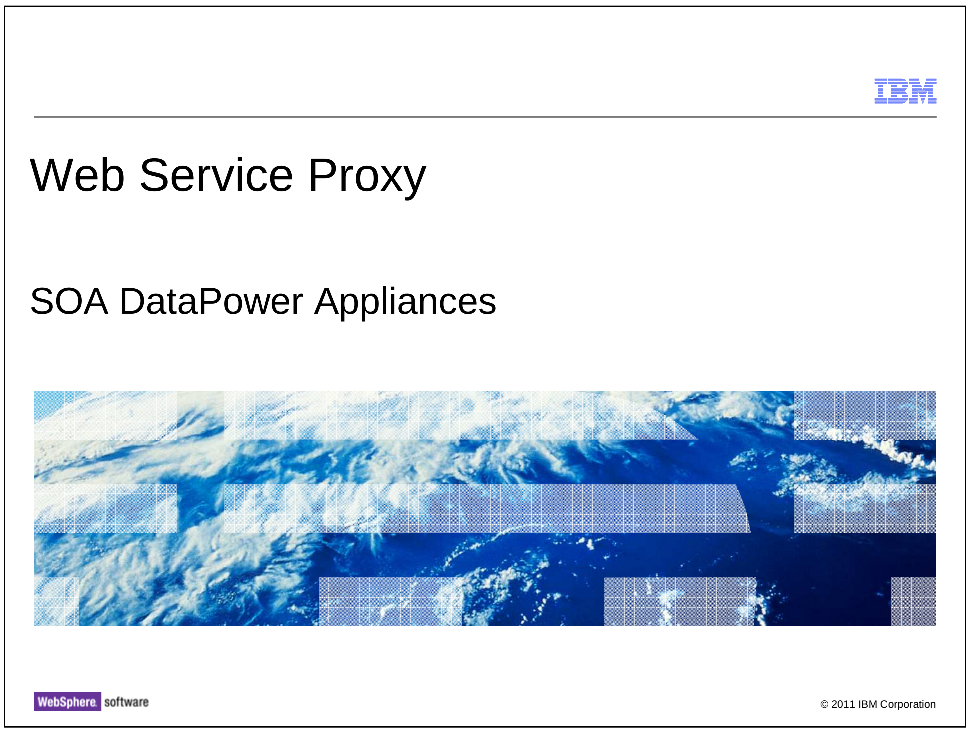


Web Service Proxy

SOA DataPower Appliances

WebSphere software

This presentation introduces the web Service Proxy of the WebSphere DataPower XE82.

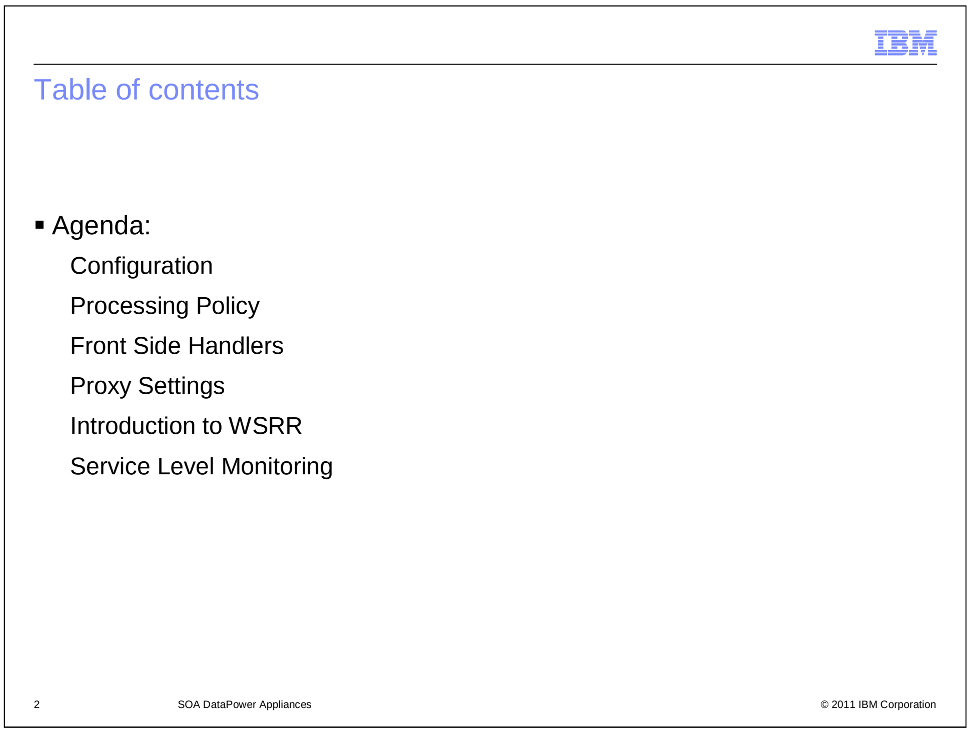
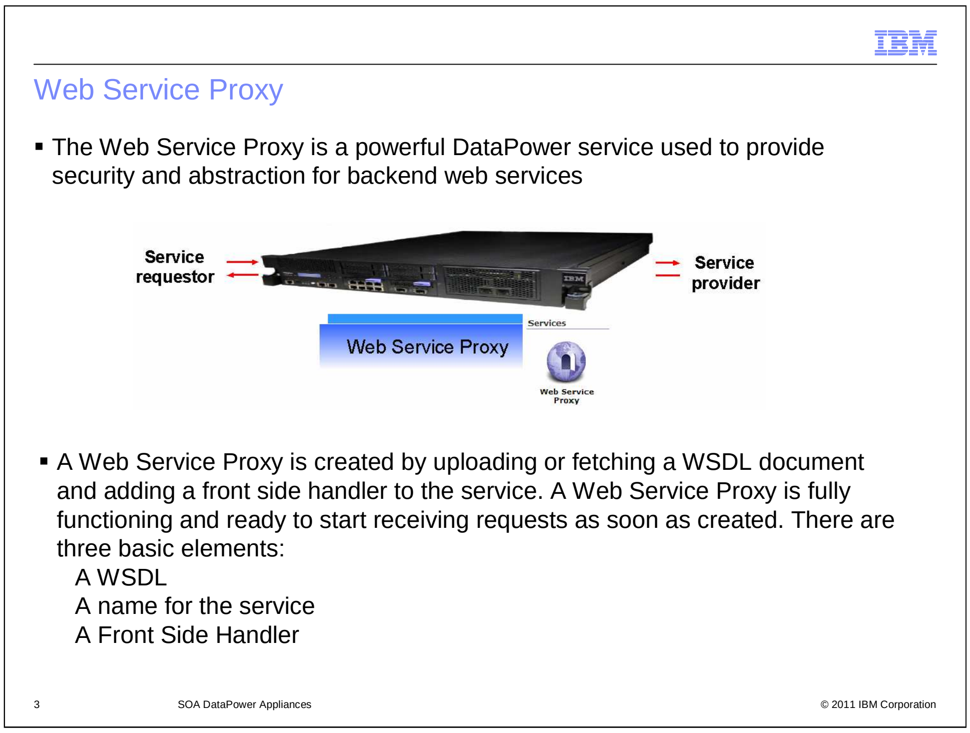


Table of contents

Introduction to Web Service Proxies and their components.

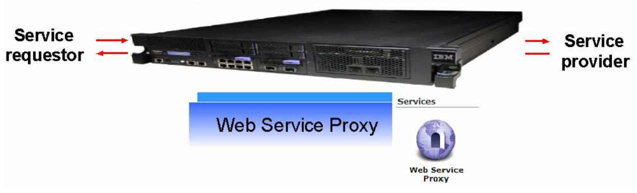
Objective: Describe a Web Service Proxy’s capabilities and provide examples of the most commonly used.

Basic knowledge of DataPower and basic Web Service terminology is assumed.



Web Service Proxy

- The Web Service Proxy is a powerful DataPower service used to provide security and abstraction for backend web services



- A Web Service Proxy is created by uploading or fetching a WSDL document and adding a front side handler to the service. A Web Service Proxy is fully functioning and ready to start receiving requests as soon as created. There are three basic elements:
  A WSDL
  A name for the service
  A Front Side Handler

The Web Service Proxy (WSP) is a powerful service that provides access to a variety of Web Service applications. Once a service is created and deployed, it is available to all the consumers that have access to it.

How do these consumers know how to call the service? Since web services are self describing, the self description is provided by the WSDL document (web services description language).

A WSDL document is an XML document in a specific format that tells the consumer everything needed to call the service.

There are several important characteristics of the Web Service Proxy. They include:

Feature rich service providing endpoint or URI abstraction,

Parser-based XML threat protection,

XML well-formedness checking,

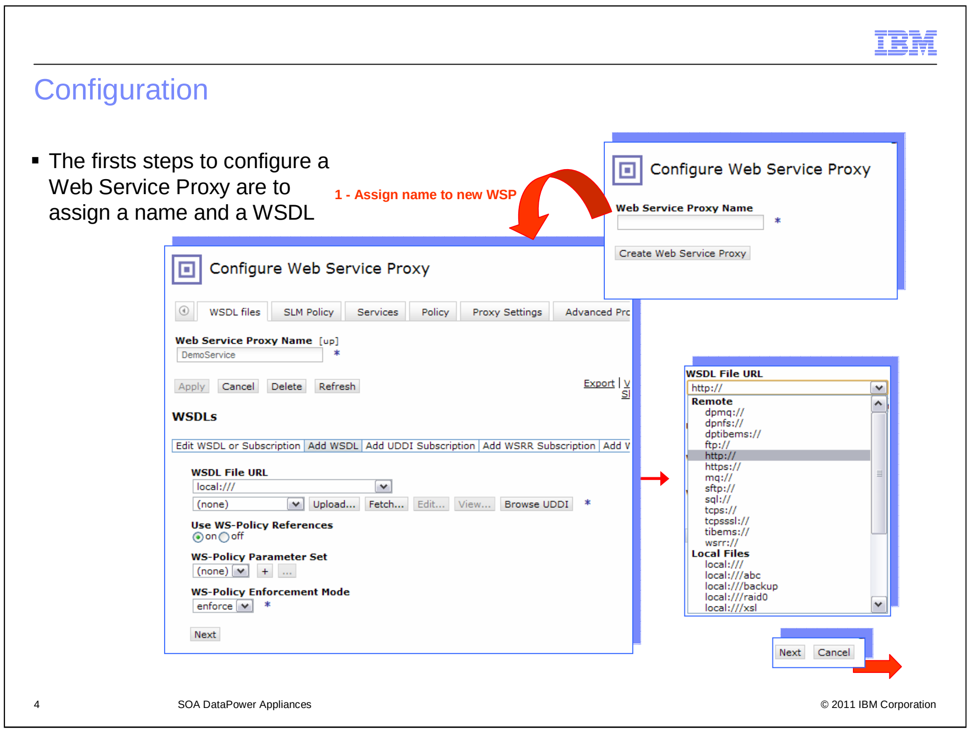
SOAP schema validation,

Payload schema validation,

Hooks for monitoring service execution, and a

Platform for building operation-level rules.

All this information is automatically available since the DataPower appliance uses the information provided within the WSDL such as endpoints, schemas, and operations to configure the Web Service Proxy.



Configuration

- The firsts steps to configure a Web Service Proxy are to assign a name and a WSDL

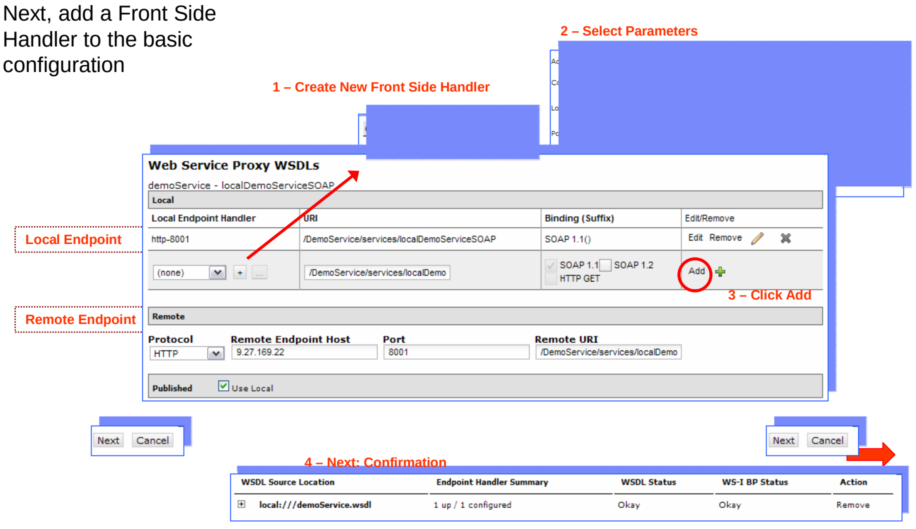
You can create a Web Service Proxy by navigating from the Control Panel in the WebGUI to the Web Service Proxy and by selecting Add. The wizard immediately prompts you for the first of the three main pieces of information required: the name. After you give the service a name, click the Create Web Service Proxy button to prompt for the WSDL, the next piece of information. The WSDL can be uploaded or fetched just as you do with any other file. Notice the option to upload and fetch the WSDL file from a remote location.

Other ways of associating a WSDL include browsing UDDI or adding UDDI and WSRR subscriptions. If required, there are also options for defining and enforcing WS-Policy sets.

WS-Policy is a specification that allows security policies to be defined for a web service that must be adhered to by the consumer. UDDI and WSRR are discussed later in this presentation.

Configuration (continued)

- Next, add a Front Side Handler to the basic configuration



Web Service Proxy WSDLs

demoService - localDemoServiceSOAP

Local

Local Endpoint Handler	URI	Binding (Suffix)	Edit/Remove
http-8001	/DemoService/services/localDemoServiceSOAP	SOAP 1.1()	Edit Remove
(none)	/DemoService/services/localDemo	✓ SOAP 1.1 ☐ SOAP 1.2 ☐ HTTP GET	Add

Remote

Protocol	Remote Endpoint Host	Port	Remote URI
HTTP	9.27.169.22	8001	/DemoService/services/localDemo

Published ✓ Use Local

WSDL Source Location	Endpoint Handler Summary	WSDL Status	WS-I BP Status	Action
local:///demoService.wsdl	1 up / 1 configured	Okay	Okay	Remove

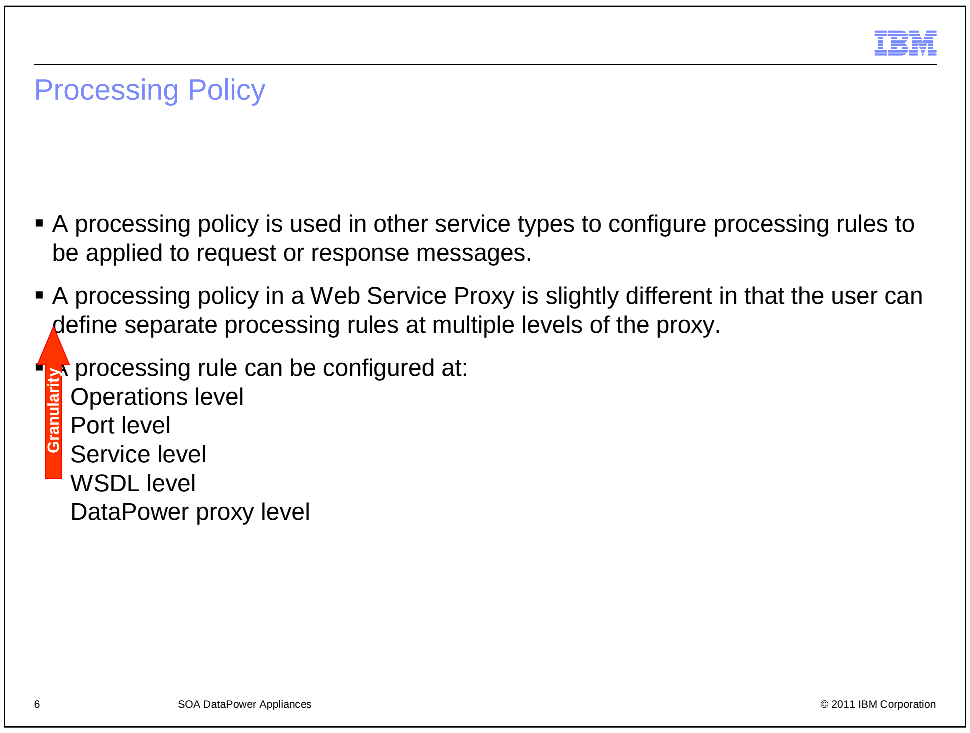
Click the Next button to create the Web Service Proxy. The final critical piece of information is the Front Side Handler that listens for requests to be processed by this service. Configure the Front Side Handler on the WSDL tab within the service configuration.

The top section is labeled local. This section is where you define the Front Side Handler or handlers which will listen for service requests.

The middle section is labeled Remote and defines the backend where ultimately the request is sent.

The bottom section on the Front Side Handler image is labeled published. This section defines the endpoint information that is published to the client if the DataPower appliance republishes this WSDL, for instance, to an external repository. This is typically configured to “use local” which means the published WSDL contains the endpoint information that you define in the top “local” section of this screen.

You only need to configure the local endpoint if you are ok with the remote provided by the WSDL. If not, you need to change the host information and URI, and protocol. An existing Front Side Handler can be selected or you can create a new one with the '+' button. After creating or selecting a Front Side Handler, you must associate a URI with it. When adding a Front Side Handler, do not forget to click the Add button!



Processing Policy

- A processing policy is used in other service types to configure processing rules to be applied to request or response messages.
- A processing policy in a Web Service Proxy is slightly different in that the user can define separate processing rules at multiple levels of the proxy.
- A processing rule can be configured at:
  - Operations level
  - Port level
  - Service level
  - WSDL level
  - DataPower proxy level

Although the simple service configuration produces a usable proxy for your web service, you will most likely want to add additional processing capabilities to your service such as AAA, message transformation, or encryption.

You are probably familiar with the DataPower appliance's processing policies and how it contains actions to be applied to a request, response, or in response to an error, but this service type offers more configuration options for the Processing policy.

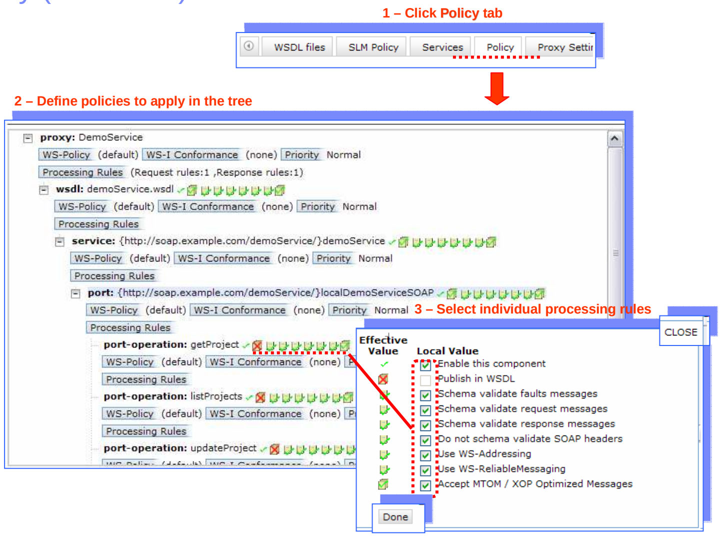
Web Service Proxy services allow separate processing rules to be configured at any level of the proxy. For example, you can apply a separate processing rule for each operation within the WSDL.

During processing, this rule hierarchy is evaluated and applied in the order shown. So for instance, if there's a request processing rule configured at the operation level and another rule configured at the WSDL level, the rule at the operation level is evaluated and started first.

When a Web Service Proxy service is created, a default request rule is created at the proxy level.

Processing Policy (continued)

- Add additional processing capabilities through the Policy tab of the WSP Configuration screen


1 – Click Policy tab
WSDL files
SLM Policy
Services
Policy
Proxy Settir
2 – Define policies to apply in the tree
3 – Select individual processing rules
CLOSE
Done

proxy: DemoService
WS-Policy (default) WS-I Conformance (none) Priority Normal
Processing Rules (Request rules:1 ,Response rules:1)
wsdl: demoService.wsdl
WS-Policy (default) WS-I Conformance (none) Priority Normal
Processing Rules
service: {http://soap.example.com/demoService/}demoService
WS-Policy (default) WS-I Conformance (none) Priority Normal
Processing Rules
port: {http://soap.example.com/demoService/}localDemoServiceSOAP
WS-Policy (default) WS-I Conformance (none) Priority Normal
Processing Rules
port-operation: getProject
WS-Policy (default) WS-I Conformance (none)
Processing Rules
port-operation: listProjects
WS-Policy (default) WS-I Conformance (none)
Processing Rules
port-operation: updateProject

Effective Value	Local Value	
✓	☑	Enable this component
☒	☐	Publish in WSDL
✓	☑	Schema validate faults messages
✓	☑	Schema validate request messages
✓	☑	Schema validate response messages
✓	☑	Do not schema validate SOAP headers
✓	☑	Use WS-Addressing
✓	☑	Use WS-ReliableMessaging
✓	☑	Accept MTOM / XOP Optimized Messages

In addition to configuring rules at the various levels of the WSDL, there are also options that can be configured at each level. You will notice a series of icons with check marks just to the right of each level of the processing policy tree.

Most of the options are self-explanatory; however, two might not be: the WS-* options WS-Addressing and WS-ReliableMessaging.

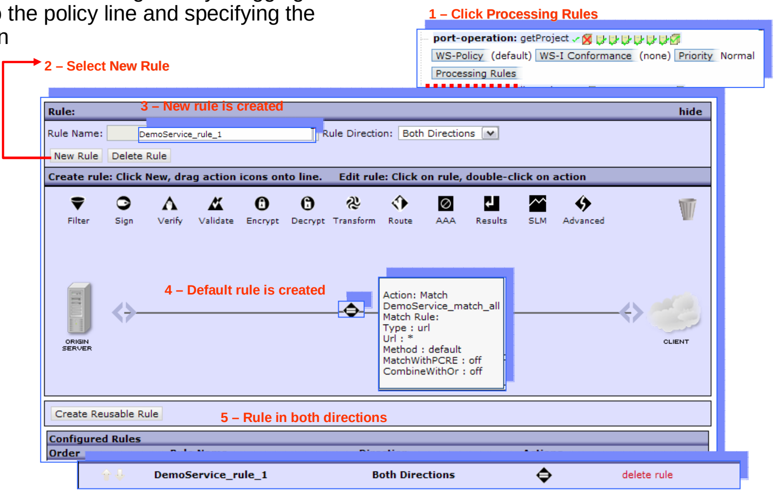
WS-Addressing is a specification that enables routing information to be carried in the SOAP header.

The second WS-* option is WS-ReliableMessaging, which allows messages to be delivered reliably between endpoints.

MTOM, or SOAP Message Transmission Optimization Mechanism, is a W3C recommendation for optimizing the electronic transmission of attachments.

Processing Policy (continued)

- Processing rules are configured by dragging actions onto the policy line and specifying the rule direction
- A default rule is created matching all URLs



port-operation: getProject
WS-Policy (default) WS-I Conformance (none) Priority Normal
Processing Rules

Rule:
Rule Name: DemoService_rule_1 Rule Direction: Both Directions
Create rule: Click New, drag action icons onto line. Edit rule: Click on rule, double-click on action

Action: Match
DemoService_match_all
Match Rule:
Type : url
Url : *
Method : default
MatchWithPCRE : off
CombineWithOr : off

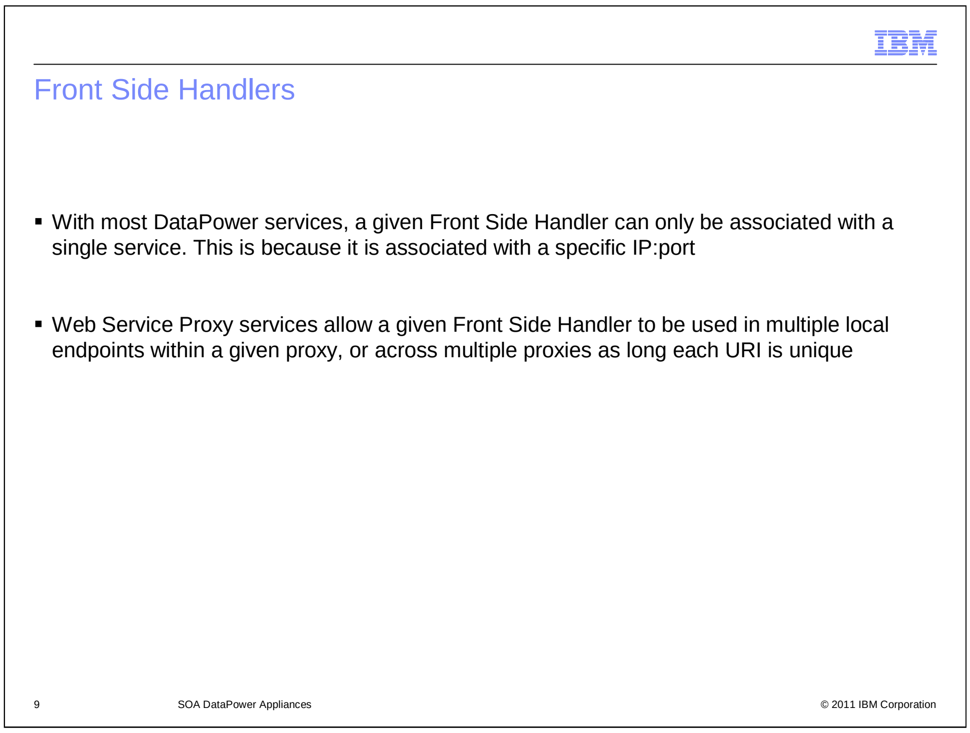
Configured Rules

Order	Rule Name	Direction	Actions
	DemoService_rule_1	Both Directions	

Processing rules are configured by dragging actions onto the policy line and specifying the rule direction.

Notice that a match rule is still required within the processing rule. A default match rule is created that matches all URLs. Though only messages destined for this service operation will reach this rule, you can further filter the inputs by creating a more specific matching rule.

After configuring the processing rule and clicking on the Apply button, the new rule is added to the service for only that specific operation.



Front Side Handlers

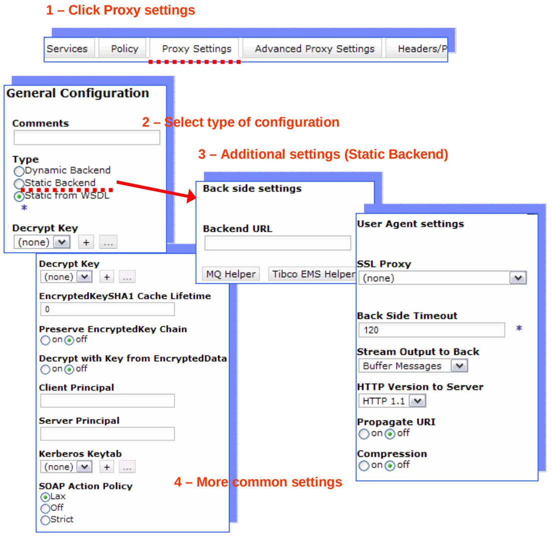
- With most DataPower services, a given Front Side Handler can only be associated with a single service. This is because it is associated with a specific IP:port
- Web Service Proxy services allow a given Front Side Handler to be used in multiple local endpoints within a given proxy, or across multiple proxies as long each URI is unique

9 SOA DataPower Appliances © 2011 IBM Corporation

An important thing to remember about configuring the Front Side Handler in your Web Service Proxy is that it can be reused. This is different from many other services you configure where your Front Side Handler must be used by only one service. It is possible for the Front Side Handler to be reused because the Web Service Proxy does not only use the IP:port combination to listen for requests. The service also includes the URI specified in the configuration and the operation being requested. This means that a Front Side Handler can be reused in the same or in a different web service proxy within a single domain as long as the combination of the IP address, port, URI, and operation are unique. This is a common thing to do when your WSDL contains multiple bindings.

Proxy Settings

- DataPower automatically creates a basic Web Service Proxy configuration by parsing a WSDL. However, some configuration options cannot be derived from a WSDL.
- Endpoint selection:
  - "Static from WSDL" will use the one from the WSDL file.
  - "Static Backend" will require to enter the Backend URL field.
  - With "Dynamic Backend", the backend server is decided within the processing policy as each request is processed
- Decrypt Key
  - In a Web Service Proxy, decrypt key is needed because WSDL compliance test is not possible if request message is encrypted.
- SOAP Action Policy
  - Indicates how strictly header specified in the WSDL will be enforced



General Configuration

Comments:

Type: Dynamic Backend / Static Backend / [✓ Static from WSDL] *

Decrypt Key: (none)

Decrypt Key: (none)

EncryptedKeySHA1 Cache Lifetime: 0

Preserve EncryptedKey Chain: on [✓ off]

Decrypt with Key from EncryptedData: on [✓ off]

Client Principal:

Server Principal:

Kerberos Keytab: (none)

SOAP Action Policy: [✓ Lax] / Off / Strict

Back side settings

Backend URL:

User Agent settings

SSL Proxy: (none)

Back Side Timeout: 120 *

Stream Output to Back: Buffer Messages

HTTP Version to Server: HTTP 1.1

Propagate URI: on [✓ off]

Compression: on [✓ off]

As you have seen, the WSDL referenced by the Web Service Proxy describes many of the details for calling the actual web service, enabling the proxy to automatically produce a base configuration. There are, however, many configuration options within the service that cannot be derived from the WSDL and that can be set at the proxy level. Many of the configuration options can be found in the proxy settings tab on the service configuration screen.

When you create Web Service Proxy, the endpoint listed in the WSDL is used to automatically populate the endpoint, or backend, within the service.

There are times when you want your service to forward requests to a different endpoint.

"Static Backend" is useful if you have multiple bindings, each requiring an endpoint definition and you want all requests to go to one single endpoint (example: http://hostname.example.com).

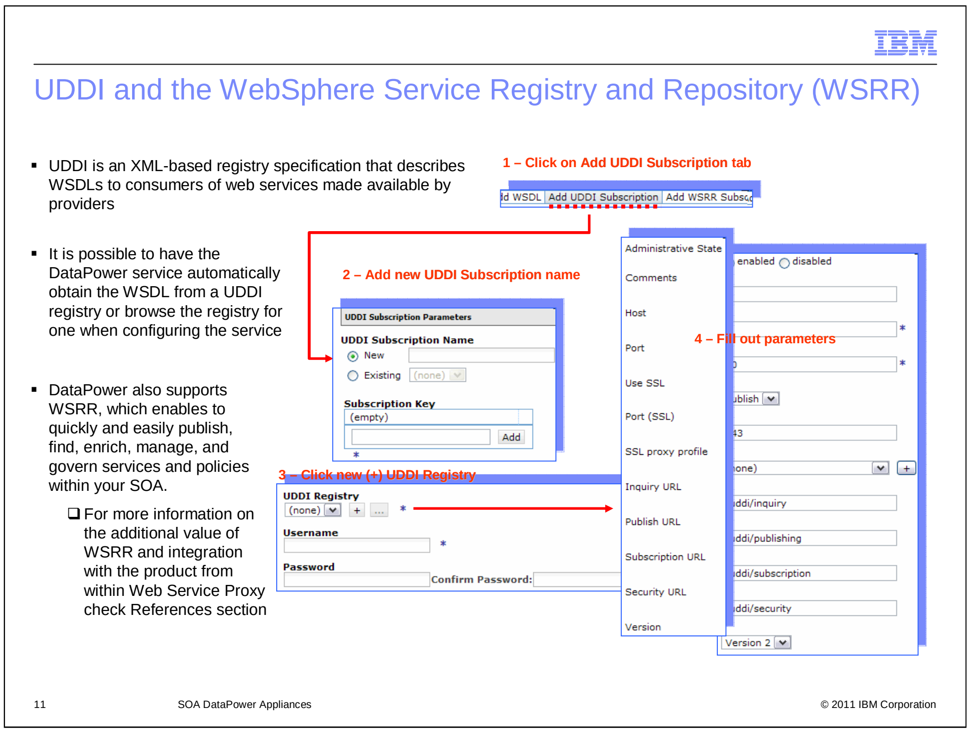
"Dynamic Backend" indicates that the backend server is determined within the processing policy as each request is processed. The flexibility of the processing policy makes it easier to implement. For example, you might have a different backend for each operation within the web service.

In all other types of services, decryption is performed with a decrypt action that is added to the request rule. This is also possible to do in a Web Service Proxy service. However it presents some issues. Before the message even gets to the Processing Policy, it is validated for WSDL compliance, which includes schema and operation validation. If the request message is encrypted, the WSDL cannot be validated.

Configuring a decrypt key within the service provides the service the appropriate key to decrypt the request message. This key is used to decrypt the message as it enters the service before it is validated for WSDL compliance. Now, the decrypted message can be validated and passed to the processing policy.

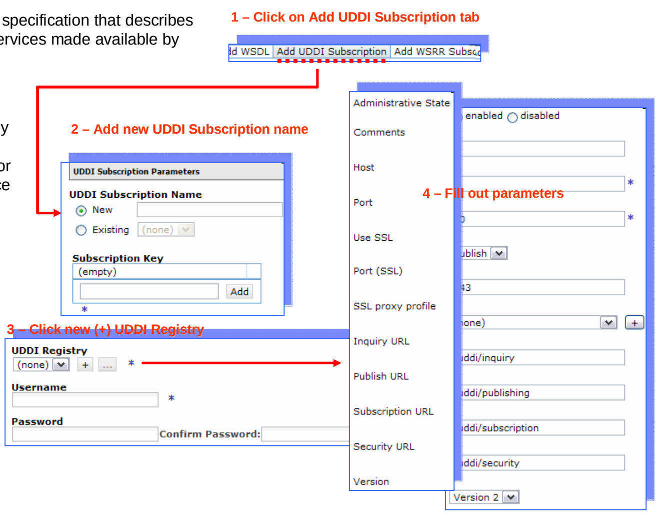
Often a WSDL specifies a soapAction parameter with a URI as its value. This indicates that a soapAction HTTP header with a corresponding value is sent along with all requests for that particular operation. The purpose of this header is to indicate the intent of the request; however, there are no restrictions on what this value must or must not be as long as it is a URI.

The option on the proxy settings tab allows you to define how strictly this header specified in the WSDL is enforced. The default value is "Lax" which indicates that an empty header is considered a match. The other two options are "strict", which indicates that the request must contain the soapAction header with an exact match, and "Off", which does not check the header at all.



UDDI and the WebSphere Service Registry and Repository (WSRR)

- UDDI is an XML-based registry specification that describes WSDLs to consumers of web services made available by providers
- It is possible to have the DataPower service automatically obtain the WSDL from a UDDI registry or browse the registry for one when configuring the service
- DataPower also supports WSRR, which enables to quickly and easily publish, find, enrich, manage, and govern services and policies within your SOA.
  - ❑ For more information on the additional value of WSRR and integration with the product from within Web Service Proxy check References section



11 SOA DataPower Appliances © 2011 IBM Corporation

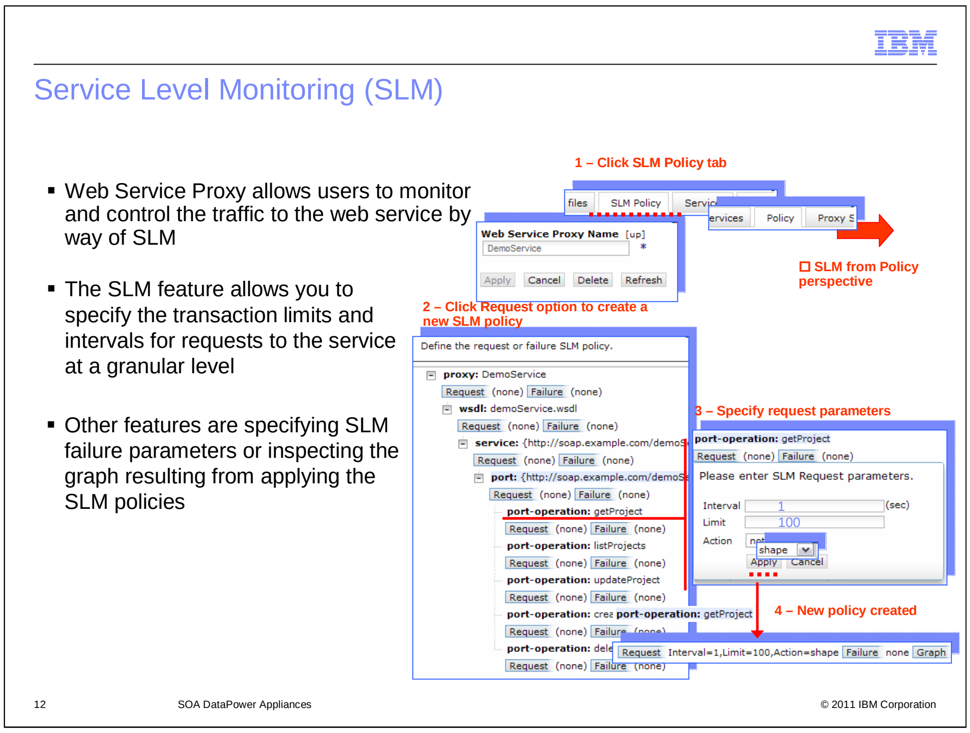
A WSDL describes a web service and provides all of the required information to the consumer. But how does the service provider publish this WSDL so people who want to consume the service can find this web service and its operations?

UDDI, or Universal Description Discovery and Integration, is an XML-based registry specification that describes how web service providers can publish a WSDL and how consumers can find it. This registry can be made available over the Internet or private networks and serves as a directory for web services.

It is possible to have the DataPower service automatically obtain the WSDL from a UDDI registry or to browse the registry for one when configuring the service.

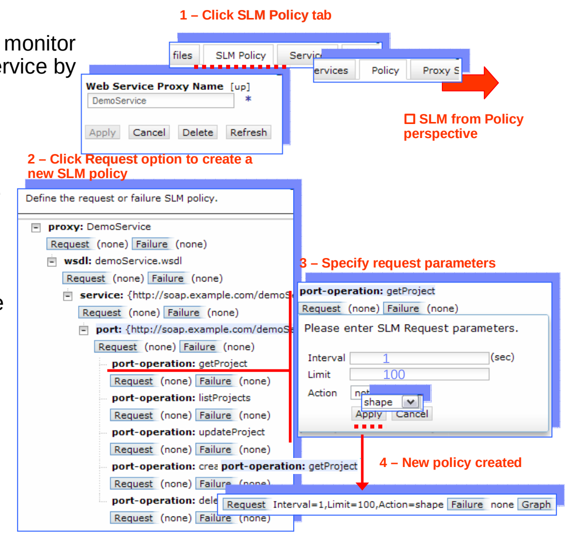
To have the DataPower appliance automatically obtain a WSDL from the UDDI registry, you must add a UDDI subscription within your service. A UDDI subscription provides the information that enables the DataPower service to retrieve the WSDL and to be notified of any updates to it. Although the UDDI subscription defines all the parameters for subscribing to a particular WSDL, it does not define the UDDI registry connection details. This is defined within the UDDI Registry.

The WebSphere Service Registry and Repository (WSRR) establishes a central point for finding and managing service metadata acquired from several sources. The sources might include service application deployments and other service metadata and endpoint registries and repositories, such as UDDI. It is where service metadata that is scattered across an enterprise is brought together to provide a single, comprehensive description of a service. Once this happens, visibility is controlled, versions are managed, proposed changes are analyzed and communicated, usage is monitored, and other parts of the SOA foundation can access service metadata with the confidence that they have found the copy of record. WSRR is an implementation of UDDI with additional features and enhancements and is intended to overcome several limitations.



Service Level Monitoring (SLM)

- Web Service Proxy allows users to monitor and control the traffic to the web service by way of SLM
- The SLM feature allows you to specify the transaction limits and intervals for requests to the service at a granular level
- Other features are specifying SLM failure parameters or inspecting the graph resulting from applying the SLM policies



Web Service Proxy Name [up]
DemoService

Define the request or failure SLM policy.

proxy: DemoService
Request (none) Failure (none)
wsdl: demoService.wsdl
Request (none) Failure (none)
service: {http://soap.example.com/demoS
Request (none) Failure (none)
port: {http://soap.example.com/demoS
Request (none) Failure (none)
port-operation: getProject
Request (none) Failure (none)
port-operation: listProjects
Request (none) Failure (none)
port-operation: updateProject
Request (none) Failure (none)
port-operation: crea
Request (none) Failure (none)
port-operation: dele
Request (none) Failure (none)

port-operation: getProject
Request (none) Failure (none)
Please enter SLM Request parameters.
Interval 1 (sec)
Limit 100
Action shape

port-operation: getProject
Request Interval=1,Limit=100,Action=shape Failure none Graph

At this point you have configured a Web Service Proxy, including processing policies and WSDL management. After this service is up and running, you might be interested in monitoring and controlling the traffic to the service.

This is made possible by a Service Level Monitor, or SLM, action that is added to your service by default. After you created your Web Service Proxy service, a request and response rule are automatically created for you at the Proxy level.

The Response rule is just an empty rule with a Match Rule and a Results action, but if you look at the request rule, the request rule contains an SLM action that looks like a simulated line graph that will enable the monitoring of the service. The nice thing about the default SLM action is that there is really no configuration required for it. The only configuration required for the SLM action is the SLM Policy. This is created by default and added to the default SLM action upon creation.

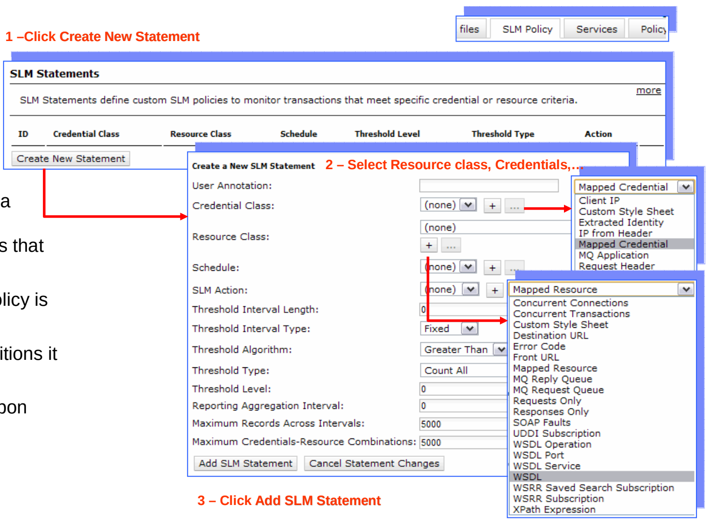
Much like the Policy tab, the SLM screen contains a hierarchy of all the WSDLs within the service down to the operations. Here it is possible to specify the transaction limits and intervals for the requests to the service at a granular level. The interval indicates the number of seconds that the transactions is counted, and the Limit indicates the maximum number of transactions for the number of seconds specified in the Interval field. So if you want to limit the number of transactions for a given operation to 100 transactions in one second span, specify 100 as the limit and 1 as the interval.

The action can be set to indicate whether the traffic should be shaped or throttled, or to write a message to the log. Each of these options can be very useful for serving different purposes. The notify action, which writes a message to the log might be used for keeping track of the peeks in traffic for capacity planning. The shape action can be used to protect your backend server from bursts of traffic over a longer period of time. The throttle action might be used to enforce a maximum number of transactions allowed in a given period of time from a specific consumer. In addition to being able to monitor and control the request traffic to the service, you can also do the same for the transactions that fail.

Let's supposed you performed a load test on your demonstration service and determined that the getProject operation is resource intensive and that the saturation or break point of the operation is reached when it receives 100 transactions per second. With this information, you can decide how to you will configure the Service Level Monitor. Because you know that it is only the getProject operation that causes the resource issue on the web service, set a limit for this operation. You must now decide what should happen after the limit is reached. Since you do not want any transactions to get discarded, you set the action to shape. Now, when the transaction limit is reached, the subsequent requests are queued and finally released to the backend at a rate that does not exceed the defined limit. This allows for a spike in traffic to be handled that   otherwise bring down the backend service or server.

Custom SLMs

- Custom SLM objects allow SLM policies to have even more control
- An SLM statement is a user-defined set of conditions and actions that specify:
  - ❑When an SLM policy is executed
  - ❑Under what conditions it will be executed
  - ❑Actions to take upon execution
  - ❑Thresholds



1 –Click Create New Statement

SLM Statements

SLM Statements define custom SLM policies to monitor transactions that meet specific credential or resource criteria. more

ID	Credential Class	Resource Class	Schedule	Threshold Level	Threshold Type	Action

Create New Statement

Create a New SLM Statement

2 – Select Resource class, Credentials,…

User Annotation:	
Credential Class:	(none)
Resource Class:	(none)
Schedule:	(none)
SLM Action:	(none)
Threshold Interval Length:	0
Threshold Interval Type:	Fixed
Threshold Algorithm:	Greater Than
Threshold Type:	Count All
Threshold Level:	0
Reporting Aggregation Interval:	0
Maximum Records Across Intervals:	5000
Maximum Credentials-Resource Combinations:	5000

Add SLM Statement · Cancel Statement Changes

3 – Click Add SLM Statement

Finally, there are custom SLM Statements. Located at the bottom of the SLM tab of the SLM configuration there is a section labeled SLM Statements. This is where you can define any custom SLM statements to be applied to transactions flowing through the service.

After clicking on the create new statement button, the configuration window opens for the new SLM statement. There are many more options than just the Interval, Limit, and Action.

To demonstrate the configuration options and how to configure this statement, let's take the example where there is an external company that uses your web service. Instead of doing one operation at a time, the company stores all operations (create projects) and kicks off a batch process at the end of the week to submit them through the create project operation. The concern is that this batch process might have an impact to other users of the service. This can be solved by adding an SLM statement to your SLM policy.

The figure shows the SLM statement configuration screen.

The first field is the user annotation, which is a place to enter a comment for this statement.

The Credential Class field refers to a credential class object that can represent a specific user credential for which the SLM policy is enforced (you can create one or select a pre-existing one)

The Mapped Credential is the result of the mapped credential phase in an AAA policy

The Extracted Identity is the result of the Extract Identity phase in an AAA policy

The Client IP is the IP address of the client making the request

The Custom Stylesheet specifies a custom XSLT stylesheet to extract the credential

The IP from Header is the IP address specified in the HTTP header

The Request Header uses the value of a specified request header.

In this example, the create project operation contains an AAA policy that extracts the identity of the client making the requests. Because the users making the request always send the same credentials, you can choose the Extracted Identity for the Credentials Type Field

The next field is the match type. Here you can specify how you want to match the credential, whether an exact match is required, a regular expression to match with, or to allow the DataPower appliance to extract and store each unique credential for the specified type. In our example you know the credential that is passed, so choose Exact for this field.

Next, the Resource Class object allows you to specify the type of request this SLM statement will apply to. This is a very exhaustive list to choose from. And because you are concerned with the create project operation, you chose the WSDL operation from the drop down.

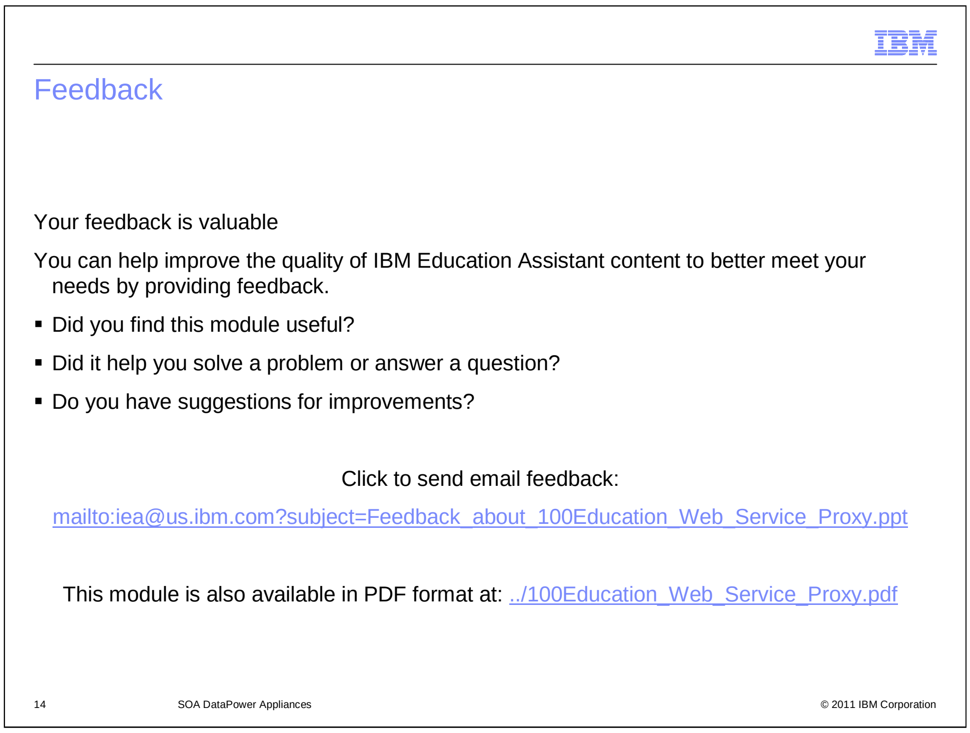
You need to choose the Match Type as well. In this case, indicate that this is a regular expression match and just provide the operation name.

The other options that you can configure are these:

Schedule: Assume that this batch process happens once every week, on a Saturday at 8 PM. In an effort to restrict any real time requests from this entity during the week, specify that this statement applies to requests on a specific day and time. At this point you have scoped this SLM statement down to the requestor, the operation, and the day and time. You can specify the actual SLM thresholds and actions.

In this SLM statement you can specify the same type of actions as you did in our first SLM policy. That is, you can specify what to do when a threshold is exceeded.

Then specify threshold interval length. This is the length in seconds that the transactions will be measured. So, to restrict this process to 20 transactions per second, specify 1 to represent the 1 second. The threshold level should be set to 20.



Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send email feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_100Education_Web_Service_Proxy.ppt

This module is also available in PDF format at: ../100Education_Web_Service_Proxy.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, disclaimer, and copyright information