front Side Handler

Input	Value
Name	`production-http-fsh` This value is an arbitrary name for the Handler. Choose a name that is easy to reference later.
Local IP Address	`DP.DATA` This address is where the service listens. The default of `0` (zero) indicates that the service is active on all addresses. Click **Select Alias** to use an alias for this value. Local host aliases help to ease migration tasks between machines.
Port Number	`2081` This integer (in the range 1 - 65535, with a default of 80) specifies the port monitored by the HTTP service. Numbers higher than 10,000 can conflict with ephemeral ports that are used by the device.
HTTP Version to Client	`HTTP 1.1` (default version) Select the HTTP version to use on the front (client) side connection. The default is HTTP 1.1. The device will fall back to version 1.0 if the client is unable to use 1.1.
Allowed Methods and Versions	Default and Get Specifies the allowed methods and versions for incoming HTTP requests. Be sure to select the **Get** check box. This method allows remote clients to retrieve the WSDL (or WSDLs) that describe the services that can be accessed through this port.
All Other Inputs	Default. See the DataPower 5.0 information center: `http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/index.jsp`

Figure 12-3 shows the configuration for the Front Side Handler.

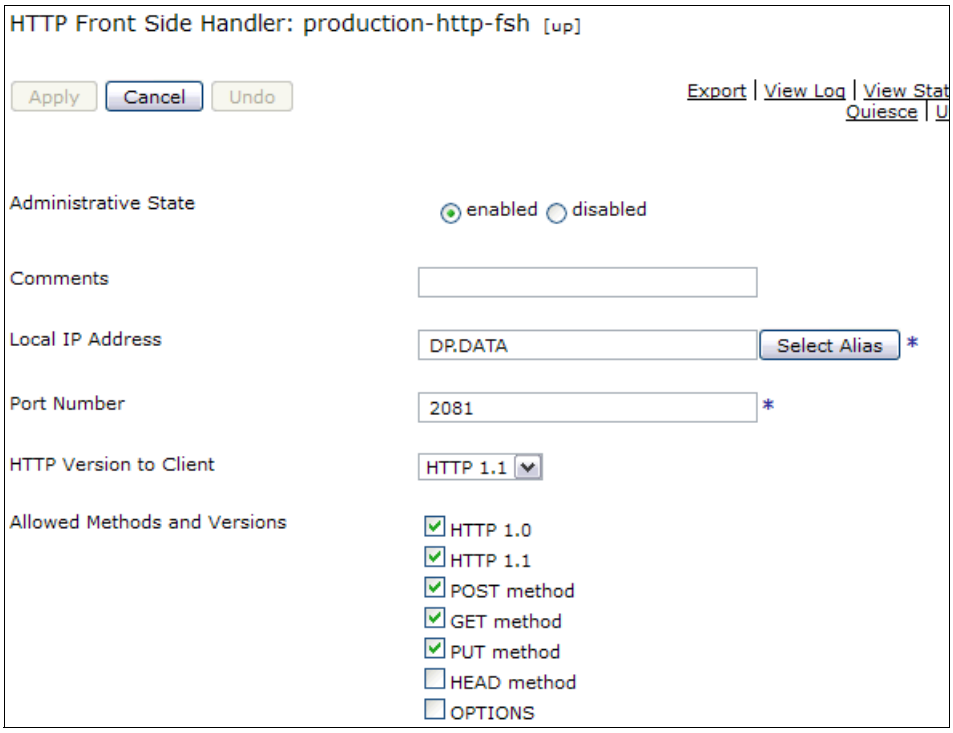


Figure 12-3 Front Side Handler

12.5 WSRR subscriptions and saved search subscription

A DataPower Web Service Proxy can obtain WSDL and other associated documents (such as schema files) by subscribing to those documents that are stored and managed on a WSRR server. DataPower supports two kinds of subscriptions: *direct reference* (WSRR subscription) and *indirect reference* (WSRR saved search subscription).

12.5.1 WSRR subscription

A WSRR subscription is a direct reference to the documents.

The configuration requires the reference to a WSRR server, the name of the WSDL or concept document, and its namespace. If there is more than one version of the service document, you must specify the version to reference.

Web Service Proxy services that are associated with the subscription are updated based on the synchronization method that is specified in the subscription. A Web Service Proxy stores a local copy of the subscribed object. Changes performed on the WSRR server to service document attributes such as the name of the WSDL or concept document, namespace, or version will require updating the WSRR subscription.

The WSRR Subscription requires a configured WSRR server with a version of 6.0 or later.

For an indirect reference to a set of WSDL files see WSRR Saved Search Subscription. This scenario employs an indirect reference to the necessary documents.

12.5.2 WSRR saved search subscription

This object provides the required configuration properties for a WSRR saved search subscription.

The saved search subscription is useful when you want to deploy web services with an indirect reference to a set of WSDL files required for Web Service Proxy operations. The configuration requires a reference to a WSRR server and the name of an existing saved search from the WSRR server. The management of the service documents is controlled from the WSRR server.

Web Service Proxy services, associated with the saved search subscription, are updated based on the synchronization method that is specified in the subscription. A Web Service Proxy virtualizes the web service endpoints, based on the WSDL files that are returned by executing the configured saved search.

This section addresses the inputs that are described in Table 12-4.

Note: These objects are displayed in line with the Web Service Proxy configuration page.

Table 12-4 WSRR Saved Search Subscription

Input	Value
WSRR Server	`production-wsrr` If the server you want is not listed, click the plus icon (+) to create a new WSRR Server object.
Saved Search Name	`ITSORedbooksTravel_WSDLs` Specifies the name of an existing saved search on the WSRR server that returns the subscribed-to resource. Click **Find a Saved Search** to see a list of all saved searches available from the selected WSRR server.
Saved Search Parameters	None Specifies the parameters for the saved search that requires parameters. The parameters are used in the query that is sent to the WSRR server. A parameter can be up to 255 characters in length. You can define a maximum of 9 parameters for a query. If a saved search requires query parameters, you must specify the required parameters. Do not define parameters for a saved search that does not require parameters. See the WebSphere Service Registry and Repository information center for information about creating a saved search with parameters or modifying an existing saved search to accept parameters: `http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/index.jsp`

Input	Value
Synchronization Method	`Automatic` (WSRR 7.5 or later) This selection causes the DataPower device to listen for notifications from the selected WSRR server that the subscription artifacts (such as a WSDL) changed. When DataPower receives this notice, it issues a request for the new data. The polling method causes the DataPower device to issue requests to synchronize its version of the data with WSRR on a regularly scheduled basis. The manual method requires a user to click **Synchronize**, available on the WSRR Subscription Status panel, which is displayed by clicking **Okay** under WSDL Status. Users can manually synchronize to the WSRR server at any time, regardless of the method chosen here. This method is useful for rapid updates or debugging.
Refresh Interval	Not used Specifies the refresh interval in seconds. Enter any value in the range 60 - 4294967. The default is 86400. Used only with the Poll method, the refresh interval is the interval in seconds between regularly-scheduled WSRR queries used to synchronize updates from the saved search result set stored on the remote WSRR server.
Fetch Policy Attachments	`On` If enabled, the subscription queries the registry for external WS-Policy PolicyAttachments that apply to the retrieved WSDLs. Retrieved policies will be processed by any Web Service Proxy that is using this subscription and is configured to allow external policy attachments.

12.6 Policy Parameter Set object

It is possible to attach a policy (WS-Policy, WS-SecurityPolicy, or WS-MediationPolicy) to a WSDL within the Web Service Proxy itself, rather than obtaining those policy files from a remote WSRR server.

When implementing SOA policy through the DataPower service, it is often necessary to provide additional information to the proxy so that it can enforce the policy. This additional information is provided through a Policy Parameter Set object.

The parameters that can be set vary depending upon the type of policy that you use. The complete list of parameters that apply to a WS-Mediation Policy are detailed in the developerWorks article named “SOA governance using WebSphere DataPower and WebSphere Service Registry and Repository, Part 1: Leveraging WS-MediationPolicy capabilities,” which is at the following location:

http://www.ibm.com/developerworks/websphere/library/techarticles/1204_burke/1204_burke.html

12.7 SLA policy details

The SLA Policy Details tab of the Web Service Proxy displays the current state of WS-MediationPolicy statement, which is in effect for the WSDLs that are used by the proxy.

Two kinds of rule sets can be attached to the WSDLs in use by the Web Service Proxy:

- WS-MediationPolicy rules, expressed as service level definitions (SLDs), and service level agreements (SLAs), can be attached to the WSDL at various points.
- Custom Policy rules, expressed as DataPower rules, can also be attached to the WSDL.

This tab displays both kinds of rules in operation.

12.7.1 SLD Definition Files

Any applicable SLDs are listed under the SLD Definitions section of the page. This section lists the policy files that are attached to each WSDL file.

The SLA Definition Files table lists the WSDL files that the service uses. Expand a WSDL file in the Document Name column to list the policies that are attached to the WSDL file. See Figure 12-4.

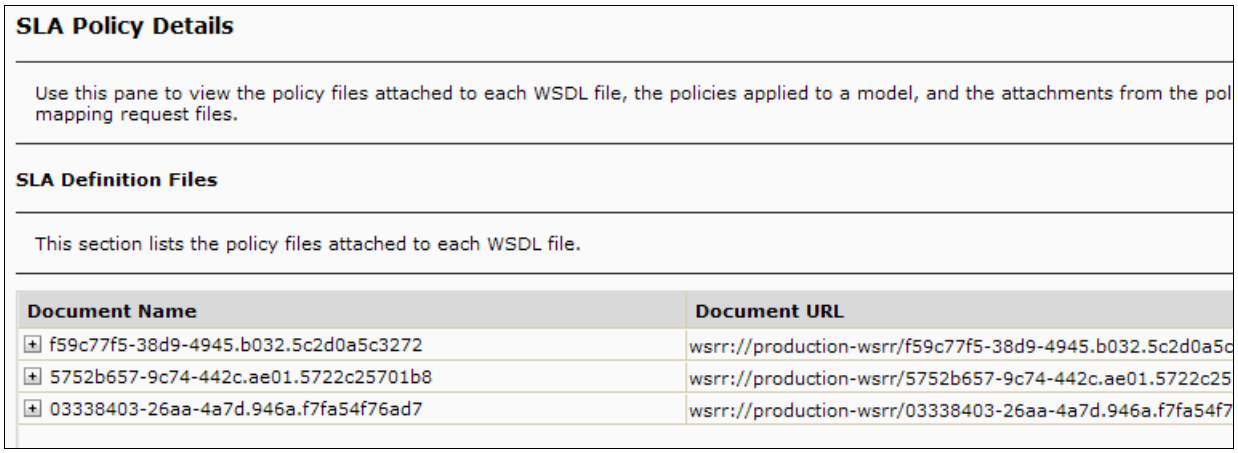


Figure 12-4 SLD definitions

A policy is identified with the policy name and ID in the Document Name column:

- If the policy name does not exist, the column contains the text `no name`.
- If the ID does not exist, the column contains the text `no ID`.

The location and file name for the attached policy are in the Document URL column.

Click the plus sign (+) beside any listed SLD file to see details about that SLD. See Figure 12-5. These details identify the document as stored on WSRR.

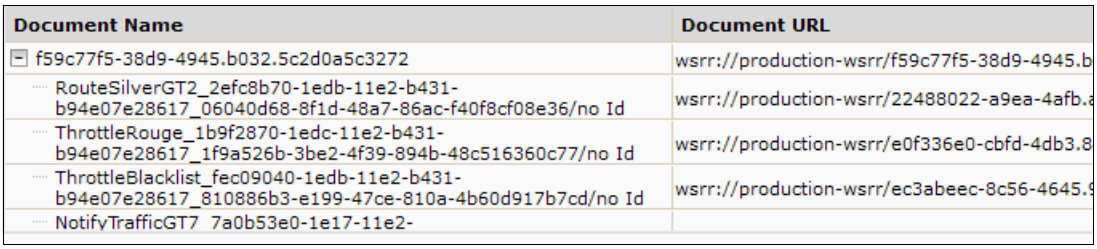


Figure 12-5 SLD expanded definitions

12.7.2 SLA policy

This section lists the three main components of the SLA Table: filter, tree, and attachments table. See Figure 12-6.

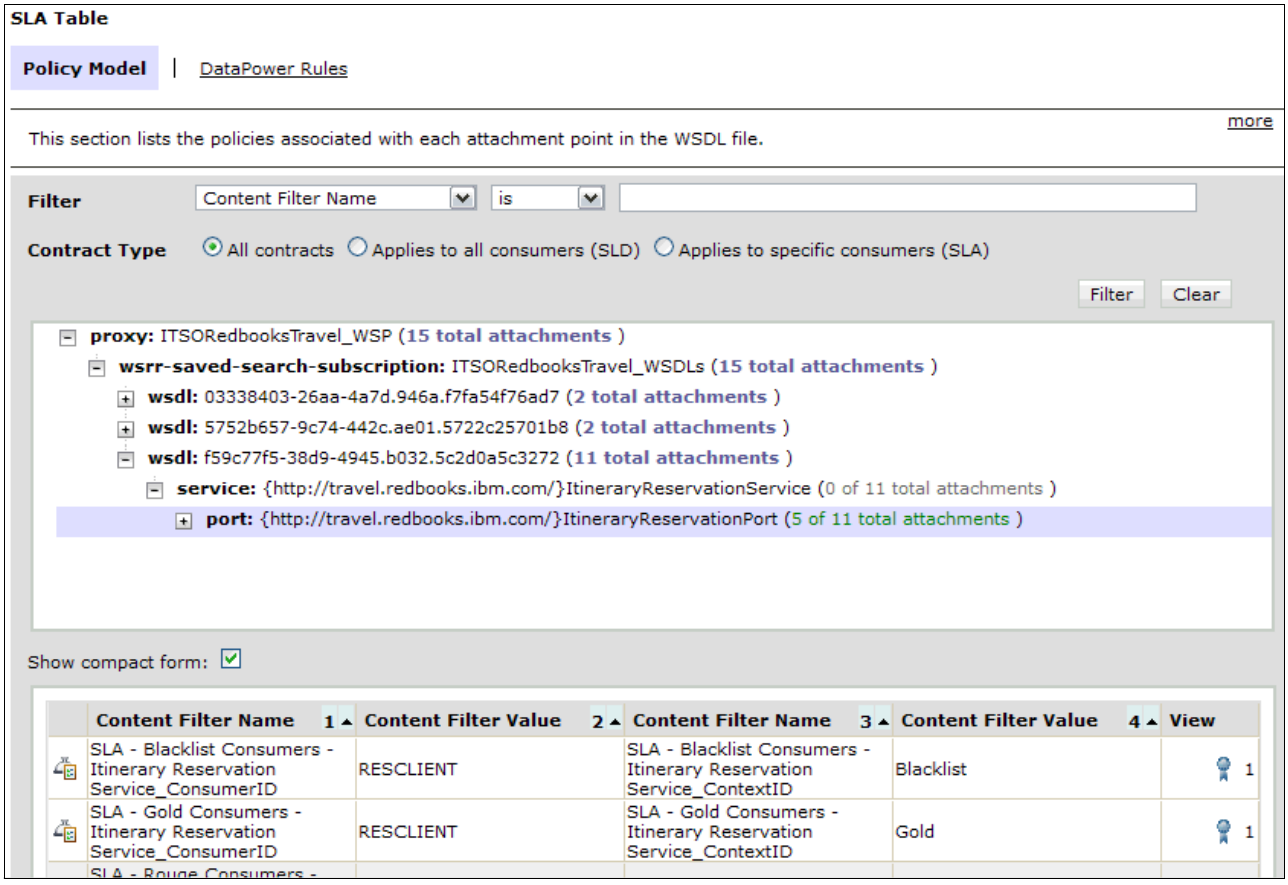


Figure 12-6 SLA Policy expanded

The filter

Use the filter to list a specific set of attachments. You can filter by content filter name, content filter number, or policy name. You can use the Contract Type option to run the filter against all policies, on SLAs only, or on SLDs only.

The tree view

The tree view displays a level for each policy attachment point in WSDL files. Each level of the tree shows how many applied attachments there are for that level. Click a level to display attachments for that level in the attachment table.

The attachment table

The attachment table displays the attachments for a specific attachment point in a WSDL file. Click an attachment to display the attachment policy. Click the same attachment again to display the cached value (the information is not updated). To update the entry, select or clear the **Show compact form** check box.

The first column in the table has an icon that indicates the type of policy:

- Default SLA indicates that the attachment is the default SLA policy. The default SLA policy is used when there are no explicit SLA rules matched to a transaction.
- SLA attachment indicates that the attachment is an SLA policy.
- SLD attachment indicates that the attachment is a service level definition (SLD).

The next four columns contain the content filter names and content filter values associated with the attachment. You can use these values in the filter.

The View column shows an icon and a number indicating the number of policies that meet the criteria. Place the cursor over the icon to see the policy name, ID, and URL.

12.7.3 DataPower Rules grid

This section lists the auto-generated DataPower rules for policy attachments. It has three main components: filter, tree, and attachments table. See Figure 12-7.

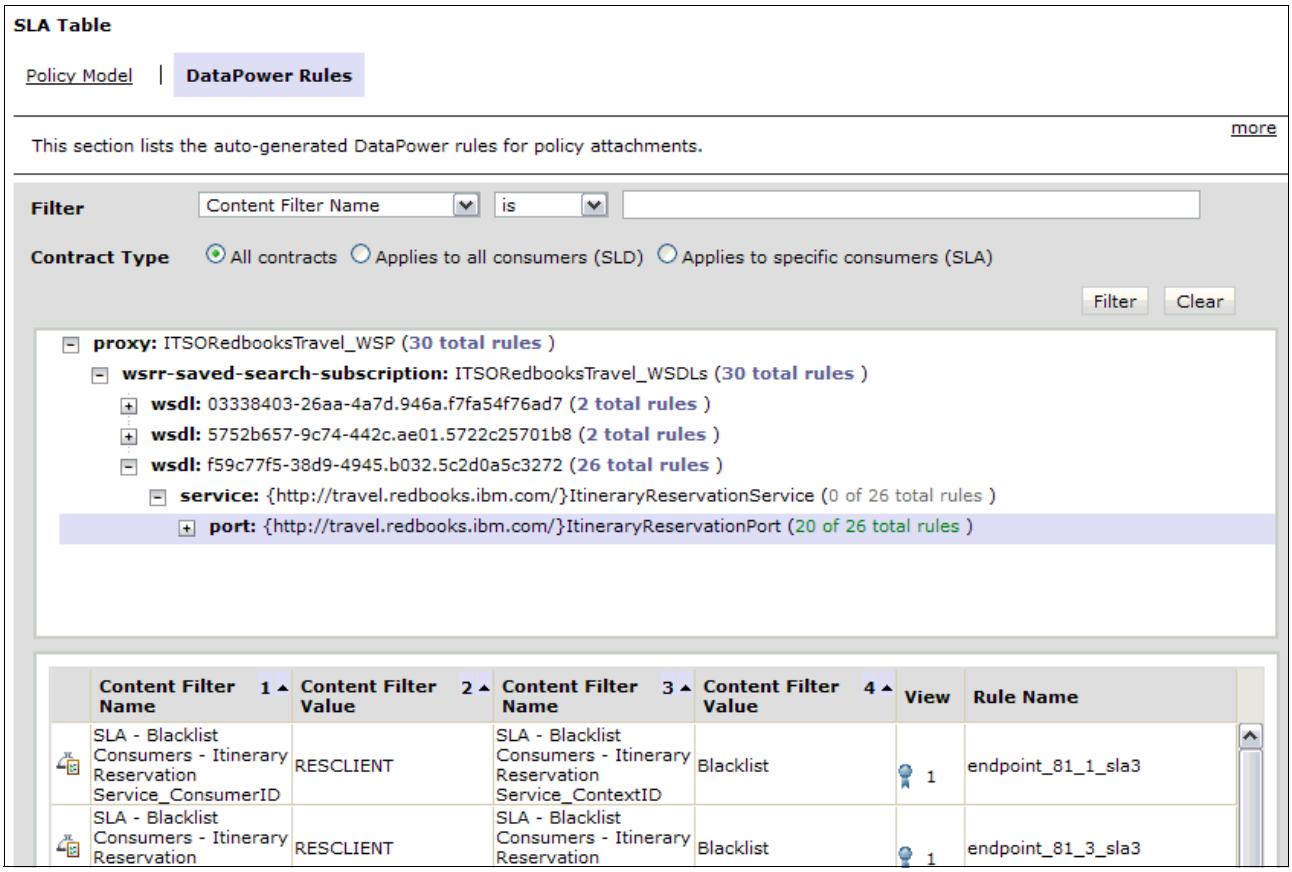


Figure 12-7 DataPower Rules

The filter

Use the filter to list a specific set of attachments. You can filter by content filter name, content filter number, or policy name. Use the Contract type option to run the filter against all policies, on SLAs only, or on SLDs only.

The tree view

The tree view displays a level for each policy attachment point in WSDL files. Each level of the tree shows how many applied attachments exist for that level. Click a level to display attachments for that level in the attachment table.

The attachment table

The attachment table displays the attachments for a specific attachment point in a WSDL file. Click an attachment to display the attachment policy. Click the same attachment again to display the cached value (the information is not updated). To update the entry, select or clear the **Show compact form** check box.

The first column in the table has an icon indicating the type of policy:

- Default SLA indicates that the attachment is the default SLA policy. The default SLA policy is used when there are no explicit SLA rules matched to a transaction.
- SLA attachment indicates that the attachment is an SLA policy.
- SLD attachment indicates that the attachment is a service level definition (SLD).

The next four columns contain the content filter names and content filter values that are associated with the attachment. You can use these values in the filter.

The View column shows an icon and a number that indicates how many policies meet the criteria. Place the cursor over the icon to see the policy name, ID, and URL.

The rules table

The rules table displays the rules for a specific attachment point in a WSDL file. Click a row to display the auto-generated request rule, response rule, and error rule. The actions in the rules use the same icons as the policy editor. To view the properties for an action, place the cursor over the action.

When no rule configuration is found, the `No rules defined` message is displayed in place of the rule and its actions. See Figure 12-8.

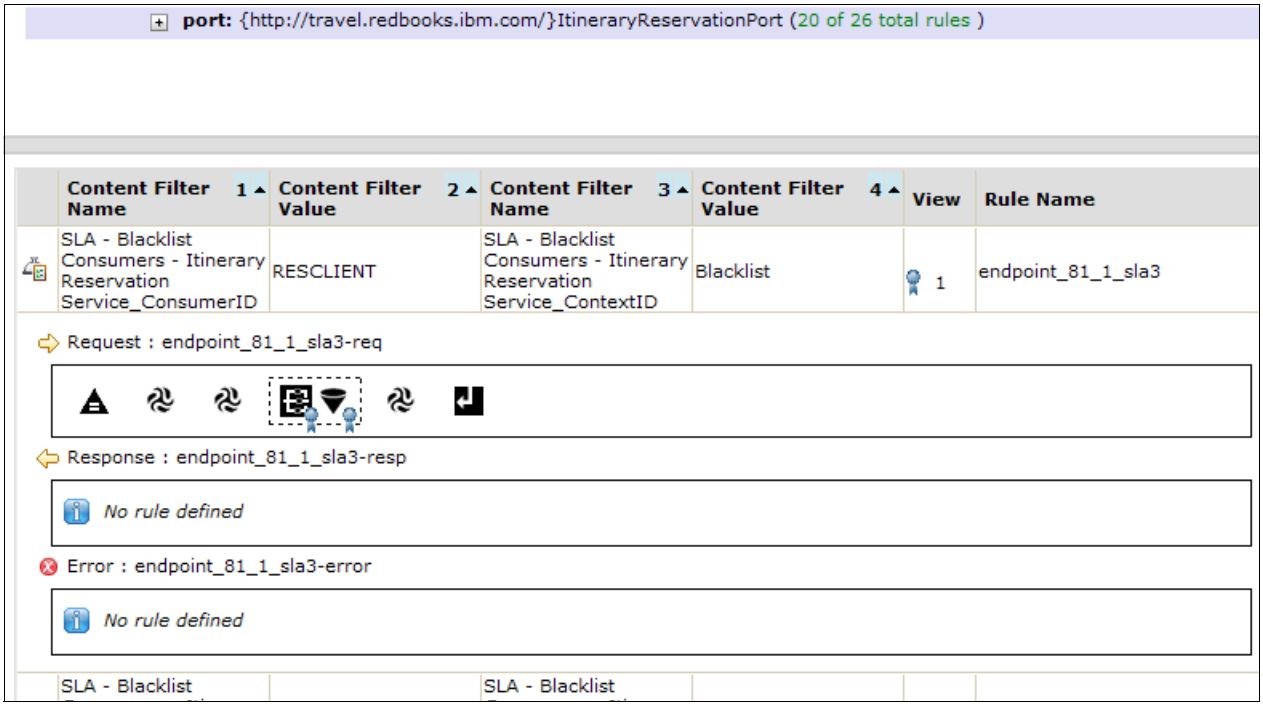


Figure 12-8 DataPower rule expanded

12.8 Scope of a policy on one or more DataPower appliances

SOA and SLA policies can be enforced by one or more DataPower devices simultaneously.

12.8.1 Applying policy to a single appliance

The previous sections in this chapter describe how to apply a policy on one device.

12.8.2 Applying policy to several appliances

The same policies can be applied to multiple DataPower devices in one of two ways:

- A single DataPower configuration pattern can be deployed to multiple devices, including the enforcement policies. A management console, such as WebSphere Appliance Management Center can be used to distribute the common configuration.
- Multiple DataPower devices can obtain configuration, including policies, from a common WSRR server.

When more than one DataPower device enforces a single SLA, administrators can choose to share real-time transaction information between the devices. See 12.8.3, “Applying policy to a DataPower cluster” on page 362 for more information about this capability.

12.8.3 Applying policy to a DataPower cluster

As usage traffic grows, more than one DataPower device may be needed to maintain the desired levels of response times, or throughput. In this case, enterprises use a cluster of DataPower devices that are sharing a load and distributing requests to back end applications. In this case, the DataPower devices must keep common statistics on items such as number of transactions per second or latency times. Otherwise, device A might not provide the same enforcement as device B, while traffic is distributed among devices.

DataPower solves this problem by using what is called SLM Peer Groups. Devices are joined into groups that share transaction statistics in real time.

Use the following steps to join devices into groups:

1. Log in to the default domain of the DataPower device.
2. Type the string `XML Management` in the search box and select **XML Management Interface** from the results. The XML Management Interface configuration panel opens (Figure 12-9 on page 363).

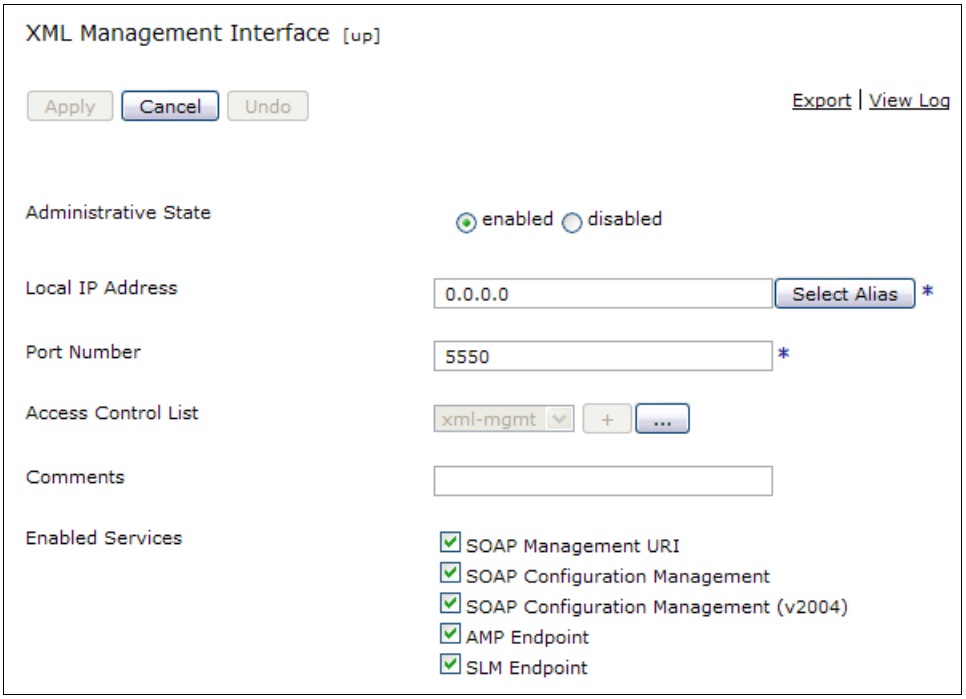


Figure 12-9 SLM Endpoint checked

3. Select the **SLM Endpoint** check box in the Enabled Services list.
4. Click **SLM** on the menu bar.
5. Set the interval for distributing SLM-related data to other members of the peer group. The default is every 10 seconds. The smallest available interval is 1 second. See Figure 12-10.

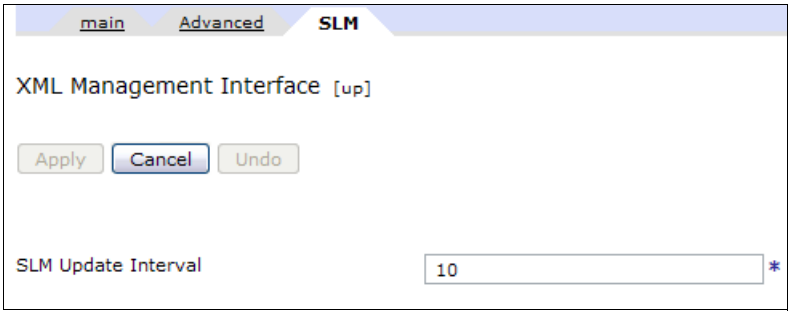


Figure 12-10 SLM Update Interval

6. Click **Apply**.

7. Type the string `Peer` in the search box. Select **Peer Group** from the results. The Peer Group configuration panel opens. This is where you add the other DataPower devices included in the cluster. See Figure 12-11.

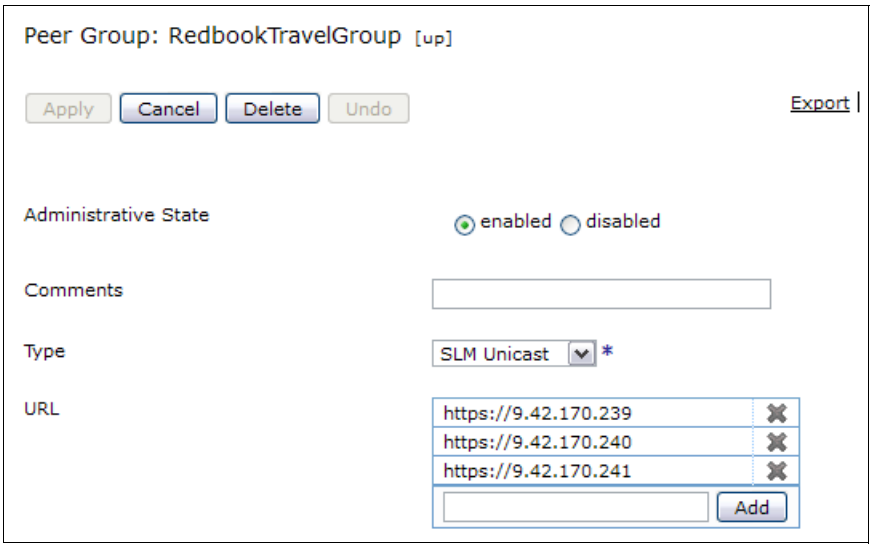


Figure 12-11 Peer Group configuration

SSL Proxy Profile: Sharing is done with SSL-enabled connections (HTTPS). You must establish an SSL Proxy Profile to handle these connections.

The SSL Proxy Profile that is used to enable these connections must be attached to the configuration of the XML Manager in use by the Web Service Proxy service. See the DataPower information center for instructions of how to create an SSL Proxy Profile and attach it to the XML Manager that is used by the Web Service Proxy:

`http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/index.jsp`

8. Create a Processing Parameter that identifies the Peer Group object and attach it to the Web Service Proxy in use. Select the **Policy** tab of the Web Service Proxy. Under the Proxy level of the Policy tree, click **WS-Policy**, as shown in Figure 12-12.

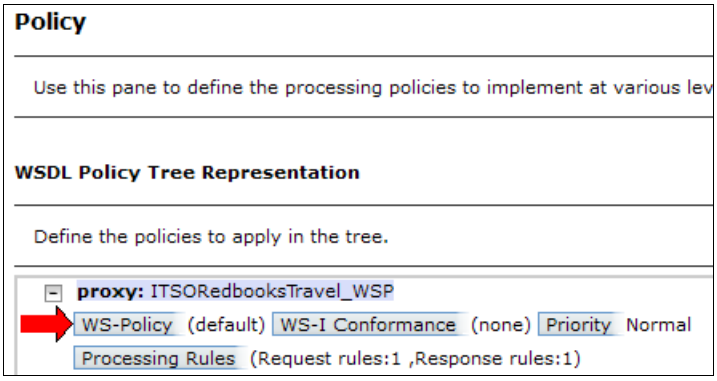


Figure 12-12 WS-Policy tab

9. When the WS-Policy tab is open, create a parameter object similar to the one in Figure 12-13. This parameter points to the Peer Group that was created earlier.

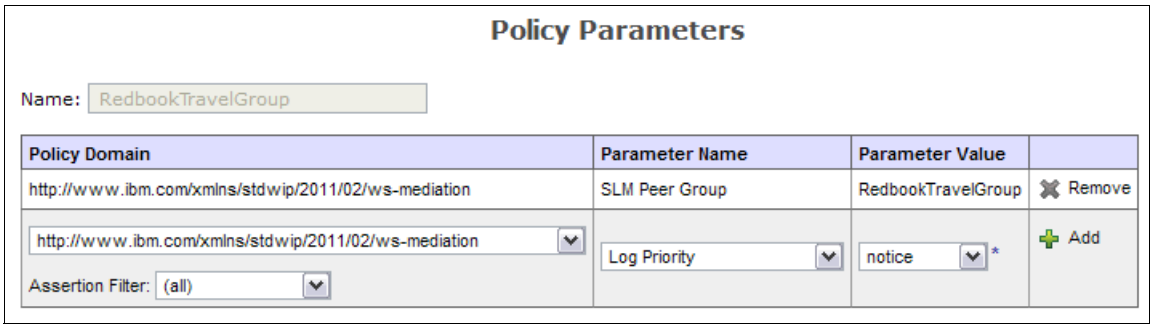


Figure 12-13 Policy Parameters

After these parameters are saved, you may attach them to the Proxy policy. Figure 12-14 shows this attachment.

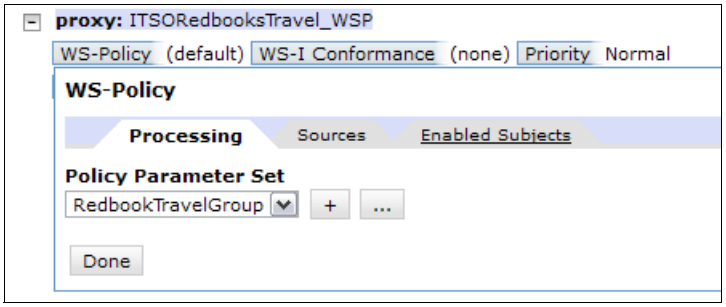


Figure 12-14 Completed WS-Policy tab

10. Click **Done** to apply the changes to the Web Service Proxy object and save the configuration. The cluster members now share common transaction statistics.

13

Creating and using custom policies

This chapter serves as a reference for the creation and use of *custom policies*. Custom policies might be necessary when the standard mediation policy capabilities that are provided in WebSphere Service Registry and Repository (WSRR) do not fulfil specific requirements.

This chapter contains the following topics:

- 13.1, “When to use a custom policy” on page 368
- 13.2, “Creating a custom policy” on page 368
- 13.3, “Importing a custom policy in WSRR” on page 382
- 13.4, “Custom policy attachment in WSRR” on page 383
- 13.5, “Importing a custom policy in DataPower” on page 385
- 13.6, “Custom policy attachment in DataPower” on page 386
- 13.7, “Custom policy enforcement” on page 388

© Copyright IBM Corp. 2013. All rights reserved.

13.1 When to use a custom policy

When policy capabilities that are provided in WSRR do not fulfill specific needs, you can implement a custom policy. Before you determining whether you need a custom policy, you need to understand what type of policy actions WSRR provides. If you can create your policy using only WSRR for the policy authoring, do that instead of using a custom policy. See 8.2, “Creating a mediation policy” on page 264 for detailed information

Tip: DataPower 5.0.0 and later provides a set of WS-MediationPolicy policies that can be used as examples to create custom mediation policies. These policies are located in the `default` domain, in the `store:///policies/templates` directory.

Future releases of WSRR will expand the WSRR policy capabilities that are not supported as ready to use. If you need to do a task with a policy that DataPower can do, but WSRR cannot, that is the perfect time to consider using a custom policy.

Consider the following examples for which a custom policy must be created:

- Security enforcement: Execute a specific authentication, authorization, and audit (AAA) action into a mediation policy.
- Security zone sign-on transformations: Use SAML (Security Assertion Markup Language) to translate security sign-on for different security zones.
- Versioning: Implement versioning to perform data mediations, validations and routing on both request and response.
- Dynamic routing: Need for dynamic content routing based on an XML parameter file or any type of routing map.
- Caching: Cache the response of a service. The policy can define the caching key calculation and also the time to live (TTL) of the caching elements.

13.2 Creating a custom policy

To create a custom policy, you must first define a domain vocabulary that allows companies to declare their own policies *assertions*.

These assertions map to custom DataPower configuration artifacts and are applied either in WSRR or in DataPower, to specified consumers at the service level agreement (SLA) level, or to service providers at the service level definition (SLD) level.

Important: After a custom policy is created, two possible options can enforce it:

- The custom policy is loaded in WSRR. It is governed and attached to an SLA or SLD in WSRR in the same manner as a mediation policy, authored in WSRR, to be retrieved through a subscription and enforced on DataPower at run time. This approach is best because it allows users to centrally manage and govern the use of the custom policy with various services.
- The custom policy is loaded directly in DataPower. It is attached at the service, port, portType, or binding levels of a WSDL file and may be associated to a specific set of consumers.

The mapping from policy assertions to configuration artifacts is performed by using an XSL style sheet, which is located on the DataPower appliance.

13.2.1 Custom policy governed in WSRR

If the custom policy is located in WSRR, then the mapping is executed after a subscription is configured between DataPower and WSRR.

After DataPower establishes a subscription with WSRR, it can retrieve WSDL files and also mediation and custom policies and policy attachments. These custom policies are then transformed into DataPower configuration artifacts. These configuration artifacts are used to enforce traffic at run time. Each of these concepts is explained in more detail in the next sections.

13.2.2 Custom policy loaded in DataPower

If the custom policy resides in DataPower, then the mapping is executed after the custom policy is attached to a WSDL file at the service, port, portType, or binding level. These configuration artifacts are used to enforce traffic at run time. Each of these concepts is explained in more detail in the next sections.

13.2.3 Actors and components

This section presents the actors and components involved in a custom policy creation process for the IBM Redbooks Travel Company. Your organization might have a different set of actors.

A custom policy creation process typically includes the following *actors*:

- Policy architect
- DataPower developer
- DevOps team

To create, administer, and enforce a custom policy requires the following *components*:

- An integrated development environment (IDE) or an XML editor to create the custom policy domain vocabulary and the custom XSL stylesheet policy. This XSL transformation contains the mapping from custom policy assertions to DataPower configuration artifacts.
- A WSRR server, on which the custom policy (an XML file) may be loaded, administred, and attached. An alternative is to load the custom policy directly on the DataPower appliance.
- A DataPower appliance, on which the XSL style sheet has to be loaded. The custom policy is enforced on the DataPöwer appliance.

Figure 13-1 illustrates the required components and actors that are involved to create, administer, attach, and enforce a custom policy.

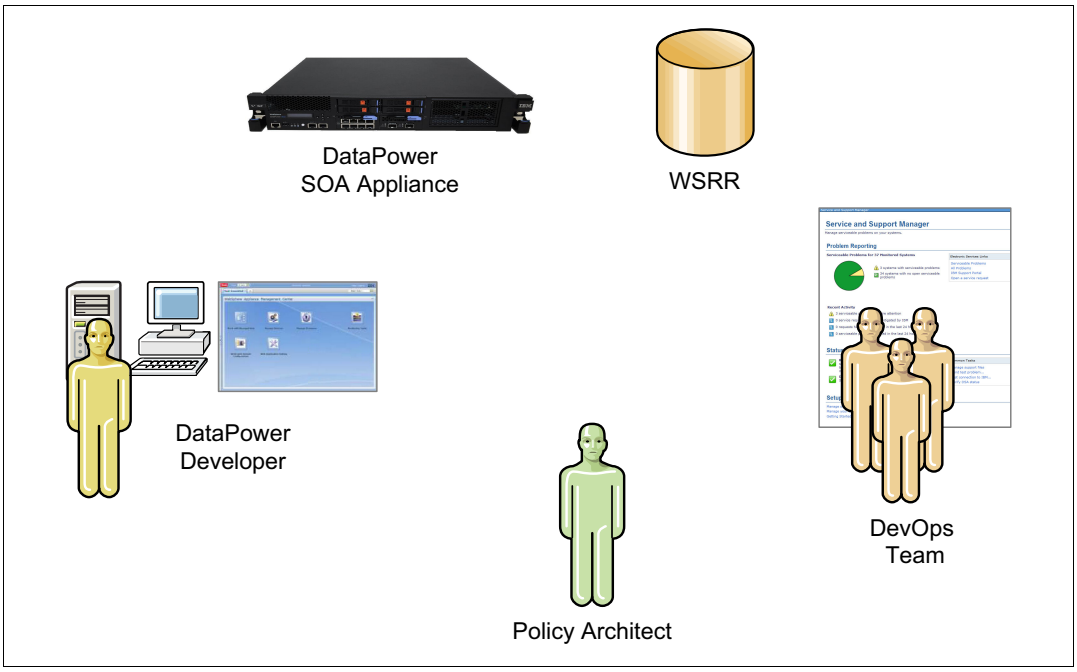


Figure 13-1 Actors and components involved in a custom policy creation

13.2.4 Steps to create a custom policy

After expressing the business need for a policy, the ordered steps to create a custom policy are shown in Figure 13-2. Anyone who creates a custom policy, which must be governed in WSRR, can follow these exact steps.

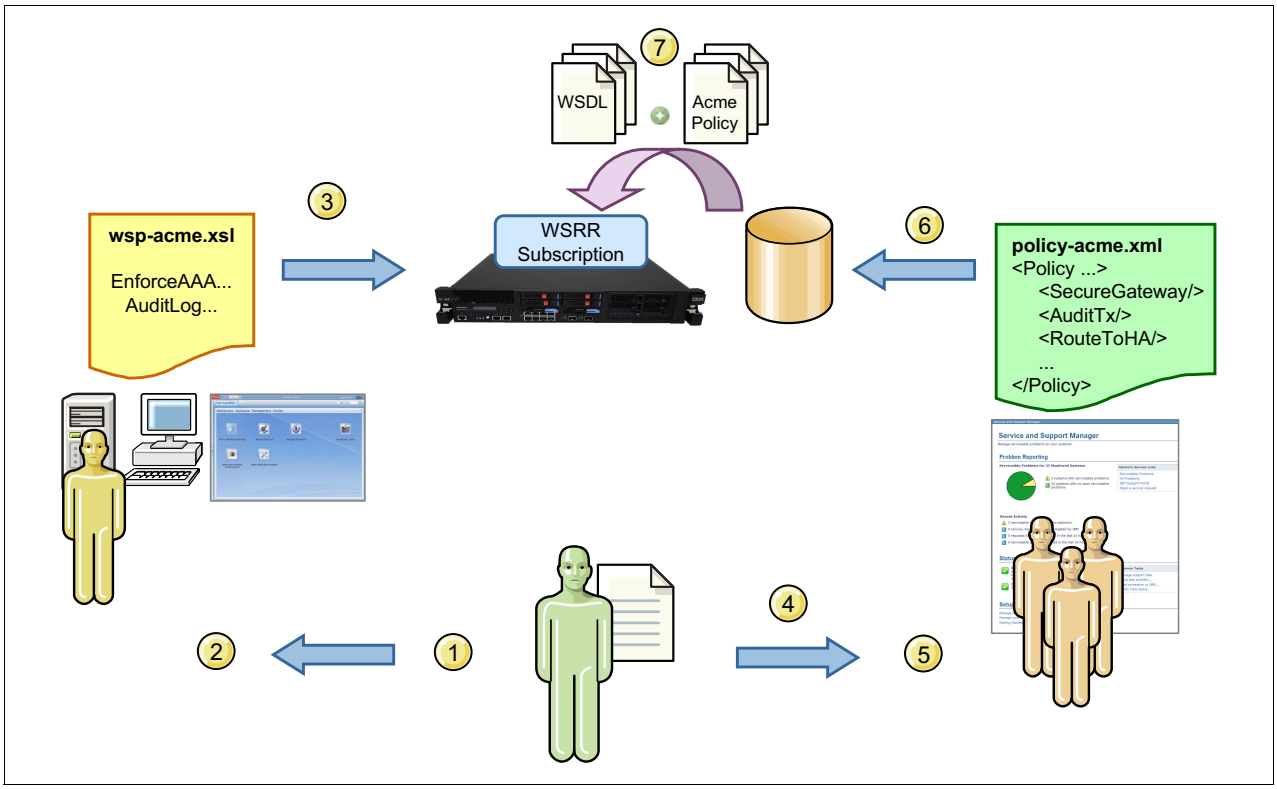


Figure 13-2 Steps to create a custom policy

To create a custom policy, the actors involved complete the following tasks as Figure 13-2 shows:

1. The Policy Architect creates a custom policy vocabulary based on the business needs.
2. The DataPower developer authors the custom policy style sheet (XSLT) as defined by the policy vocabulary.
3. The DataPower developer deploys the custom policy style sheet to the DataPower appliance.
4. The Policy Architect publishes the policy vocabulary specification for use by the DevOps team.
5. The DevOps team authors policy documents based on the custom policy assertions.
6. The DevOps team publishes, manages, and attaches the policies in WSRR.
7. The WSRR Subscription or WSRR Saved Search Subscription is synchronized to add the new policy for enforcement. This step can be executed automatically, using a notification mechanism between WSRR and DataPower.

Important: Create a custom policy if *and only if* a standard WSRR mediation policy cannot be used.

13.2.5 Custom policy domain

A *custom policy domain* consists of the following components:

- A dedicated namespace
- A vocabulary (or grammar) bound to this namespace

After you create a namespace and a vocabulary, you can implement a custom policy. This policy is based on XML. Therefore, you can generate an XML Schema Definition (XSD) of the custom policy. The XSD is useful if you want to do the additional step of validating usage of the custom policy. The XSD may also be used to generate documentation or user interfaces of the available policy assertions of a custom policy.

The vocabulary that is created must remain simple and easy to use for the person in charge of creating custom policies, typically a Policy Architect. A practice that works well is to generate documentation regarding the capabilities of a custom policy to help improve the simplicity of its implementation.

Examples of custom policy domain creation are in 5.5, “Creating custom policy domain and assertions for versioning” on page 134 and 6.3, “Creating custom policy domain and assertion for security” on page 194.

13.2.6 Custom policy XSL style sheet

An author of a custom policy must know several concepts about a custom policy XSL style sheet.

Introduction to custom policy style sheet

Use a custom policy style sheet to generate DataPower configuration artifacts based on custom policy assertions.

Figure 13-3 on page 373 presents the creation process of the DataPower configuration artifacts. The consumption of custom policies is supported only on DataPower Web-Service proxies.

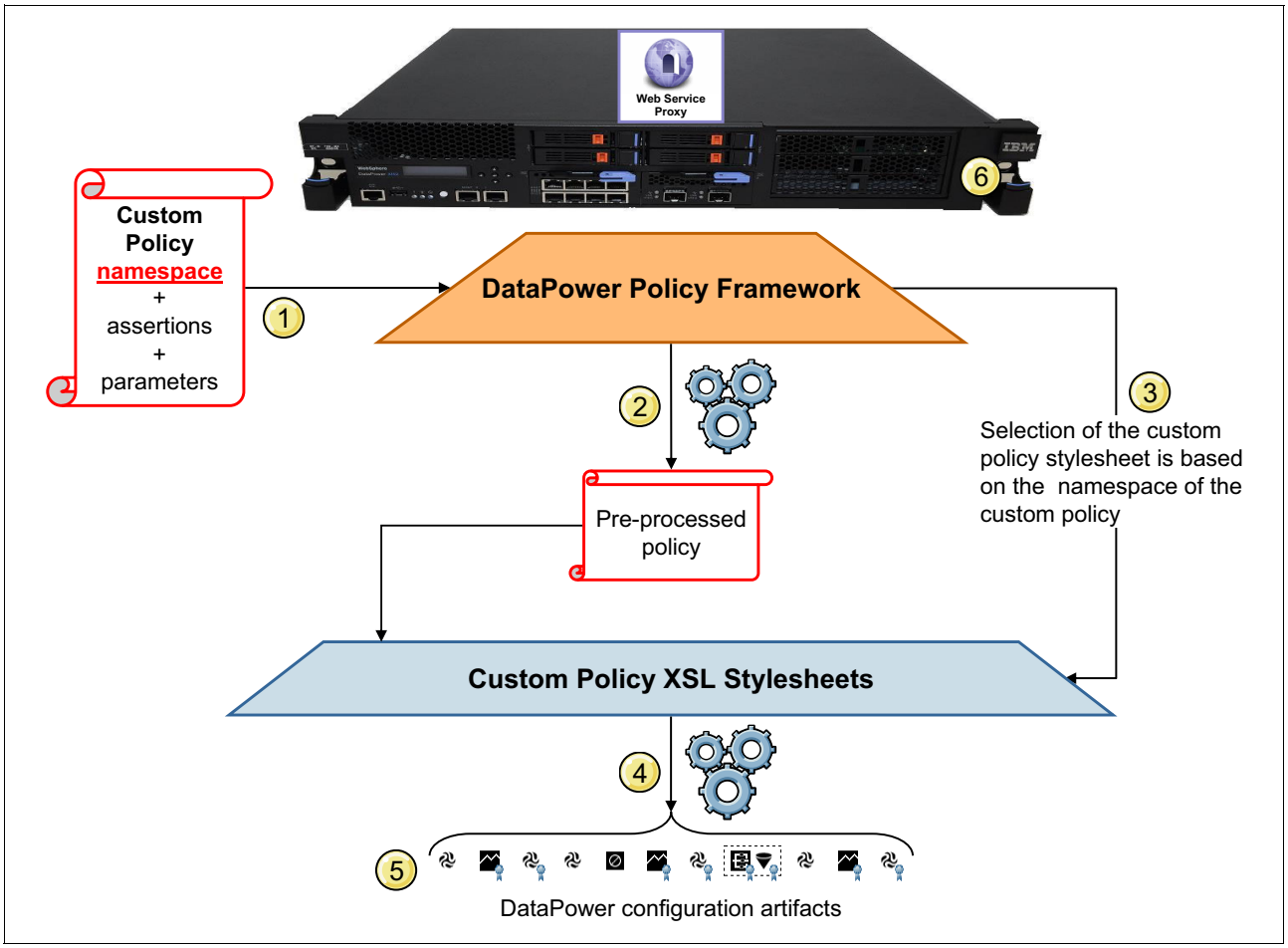


Figure 13-3 From custom policy to DataPower configuration artifacts

The following steps are run during the DataPower configuration artifacts creation process:

1. A custom policy is created. This policy can be governed in WSRR or can be loaded on a DataPower SOA appliance. If it is governed in WSRR, a subscription is required in DataPower to retrieve the WSDL files and also policies.
2. The DataPower policy framework as part of the SOA Policy Solution (introduced with DataPower 5.0.0) is responsible for transforming the custom policy into a DataPower internal pre-processed policy based on XML.
3. The DataPower policy framework is also responsible for selecting the correct custom policy XSL style sheet based on the custom policy namespace.
4. The custom policy style sheet is used to transform the pre-processed policy into a set of DataPower configuration artifacts.
5. The DataPower configuration artifacts are created on specific processing rules (request or response as defined in the custom policy style sheet). These processing rules are executed before the completion of processing rules defined on the `Policy` tab of a Web Service Proxy.

A custom policy stylesheet example is provided on DataPower appliances using firmware 5.0.0 or later. You can find this style sheet (`jk-example.xsl`) in the `default` domain in the `store:///policies/custom` directory.

In this chapter, an example of custom policy XSL style sheet is also provided. This style sheet is based on the `jk-example.xsl` file. Example 13-1 shows the `itso.customPolicy.example.xsl` file.

Example 13-1 itso.customPolicy.example.xsl

```
<?xml version="1.0"?>
<!-- +
     |*****************************************************
     |*** Author: ITSO - gauci@fr.ibm.com
     |*** file: itso.customPolicy.example.xsl
     |*** Description: Custom policy XSL styleheet example
     |*** Revision: 1.0: Initial version
     |*****************************************************
    +-->

    <xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:func="http://exslt.org/functions"
        xmlns:dpe="http://www.datapower.com/extensions"
        xmlns:dppolicy="http://www.datapower.com/policy"
        xmlns:dpconfig="http://www.datapower.com/param/config"
        xmlns:dpfunc="http://www.datapower.com/extensions/functions"
        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:myNamespace="http://itso.redbooks.ibm.com/myNamespace"

        extension-element-prefixes="dpe"
        exclude-result-prefixes="dpe dpconfig dppolicy wsp myNamespace">

        <!-- ******************************************************* -->
        <!-- (1) Declare the policy domain this stylesheet implements -->
        <!-- ******************************************************* -->

        <dpe:summary xmlns="">
            <dppolicy:domain>http://itso.redbooks.ibm.com/myNamespace</dppolicy:domain>
            <operation>xform</operation>
            <description></description>
        </dpe:summary>

    <!-- the policy domain namespace the stylesheet is executed for -->
    <xsl:variable name="seqno"
select="/dppolicy:request/dppolicy:header/dppolicy:SequenceNo"/>
    <xsl:variable name="nsuri"
select="/dppolicy:request/dppolicy:sequence/DomainNamespace[position()=$seqno]/@uri"/>

    <!-- The following global variables represent the input document -->
    <!-- header with aux information -->
    <xsl:variable name="header" select="/dppolicy:request/dppolicy:header"/>
    <!-- configured policy bindings defined as dpe:param above (Policy Parameters)-->
    <xsl:variable name="bindings" select="/dppolicy:request/dppolicy:bindings"/>
    <!-- ws-policy alternative -->
    <xsl:variable name="policy" select="/dppolicy:request/dppolicy:policy"/>
    <!-- previously generated configuration for this policy alternative -->
    <xsl:variable name="configuration" select="/dppolicy:request/dppolicy:configuration"/>
    <!-- general notepad to pass information between processing steps of one alternative
-->
    <xsl:variable name="notepad" select="/dppolicy:request/dppolicy:notepad"/>

    <!-- helper variables -->
    <xsl:variable name="rD" select="$header/dppolicy:RequestDomain"/>
    <xsl:variable name="lT" select="$header/dppolicy:LogType"/>
```

```
    <xsl:variable name="lC" select="$header/dppolicy:LogClass"/>
    <xsl:variable name="rO" select="$header/dppolicy:RequestObject"/>

    <!--+
       |*****************************
       |*** Matching Template
       |*** Element: ROOT
       |*****************************
       +-->
    <xsl:template match="/">

    <!-- Log -->
      <xsl:message dpe:priority="error">Entering the root template</xsl:message>

        <!-- Get the Input context name -->
        <xsl:choose>
            <xsl:when test="string-length($notepad/shared-context/in-context) > 0">
                <dpe:set-local-variable name="'input-context'"
value="$notepad/shared-context/in-context"/>
                <xsl:message dpe:priority="debug" dpe:domain="{$rD}" dpe:type="{$lT}"
dpe:class="{$lC}" dpe:object="{$rO}">Input context=<xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:message>
            </xsl:when>
            <xsl:otherwise>
                <xsl:message terminate="yes" dpe:priority="error" dpe:domain="{$rD}"
dpe:type="{$lT}" dpe:class="{$lC}" dpe:object="{$rO}">Cannot find the input
context</xsl:message>
            </xsl:otherwise>
        </xsl:choose>

        <!-- Get the Output context name -->
        <xsl:choose>
            <xsl:when test="string-length($notepad/shared-context/out-context) > 0">
                <dpe:set-local-variable name="'output-context'"
value="$notepad/shared-context/out-context"/>
                <xsl:message dpe:priority="debug" dpe:domain="{$rD}" dpe:type="{$lT}"
dpe:class="{$lC}" dpe:object="{$rO}">Output context=<xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:message>
            </xsl:when>
            <xsl:otherwise>
                <xsl:message terminate="yes" dpe:priority="error" dpe:domain="{$rD}"
dpe:type="{$lT}" dpe:class="{$lC}" dpe:object="{$rO}">Cannot find the output
context</xsl:message>
            </xsl:otherwise>
        </xsl:choose>

         <!-- process single alternative -->
         <xsl:apply-templates select="$policy/*[local-name()='All']"/>

    </xsl:template>

   <!--+
       |*****************************
       |*** Matching Template
       |*** Element: All
       |*****************************
       +-->
   <!-- process each assertion in the alternative -->
    <xsl:template match="*[local-name()='All']">
```

```
        <xsl:apply-templates mode="assertion"/>

    </xsl:template>

     <!-- my domain assertions -->

    <!--+
       |*****************************
       |*** Matching Template
       |*** Element: myNamespace:MyAssertion
       |*** Mode: assertion
       |*****************************
      +-->
    <xsl:template mode="assertion" match="myNamespace:MyAssertion">

        <!-- Log -->
        <xsl:message dpe:priority="error">Enter custom policy template - MyAssertion:
PolicyID=<xsl:value-of select="$header/dppolicy:PolicyID"/></xsl:message>

        <!-- ********************************************************************* -->
        <!-- (5) The configuration for assertion MyAssertion                -->
        <!-- ********************************************************************* -->
        <!-- generate processing action to execute (assertionNo = order of processing) -->
        <xsl:variable name="config">
            <dppolicy:config uri="{$nsuri}" assertionNo="{position()}"
direction="{'request-rule'}">

                <!-- Create StylePolicyAction element here. -->

                <!-- Identity transformation example -->
                <xsl:variable name="actionNameXForm"
select="concat($header/dppolicy:PolicyID,'-xform-', position(),'-Example')"/>
                <xsl:element name="StylePolicyAction">
                    <xsl:attribute name="name"><xsl:value-of
select="$actionNameXForm"/></xsl:attribute>
                    <xsl:element name="Type"><xsl:text>xform</xsl:text></xsl:element>
                    <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                    <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                    <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                    <xsl:element
name="Transform"><xsl:text>store:///identity.xsl</xsl:text></xsl:element>
                    <xsl:element
name="OutputType"><xsl:text>default</xsl:text></xsl:element>
                </xsl:element>

                <!-- Validate action example -->
                <xsl:variable name="actionNameValidate"
select="concat($header/dppolicy:PolicyID,'-validate-', position(),'-Example')"/>
                <xsl:element name="StylePolicyAction">
                    <xsl:attribute name="name"><xsl:value-of
select="$actionNameValidate"/></xsl:attribute>
                    <xsl:element name="Type"><xsl:text>validate</xsl:text></xsl:element>
                    <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                    <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                    <xsl:element name="Output"><xsl:text>NULL</xsl:text></xsl:element>
```

```
                    <!-- WSDL file that validates incoming request messages -->
                    <xsl:element
name="WsdlURL"><xsl:text>local:///myWSDLExample.wsdl</xsl:text></xsl:element>
                    <xsl:element
name="SOAPValidation"><xsl:text>body</xsl:text></xsl:element>
                </xsl:element>

                <!-- Set-Variable action example -->
                <xsl:variable name="actionNameSetvar"
select="concat($header/dppolicy:PolicyID,'-setvar-', position(),'-Example')"/>
                <xsl:element name="StylePolicyAction">
                    <xsl:attribute name="name"><xsl:value-of
select="$actionNameSetvar"/></xsl:attribute>
                    <xsl:element name="Type"><xsl:text>setvar</xsl:text></xsl:element>
                    <xsl:element name="PolicyKey"><xsl:value-of
select="$policyKey"/></xsl:element>
                    <xsl:element name="Input"><xsl:value-of
select="dpe:local-variable('input-context')"/></xsl:element>
                    <xsl:element name="Output"><xsl:value-of
select="dpe:local-variable('output-context')"/></xsl:element>
                    <xsl:element
name="NamedInOutLocationType"><xsl:text>default</xsl:text></xsl:element>
                    <xsl:element
name="Variable"><xsl:text>var://context/example/data</xsl:text></xsl:element>
                    <!-- value of the target endpoint -->
                    <xsl:element name="Value"><xsl:text>example of a static value for
variable assignement</xsl:text></xsl:element>
                </xsl:element>

        </dppolicy:config>

           <xsl:message dpe:priority="error">Exit template MyAssertion</xsl:message>
      </xsl:variable>

        <!-- Send the config to the console, whatever will fit (roughly 3k) -->
        <xsl:message dpe:priority="error">Exit template MyAssertion - Config:
           <xsl:copy-of select="$config"/>
        </xsl:message>

        <!-- Send the config to the temporary://  -->
        <xsl:variable name="filename"
select="concat('MyAssertion-config-',$header/dppolicy:PolicyID, '.xml')"/>
        <dpe:dump-nodes file="$filename" nodes="$config"/>

        <!-- Send the config to output -->
        <xsl:copy-of select="$config"/>

        <!-- Log -->
      <xsl:message dpe:priority="error">Exiting the MyAssertion template</xsl:message>

    </xsl:template>

    <!-- default catch all -->
    <xsl:template match="text()"/>


</xsl:stylesheet>
```

This example style sheet contains the creation of three actions on a request processing rule:

- A `Transform` action: The identity transformation is applied in the provided example.
- A `Validate` action: An artificial WSDL is applied to perform validation of a message body.
- A `Set-Variable` action: The action assigns a static string value to the context variable `var://context/example/data`, implementing a custom policy XSL style sheet

Steps to implement a custom policy XSL style sheet

To implement a custom policy XSL style sheet:

1. Choose a custom policy domain namespace. This namespace is used by DataPower to identify custom policy assertions.

   Tip: A good approach is to add a date or version to the custom policy namespace. This approach can ease the support of future versions of the policy domain.

2. Decide on a vocabulary for policy domain assertions based on business requirements. Create place holder templates for each policy assertions.
   a. DataPower Policy parameters might be necessary to provide additional information to correctly bind the assertion into the configuration.
   b. Create the style sheet that translates the policy assertions into the configuration.

   Preferred practice: Make a copy of the `itso.customPolicy.example.xsl` or `jk-example.xsl` file and use it as a starting point.

3. Create the DataPower processing actions that implement the policy assertion behavior you want. This task must be done on a DataPower appliance, by configuration.
4. Export the DataPower configuration from the previous step. This configuration includes the processing actions that are needed to implement custom policy assertions. Extract the applicable configuration (`StylePolicyAction` objects and prerequisite objects, if any) for each policy assertion from the export package for use in step 5. The `export.xml` file in a ZIP file that is exported by DataPower includes the XML definition of the configuration objects.
5. Apply the extracted configuration to each policy assertion template as required to implement each policy assertion.
6. After the implementation for the assertions is complete, copy the style sheet that is created in step 2a to the DataPower appliance in the `store:///policies/custom` directory, and copy any referenced files (for example style sheets, XML files, or certificates) to their respective locations.
7. When all the files are in place on the DataPower appliance, you can use a custom policy on a Web Service proxy. You can load a custom policy on a DataPower appliance or on a WSRR server.

Anatomy of a custom policy mapping style sheet

The `isto.customPolicy.example.xsl` style sheet that is described in this section presents the anatomy of a custom policy XSL transformation. You can use this style sheet as an example of how to convert a policy into the DataPower configuration objects that are necessary to enforce the policy assertions.

Custom policy assertions are defined through a custom policy vocabulary.

The first part of the `isto.customPolicy.example.xsl` consists of the custom policy namespace declaration. This namespace must be used in the custom policy related to the current custom policy style sheet.

Example 13-2 shows the prefixes and namespace declaration of the `isto.customPolicy.example.xsl` style sheet.

Example 13-2 Prefix and namespace declaration

```
<?xml version="1.0"?>
<!-- +
     |*****************************************************
     |*** Author: ITSO - gauci@fr.ibm.com
     |*** file: itso.customPolicy.example.xsl
     |*** Description: Custom policy XSL styleheet example
     |*** Revision: 1.0: Initial version
     |*****************************************************
   +-->

   <xsl:stylesheet version="1.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:func="http://exslt.org/functions"
       xmlns:dpe="http://www.datapower.com/extensions"
       xmlns:dppolicy="http://www.datapower.com/policy"
       xmlns:dpconfig="http://www.datapower.com/param/config"
       xmlns:dpfunc="http://www.datapower.com/extensions/functions"
       xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
       xmlns:myNamespace="http://itso.redbooks.ibm.com/myNamespace"

       extension-element-prefixes="dpe"
       exclude-result-prefixes="dpe dpconfig dppolicy wsp myNamespace">

       <!-- ******************************************************* -->
       <!-- (1) Declare the policy domain this stylesheet implements -->
       <!-- ******************************************************* -->

       <dpe:summary xmlns="">
    <dppolicy:domain>http://itso.redbooks.ibm.com/myNamespace</dppolicy:domain>
           <operation>xform</operation>
           <description></description>
       </dpe:summary>
...
```

The `myNamespace` prefix and the `http://itso.redbooks.ibm.com/myNamespace` namespace must be replaced when a new custom policy is created.

The second important part of the `isto.customPolicy.example.xsl` style sheet consists of a set of global variables and two matching templates, which must not be modified. The two matching templates are as follows:

- `<xsl:template match="/">`

  Match the root element of the preprocessed custom policy.

- `<xsl:template match="*[local-name()='All']">`

  Match the <wsp:All> element of the preprocessed custom policy.

In this second part, you may declare any required DataPower policy parameters that are necessary for binding and associating policy assertions to configuration. These parameters can be used in DataPower through WS-Policy Parameter Set objects.

An example of policy parameter use is in 5.7.3, “Creating a Web Service Proxy for versioning in DataPower” on page 166.

Example 13-3 shows a policy parameter declaration.

Example 13-3 DataPower policy parameter declaration

```
...
<!-- Specify the WSRR Server to use to retrieve WSDL files for 'validate' actions
-->
    <dpe:param name="dpconfig:WSRR-Server" type="dmReference" reftype="WSRRServer"
xmlns="">
<dppolicy:assertion>{http://itso.ibm.com/ibmredbooks-travelcompany/versioning/2012
-11}Versioning</dppolicy:assertion>
        <display>WSRR-Server</display>
        <description>Specify which WSRR server to use to retrieve WSDL files for
validations based on WSDL</description>
    </dpe:param>
...
```

Figure 13-4 shows the configuration panel on a WS-Policy Parameter Set object in DataPower, to specify the value of the parameter from Example 13-3:

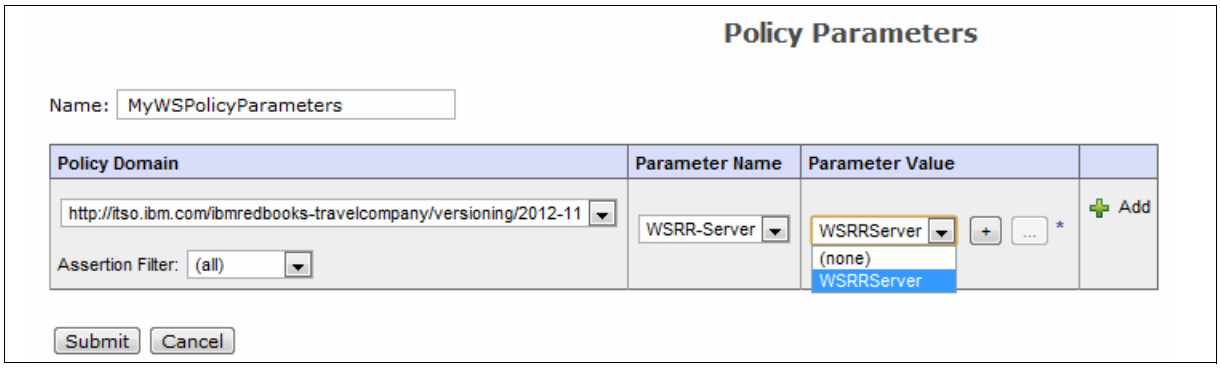


Figure 13-4 WS-Policy parameter set in DataPower

The third part of the `isto.customPolicy.example.xsl` style sheet consists of specific matching template, based on a custom policy assertion (`isto:MyAssertion`).

It is of course possible to add as many matching templates as needed. Every custom policy assertion must be matched and treated through its own matching template.

Important: The configuration objects that a custom policy style sheet generates are the processing actions (configuration object type: `StylePolicyAction`) that are needed to enforce policy assertions and the objects that are referenced by these processing actions.

The processing action objects that are generated by each policy assertion are added automatically to a processing rule (object type: `Rule`) by the DataPower policy framework. Thus, a custom policy style sheet must not create its own processing rule.

Multiple policy assertions can be rendered on the same automatically generated processing rule. Therefore, the Input and Output contexts that are used by the processing actions for each assertion must share DataPower context correctly. A shared context is handled in a custom policy style sheet by using the input and output contexts. Values of input and output contexts are stored in the following local variables:

- `dpe:local-variable('input-context')`
- `dpe:local-variable('output-context')`

A policy assertion might not care about the input or output context. However, all assertions must, at a minimum, route the input context to the output context by using an identity transformation to preserve the integrity of the contexts.

DataPower attempts to render the configuration for assertions, in the order that they appear in a custom policy, by using the `assertionNo` attribute for the `dppolicy:config` element. Thus, handle this attribute properly. A custom policy stylesheet sets `assertionNo` attribute based on the position of the assertion within a custom policy. Therefore, at run time, the execution order of the DataPower configuration artifacts that are created by the custom policy style sheet is the same as the order of the policy assertions of the custom policy.

Important: Be careful when using for-each loops to iterate through policy assertions. The position, which is calculated by using the position() XSL function of each assertion in the for-each loop, is an integer value 1.

Uploading a custom policy XSL style sheet to a DataPower appliance

Upload the XSL style sheet in the `default` domain, in the `store:///policies/custom` directory, as shown in Figure 13-5.

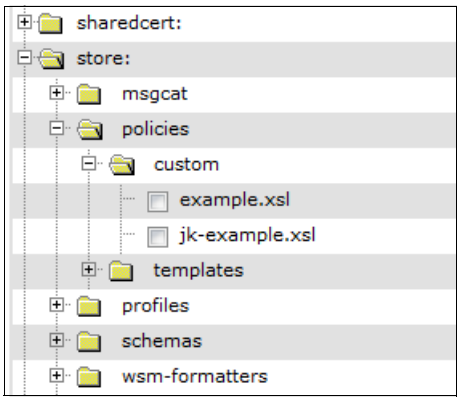


Figure 13-5 Custom policy XSL stylesheet directory

Use one of the following methods to upload an XSL style sheet on a DataPower device:

- Manual upload, by using the web GUI
- SOAP Management (SOMA) API, through the XML management interface of a DataPower SOA appliance
- Command-line interface (CLI) wrapped into a shell script

For information, related to SOMA API, see *WebSphere DataPower SOA Appliance: The XML Management Interface*, REDP-4446.

To verify if the custom versioning policy style sheet is parsed properly, check the `Policy Domains Supported` interface. You can access this interface through the Status menu by selecting **Status** → **Web-Service** → **Policy Domains Supported**. This status interface provides the information shown in Figure 13-6:

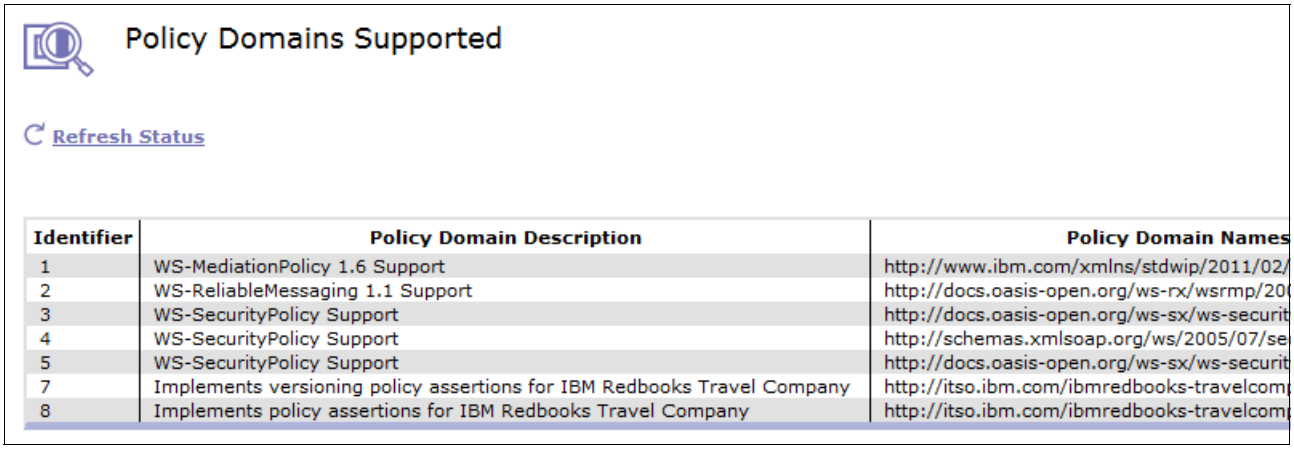


Figure 13-6 Policy Domains Supported user interface

13.3 Importing a custom policy in WSRR

After you create a custom policy, you can upload it in WSRR by using the same method as you use for any other policy file. You can also manage the custom policy and attach SLA and SLD levels as a standard mediation policy.

A custom policy and a standard mediation policy, created in WSRR, have the following differences:

- A custom policy is created outside WSRR, by using an XML editor or an IDE.
- A custom policy cannot be displayed in WSRR. However, you can download the policy from WSRR and open it in an XML editor.

To upload a custom policy in WSRR:

1. Connect to WSRR by using the business space to access the service registry for operations.
2. Click the **Load Documents** link provided in the Service Registry Actions widget, as shown in Figure 13-7.

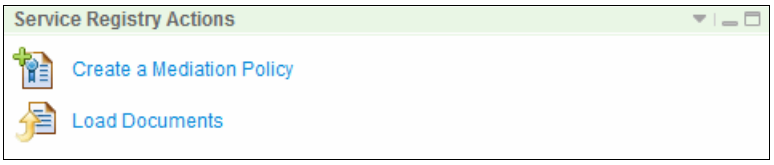


Figure 13-7 Load a custom policy document in WSRR

3. In the Load Documents window, click **Browse** to locate the custom policy to load, and then select the **Policy** document type, as shown in Figure 13-8.

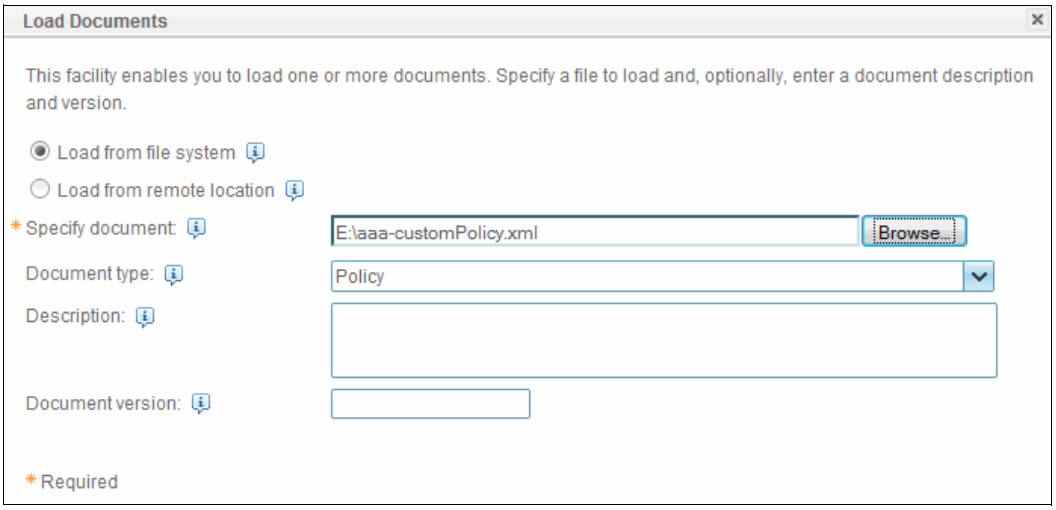


Figure 13-8 Document type specification when loading a custom policy document into WSRR

4. Enter a description and provide a document version to the policy that is loaded on WSRR, as shown in Figure 13-9.

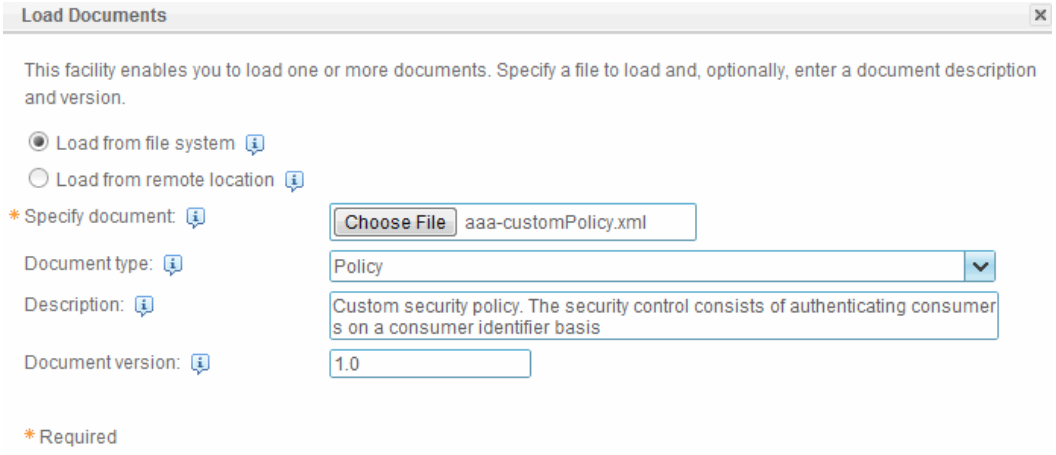


Figure 13-9 Description and version when loading a custom policy document into WSRR

13.4 Custom policy attachment in WSRR

A custom policy can be attached in WSRR with the same capabilities as a mediation policy. Assuming a custom policy is already loaded in WSRR, to attach a custom policy at SLA or SLD levels, use the following steps:

1. Connect to WSRR by using the business space to access the service registry for operations.
2. Select the custom policy that must be attached. The custom policy is displayed, as shown in Figure 13-10 on page 384.

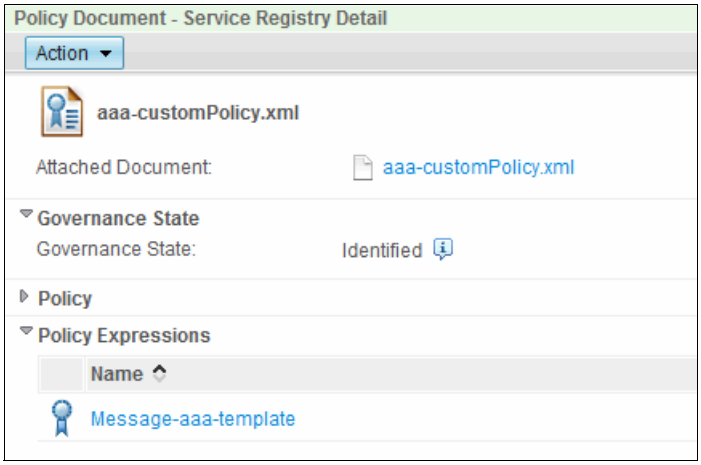


Figure 13-10 Custom policy in WSRR: Source document

3. Click the name provided in the Policy Expression section (Figure 13-11).

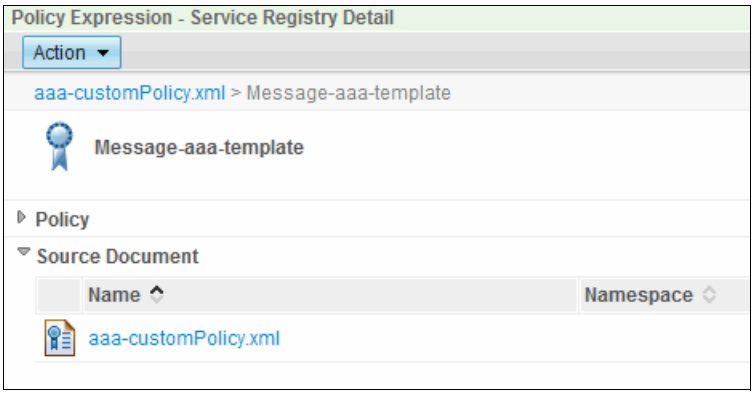


Figure 13-11 Custom policy in WSRR: Policy Expression

4. Select **Action** → **Manage Policy Attachments,** as shown in Figure 13-12.

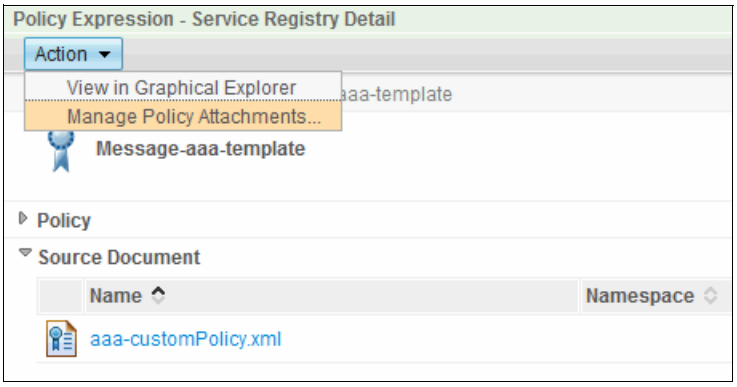


Figure 13-12 Manage a custom policy attachment in WSRR

5. Select **Attach to Specific Items** if you want to attach the custom policy to a specific SLA or SLD. Select **Attach to Items using a Query** if you want to attach the custom policy to a set of SLAs or SLDs, as shown in Figure 13-13.

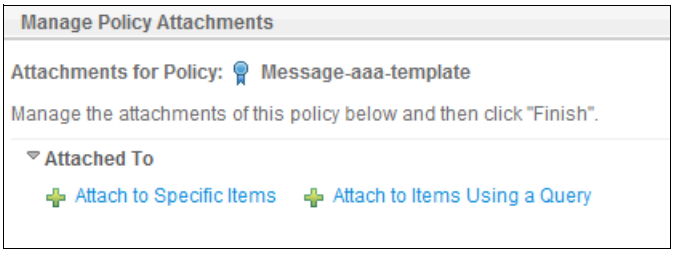


Figure 13-13 Attach a custom policy in WSRR

After the policy is attached at the correct level, it must be deployed from the WSRR Governance Master to the WSRR instance of the target environment. See Chapter 3, “Policy traffic management, provider only, with operations” on page 35 for details about policy promotion between different environments.

13.5 Importing a custom policy in DataPower

A policy file is imported on a DataPower device as any other file. Use the following steps to upload a custom policy document on a DataPower appliance:

1. Connect to a DataPower appliance using your credentials (login and password). From the log-in window, select the domain in which the custom policy file must be loaded.
2. Click the File Management icon in the control panel, as shown in Figure 13-14:

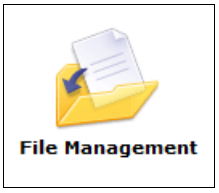


Figure 13-14 File Management icon

3. Select the directory in which the custom policy must be loaded and click the **Actions** link to the right of the directory name, as shown in Figure 13-15, and select the **Upload Files** option.

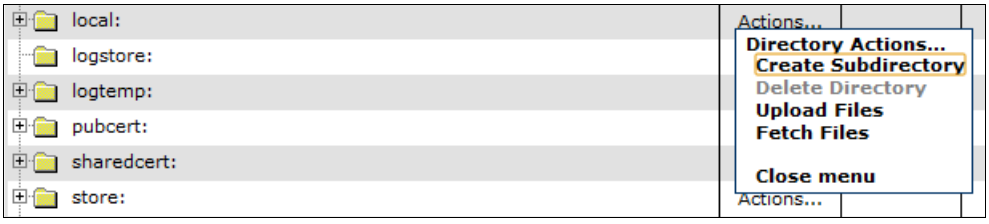


Figure 13-15 Actions menu to upload a file

4. Choose the policy file you want to upload and then click **Upload**, as shown in Figure 13-16.

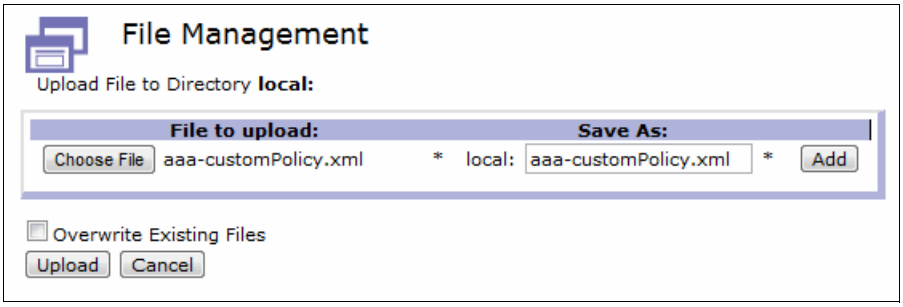


Figure 13-16 Upload a custom policy document

Select the **Overwrite Existing Files** option if necessary.

13.6 Custom policy attachment in DataPower

This section describes the attachment of a custom policy document on a Web Service Proxy in DataPower.

Use the following DataPower configuration steps to attach the custom policy to a service in DataPower:

1. Edit the Web Service Proxy object that contains the WSDL of the services for which a custom policy must be attached.
2. Select the **Policy** tab of the Web Service Proxy
3. On the WSDL Policy Tree Representation section, select the WSDL on which the custom policy must be attached.
4. Select the port, portType, binding, or service level in the WSDL policy tree representation of a service.
5. Click the **WS-Policy** link at the level you want to attach the custom policy, as shown in Figure 13-17:

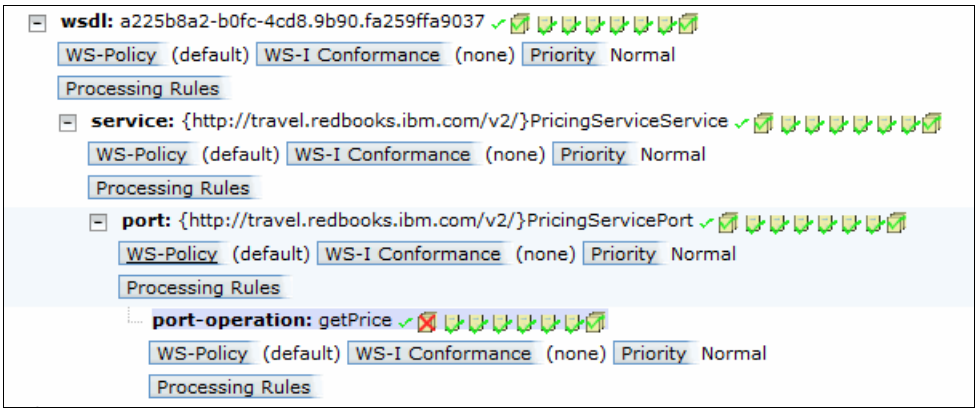


Figure 13-17 WS-Policy link at the port, portType, binding or service level

6. Select the **Sources** tab and select the custom policy you want to attach. Also specify the policy identifier of the custom policy, as shown in Figure 13-18.

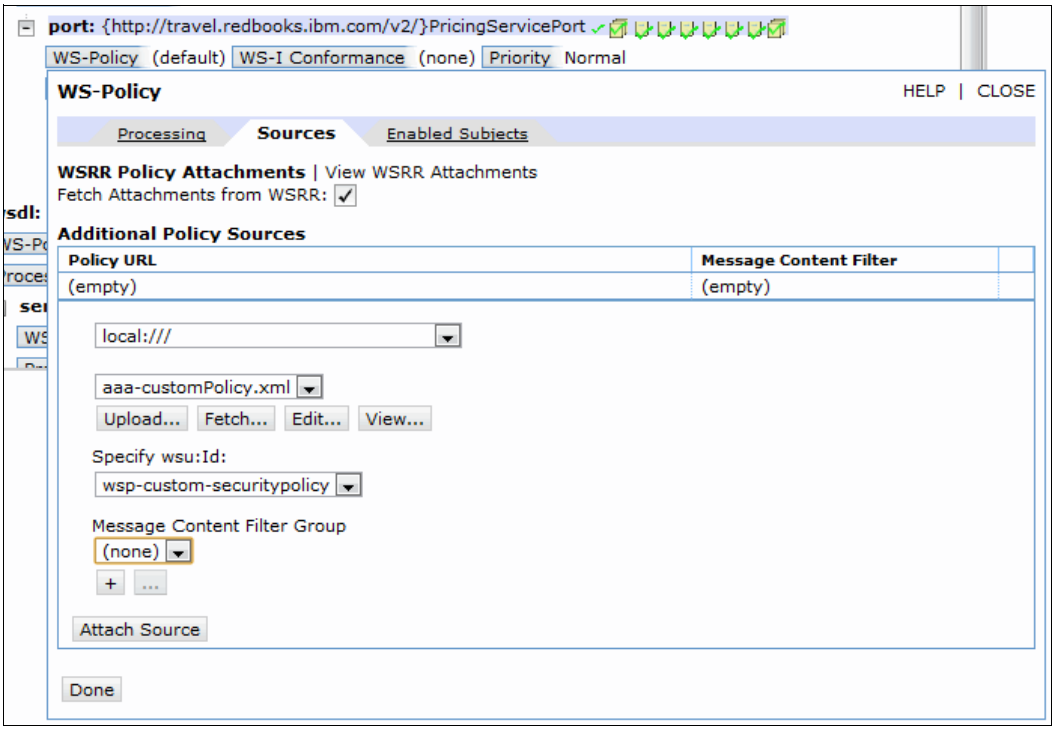


Figure 13-18 Sources tab to attach a custom policy

7. If the policy must be enforced for specific consumers, you can create a message content filter group. Click the plus sign (**+**) beside the Message Content Filter Group label in this case, shown in Figure 13-19.

   Consumers for which a custom policy must be enforced are identified through a consumer identifier. This identifier can be extracted either from an HTTP header or from an XML content by using a dedicated XPath expression.

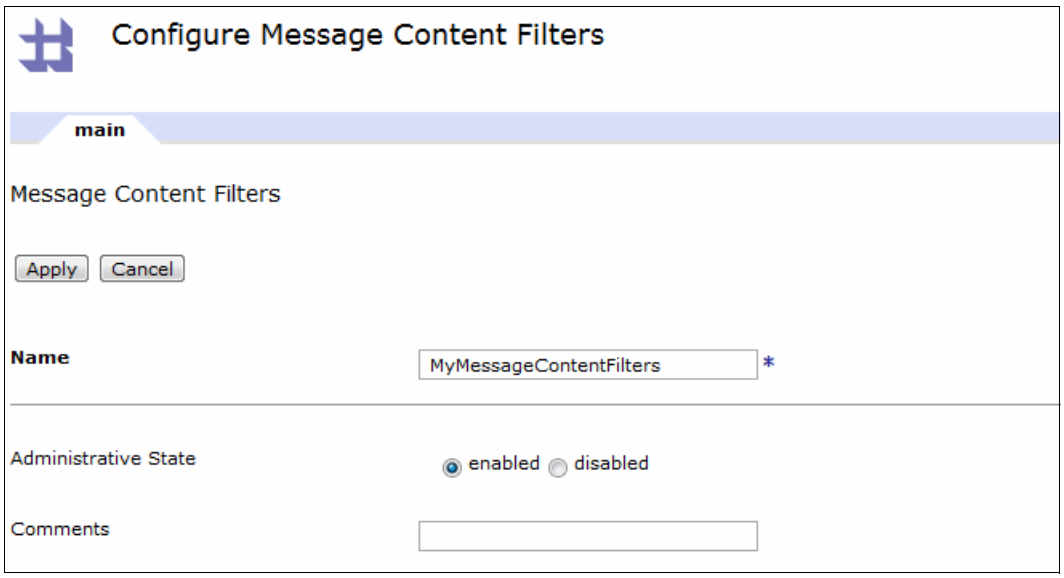


Figure 13-19 Message content type filters configuration panel

8. Click **Attach Source** to attach the custom policy, as shown in Figure 13-20. If a message content filter is created and associated to the custom policy attachment, then only consumers defined through this Message Content Filter object are enforced in regard with the selected policy. If no message content filter is selected, then the custom policy is enforced for all consumers that access the service.

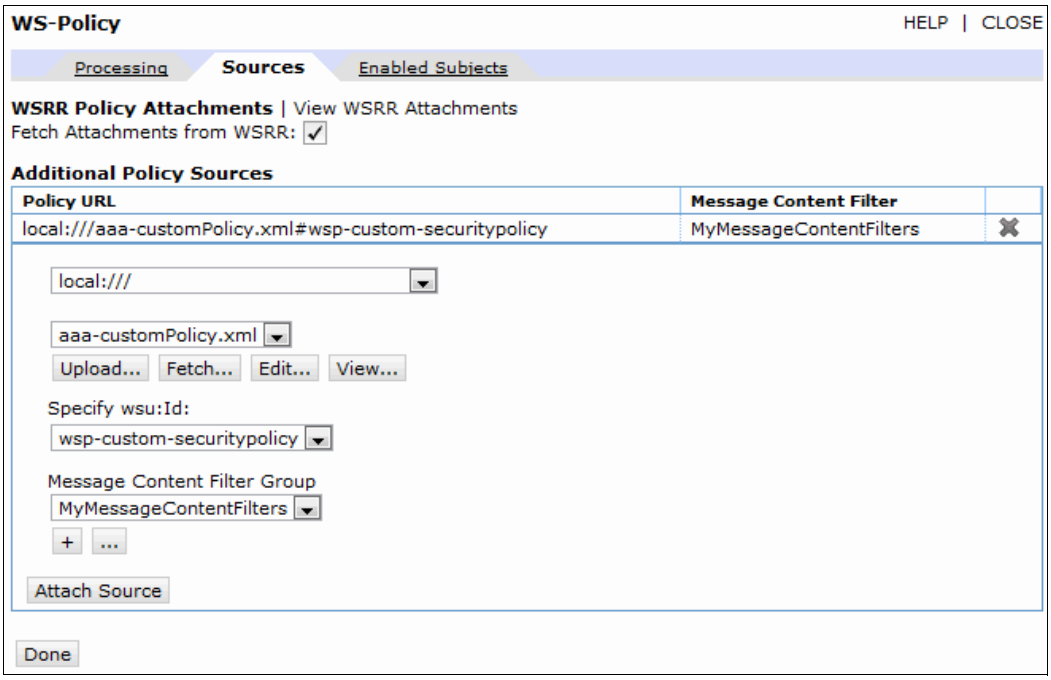


Figure 13-20 Attaching a custom policy in DataPower

13.7 Custom policy enforcement

After DataPower established a subscription with WSRR, it can retrieve WSDL files and also mediation and custom policies. Custom policies are transformed into DataPower configuration *artifacts*. These configuration artifacts are used to enforce traffic at run time.

Artifacts are displayed in DataPower from the SLA Policy tab of a Web Service Proxy. The SLA Policy tab contains an SLA table, which displays the following sets of information:

- Policy model
- DataPower rules

Custom policy enforcement detailed examples are presented in Chapter 5, “Versioning with custom policy” on page 119 and Chapter 6, “Security using custom policy” on page 187.

14

ITCAM as policy monitoring point

This chapter describes the IBM Tivoli Composite Application Manager (ITCAM) for SOA component, its architecture, and its behavior. The chapter also includes details about configuration and troubleshooting.

This chapter contains the following topics:

- 14.1, “Overview of ITCAM as the PMP” on page 390
- 14.2, “Overview of Tivoli architecture” on page 391
- 14.3, “ITCAM for Applications processes policy and management updates from WSRR” on page 393
- 14.4, “Configuring for integration of WSRR and ITCAM for Applications” on page 394
- 14.5, “Troubleshooting the installation” on page 396
- 14.6, “Troubleshooting and tracing the integration” on page 396
- 14.7, “Operations notification and upstream integration with other Tivoli products” on page 409
- 14.8, “ITCAM monitoring agent for DataPower policies” on page 410

© Copyright IBM Corp. 2013. All rights reserved.

14.1 Overview of ITCAM as the PMP

The current portfolio of Tivoli SOA security and infrastructure management products include IBM Tivoli Composite Application Manager for Transactions and IBM SmartCloud Application Performance Management (APM), both of which contain an agent named IBM Tivoli Composite Application Manager (ITCAM) for SOA.

ITCAM for Transactions and IBM SmartCloud APM both reside on an IBM Tivoli Monitoring 6.x framework, which enables the ITCAM for SOA agent to benefit from the common capabilities of the framework.

ITCAM for SOA interfaces with web services that are deployed across most popular SOA and web services platforms, starting with WebSphere Application Server, WebSphere Application Server CE, WebSphere Business Integration Server Foundation, WebSphere Process Server, WebSphere Enterprise Service Bus (WebSphere ESB), DataPower SOA appliances, IBM CICS® TS 3.1 and also BEA WebLogic, Microsoft .NET, SAP NetWeaver, and JBoss. Initial protocol support includes SOAP/HTTP(S), SOAP/JMS,CICS, and WebSphere Message Broker. This interfacing enables the management of services as first-class resources.

ITCAM for SOA enables service metrics and alerts to be shown alone or combined in the Tivoli Enterprise Portal with those from other Tivoli monitoring products. You can drill down from services to application components to identify failures.

ITCAM for SOA provides built-in and extensible alerts, situations, workflows, and managed mediation primitives to enable powerful automation scenarios. Service metrics, alerts, and automation workflows provided by ITCAM for SOA and other Tivoli products can be used by the automation framework in Tivoli Monitoring 6.x to actively manage service requests, for example by rejecting some incoming requests or provisioning additional servers during periods of heavy load.

For WebSphere ESB Service Component Architecture (SCA) run time, ITCAM for SOA provides SCA-based mediation primitives for enhancing management functions (monitoring, logging, routing, and transformation), and you can enable or disable at run time without reconfiguration or redeployment.

The service metrics and alerts that are displayed in ITCAM for SOA provide a rich and many-layered source of information that can help to reduce the time and skills required for problem root-cause analysis and resolution. For example, you can drill down from the service layer to the application component layer to understand why a particular web service implementation (perhaps mapped to an EJB method on a particular server) has slowed down.

ITCAM for SOA can be used by IT operators to manage real-time alerts, by SOA architects to view the actual runtime relationships of service flows, and by SOA developers to provide a deep understanding of service behavior and performance.

Tivoli Monitoring or ITCAM: Technically, IBM Tivoli Management provides the framework and common GUI for all Tivoli Monitoring components. ITCAM for SOA is only one of many such applications. Therefore, the proper terminology is *Open the Tivoli Monitoring console and access the ITCAMforSOA agent* or *Create a situation in Tivoli Monitoring that monitors the ITCAM agent*. Practically speaking, the two phrases are interchangeable and are used in this book in a manner intended to be the simplest to read.

Beyond the monitoring capabilities, ITCAM also interfaces with WSRR as the executor for WSRR policies.

14.2 Overview of Tivoli architecture

This section describes the physical architecture of the IBM Tivoli Monitoring and ITCAM stack and how it interfaces with the other parts of the Tivoli portfolio.

14.2.1 IBM Tivoli Monitoring Architecture

The Tivoli Monitoring architecture consists of the following components:

- A Tivoli Enterprise Portal Server which provides visualization and is the only component with which users interact. It also hosts the added GUI capabilities which ITCAM for SOA provides, such as services flows.
- A Hub Tivoli Enterprise Management Server provides centralized management services. All other components access the Hub Tivoli Enterprise Monitoring Server to update it.
- A Tivoli Data Warehouse stores historical performance data.
- Multiple Remote Tivoli Enterprise Management Servers are a scalability layer, a security layer, or both a scalability and security layer between the agents and the Hub Tivoli Enterprise Monitoring Server.
- A variety of agents which are deployed on the monitored severs.

  Tivoli Monitoring agents also have agentless capability so they can remotely monitor servers or devices that cannot have an agent installed directly on them, such as DataPower.

Figure 14-1 shows an example architecture.

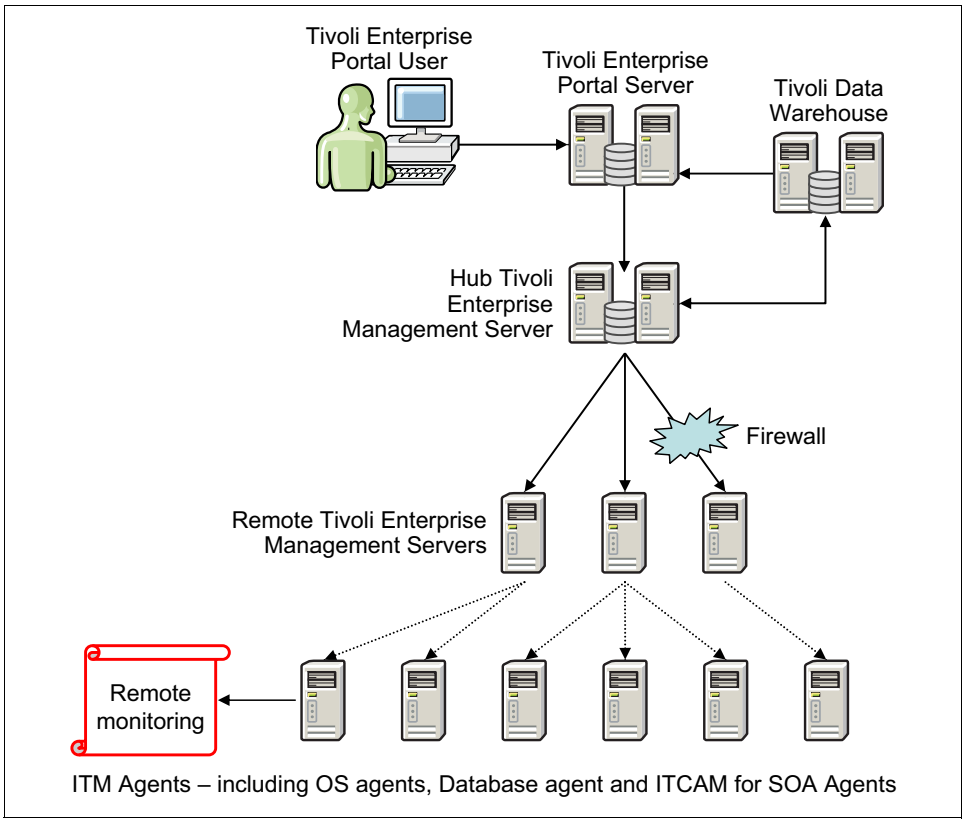


Figure 14-1 Sample Tivoli Monitoring architecture

Because Tivoli Monitoring gathers information from a wide variety of sources that are relevant to the SOA environment, it holds a complete picture of the performance and health of the environment, as shown in Figure 14-2.

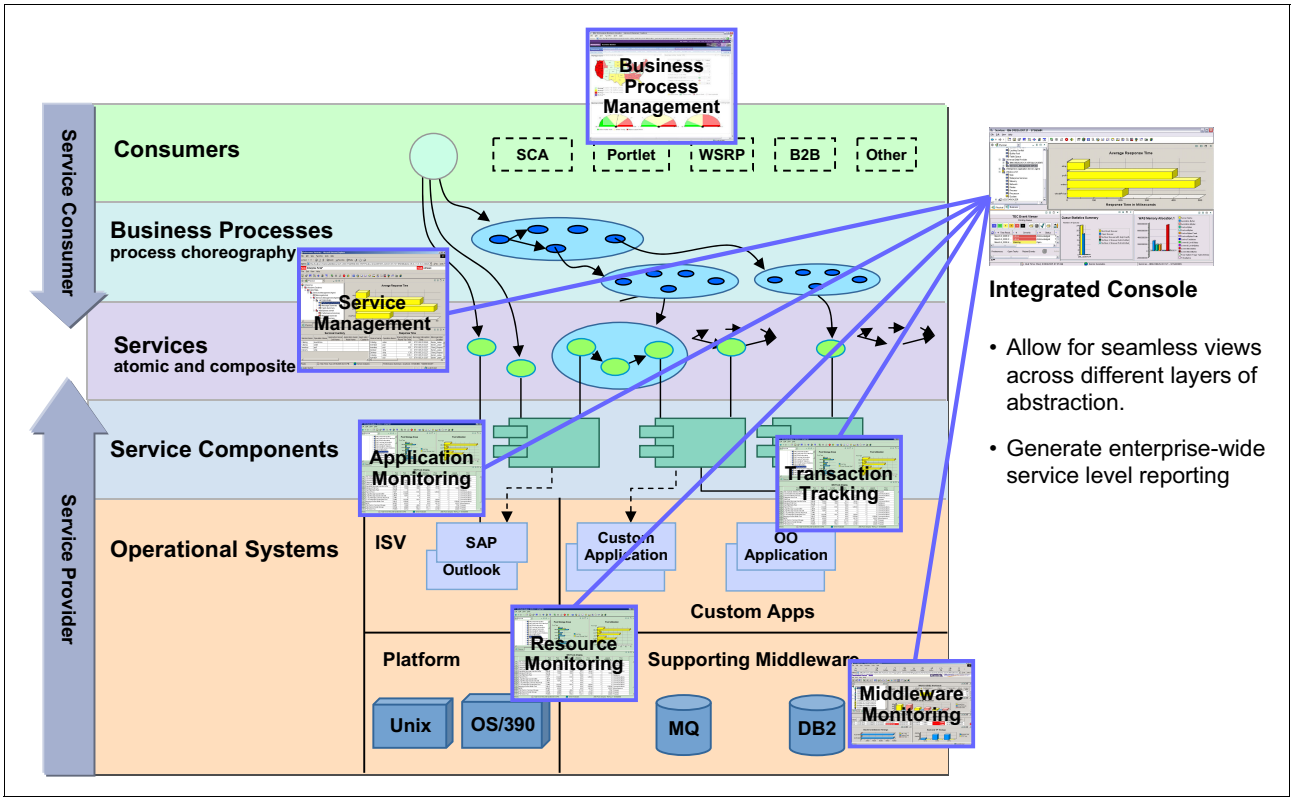


Figure 14-2 Tivoli Monitoring information resources

14.2.2 ITCAM for SOA architecture

All Tivoli Monitoring agents have two components:

- An active agent that is deployed (per best practices) adjacent to the Monitoring target.
- Plug-ins and updates to the Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server that enable them to manage and display the information the agent is gathering.

ITCAM for SOA is an agent (agent code D4) that is slightly more complex than most. The active agent is split into a datacollector, which accesses the target, and an agent component, which interfaces with the Tivoli Enterprise Monitoring Server. Depending on the target and the OS architecture, the DataCollector can be a separate thread or process from the standard agent.

The Tivoli Enterprise Portal Server component of ITCAM for SOA is an extension to the standard Tivoli Monitoring database that contains information about the SOA process flows. These are called the SOA Domain Management Server (SDMS) and Tivoli Common Object Repository. The ITCAM for SOA Tivoli Enterprise Portal Server component also includes the interface to WSRR (as detailed in 14.3, “ITCAM for Applications processes policy and management updates from WSRR” on page 393).

14.3 ITCAM for Applications processes policy and management updates from WSRR

Figure 14-3 demonstrates the runtime integration flow between WSRR and ITCAM. This flow does not show the initial setup which includes such tasks as creation of the ITCAM subscription. These tasks are covered in 14.2, “Overview of Tivoli architecture” on page 391.

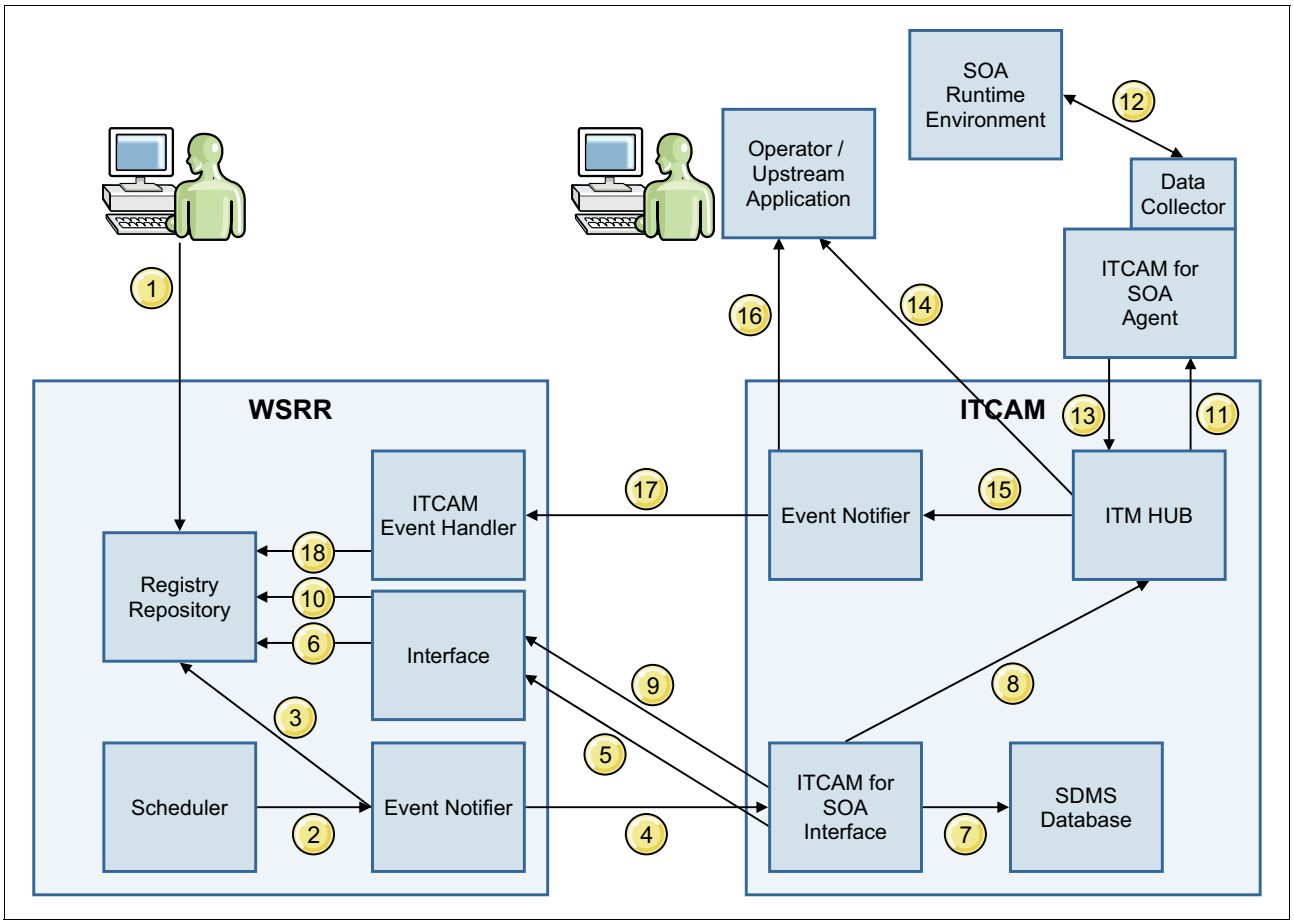


Figure 14-3 Integration flow between WSRR and ITCAM

The following steps describe the flow shown in Figure 14-3.

1. The user updates WSRR, creating, modifying, or deleting a policy.
2. The Scheduler periodically activates the Event Notifier.
3. The Event Notifier checks whether any events have a subscriber.
4. The Event Notifier updates the ITCAM subscriber (there might be multiple instances of ITCAM subscribing to a single WSRR).
5. ITCAM queries WSRR for details.
6. WSRR extracts the relevant objects information.
7. ITCAM checks the rules file to see what response is required (creation of situation, deletion of situation, ignore event, and so on). For the purpose of the flow demonstration, assume that a new situation is to be created.
8. ITCAM creates the situation (or performs whatever other operation is required).

9. ITCAM updates WSRR of the new situation.
10. WSRR updates the object properties (for example, the itmSituationIdentifier field is added and populated).
11. The situation is uploaded to the ITCAMforSOA agent.
12. The ITCAMforSOA agent monitors the SOA runtime environment (WebSphere Message Broker, DataPower, WebSphere Application Server, and so on).
13. If a policy threshold is violated, the situation fires and Tivoli Monitoring infrastructure is updated.
14. Operators are alerted in the Tivoli Monitoring console and any automatic scripts are run, including email update or auto-fix.
15. Tivoli Monitoring sends the event to the Tivoli Monitoring notifying service.
16. The Tivoli Monitoring notifier may forward the event to an upstream integration product such as Tivoli Omnibus.
17. The Tivoli Monitoring notifier may forward the back event to WSRR.
18. The ITCAM listener updates the WSRR repository with the updated parameter.

14.4 Configuring for integration of WSRR and ITCAM for Applications

This section presents an overview and several best practices for the WSRR and ITCAM configuration. Full details are in *ITCAM for SOA WSRR Integration Guide*, which is located at the following website:

`http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/WSRR_Integration_Guide_72/soawsrr.html`

The configuration consists of three stages:

- Exchange of security certificates. This stage enables the products to exchange information.
- In WSRR, activation of the following items:
  - `SubscriptionNotifier`
  - `SubscriptionNotifierPluginScheduler`

  Ready to use, the WSRR notifier is configured correctly to post HTTP events. The plug-in is inactive and must be activated by editing the configuration file.

Notes:

- Setting the frequency of the plug-in can affect both WSRR and ITCAM performance, as they update each other.
- Using the scheduler means that ITCAM will be creating or deleting situations in production environments automatically. It might be preferable to deactivate the scheduler and manually perform the synchronization with the following script:

  `kd4WSRRITMSynchronization`

- In ITCAM, registry of the `wsrr_sdms_config.xml` configuration file and subscription to WSRR notifier.

  This file is divided into four sections:

  - Tivoli Monitoring Instance, which contains information regarding login to the Tivoli Monitoring.
  - Situation Configuration, which contains the default configuration of situations created by WSRR.
  - Aliases, which makes the fourth section more readable.
  - WSRR Instances, which contains the process rules that determine how ITCAM processes events from WSRR. These process rules are configured to operate with the standard governance enablement profile (GEP). If you modify the GEP or create your own profile, this section (and the previous one) must be modified to match.

    This section is also where you configure classifications that will define whether or not a situation will be created. For example, the configuration can indicate that only SLDs that are attached to an endpoint that is classified as *production* can generate situations.

Use the following preferred practices for the `wsrr_sdms_config.xml` file:

- Ready to use, the `situation.tec.forwarding` parameter is deactivated. Set the value to `true` to send events back to WSRR or upstream to Omnibus.
- If you set `situation.tec.forwarding` to `true`, make sure that `situation.tec.destination` is correct. The best approach is to create a new destination in ITCAM, using `tacmd createeventdist`, and letting ITCAMforSOA use that destination instead of the default (`0`).
- Setting the **`takeaction`** commands depends on your use for them. If you want to use these commands to run a script that automatically fixes the problem, then the best location for these commands is in the agent. If you want to use **`takeaction`** to run a script that will send an email, then set the value to `tems` so that the mailing script runs from a central location and not from each and every server that has an ITCAMforSOA agent.
- Set `takeaction.eachinterval` to `true` if you want to do an action multiple times when a policy threshold is passed, for as long as the problem remains. If you want to run an action only when the threshold violation first occurs, leave the setting at `false`. For example, ITCAM detected that a webservice is running slowly and raises an alert. If, after five minutes. the web service performance improves to within the policy threshold, then the situation closed automatically. Otherwise, it remained open. If you want to send an email, the best approach is probably to send the email only the first time the problem is detected. If you want to run an auto-fix script, run it multiple times until the problem is repaired.
- The `situation.sampling.interval` parameter is the only parameter that can be overridden by the WSRR policy itself. If you do not set a value in the policy (“Defining ITCAM specific properties” on page 291) then this value will be used.

Be careful when editing the `wsrr_subscription.xml` file, because the error checking that ITCAM does, when the script is run, is basic.

14.5 Troubleshooting the installation

Several issues might arise during installation of ITCAM for SOA. The following named technotes supplement the information center troubleshooting documentation:

- SOA Operational Flow and Sun JVM:

  http://www-304.ibm.com/support/docview.wss?uid=swg21429283

- ITCAM for SOA Topology Workspaces unavailable after Tivoli Monitoring is upgraded to 6.2.3 or later:

  http://www.ibm.com/support/docview.wss?uid=swg21567013

- ITCAM for SOA: After remote deploy of the agent, the host name is lost and the agent does not function correctly:

  http://www-304.ibm.com/support/docview.wss?uid=swg21381579

- ITCAM for SOA workspace is unusable if ITCAM for SOA 7.1.1 or its fix packs are installed after Tivoli Monitoring 6.2.2 fix pack 4:

  http://www.ibm.com/support/docview.wss?uid=swg21504213

- ITCAM for SOA 7.1.x with a corrupted SDMS database:

  http://www.ibm.com/support/docview.wss?uid=swg21381100

- ITCAM for SOA: WSRR Event Handling Integration Fails:

  http://www.ibm.com/support/docview.wss?uid=swg21381160

14.6 Troubleshooting and tracing the integration

The primary purpose of tracing and troubleshooting is to determine whether situations are being created from policies.

For situations to be created, the following requirements must be fulfilled:

- The WSRR-ITCAM integration must be configured without errors.
- The policy must be in the correct governance state.
- The SLD must be in the correct governance state.
- The endpoints must be online.
- If any classifications were chosen in the SDMS configuration file, the relevant components in WSRR must have the same classifications.

If you suspect that situations are not being created, the simplest test is to run the **kd4WSRRITMSynchronization** script to enforce immediate synchronization between WSRR and ITCAM and then read the trace logs.

14.6.1 Situation verification

If only a few policies exist, detecting whether they were created in ITCAM can be easy. However, if a large number of policies exists, then it is easier if they are listed.

After loading the named queries, found in the extended downloadable material that is attached to this book, it is possible to view the policies that were successfully converted into ITCAM situations.

Running the following query produces an XML file with the list of policies, as shown in Example 14-1:

```
https://WSRRServer:9443/WSRR/8.0/Metadata/XML/Query/SucessfulITCAMPolicies
```

Example 14-1 Response of SucessfulITCAMPolicies

```
<resources>
<resource>
<properties>
<property name="primaryType" value=""/>
<property name="bsrURI" value="b0d521b0-e370-4008.a9e0.6297a362e046"/>
<property name="_sdoType" value="PolicyExpression"/>
<property name="classificationURIs" value="http://www.w3.org/ns/ws-policy
http://www.ibm.com/xmlns/prod/serviceregistry/lifecycle/v6r3/LifecycleDefinition#S
OAPolicyLifecycle_Approved
http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring"/>
<property name="description" value=""/>
<property name="name" value="urn:MonRequestorBusy"/>
</properties>
</resource>
<resource>
<properties>
<property name="primaryType" value=""/>
<property name="bsrURI" value="b82737b8-ca58-48fc.94bd.bfac60bfbdb9"/>
<property name="_sdoType" value="PolicyExpression"/>
<property name="classificationURIs" value="http://www.w3.org/ns/ws-policy
http://www.ibm.com/xmlns/prod/serviceregistry/lifecycle/v6r3/LifecycleDefinition#S
OAPolicyLifecycle_Approved
http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring"/>
<property name="description" value=""/>
<property name="name" value="urn:ClientFaulted"/>
</properties>
</resource>
</resources>
```

Example 14-1 shows that two policies were translated into ITCAM situations.

The named query, `AllAttachedITCAMPolicies`, lists all of the policies that are attached to an SLD. Assuming that all SLDs are in the `subscribable` state, that their endpoints are `online`, and that the policy itself is in the correct state (`monitor` or `active`), then both queries should return the same policies.

Within ITCAM itself, the situations can be seen in the Situation Editor, under Services Management Agent Environment or by running the following commands:

```
tacmd login -s <ITCAMServerName> -u <ITCAMUser> -p <ITCAMPassword>
tacmd listsit -t d4
```

Both techniques will display both of the situations that were created manually within ITCAM and the situations that were created by WSRR. Remember that WSRR situations all have the following name:

SLD - Policy

14.6.2 Trace levels in WSRR

Use the following steps to set the trace level in WSRR to follow the ITCAM integration in the WebSphere Integrated Console:

1. Select **Troubleshooting** → **Logs and trace** → **server1** → **Diagnostic Trace** and then select **server1.**
2. Choose either the **Configuration** or **Runtime** tab, depending on whether this change is short or long term:
   - Changes in the Configuration tab become active after recycling the WSRR server, but will be permanent thereafter.
   - Changes in the Runtime tab become active immediately, but will revert to the previous version after a restart.
3. After you select a tab, click **Change log detail levels**.
4. Navigate to the log level or type the values in the text box, as shown in Figure 14-4.

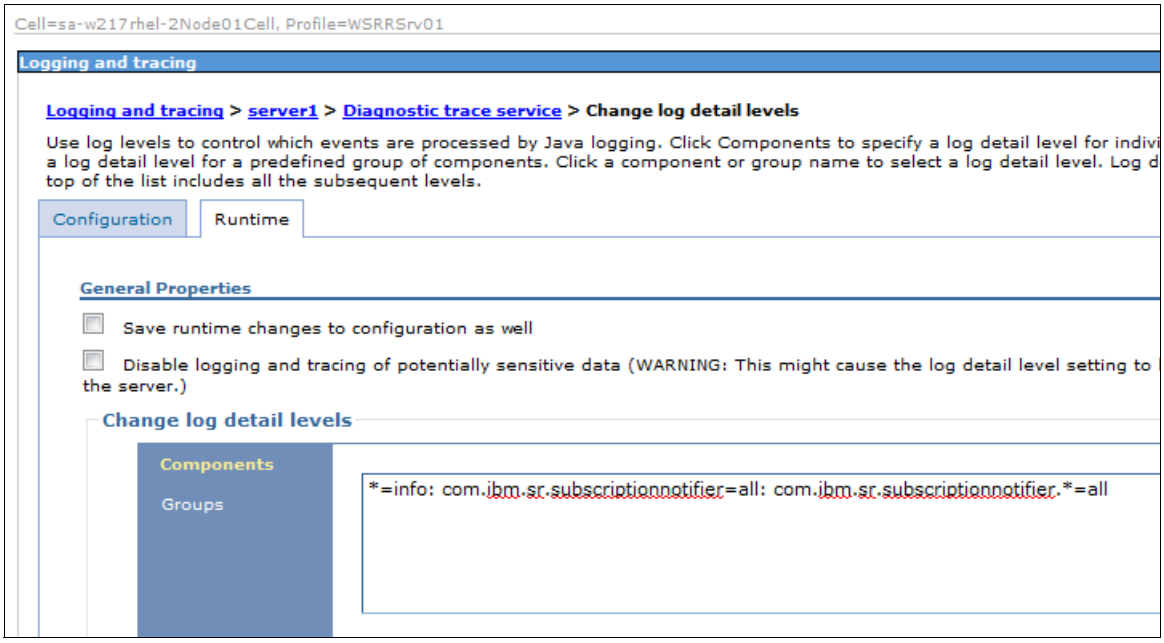


Figure 14-4 WSRR trace settings

5. Click **OK.** The trace file is placed in the WebSphere Application Server profile log directory, such as in the following examples:

   `/opt/ibm/WebSphere/WSRR/v8.0/profiles/WSRRSrv01/logs/server1/`

14.6.3 Trace levels in ITCAM

Use the following steps to set the trace level in WSRR to follow the ITCAM integration in the WebSphere Integrated Console:

1. Select **Troubleshooting** → **Logs and trace** → **server1** → **Diagnostic Trace** and then select **ITMServer.**
2. Choose either the **Configuration** or **Runtime** tab, depending on whether this change is short or long term:
   - Changes in the Configuration tab become active after recycling the WSRR server, but will be permanent thereafter.
   - Changes in the Runtime tab become active immediately, but will revert to the previous version after a restart.
3. After you select a tab, click **Change log detail levels.**
4. Navigate to the log level or type the values in the text box, as shown Figure 14-5.

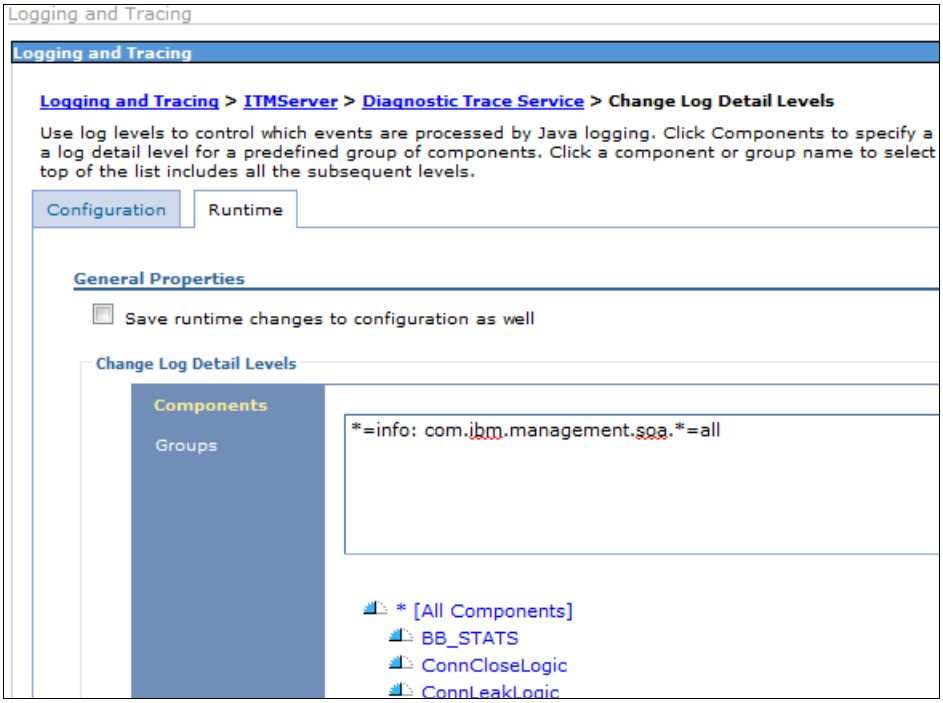


Figure 14-5 ITCAM trace settings

5. Click **OK.** The trace file is placed in the Tivoli Enterprise Portal WebSphere Application Server profile log directory, for example, `C:\IBM\ITM\CNPSJ\profiles\ITMProfile\logs\ITMServer\`.

Reading traces: When reading the logs, an easier approach might be to load them into a text editor that can emphasize XML tags so that WSRR information is easier to find.

14.6.4 Example traces

This section has examples of traces for invalid character, missing situation field, nonexistent table, and valid policy.

Invalid character

In this log (Example 14-2), the `TooManyMessages` policy contains an invalid character that the ITCAM cannot parse. In a text editor, the space between `urn:` and `TooManyMessages` appears to contain a non-printable character that the Simple API for XML (SAX) parser cannot handle.

The solution is to fix the name of the policy or recreate it.

Example 14-2 Invalid character error

```
[11/7/12 10:26:11:355 EST] 00000028 WSRRNotificat I   KD4DM0480I Processing the WSRR
notification [<?xml version="1.0" encoding="utf-8"?>
<body:resources
xmlns:body="http://www.ibm.com/xmlns/prod/serviceregistry/HttpPostNotifierPluginMsgBody">
  <body:resource bsrURI="80e3f780-3654-448f.b4d4.946ca894d4b7"
correlationId="Correlation_SOAPolicy" securityToken="SecurityToken_SOAPolicy"
type="Subscription">
    <body:notificationResource event="ATTACH" resourceBsrURI="" resourceName=""
resourceToSubscribedRelationship="attachedPolicy" resourceType="PolicyExpression"
subscribedBsrUri="1e5e791e-9701-417f.8bd2.07fe5807d2d6" subscribedName="SLD - Itinerary
Availability"
subscribedPrimaryType="http://www.ibm.com/xmlns/prod/serviceregistry/profile/v6r3/Governanc
eEnablementModel#ServiceLevelDefinition" subscribedType="GenericObject">
      <policyUri>urn:TooManyMessages</policyUri>
    </body:notificationResource>
  </body:resource>
</body:resources>]
[11/7/12 10:26:11:358 EST] 00000028 WSRRNotificat E   KD4DM0487E Failed to parse the WSRR
notification, reason : [org.xml.sax.SAXException: KD4DM0209F Validation fatal. Line number:
5, column number 22, message An invalid XML character (Unicode: 0xe) was found in the
element content of the document..]

com.ibm.management.soa.dms.wsrrsync.xml.sax.ParsingException: org.xml.sax.SAXException:
KD4DM0209F Validation fatal. Line number: 5, column number 22, message An invalid XML
character (Unicode: 0xe) was found in the element content of the document..
   at
com.ibm.management.soa.dms.wsrrsync.xml.sax.GenericSAXParser.parse(GenericSAXParser.java:15
2)
   at
com.ibm.management.soa.dms.wsrrsync.subscription.event.WSRRNotificationParser.parse(WSRRNot
ificationParser.java:101)
   at
com.ibm.management.soa.dms.wsrrsync.servlet.WSRRNotificationServlet.doPost(WSRRNotification
Servlet.java:297)
   at javax.servlet.http.HttpServlet.service(HttpServlet.java:763)
   at javax.servlet.http.HttpServlet.service(HttpServlet.java:856)
   at com.ibm.ws.webcontainer.servlet.ServletWrapper.service(ServletWrapper.java:1152)
   at com.ibm.ws.webcontainer.servlet.ServletWrapper.handleRequest(ServletWrapper.java:592)
   at
com.ibm.ws.wswebcontainer.servlet.ServletWrapper.handleRequest(ServletWrapper.java:526)
   at
com.ibm.ws.webcontainer.servlet.CacheServletWrapper.handleRequest(CacheServletWrapper.java:
90)
   at com.ibm.ws.webcontainer.WebContainer.handleRequest(WebContainer.java:764)
   at com.ibm.ws.wswebcontainer.WebContainer.handleRequest(WebContainer.java:1478)
```

```
   at com.ibm.ws.webcontainer.channel.WCChannelLink.ready(WCChannelLink.java:133)
   at
com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.handleDiscrimination(HttpInboundLink.j
ava:450)
   at
com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.handleNewRequest(HttpInboundLink.java:
508)
   at
com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.processRequest(HttpInboundLink.java:29
6)
   at com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.ready(HttpInboundLink.java:270)
   at
com.ibm.ws.tcp.channel.impl.NewConnectionInitialReadCallback.sendToDiscriminators(NewConnec
tionInitialReadCallback.java:214)
   at
com.ibm.ws.tcp.channel.impl.NewConnectionInitialReadCallback.complete(NewConnectionInitialR
eadCallback.java:113)
   at
com.ibm.ws.tcp.channel.impl.AioReadCompletionListener.futureCompleted(AioReadCompletionList
ener.java:165)
   at com.ibm.io.async.AbstractAsyncFuture.invokeCallback(AbstractAsyncFuture.java:217)
   at
com.ibm.io.async.AsyncChannelFuture.fireCompletionActions(AsyncChannelFuture.java:161)
   at com.ibm.io.async.AsyncFuture.completed(AsyncFuture.java:136)
   at com.ibm.io.async.ResultHandler.complete(ResultHandler.java:196)
   at com.ibm.io.async.ResultHandler.runEventProcessingLoop(ResultHandler.java:751)
   at com.ibm.io.async.ResultHandler$2.run(ResultHandler.java:881)
   at com.ibm.ws.util.ThreadPool$Worker.run(ThreadPool.java:1551)
Caused by: org.xml.sax.SAXException: KD4DM0209F Validation fatal. Line number: 5, column
number 22, message An invalid XML character (Unicode: 0xe) was found in the element content
of the document..
   at
com.ibm.management.soa.dms.wsrrsync.xml.sax.GenericSAXHandler.fatalError(GenericSAXHandler.
java:281)
   at org.apache.xerces.util.ErrorHandlerWrapper.fatalError(Unknown Source)
   at org.apache.xerces.impl.XMLErrorReporter.reportError(Unknown Source)
   at org.apache.xerces.impl.XMLErrorReporter.reportError(Unknown Source)
   at org.apache.xerces.impl.XMLScanner.reportFatalError(Unknown Source)
   at
org.apache.xerces.impl.XMLDocumentFragmentScannerImpl$FragmentContentDispatcher.dispatch(Un
known Source)
   at org.apache.xerces.impl.XMLDocumentFragmentScannerImpl.scanDocument(Unknown Source)
   at org.apache.xerces.parsers.XML11Configuration.parse(Unknown Source)
   at org.apache.xerces.parsers.XML11Configuration.parse(Unknown Source)
   at org.apache.xerces.parsers.XMLParser.parse(Unknown Source)
   at org.apache.xerces.parsers.AbstractSAXParser.parse(Unknown Source)
   at org.apache.xerces.jaxp.SAXParserImpl.parse(Unknown Source)
   at javax.xml.parsers.SAXParser.parse(Unknown Source)
   at
com.ibm.management.soa.dms.wsrrsync.xml.sax.GenericSAXParser.parse(GenericSAXParser.java:14
1)
   ... 25
more
```

Missing situation field

In this log (Example 14-3), the `Identity` policy does not contain a required field. Although the policy is a valid WSRR policy, ITCAM requires a field for this specific table.

The solution is to add the missing field.

Example 14-3 Missing situation field error

```
[10/26/12 14:45:06:629 EDT] 00000e45 WSRRClientBas 1    executeQuery : executing query
[https://sa-w217rhel-2.itso.ral.ibm.com:9443/WSRR/7.5/Content/ee80faee-8130-4053.b672.52099
0527272]
[10/26/12 14:45:06:634 EDT] 00000e45 WSRRClientBas 2   Opening connection to
sa-w217rhel-2.itso.ral.ibm.com:9443...
[10/26/12 14:45:06:635 EDT] 00000e45 WSRRClientBas 2   Starting SSL handshake...
[10/26/12 14:45:06:680 EDT] 00000e45 WSRRClientBas 2   No errors, certificate is already
trusted
[10/26/12 14:45:07:016 EDT] 00000e45 WSRRClientBas 1    executeQuery : query result [
<?xml version="1.0"?>
<wsp:Policy xmlns:wsp="http://www.w3.org/2006/07/ws-policy"
xmlns:wsrr="http://www.ibm.com/xmlns/prod/serviceregistry/6/2/wspolicy"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.x
sd" xmlns:wsmon="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring"
wsrr:policyClass="WSMNPolicyClass" Name="urn:Identity"
wsrr:policyClassDomain="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring">
  <wsmon:Rule>
    <wsmon:Condition>
      <wsmon:Expression>
        <wsmon:Services_Inventory_ReqID_610>
          <wsmon:AllOfAssertion_SIReqID>
            <wsmon:Msg_Count
                 wsmon:expression="GT"
                 wsmon:value="4"/>
            <wsmon:Msg_Count
                 wsmon:expression="LT"
                 wsmon:value="66"/>
          </wsmon:AllOfAssertion_SIReqID>
        </wsmon:Services_Inventory_ReqID_610>
      </wsmon:Expression>
    </wsmon:Condition>
    <wsmon:Action/>
  </wsmon:Rule>
</wsp:Policy>
] for query
[https://sa-w217rhel-2.itso.ral.ibm.com:9443/WSRR/7.5/Content/ee80faee-8130-4053.b672.52099
0527272]
[10/26/12 14:45:07:016 EDT] 00000e45 WSRRSOASyncMa 1    getValidSLAPolicyAttachments : The
policy received from WSRR : [
<?xml version="1.0"?>
<wsp:Policy xmlns:wsp="http://www.w3.org/2006/07/ws-policy"
xmlns:wsrr="http://www.ibm.com/xmlns/prod/serviceregistry/6/2/wspolicy"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.x
sd" xmlns:wsmon="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring"
wsrr:policyClass="WSMNPolicyClass" Name="urn:Identity"
wsrr:policyClassDomain="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring">
  <wsmon:Rule>
    <wsmon:Condition>
      <wsmon:Expression>
        <wsmon:Services_Inventory_ReqID_610>
          <wsmon:AllOfAssertion_SIReqID>
            <wsmon:Msg_Count
```

```
                wsmon:expression="GT"
                wsmon:value="4"/>
            <wsmon:Msg_Count
                wsmon:expression="LT"
                wsmon:value="66"/>
          </wsmon:AllOfAssertion_SIReqID>
        </wsmon:Services_Inventory_ReqID_61O>
      </wsmon:Expression>
    </wsmon:Condition>
    <wsmon:Action/>
  </wsmon:Rule>
</wsp:Policy>
]
[10/26/12 14:45:07:017 EDT] 00000e45 WSRRPolicyPar I   No
{http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring}Sampling interval found.  The
default sampling interval will be used.
[10/26/12 14:45:07:019 EDT] 00000e45 WSRRPolicyAct I   Found and ignoring empty
<wsmon:Action> element
[10/26/12 14:45:07:021 EDT] 00000e45 SystemErr     R   java.lang.Exception: Error found in
formula "(*VALUE Services_Inventory_ReqID_61O.Msg_Count *GT 4 *AND *VALUE
Services_Inventory_ReqID_61O.Msg_Count *LT 66)".  A situation relating to the Services
Inventory ReqID Table must contain a value for the RequesterID attribute.
[10/26/12 14:45:07:021 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.policy.PolicyFormulaValidator.validateSIReqIDSituation(
PolicyFormulaValidator.java:203)
[10/26/12 14:45:07:021 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.policy.PolicyFormulaValidator.validateFormula(PolicyFor
mulaValidator.java:76)
[10/26/12 14:45:07:021 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.policy.WSRRPolicyParser.parseMonitoringPolicy(WSRRPolic
yParser.java:132)
[10/26/12 14:45:07:021 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.getValidSLAPolicyAttachments(WS
RRSOASyncManager.java:2414)
[10/26/12 14:45:07:021 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.createOrUpdateSESituations(WSRR
SOASyncManager.java:2080)
[10/26/12 14:45:07:022 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.resyncSLESituations(WSRRSOASync
Manager.java:3495)
[10/26/12 14:45:07:022 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.processWSRRNotification(WSRRSOA
SyncManager.java:297)
[10/26/12 14:45:07:022 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRPolicySyncBean.processWSRRNotification(WSRRPol
icySyncBean.java:89)
[10/26/12 14:45:07:022 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.EJSRemoteStatelessWSRRPolicySync_ca672eb8.processW
SRRNotification(Unknown Source)
[10/26/12 14:45:07:022 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs._WSRRPolicySync_Stub.processWSRRNotification(_WSRR
PolicySync_Stub.java:269)
[10/26/12 14:45:07:022 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.servlet.WSRRResyncTask.execute(WSRRResyncTask.java:187)
[10/26/12 14:45:07:022 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.servlet.WSRRResyncTask.run(WSRRResyncTask.java:144)
[10/26/12 14:45:07:022 EDT] 00000e45 SystemErr     R   at
java.util.concurrent.ThreadPoolExecutor$Worker.runTask(ThreadPoolExecutor.java:678)
[10/26/12 14:45:07:022 EDT] 00000e45 SystemErr     R   at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:703)
```

```
[10/26/12 14:45:07:022 EDT] 00000e45 SystemErr     R   at
java.lang.Thread.run(Thread.java:811)
[10/26/12 14:45:07:022 EDT] 00000e45 WSRRSOASyncMa E   KD4DM0433E A problem was encountered
with parsing the policy, reason [Error found in formula "(*VALUE
Services_Inventory_ReqID_610.Msg_Count *GT 4 *AND *VALUE
Services_Inventory_ReqID_610.Msg_Count *LT 66)".  A situation relating to the Services
Inventory ReqID Table must contain a value for the RequesterID attribute.], continuing to
process...
                                java.lang.Exception: Error found in formula "(*VALUE
Services_Inventory_ReqID_610.Msg_Count *GT 4 *AND *VALUE
Services_Inventory_ReqID_610.Msg_Count *LT 66)".  A situation relating to the Services
Inventory ReqID Table must contain a value for the RequesterID attribute.
   at
com.ibm.management.soa.dms.wsrrsync.policy.WSRRPolicyParser.parseMonitoringPolicy(WSRRPolic
yParser.java:139)
   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.getValidSLAPolicyAttachments(WS
RRSOASyncManager.java:2414)
   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.createOrUpdateSESituations(WSRR
SOASyncManager.java:2080)
```

Nonexistent table

In the next case (Example 14-4), a policy was created against the `Message Arrival Threshold Table`. The problem arises when the ITCAM is version 7.1.1.3 because WSRR can create situations only against that table from version 7.2 and later. Note that the trace log does not show the ITCAM version.

The solution is to either ignore the policy or upgrade ITCAM.

Example 14-4 Nonexistent table error

```
[10/26/12 14:45:05:841 EDT] 00000e45 WSRRClientBas 1    executeQuery : executing query
[https://sa-w217rhel-2.itso.ral.ibm.com:9443/WSRR/7.5/Content/c8c1e8c8-d2e3-43fd.b41b.2ab66
42a1b52]
[10/26/12 14:45:05:846 EDT] 00000e45 WSRRClientBas 2   Opening connection to
sa-w217rhel-2.itso.ral.ibm.com:9443...
[10/26/12 14:45:05:847 EDT] 00000e45 WSRRClientBas 2   Starting SSL handshake...
[10/26/12 14:45:05:893 EDT] 00000e45 WSRRClientBas 2   No errors, certificate is already
trusted
[10/26/12 14:45:06:232 EDT] 00000e45 WSRRClientBas 1    executeQuery : query result [
<?xml version="1.0"?>
<wsp:Policy
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.x
sd" xmlns:wsrr="http://www.ibm.com/xmlns/prod/serviceregistry/6/2/wspolicy"
xmlns:wsp="http://www.w3.org/ns/ws-policy"
xmlns:wsmon="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring"
wsrr:policyClass="WSMNPolicyClass" Name="urn:MessageArrivalThreshold "
wsrr:policyClassDomain="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring">
  <wsmon:Rule>
    <wsmon:Condition>
      <wsmon:Expression>
        <wsmon:Message_Arrival_Threshold_Table_610>
          <wsmon:AllOfAssertion_MAT>
            <wsmon:Message_Count
                 wsmon:expression="EQ"
                 wsmon:value="10"/>
          </wsmon:AllOfAssertion_MAT>
        </wsmon:Message_Arrival_Threshold_Table_610>
```

```
      </wsmon:Expression>
    </wsmon:Condition>
    <wsmon:Action>
      <wsmon:ITM>
            <wsmon:Informational></wsmon:Informational>
      </wsmon:ITM>
    </wsmon:Action>
  </wsmon:Rule>
</wsp:Policy>
] for query
[https://sa-w217rhel-2.itso.ral.ibm.com:9443/WSRR/7.5/Content/c8c1e8c8-d2e3-43fd.b41b.2ab66
42a1b52]
[10/26/12 14:45:06:233 EDT] 00000e45 WSRRSOASyncMa 1    getValidSLAPolicyAttachments : The
policy received from WSRR : [
<?xml version="1.0"?>
<wsp:Policy
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.x
sd" xmlns:wsrr="http://www.ibm.com/xmlns/prod/serviceregistry/6/2/wspolicy"
xmlns:wsp="http://www.w3.org/ns/ws-policy"
xmlns:wsmon="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring"
wsrr:policyClass="WSMNPolicyClass" Name="urn:MessageArrivalThreshold "
wsrr:policyClassDomain="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring">
  <wsmon:Rule>
    <wsmon:Condition>
      <wsmon:Expression>
        <wsmon:Message_Arrival_Threshold_Table_610>
          <wsmon:AllOfAssertion_MAT>
            <wsmon:Message_Count
                 wsmon:expression="EQ"
                 wsmon:value="10"/>
          </wsmon:AllOfAssertion_MAT>
        </wsmon:Message_Arrival_Threshold_Table_610>
      </wsmon:Expression>
    </wsmon:Condition>
    <wsmon:Action>
      <wsmon:ITM>
            <wsmon:Informational></wsmon:Informational>
      </wsmon:ITM>
    </wsmon:Action>
  </wsmon:Rule>
</wsp:Policy>
]
[10/26/12 14:45:06:233 EDT] 00000e45 WSRRPolicyPar I   No
{http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring}Sampling interval found.  The
default sampling interval will be used.
[10/26/12 14:45:06:235 EDT] 00000e45 WSRRPolicyCon I   Problem found in element
<wsmon:Message_Arrival_Threshold_Table_610>
[10/26/12 14:45:06:237 EDT] 00000e45 WSRRPolicyCon I   Found and ignoring unexpected node:
{http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring}Message_Arrival_Threshold_Table_610
[10/26/12 14:45:06:239 EDT] 00000e45 SystemErr     R   java.lang.Exception: XML is
malformed - no valid {http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring}KD4 Table
element found
[10/26/12 14:45:06:239 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.policy.WSRRPolicyConditionParser.processExpression(WSRR
PolicyConditionParser.java:192)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.policy.WSRRPolicyConditionParser.processCondition(WSRRP
olicyConditionParser.java:125)
```

```
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.policy.WSRRPolicyConditionParser.getCondition(WSRRPolic
yConditionParser.java:107)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.policy.WSRRPolicyParser.processRule(WSRRPolicyParser.ja
va:239)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.policy.WSRRPolicyParser.buildMonitoringPolicyFromXML(WS
RRPolicyParser.java:149)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.policy.WSRRPolicyParser.parseMonitoringPolicy(WSRRPolic
yParser.java:125)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.getValidSLAPolicyAttachments(WS
RRSOASyncManager.java:2414)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.createOrUpdateSESituations(WSRR
SOASyncManager.java:2080)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.resyncSLESituations(WSRRSOASync
Manager.java:3495)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.processWSRRNotification(WSRRSOA
SyncManager.java:297)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRPolicySyncBean.processWSRRNotification(WSRRPol
icySyncBean.java:89)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs.EJSRemoteStatelessWSRRPolicySync_ca672eb8.processW
SRRNotification(Unknown Source)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.ejbs._WSRRPolicySync_Stub.processWSRRNotification(_WSRR
PolicySync_Stub.java:269)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.servlet.WSRRResyncTask.execute(WSRRResyncTask.java:187)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
com.ibm.management.soa.dms.wsrrsync.servlet.WSRRResyncTask.run(WSRRResyncTask.java:144)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
java.util.concurrent.ThreadPoolExecutor$Worker.runTask(ThreadPoolExecutor.java:678)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:703)
[10/26/12 14:45:06:240 EDT] 00000e45 SystemErr     R   at
java.lang.Thread.run(Thread.java:811)
[10/26/12 14:45:06:240 EDT] 00000e45 WSRRSOASyncMa E   KD4DM0433E A problem was encountered
with parsing the policy, reason [XML is malformed - no valid
{http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring}KD4 Table element found], continuing
to process...
                                java.lang.Exception: XML is malformed - no valid
{http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring}KD4 Table element found
   at
com.ibm.management.soa.dms.wsrrsync.policy.WSRRPolicyParser.parseMonitoringPolicy(WSRRPolic
yParser.java:139)
   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.getValidSLAPolicyAttachments(WS
RRSOASyncManager.java:2414)
   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.createOrUpdateSESituations(WSRR
SOASyncManager.java:2080)
```

```
   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.resyncSLESituations(WSRRSOASync
Manager.java:3495)
   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRSOASyncManager.processWSRRNotification(WSRRSOA
SyncManager.java:297)
   at
com.ibm.management.soa.dms.wsrrsync.ejbs.WSRRPolicySyncBean.processWSRRNotification(WSRRPol
icySyncBean.java:89)
   at
com.ibm.management.soa.dms.wsrrsync.ejbs.EJSRemoteStatelessWSRRPolicySync_ca672eb8.processW
SRRNotification(Unknown Source)
   at
com.ibm.management.soa.dms.wsrrsync.ejbs._WSRRPolicySync_Stub.processWSRRNotification(_WSRR
PolicySync_Stub.java:269)
   at
com.ibm.management.soa.dms.wsrrsync.servlet.WSRRResyncTask.execute(WSRRResyncTask.java:187)
   at
com.ibm.management.soa.dms.wsrrsync.servlet.WSRRResyncTask.run(WSRRResyncTask.java:144)
   at java.util.concurrent.ThreadPoolExecutor$Worker.runTask(ThreadPoolExecutor.java:678)
   at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:703)
   at java.lang.Thread.run(Thread.java:811)
```

Valid policy

The log in Example 14-5 shows no error. For a valid policy, the log in the example is what you can expect to see in the log, that is, no error.

Example 14-5 No error

```
[10/26/12 14:45:06:243 EDT] 00000e45 WSRRClientBas 1    executeQuery : executing query
[https://sa-w217rhel-2.itso.ral.ibm.com:9443/WSRR/7.5/Content/0703a807-15b7-478f.aaeb.a8684
1a8eb5e]
[10/26/12 14:45:06:248 EDT] 00000e45 WSRRClientBas 2   Opening connection to
sa-w217rhel-2.itso.ral.ibm.com:9443...
[10/26/12 14:45:06:249 EDT] 00000e45 WSRRClientBas 2   Starting SSL handshake...
[10/26/12 14:45:06:294 EDT] 00000e45 WSRRClientBas 2   No errors, certificate is already
trusted
[10/26/12 14:45:06:628 EDT] 00000e45 WSRRClientBas 1    executeQuery : query result [
<?xml version="1.0"?>
<wsp:Policy xmlns:wsp="http://www.w3.org/2006/07/ws-policy"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.x
sd" xmlns:wsrr="http://www.ibm.com/xmlns/prod/serviceregistry/6/2/wspolicy"
xmlns:wsmon="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring"
wsrr:policyClass="WSMNPolicyClass" Name="urn:testMonPolicy"
wsrr:policyClassDomain="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring">
  <wsmon:Rule
       wsmon:SamplingInterval="60">
    <wsmon:Condition>
      <wsmon:Expression>
        <wsmon:Services_Inventory_610>
          <wsmon:AllOfAssertion_SI>
            <wsmon:Msg_Count
                 wsmon:expression="GE"
                 wsmon:value="5"/>
          </wsmon:AllOfAssertion_SI>
        </wsmon:Services_Inventory_610>
      </wsmon:Expression>
    </wsmon:Condition>
    <wsmon:Action>
```

```
      <wsmon:ITM>
             <wsmon:Warning></wsmon:Warning>
      </wsmon:ITM>
    </wsmon:Action>
  </wsmon:Rule>
</wsp:Policy>
] for query
[https://sa-w217rhel-2.itso.ral.ibm.com:9443/WSRR/7.5/Content/0703a807-15b7-478f.aaeb.a8684
1a8eb5e]
[10/26/12 14:45:06:628 EDT] 00000e45 WSRRSOASyncMa 1    getValidSLAPolicyAttachments : The
policy received from WSRR : [
<?xml version="1.0"?>
<wsp:Policy xmlns:wsp="http://www.w3.org/2006/07/ws-policy"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.x
sd" xmlns:wsrr="http://www.ibm.com/xmlns/prod/serviceregistry/6/2/wspolicy"
xmlns:wsmon="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring"
wsrr:policyClass="WSMNPolicyClass" Name="urn:testMonPolicy"
wsrr:policyClassDomain="http://www.ibm.com/xmlns/stdwip/2011/02/ws-monitoring">
  <wsmon:Rule
       wsmon:SamplingInterval="60">
    <wsmon:Condition>
      <wsmon:Expression>
        <wsmon:Services_Inventory_610>
          <wsmon:AllOfAssertion_SI>
            <wsmon:Msg_Count
                 wsmon:expression="GE"
                 wsmon:value="5"/>
          </wsmon:AllOfAssertion_SI>
        </wsmon:Services_Inventory_610>
      </wsmon:Expression>
    </wsmon:Condition>
    <wsmon:Action>
      <wsmon:ITM>
             <wsmon:Warning></wsmon:Warning>
      </wsmon:ITM>
    </wsmon:Action>
  </wsmon:Rule>
</wsp:Policy>
]
[10/26/12 14:45:06:629 EDT] 00000e45 WSRRSOASyncMa 1    getValidSLAPolicyAttachments : The
contents of monitoring policy : [
table:  Services_Inventory_610
formula:  (*VALUE Services_Inventory_610.Msg_Count *GE 5)
sampling Interval:  60
advice:
state:  Warning
systemCommand:
notifyAction:   {}
```

14.7 Operations notification and upstream integration with other Tivoli products

ITCAM, by itself, does not have a built-in capability to send email or open trouble tickets. There are two options to notify operators of events:

- The ITCAM portal itself, which shows the situations that fired.
- Running an automatic action when a situation fires that will run a script to send an email, SMS, or any other option. This option requires the development of special integration tools by the customer.

However, ITCAM is part of the Tivoli solution suite and if advanced notification or other capabilities are required (such as “send a mail if over 10 business services are slow” or “open one trouble ticket for 5 similar events”) then integration with other tools, such as Tivoli Netcool/OMNIbus and Tivoli Netcool/ Impact, are required.

Tivoli Netcool/OMNIbus can be considered a *manager of managers*. Many customers use IBM Tivoli Netcool/OMNIbus to manage tens of millions of events daily. The software can be deployed in a distributed, parallel, or hierarchical fashion to support complex operations environments that span diverse geographic boundaries. Because it couples scalability with a flexible architecture, the software can deliver robust event management to support environments of any size. When the software, such as ITCAM for SOA, fires events, they are processed in the ObjectServer, a high-speed, in-memory database that collects events from across the infrastructure in real time. Netcool/OMNIbus then eliminates duplicate events and filters events through an advanced problem escalation engine. Information from outside ITCAM can be used to hone in on the most critical problems and even automate the isolation and resolution of those problems.

Another component that can be used upstream is IBM Tivoli Netcool/Impact. This component enriches OMNIbus and streamlines event and alert management, business service management, and incident and problem management processes by enabling context-driven correlation. It provides real-time access to actionable intelligence, and improves efficiency through automation. Impact can catch an ITCAM for SOA event that affects a specific service, query WSRR as to the business owner, and access the organizational LDAP to get the updated contact details. This information is inserted into the notification event, which the operations team receives. In this way, the team is provided with extra information that can help them handle the problem faster.

Impact can also dramatically reduce event and incident volumes by suppressing maintenance events, non-service impacting events, and false alarms, and by contextually correlating multiple events into a single actionable event.

IBM Tivoli Business Service Manager uses the events that are collected by ITCAM, OMNIbus, and Impact and presents the integrated visibility, control, and automation that lines of businesses and IT operations need to help improve efficiency, reduce costs, and assure services.

Tivoli Business Service Manager is a portal that shows business dashboards, which provide real-time balanced scorecards and KPIs for the lines of business. This information is needed for making informed decisions and effectively managing profit and loss, including vital indicators such as revenue, transaction, customer, bottlenecks, and call volume.

The operational dashboards in Tivoli Business Service Manager provide visibility into real-time service health and integrity, service dependencies and KPIs that are needed to

deliver against service and operational objectives, including SLA tracking, impact and root-cause analysis, event views, business activity, and process workflow metrics.

ITCAM-OMNIbus integration is beyond the scope of this book and must be designed by a Tivoli architect. The basic integration is detailed in the information center:

http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3fp1/itm623FP1_install601.htm?path=3_0_3_0_9_1#omnibus_install

The SDMS configuration file can set all WSRR situations to be forwarded to OMNIbus by setting the `situation.tec.destination` parameter to point to OMNIbus, as detailed in 14.4, “Configuring for integration of WSRR and ITCAM for Applications” on page 394.

In this case, to continue receiving the ITCAM situations in WSRR, you must either set the SDMS to forward events to two locations (OMNIbus and WSRR) or configure OMNIbus to forward the ITCAM for SOA events to WSRR. This step is slightly more work at the beginning, but simplifies long-term support. Details are at the following website:

http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/twsr_conftivolinetcoolomnibus.html

14.8 ITCAM monitoring agent for DataPower policies

Part of the extended downloadable material for this book is a Tivoli Monitoring agent that can collect WS-Management information from DataPower. Essentially, the agent can collect performance metrics on the policies that DataPower is running. This information can be used to create situations to alert when specific policies are triggered or triggered too often. It can also be saved in the Tivoli Monitoring data warehouse, and long term information can be gathered, for example, policies that were triggered over the last six months.

After the agent is installed, the DataPower must be configured to allow the agent to access it and the agent must be configured to access the DataPower.

Use the following steps to allow the ITCAM DataPower collector to request the DataPower device that is using invocations based on WS-Management:

1. Connect to the `default` domain of the DataPower appliance using the administrator account.
2. Navigate to the XML Management Interface (Figure 14-6 on page 411) by selecting **Network** → **Management** → **XML Management Interface**.

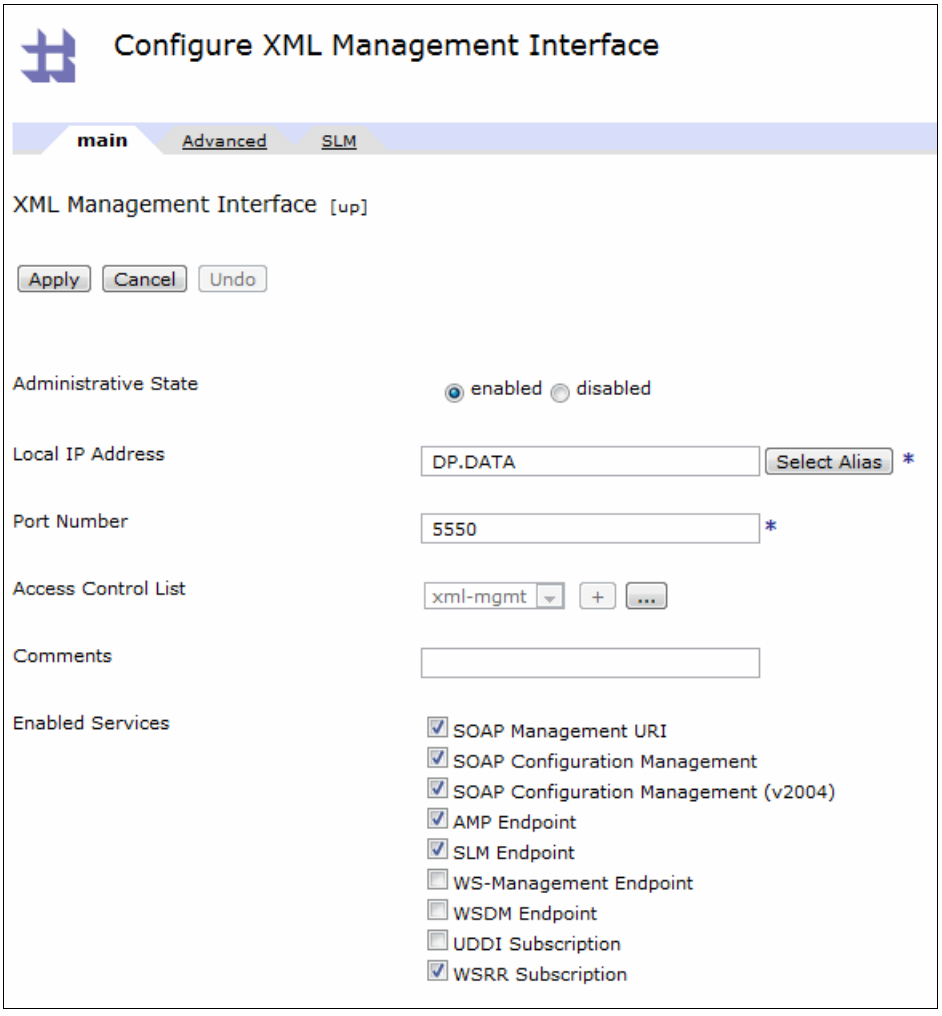


Figure 14-6 Main panel of the XML Management Interface

3. In the list of available Enabled Services properties, select the **WS-Management Endpoint** check box, as show in Figure 14-7.

Figure 14-7 WS-Management Endpoint selected

4. Enable the XML Management Interface service, as shown in Figure 14-8.

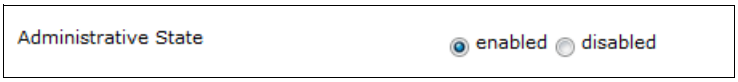


Figure 14-8 XML management interface enabled

5. Click **Apply** to save your changes.
6. Click the **Save Config** link to make your configuration persistent.

Configuring the agent is performed as for any ITCAM agent, during initial installation or by reconfiguring the agent. as shown in Figure 14-9.

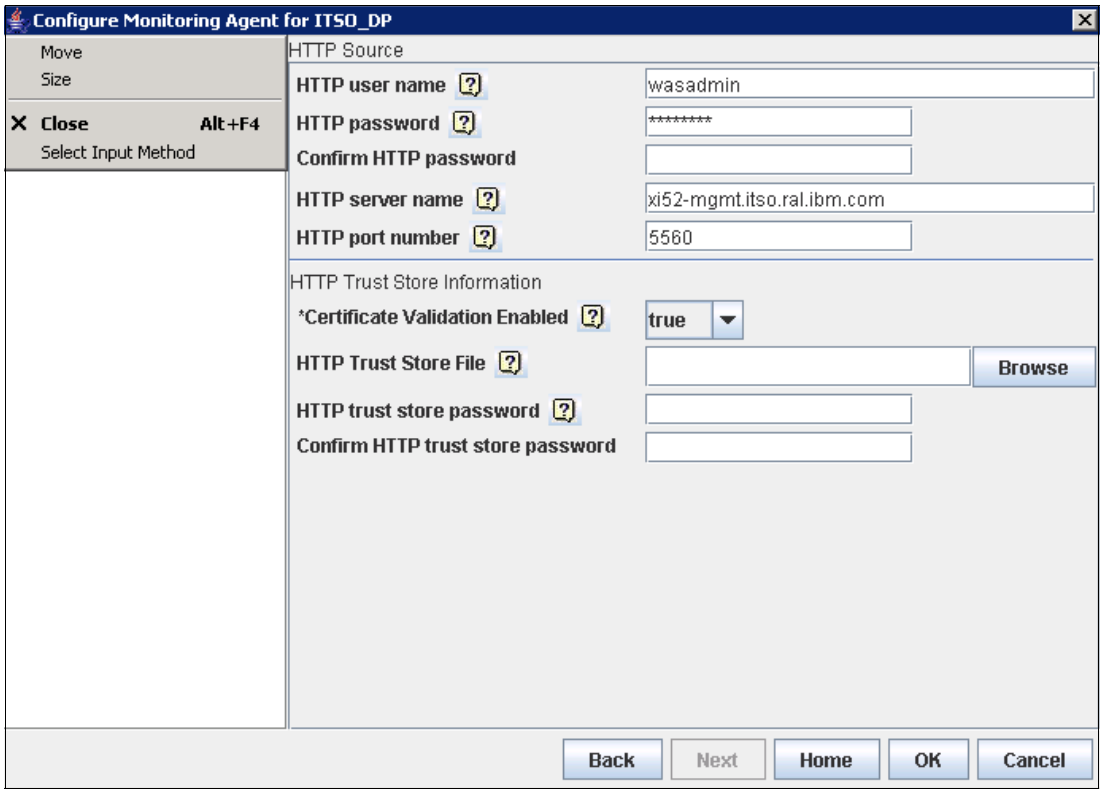


Figure 14-9 Reconfigure the ITCAM agent for DataPower

Part 5

Appendixes

This part contains the following appendix information:

- Appendix A, “Implementing a SOA Policy Solution flow of work” on page 415
- Appendix B, “ITCAM monitoring attribute tables” on page 421
- Appendix C, “Additional material” on page 427

© Copyright IBM Corp. 2013. All rights reserved.

A

Implementing a SOA Policy Solution flow of work

This appendix provides the flow of activities necessary for the reasonably proficient practitioner to use the files from this book to create a proof of concept (PoC) or proof of technology (PoT).

This appendix contains the following topics:

- A.1, “Installing the samples from this book” on page 416
- A.2, “Validating installation of each individual product in the SOA Policy Solution” on page 417
- A.3, “Integrating products in the SOA Policy Solution” on page 418
- A.4, “Governing of services” on page 419
- A.5, “Creating Policies” on page 420
- A.6, “Attaching policies” on page 420
- A.7, “Promotion within WSRR” on page 420
- A.8, “Running the client to validate the services” on page 420

© Copyright IBM Corp. 2013. All rights reserved.

A.1 Installing the samples from this book

A file that contains the examples that are described in Part 2, “Policy examples” on page 33 is at the following website:

ftp://www.redbooks.ibm.com/redbooks/SG248101/

You can use the sample services in the file or use your own. A good approach is to use the sample services in the file to gain familiarity with the SOA Policy Solution capabilities and then start to use your own services after gaining that expertise.

The file contains the following set of capabilities:

- IBM Tivoli Composite Application Manager (ITCAM)

  The ITCAM agent in the file displays information about the DataPower policy use. It accesses the WS-Management interface of the DataPower and that information can be used as standard ITCAM data. See 14.8, “ITCAM monitoring agent for DataPower policies” on page 410 for more details.

- IBM WebSphere Service Registry and Repository (WSRR)

  Included in the additional web material that accompanies this book are files that can be imported into your three WSRR instances. See “Import objects into WebSphere Service Registry and Repository” on page 433 for details about importing them.

- IBM WebSphere DataPower

  The following artifacts are available in the additional web material that accompanies this book:

  - Custom policies for versioning and security use cases:
    - Custom policy `aaa-customPolicy.xml` file is used in Chapter 6, “Security using custom policy” on page 187.
    - Custom policy `versioning-customPolicy.xml` file is used in Chapter 5, “Versioning with custom policy” on page 119.
  - Custom policy style sheets used for versioning and security use cases:
    - Custom policy style sheet `itso.customPolicy.aaa.xsl` is used in Chapter 6, “Security using custom policy” on page 187.
    - Custom policy style sheet `itso.customPolicy.versioning.xsl` file is used in Chapter 5, “Versioning with custom policy” on page 119.
  - Custom policy style sheet `itso.customPolicy.example.xsl` is used as an example in Chapter 13, “Creating and using custom policies” on page 367.
  - Custom style sheet `itso.pricingService.v1-v2.xsl` file used to perform data mediation between version 1 and version 2 (in both directions) of the Pricing service. This style sheet is used in Chapter 5, “Versioning with custom policy” on page 119.
  - DataPower AAA `AAAInfo.xml` information file is used in the authentication step of the DataPower AAA Policy in Chapter 6, “Security using custom policy” on page 187.

- IBM WebSphere Application Server

  Included in the additional web material that accompanies this book is an EAR file with sample web services and a web-based client for these services. See “Deploy sample web services into WebSphere Application Server” on page 428 for instructions about deploying the services.

A.2 Validating installation of each individual product in the SOA Policy Solution

You must validate that all four products listed in the previous section are correctly installed and working correctly. The following sections provide validation information. Also, see the information centers to learn more about the products.

In some instances, you might decide not to use WebSphere DataPower or ITCAM. For example, you can write monitoring policy only using WSRR with services that are running on an application server, and monitoring that is taking place on ITCAM with no WebSphere DataPower usage. In a similar manner, you can write mediation policy only using WSRR with services running on an application server and transaction mediation taking place on WebSphere DataPower. In those instances, validating proper installation of WebSphere DataPower or ITCAM is obviously not necessary.

A.2.1 IBM Tivoli Composite Application Manager

IBM Tivoli Composite Application Manager (ITCAM) installation consists of three stages:

1. Installation of IBM Tivoli Monitoring base:

   http://pic.dhe.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.3fp1/itm623fp1_qsg_en.htm

2. Installation of ITCAM for SOA:

   http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_install_guide/part1.html

3. Installation of ITCAM for SOA agent to collect data from SOA Application Server and DataPower:

   - http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_install_guide/part3a.html
   - http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_install_guide/part2.html
   - http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_install_guide/configdatapower.html

Verification of the installation is described at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_install_guide/verifyinstall.html

For troubleshooting information, see 14.5, “Troubleshooting the installation” on page 396.

After you complete these steps, you see performance data in the ITCAM portal.

A.2.2 IBM WebSphere Service Registry and Repository

WebSphere Service Registry and Repository (WSRR) is an application that runs on WebSphere Application Server and persists its data in a database, typically DB2. The information center has helpful information about installation:

http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.intinst.doc/roadmap_form_wsrr.html

For a typical WSRR installation, WebSphere Application Server and DB2 are installed.

A.2.3 IBM WebSphere DataPower

WebSphere DataPower appliances simplify, govern, and optimize the delivery of services and applications, and enhance the security of XML and IT services. They extend the capabilities of an infrastructure by providing a multitude of functions. The DataPower SOA appliances information center provides the required information about how to install every kind of appliances in regard to its hardware generation (model XI52, type 7199 with firmware 5.0.0 is used in this book). See the following sources for more information:

- IBM WebSphere DataPower 5.0 information center:

  http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/index.jsp

- Installation steps of a DataPower SOA appliance:

  http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/nav/0_0

A.3 Integrating products in the SOA Policy Solution

This section addresses where to find the information you need to complete the configuration of the products in the SOA Policy Solution so that the policies and services automatically flow where they are necessary.

A.3.1 ITCAM with WSRR

ITCAM and WSRR integration consists of two stages:

1. Configuring the ITCAM-WSRR integration to create situations from policies:

   http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/WSRR_Integration_Guide_72/soawsrr.html

2. Configuring the WSRR listener to receive events from ITCAM:

   http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/cwsr_itcamforsoaintegration.html

Further configuration details are in 14.4, “Configuring for integration of WSRR and ITCAM for Applications” on page 394.

There is no way to perform a full integration test without performing the full work of creating a policy, SLD, and so on. This information is discussed in 7.3.1, “Steps to create a policy” on page 240. Until then, running the **kd4WSRRITMSynchronization** script and ensuring that no errors are returned is the best option.

Likewise, to test that the ITCAM listener is operating in WSRR, the simplest test is to use Telnet to connect to the correct port, send a test message, and check that the message appears in the EIF log of the listener.

A.3.2 IBM WebSphere DataPower with ITCAM and WSRR

When integrating with a monitoring product, the IBM WebSphere DataPower appliance is the data collector and the monitoring product is the data consumer.

To provide transaction data for a monitoring product such as ITCAM for SOA, you must enable the appliance to process the Web Services Management (WS-Management) request by configuring the XML Management Interface of a DataPower appliance.

In each application domain on a DataPower appliance, the Web Services Management Agent collects transaction data. Each application domain contains a Web Services Management Agent that can be configured to meet domain-specific requirements.

For detailed information about DataPower integration with ITCAM for SOA, see the following website:

http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/index.jsp?topic=%2Fcom.ibm.dp.xi.doc%2Fwsm_collectingwsmdata.html

For a detailed description of DataPower and WSRR integration, see Chapter 12, “DataPower policy enforcement point configuration” on page 347.

A.4 Governing of services

Services have a governance state that must be transitioned appropriately to be deployed from WSRR to the run times. Be aware of this information and take appropriate action to successfully make those transitions.

A.4.1 IBM WebSphere Service Registry and Repository

To correctly govern a service in WSRR, several objects must be created and transitioned to the correct state. See 3.3, “Govern existing services” on page 56 for details.

A.4.2 IBM WebSphere DataPower

A DataPower Web Service Proxy can obtain WSDL and other associated documents (such as schema files) by subscribing to those documents stored and managed on a WSRR server.

See 12.5, “WSRR subscriptions and saved search subscription” on page 354 for details.

A.4.3 IBM Tivoli Composite Application Manager

ITCAM requires that the SLD is in the “governance subscribable” state and the monitoring policy must be in the correct governance state.

See Chapter 14, “ITCAM as policy monitoring point” on page 389 for details.

A.5 Creating Policies

Mediation polices, used to control DataPower, are created in the WSRR Business Space interface; see 8.2, “Creating a mediation policy” on page 264.

Monitoring policies, used to create situations in ITCAM for SOA, are created in the WSRR web UI; see 9.2.2, “Creating a monitoring policy” on page 282.

A.6 Attaching policies

A policy, whether it is a mediation policy, a monitoring policy, or a custom policy is of little use by itself. A policy is most useful when attached to something, implying that the rules of the policy should be applied to the object to which the policy is attached. In WSRR, a policy may be attached to any object. However for the attachment of a mediation policy to mean something to DataPower or ITCAM for SOA, it must be attached to either a service level agreement (SLA) or a service level definition (SLD). Chapter 10, “Attaching a policy to a service” on page 313 provides details.

A.7 Promotion within WSRR

A full production deployment consists of multiple registries to support the various stages of the service development and testing lifecycle. This means that you can keep the service metadata for your staging and production environments in separate registries to more effectively control what will happen when a service goes into production. Those services that are using the SOA service lifecycle can have a read-only WSRR for each environment. For more detail, see 3.3.6, “Promoting the services to WSRR run times” on page 68.

A.8 Running the client to validate the services

How to invoke the sample services is described in “Using the sample Web Service Test Client” on page 431. Ensure the endpoint is updated to use the DataPower Web Service Proxy, as described.

B

ITCAM monitoring attribute tables

This appendix provides information about the attributes that can be used in the monitoring policy. Although this appendix does not provide complete details for the attributes, it does have links to the information center for IBM Tivoli Monitoring where you can find additional information for the attribute tables. Descriptions for the main attributes per table that are normally considered for use in a SOA Policy Solution monitoring policy are provided.

This appendix contains the following topics:

- B.1, “Attribute table descriptions” on page 422
- B.2, “Endpoint Inventory attributes” on page 423
- B.3, “Fault Log_610 attributes” on page 423
- B.4, “Message Arrival Threshold_610 attributes” on page 424
- B.5, “Services Inventory_610 attributes” on page 425
- B.6, “Services Inventory Requester Identity_610 attributes” on page 426

© Copyright IBM Corp. 2013. All rights reserved.

B.1 Attribute table descriptions

The attribute table descriptions, with the exception of the Endpoint_Inventory_attributes, contain a short name with the title *Term reported for historical data*. This attribute name is the one you should use in the monitoring policy in WSRR. Subsequent sections of this appendix provide additional information for each attribute table.

Specifically, the SOA Policy Solution supports the following tables:

- Endpoint Inventory attributes: This attribute group describes the attributes that make up the Endpoint Inventory table, which is used in the Endpoint Performance Summary workspace. The Endpoint Inventory table contains metrics for a Service Endpoint. This table is populated when the agent receives metric files that contain an endpoint address.
- Fault Log_610 attributes: This attribute group defines the data that make up each SOAP fault that is received by the data collector. Use these attributes to create situations that monitor SOAP errors received by the data collector.
- Message Arrival Threshold_610 attributes: This attribute group represents the computed data table, which is used to create situations based on the arrival rate of messages that monitor message arrival data, such as the number of messages that arrive during a specified time interval.
- Services Inventory_610 attributes: This attribute group provides data about current service inventory. It also contains aggregate metric data. Use the Services Inventory_610 attributes to track your services inventory and ensure that settings do not reach or exceed predefined thresholds.
- Services Inventory Requester Identity_610 attributes: This attribute group augments the Services Inventory_610 attributes by providing further detail for viewing service metrics and their relationships. Use the Services Inventory Requester Identity_610 attributes to track the requester identities that are requesting your services, and to ensure that the use of services for any requester identity does not exceed predefined thresholds.

Attribute table release: The previous attribute tables may be used with ITCAM for Applications Release 7.2 or later. For Release 7.1.1, only the Fault Log, Services Inventory, and Services Inventory Requester Identity attribute tables may be used.

B.2 Endpoint Inventory attributes

The Endpoint Inventory attributes table requires ITCAM for Applications version 7.2 or later. A description for the Endpoint Inventory attributes can be found at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_user_guide_and_help/attr_kd43ei.html

The attribute names used are exactly the same as those for Services_Inventory_attributes, as shown in “Services Inventory_610 attributes” on page 425 except for the `new` field which describes the endpoint for the service. The Endpoint Address must be specified to use this table.

The Endpoint Address is the address of the service endpoint that the message was received on or set to. The valid format is an alphanumeric string, with a maximum of 256 characters. The short name is UENDPOINT.

This attribute table is like the Services Inventory table, except it is divided by endpoint. If the service has only one endpoint, then use the Services Inventory table. If, however, you need to delineate within multiple endpoints for a service, use the Endpoint Inventory attributes table. Make sure that the agent is actually filling in the Endpoint Address for this attribute table to be useful.

See “Services Inventory_610 attributes” on page 425 for additional information.

B.3 Fault Log_610 attributes

The Fault Log table is useful when a specific fault code or fault string is a cause for operations to take an action such as restarting a server, or is at least severe enough that operations should be notified.

A description of the table is at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_user_guide_and_help/attr_kd42et.html

Fields of particular interest are as follows:

- UFLTCODE: The SOAP fault code for which the message is to be rejected. If a specific fault code requires action, then specify this attribute in the policy.
- UFLTSTR: The SOAP fault string for which the message is to be rejected. If a specific fault string requires action, specify this attribute in the policy.

The following website includes the table and field names for the data warehouse, if the information will be stored long-term in ITCAM:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamsoa.doc/kd4ugmst211.htm?path=2_15_0_2_17_6#kd42ettable

B.4 Message Arrival Threshold_610 attributes

The Message Arrival Threshold attributes, as its name implies, is useful for monitoring the message arrival rate for a provider service.

A description of the table is at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_user_guide_and_help/attr_kd42at.html

Fields of particular interest are as follows:

- MSGLMT: The threshold for the number of messages that are allowed within the specified time interval.
- TMINT: The sliding time interval for which the condition is applied, specified in seconds. The minimum allowed interval is 60 seconds; the value specified is rounded up to the next closest 30-second increment.

This table is one of the few ITCAM tables that allows the user to specify a time interval. Using a 60-second time interval allows the message arrival rate per minute to be used. This is especially useful when there are busy times that need to be monitored at a finer granularity. In any case, the user can certainly specify a larger time interval if it makes sense to do so.

The following website includes the table and field names for the data warehouse, if the information will be stored long-term in ITCAM:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamsoa.doc/kd4ugmst213.htm?path=2_15_0_2_17_8#kd42attable

B.5 Services Inventory_610 attributes

The Services Inventory attributes are used most often in the monitoring policy, because it summarizes the metrics for the entire service. A description of the table is at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_user_guide_and_help/attr_kd42it.html

Fields of particular interest is as follows:

- AVGETIME: The average elapsed round-trip time, in milliseconds. This attribute is useful in situations where a service level agreement (SLA) is in place for a service for response time and the SLA is being violated.
- FLTCOUNT: The number of faults observed during this interval. If there are an excessive number of faults (In some cases, 1 is considered excessive), then operations should be notified.
- MAXETIME: The longest elapsed time, in milliseconds, of any message observed during this monitoring interval. Although the average amount of time will probably be used more often in monitoring policy, it is sometimes useful to also consider the highest response time. When this number becomes excessive, it indicates that a server is unable to keep up, and operations needs to be notified.
- MAXMSGLEN: The length of the longest message, in bytes, observed during this monitoring interval. This attribute might be useful in identifying that applications exist that are not sending the correct message headers, or can be indicative of a denial of service attack.
- MINMSGLEN: The length of the shortest message, in bytes, observed during this monitoring interval. This attribute might be useful in identifying that certain applications are not sending the correct message headers, or can be indicative of a denial of service attack.
- MSGCOUNT: The number of messages observed during this interval. This attribute is useful in situations where a service level agreement (SLA) is in place for a service for number of messages being received and that SLA is being violated.
- VLDRSPPCT: The percentage of response messages observed during this interval that were not faults. This attribute is useful when a percentage of valid transactions is expected.

The following website includes the table and field names for the data warehouse, if the information will be stored long-term in ITCAM:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamsoa.doc/kd4ugmst220.htm?path=2_15_0_2_17_15#kd42ittable

B.6 Services Inventory Requester Identity_610 attributes

The Services Inventory Requester Identity attributes are the same as the Services Inventory with the exception that it contains more rows. Use the Services Inventory Requester Identity attributes to keep track of the requester identities that are requesting your services and to ensure that their use of these services do not exceed predefined thresholds.

The Services Inventory Requester Identity table contains one row per unique combination of requester identity, service port name, service port namespace, operation name, operation namespace, and service type (provider or requester), for each five-minute time interval. It should be used when it is necessary to write monitoring policy on a consumer-provider pair.

A description of the table can be found at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_user_guide_and_help/attr_kd42jt.html

The following field is of particular interest:

- UREQID: The identity of the service requester. The format of this attribute is an alphanumeric text string. Its format will be as created by the agent creating the information for ITCAM. This field is mandatory; without entering it, the situation will not be created.

The following website includes the table and field names for the data warehouse, if the information will be stored long-term in ITCAM:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamsoa.doc/kd4ugmst221.htm?path=2_15_0_2_17_16#kd42jttable

C

Additional material

Additional material can be downloaded from the Internet as described in the following sections.

C.1 Locating the web material

The web material that is associated with this book is available in softcopy on the Internet from the IBM Redbooks web server:

ftp://www.redbooks.ibm.com/redbooks/SG248101/

Alternatively, you can go to the IBM Redbooks website:

ibm.com/redbooks

Select **Additional materials** and open the directory that corresponds with the IBM Redbooks form number, SG248101.

C.2 Using the web material

The additional web material that accompanies this book includes the following files:

- `RedbooksSampleServices.ear`

  EAR file for deployment to WebSphere Application Server with sample web services.
- `GovMasterWSRR.zip`

  WSRR objects to be imported in the your Governance Master WSRR
- `StagingWSRR.zip`

  WSRR objects to be imported in the your Staging WSRR
- `ProductionWSRR.zip`

  WSRR objects to be imported in the your Production WSRR

© Copyright IBM Corp. 2013. All rights reserved.

- `Policies.zip`

  Policies to be imported in the your Governance Master WSRR

- `SOA Policy Solution Business Case ROI.xls`

  Spreadsheet that is used to calculate the return on investment (ROI) for the SOA Policy Solution at IBM Redbooks Travel Company, as shown in Chapter 2.2, “Creating the business case” on page 28.

C.2.1 Downloading and extracting the web material

Create a subdirectory (folder) on your workstation, and download the contents of the web material file into this folder. Extract the `SG248101.zip` file into this folder to access the sample application files. Move the `SOA Policy Solution Business Case ROI.xls` file into a separate folder if you will be calculating the return on investment of using the SOA Policy Solution.

C.2.2 Deploy sample web services into WebSphere Application Server

Before you can deploy the `RedbooksSampleServices.ear` file, you must have an instance of a WebSphere Application Server profile. This profile can be the same one that you are using for WSRR, or a separate one. When writing this book, a separate profile in the same installation as WSRR was used. See the WebSphere Application Server information center for details about profiles:

`http://www14.software.ibm.com/webapp/wsbroker/redirect?version=matt&product=was-nd-dist&topic=TproProfiles`

Security does not have to be enabled for the server; the services still function if security is enabled.

Complete the following steps:

1. Assuming you created a profile and started the server, navigate to the administrative console, at a URL such as `http://myserver:9060/ibm/console/`.
2. On the tree menu on the left, navigate to **Applications** → **Application Types** → **WebSphere enterprise applications**. The Enterprise Applications page opens.
3. Click **Install**.
4. On the next page, click **Browse**, next to Full path, and go to `RedbooksSampleServices.ear` where you extracted your ZIP file.
5. Click **Next**.
6. On the next page, ensure Fast Path is selected and click **Next**.
7. Unless you have a reason to specify individual settings, click **Step 4**, and then click **Finish**.

8. After the application is successfully installed, a display similar to the one in Figure C-1opens. Click **Save**.

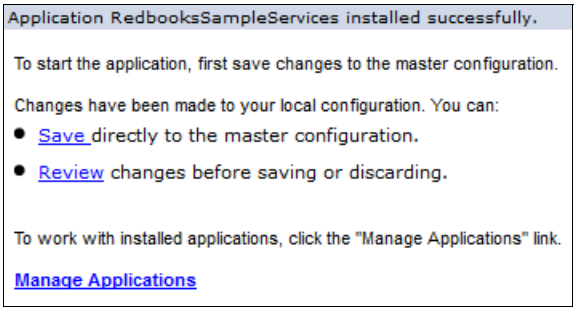


Figure C-1 Installation successful message

This application is now successfully installed. However, some additional configuration is required and then the application must be started.

9. Click **RedbooksSampleServices** as shown in Figure C-2.

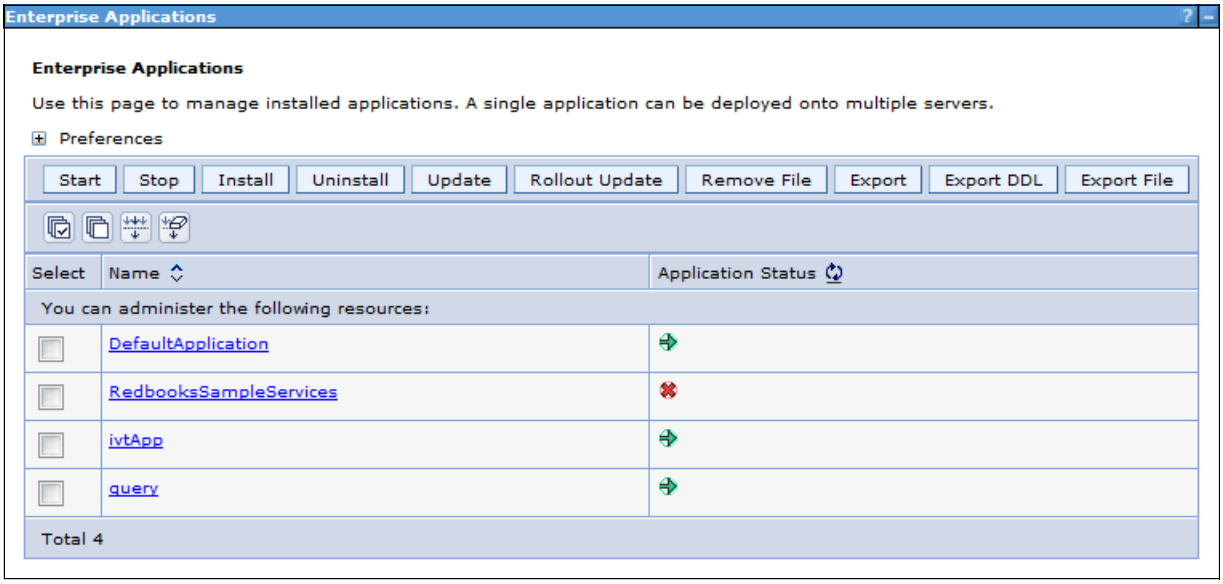


Figure C-2 List of installed applications in WebSphere Application Server

10. Under Modules on the right, click **Manage Modules**.

11. In the list of modules, as shown in Figure C-3, click **redbooksWeb**.

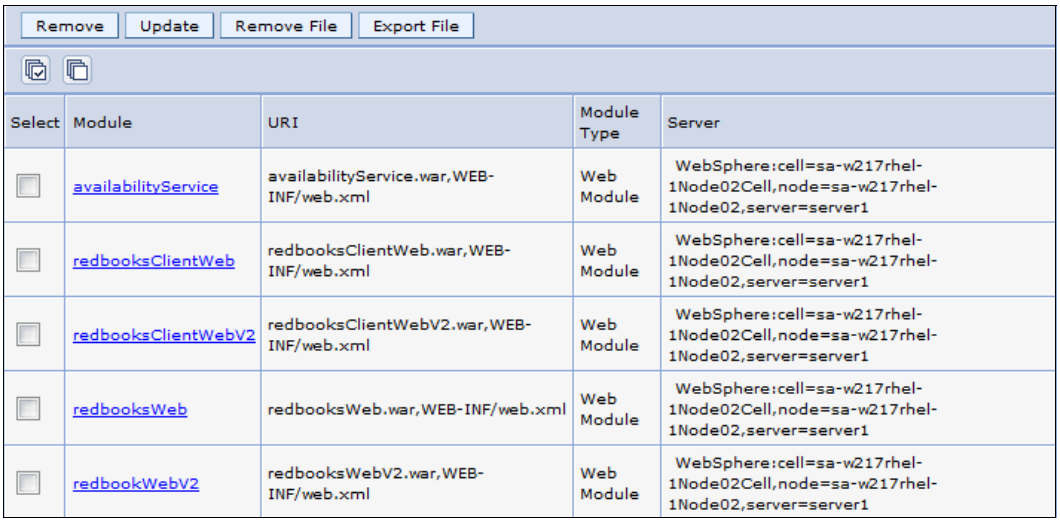


Figure C-3 List of modules in the RedbooksSampleServices application

12. Under Web Services Properties, click **Web services client bindings**.
13. The window shown in Figure C-4 displays. If this is not what you see, verify that you clicked the correct module.

Figure C-4 Web services client bindings for the redbooksWeb module

14. Under Port Information, click **Edit** to open the Port Information for Web service ItineraryAvailabilityService, as shown in Figure C-5.

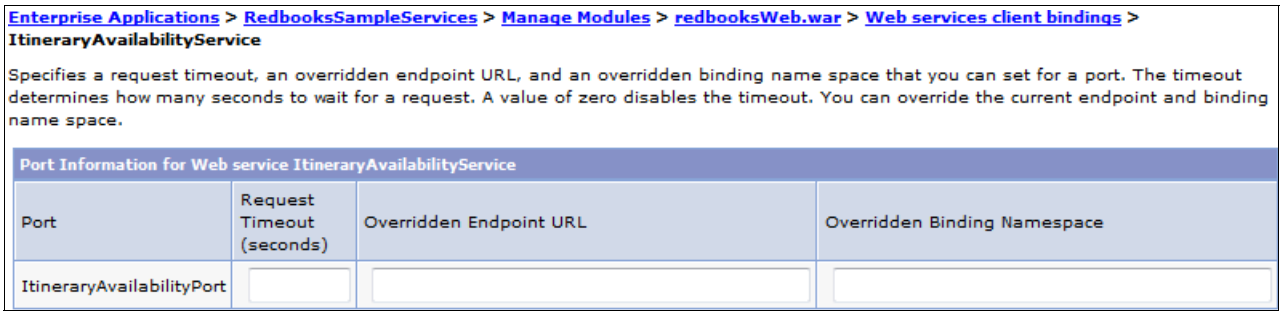


Figure C-5 Port Information for Web service ItineraryAvailabilityService

As mentioned previously, the Itinerary Reservation Service is a consumer of the Itinerary Availability Service. It is here that you specify the endpoint that it will use to do this. If you do not specify this, the Add operation on the Itinerary Reservation Service can fail.

15. If you set up DataPower and want to call the Itinerary Availability Service through DataPower, then under Overridden Endpoint URL enter the DataPower proxy URL, which is in the following format:

    `http://<datapowerHost>:<frontSideHandlerPort>/redbooksTravelAvailability/ItineraryAvailabilityService`

    Where `datapowerHost` is the host name of your DataPower appliance and `frontSideHandlerPort` is the port number of your front side handler, for example:

    `http://xi52.itso.ral.ibm.com:2081/redbooksTravelAvailability/ItineraryAvailabilityService`

    Alternatively, if you want the Itinerary Reservation Service to call the Itinerary Availability Service directly, then under Overridden Endpoint URL enter the service's actual URL, which is in the following format:

    `http://<wasHost>:<wasPort>/redbooksTravelAvailability/ItineraryAvailabilityService`

    In the URL, `wasHost` is the host name of the server where your WebSphere Application Server is running, and `wasPort` is the port number it is listening on for HTTP connections, for example:

    `http://sa-w217rhel-1.itso.ral.ibm.com:9080/redbooksTravelAvailability/ItineraryAvailabilityService`

16. Click **OK**.
17. Click **Save**.
18. Click **Enterprise Applications** in the breadcrumb.
19. Select the check box next to **RedbooksSampleServices** and then click **Start**.

You have now successfully deployed all of the sample services supplied with this book.

Using the sample Web Service Test Client

You can get access to web clients for the four services (Itinerary Availability Service, Itinerary Reservation Service, Pricing Service and Pricing Service V2) by using a URL in the following format:

`http://<wasHost>:<wasPort>/redbooksTravelClient/`

In this format, `wasHost` is the host name of the server where your WebSphere Application Server is running, and `wasPort` is the port number it is listening on for HTTP connections. Consider the following example:

`http://sa-w217rhel-1.itso.ral.ibm.com:9080/redbooksTravelClient/`

Figure C-6 shows the client in a Firefox browser.

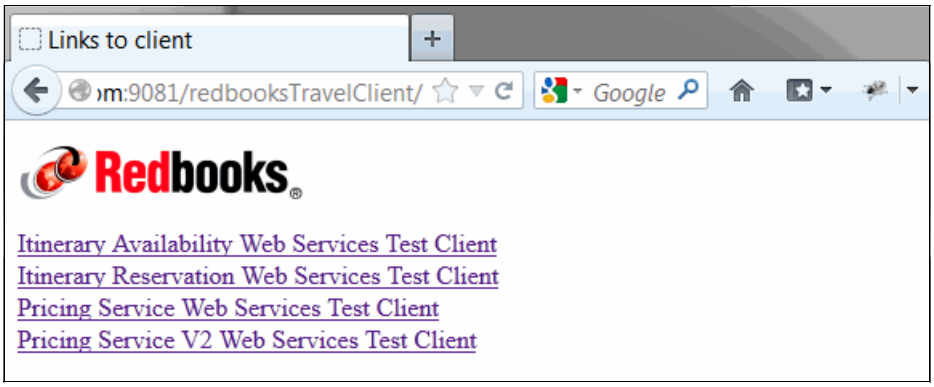


Figure C-6 Web client for the sample services

Clicking any of these links takes you to the client for each service, for example the Itinerary Availability Web Service Client shown in Figure C-7 on page 433. They are all similar and include most values as pre-completed, although not necessarily with the values you want.

The default value for the endpoint should only have the host name and port changed. If you want to call the service directly, then *localhost* is acceptable because it is relative to where the client is running, not your browser, and the client and server are running on the same WebSphere Application Server. If, however, you want to call through DataPower, you must change the endpoint's host and port to be that of your DataPower appliance, the Front Side Handler you configured.

For example, the following first line might become the second line:

```
http://localhost:9080/redbooksTravelAvailability/ItineraryAvailabilityService
http://xi52.itso.ral.ibm.com:2081/redbooks/ItineraryAvailabilityService
```

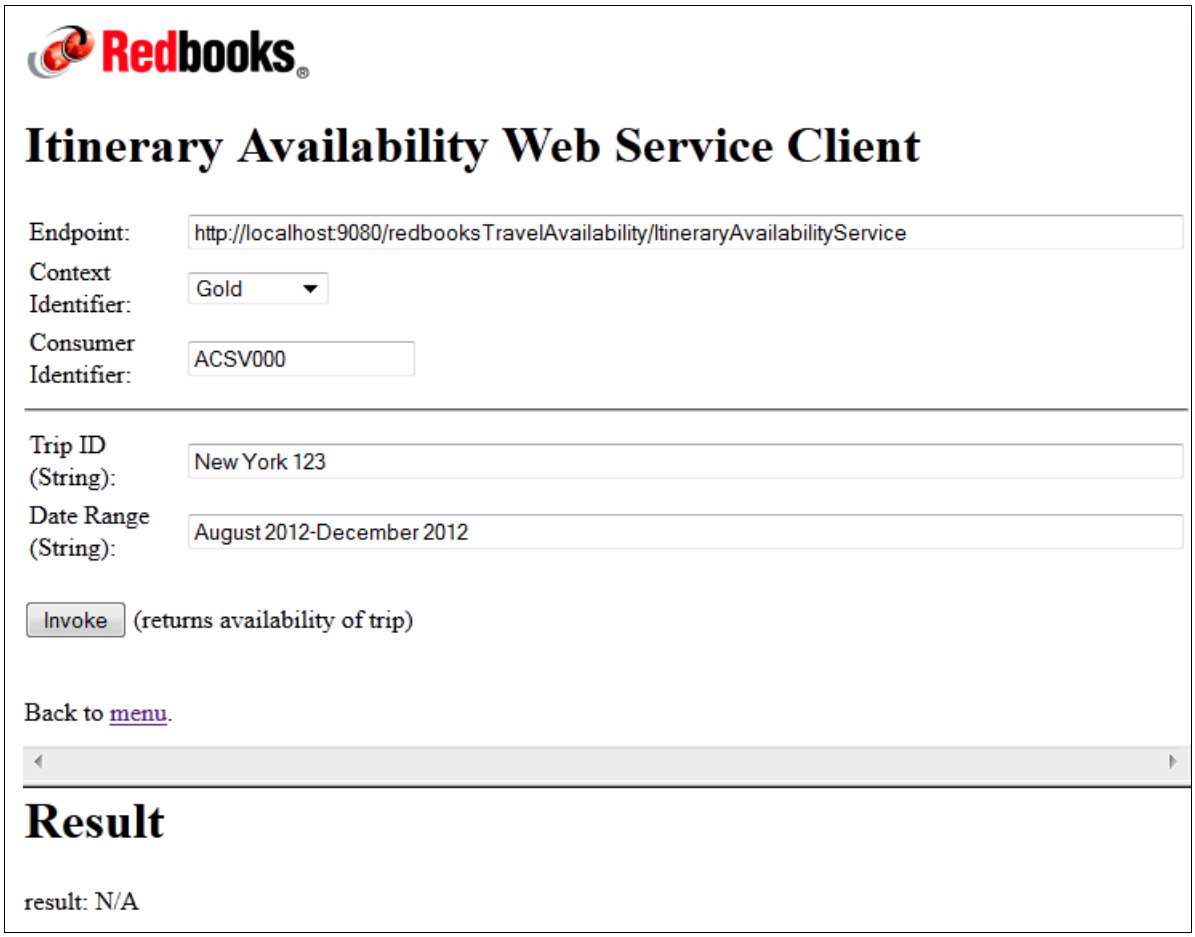


Figure C-7 Itinerary Availability Web Service Client

The client allows you specify the consumer identifier and context identifier that are inserted into the SOAP header.

The actual values that are used for all of the parameters of the operations are irrelevant except for the Number of people for the Pricing Service. This value of this parameter determines how long, in seconds, the service takes to respond to invocations.

C.2.3 Import objects into WebSphere Service Registry and Repository

The ZIP files (`GovMasterWSRR.zip`, `StagingWSRR.zip`, `ProductionWSRR.zip`, and `Policies.zip`) must be imported into the registries (Governance Master, Staging, and Production), as indicated in Table C-1.

Table C-1 Where to import WSRR files

File	WSRR to import into
`GovMasterWSRR.zip`	Governance Master
`StagingWSRR.zip`	Staging
`ProductionWSRR.zip`	Production
`Policies.zip`	Production

Complete the following steps in WSRR to import the objects. Repeat the steps for each file, ensuring you import each into the correct instance of WSRR.

1. Navigate to the web UI for the relevant instance of WSRR at a URL such as the following example:

   `https://<wsrrHost>:<wsrrPort>/ServiceRegistry/`

   In this URL, `wsrrHost` is the host name of the server running WSRR, and `wsrrPort` is the port number of the instance of WebSphere Application Server that is running WSRR, for example:

   `https://sa-w217rhel-2.itso.ral.ibm.com:9443/ServiceRegistry/`

2. Enter the login credentials and click **Login**.
3. If necessary, switch to the Administrator perspective in the top-right of the page.
4. Under Actions, select **Import,** click **Browse**, and select one of the ZIP files you want to import, for example the `StagingWSRR.zip` file.
5. Click **OK**. After a time, a panel similar to Figure C-8 opens. Do not be concerned if it shows only one file as being imported; all of the others were also imported.

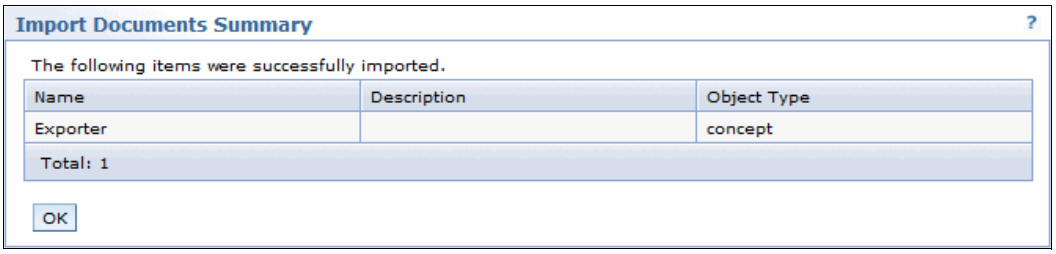


Figure C-8 Import Documents Summary

You are now finished loading files into WSRR and can follow the instructions in the book from the start, to attach policies and complete the set up of the sample environment.

C.3 Set up and use the Sample SOA Policy Pattern Business Case ROI spreadsheet

See the instructions in Chapter 2.2, “Creating the business case” on page 28 for constructing your own business case. Also see the sample results in Chapter 2.3, “Business case results” on page 29.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- *IBM WebSphere DataPower SOA Appliances Part II: Authentication and Authorization*, REDP-4364
- *WebSphere DataPower SOA Appliance: The XML Management Interface*, REDP-4446

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Online resources

These websites are also relevant as further information sources:

- IBM WebSphere DataPower Version 5 information center:

  http://pic.dhe.ibm.com/infocenter/wsdatap/v5r0m0/index.jsp
- IBM WebSphere Service Registry and Repository Version 8.0 information center:

  http://pic.dhe.ibm.com/infocenter/sr/v8r0/index.jsp
- IBM Tivoli Composite Application Manager for Applications information center:

  http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=%2Fcom.ibm.itcama.doc_7.1%2Fwelcome_itcamfapps71.html
- Configuring data collection, DataPower SOA Appliance:

  http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfapps_soa.doc_72/soa_install_guide/configdatapower.html
- WebSphere Service Registry and Repository Business Space:

  http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/twsr_getting_started_with_wsrr_business_space.html
- Service level definition:

  http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/rwsr_gep_service_level_definition.html
- Governance enablement profile, customizing runtime environments:

  http://pic.dhe.ibm.com/infocenter/sr/v8r0/topic/com.ibm.sr.doc/cwsr_runtime_env.html

© Copyright IBM Corp. 2013. All rights reserved.

- Lifecycles in the governance enablement profile:

  http://pic.dhe.ibm.com/infocenter/sr/v8r0/index.jsp?topic=%2Fcom.ibm.sr.doc%2Frwsr_gep_life_cycles.html

- Best practices for web services versioning:

  http://www.ibm.com/developerworks/webservices/library/ws-version

- Enhancements to integration with WSRR in ITCAM for SOA version 7.1.1 Fix Pack 3; the following site requires a user ID and password:

  http://www.ibm.com/developerworks/wikis/display/tivolimediagallery/Enhancements+to+integration+with+WSRR+in+ITCAM+for+SOA+version+7.1.1+Fix+Pack+3

- SOA governance using WebSphere DataPower and WebSphere Service Registry and Repository, Part 1: Leveraging WS-MediationPolicy capabilities:

  http://www.ibm.com/developerworks/websphere/library/techarticles/1204_burke/1204_burke.html

- SOA Operational Flow and Sun JVM:

  http://www-304.ibm.com/support/docview.wss?uid=swg21429283

- ITCAM for SOA Topology Workspaces unavailable after Tivoli Monitoring is upgraded to 6.2.3 or later:

  http://www.ibm.com/support/docview.wss?uid=swg21567013

- ITCAM for SOA: After remote deploy of the agent, the host name is lost and the agent does not function correctly:

  http://www-304.ibm.com/support/docview.wss?uid=swg21381579

- ITCAM for SOA workspace is unusable if ITCAM for SOA 7.1.1 or its fix packs is installed after Tivoli Monitoring 6.2.2 fix pack 4:

  http://www.ibm.com/support/docview.wss?uid=swg21504213

- ITCAM for SOA 7.1.x with a corrupted SDMS database:

  http://www.ibm.com/support/docview.wss?uid=swg21381100

- ITCAM for SOA: WSRR Event Handling Integration Fails:

  http://www.ibm.com/support/docview.wss?uid=swg21381160

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

SOA Policy, Service Gateway, and SLA Management

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

SOA Policy, Service Gateway, and SLA Management

Explore the SOA Policy Solution by using established examples

Learn how to centralize and reuse runtime policies

See how to dynamically apply standardized policies

This IBM Redbooks publication teaches you how to automate your runtime policy by using a centralized policy management system. The SOA Policy Solution provides a centralized policy administration, enforcement, and monitoring for runtime policies that enable traffic management for service level agreement enforcement, service mediation, and other customized policies. Policies can be defined once and reused among multiple services, thus enabling a standardized, consistent approach to a runtime policy that saves time and money for implementation and maintenance of non-functional requirements for the enterprise and assists with faster time to market.

Business users can use the SOA Policy Solution to help create the service level agreements for their business services to deliver on promises for business performance. IT Architects can use the SOA Policy Solution to architect the policy solution patterns that standardize the runtime policy usage at their organization. Developers select specific policy patterns to implement the non-functional requirements that are associated with their projects. Operations groups provide information about operation needs and create standardized monitoring policy for operational action at run time.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-8101-00 ISBN 0738437824